FAST ONLINE CONSTRAINED OPTIMIZING CONTROLLERS

A THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF MASTER OF PHILOSOPHY
IN THE FACULTY OF SCIENCE AND ENGINEERING

2018

By
AWO KING-HANS
School of Electrical and Electronic Engineering

Contents

Nomenclature **9**

Abstract **11**

Declaration **12**

Copyright **13**

Acknowledgements **14**

1 Introduction **15**
1.1 Background and motivation 15
1.2 Recent trends in Fast Model Predictive Control 17
1.3 Objective of the Thesis 20
1.4 Thesis Outline 20
1.5 Contributions 21
1.6 Publications 23
1.7 Summary 23

2 Optimization **24**
2.1 Unconstrained optimization 25
2.1.1 Optimality Conditions 26
Necessary conditions 26
Sufficient conditions 26
2.2 Constrained optimization 29
2.2.1 Optimality conditions 30
2.2.2 Quadratic programming 31
2.2.2.1 Interior point for convex QP 32
2.2.2.2 Active set for convex QP 36

Comparison of algorithms 39
2.3 Summary 40

3 IMC Anti-windup techniques **41**
3.1 Windup and Directionality 41
3.1.1 Windup 41
3.1.2 Directionality 43
3.2 Internal Model Control (IMC) 45
3.2.1 IMC Design 47
3.2.1.1 Step 1: Model Factorization 48
3.2.1.2 Step 2: Robustness Filter Design 48
3.3 Modified IMC Anti-windup (MIMC) 49
3.3.1 Factorization 1 51
3.3.2 Factorization 2 52
3.3.3 Summary of the Modified IMC algorithm 52
3.4 TMIA Control Structure 54
3.4.1 TMIA control algorithm 59
3.5 Extension to Rate Constraints 60
3.5.1 Rate saturation 60
3.5.2 TMIA extension 61
3.6 Summary 63

4 TMIA control of a Quadruple Tank process **65**
4.1 The Quadruple Tank Process 65
4.1.1 Tank configurations 67
4.1.2 Hardware Peripherals 68
4.2 Control Platforms 69
4.2.1 Quanser Real-Time Control (QUARC) 70
4.2.2 Siemens PLC 72
4.3 Plant Model 72
4.4 TMIA control design 73
4.4.1 Minimum phase 73
4.4.2 Non-minimum phase 76
4.5 Summary 80

5 PLC implementation **81**
5.1 Siemens S7 PLC . . . 81
5.1.1 PLC communication setup . . . 83
5.1.1.1 Program Coding and download using the PC/PG interface . . . 84
5.1.1.2 HMI via MATLAB-OPC communication . . . 85
5.1.2 Program Structure . . . 87
5.1.3 PLC code . . . 88
5.1.4 Optimization of memory utilization . . . 89
5.1.5 QP Algorithm implementation . . . 91
5.1.6 Computational considerations . . . 91
5.2 PLC implementation results . . . 92
5.3 Summary . . . 95

6 Experimental Results - Quadruple Tank Process **96**
6.1 Performance . . . 96
6.1.1 Minimum phase . . . 97
Profile 1: . . . 97
Profile 2: . . . 100
Profile 3: . . . 100
6.1.1.1 Minimum phase comparison . . . 105
6.1.2 Non-minimum phase . . . 105
6.2 Comparison with MPC . . . 115
6.3 Summary . . . 118

7 Conclusions and Future work **119**
7.1 Conclusions . . . 119
7.2 Future directions . . . 120
7.2.1 PLC coding using Text-based programming . . . 120
7.2.2 Rate constrained TMIA controller for fast systems . . . 120
7.2.3 Comparison with MPC . . . 121

Bibliography **122**

List of Tables

5.1 Comparison of QP algorithms (minimum phase). 92

6.1 Performance comparison Profile 1 (minimum phase) 105

List of Figures

2.1 Convex and Non-convex set 25

3.1 Anti-windup compensation 42
3.2 Directionality compensation 43
3.3 Conventional IMC Structure 46
3.4 IMC Antiwindup Structure 50
3.5 Modified IMC Antiwindup Structure 51
3.6 TMIA Control Structure 55
3.7 Input rate block diagram 61
3.8 Modified TMIA controller 63
3.9 Snippet of TMIA controller modification in MATLAB 63

4.1 Schematic diagram of the Quadruple Tank process [1] 66
4.2 Quadruple Tank process connections - minimum phase [2]. 67
4.3 Snapshot of the Quadruple Tank process with connections to a PLC and Quanser hardware. Top right corner is a Siemens SIMATIC S7-300 CPU 314C-2 PN/DP PLC. Top left corner is a Quanser Data Acquisition board. Top middle consists of Analog Sensors Adapter (above) and Voltage Amplifier (beneath). 69
4.4 Communication setup with MATLAB-QUARC platform. 70
4.5 SIMULINK block diagram of TMIA control of plant in real time using the QUARC environment 71
4.6 TMIA implementation with additional bandwidth filter 74
4.7 Singular value plots 75
4.8 Simulation of closed loop responses for minimum and non-minimum phase. 79

5.1 S7-300 CPU 314C-2 PN/DP PLC. 83
5.2 PLC communication setup. 84

5.3 Sample Siemens PLC program in run time showing online diagnostics. 85
5.4 Real time MATLAB-OPC communication with PLC via KepserverEX5 86
5.5 OPC Read/Write to KepserverEX5 channels. 87
5.6 PLC code call structure. 90
5.7 PLC Memory usage (TMIA controller using Interior point method) . . 93
5.8 PLC code: Number of QP iterations per scan cycle and TMIA program execution time for OB35 block (Interior point algorithm). 94
5.9 PLC online diagnostics: Longest and shortest execution time 94

6.1 Controller comparisons for Profile 1 (TMIA vs IMC) 98
6.2 Controller comparisons for Profile 1 (TMIA vs MIMC) 99
6.3 Controller comparisons for Profile 2 (TMIA vs IMC) 101
6.4 Controller comparisons for Profile 2 (TMIA vs MIMC) 102
6.5 Controller comparisons for Profile 3 (TMIA vs IMC) 103
6.6 Controller comparisons for Profile 3 (TMIA vs MIMC) 104
6.7 Profile 1: Real-time implementation of TMIA controller for non-minimum phase. 106
6.8 Profile 1: Real-time implementation of MIMC controller for non-minimum phase. 107
6.9 Profile 1: Real-time implementation of IMC controller for non-minimum phase. 108
6.10 Profile 2: Real-time implementation of TMIA controller for non-minimum phase. 109
6.11 Profile 2: Real-time implementation of MIMC controller for non-minimum phase. 110
6.12 Profile 2: Real-time implementation of IMC controller for non-minimum phase. 111
6.13 Profile 3: Real-time implementation of TMIA controller for non-minimum phase. 112
6.14 Profile 3: Real-time implementation of MIMC controller for non-minimum phase. 113
6.15 Profile 3: Real-time implementation of IMC controller for non-minimum phase. 114
6.16 Quadruple Tank control in minimum phase with MPC controller ($P = 20$, $M = 2$) and TMIA controller. 116

6.17 Quadruple Tank control in non-minimum phase with MPC controller ($P = 100$, $M = 2$) and TMIA controller. 117

Nomenclature

Acronyms

CPU	Central Processing Unit
IMC	Internal Model Control
IO	Input-Output
KKT	Karush Kuhn Tucker
MIMC	Modified Internal Model Control
MIMO	Multiple Input Multiple Output
MPC	Model Predictive Control
MPI	Multi-Point Interface
MV	Manipulated Variable
OLE	Object Linking and Embedding
OPC	OLE for Process Control
PAC	Programmable Automation Controller
PID	Proportional Integral Derivative
PLC	Programmable Logic Controller
PRBS	Pseudo Random Binary Sequence
SISO	Single Input Single Output
TMIA	Two-stage Multivariable IMC Anti-windup
QP	Quadratic Program

Mathematical Symbols

:	such that
$\rightarrow$	tends to
$\in$	is an element of
$\mapsto$	maps to
:=	defined as
$\mathfrak{R}$	Real space
$\mathfrak{R}^n$	Real space of dimension n
$\forall$	for all
$\exists$	there exists

Abstract

Research in recent years has focused on the application of more advanced control technologies in industrial controllers, however the low computing power of standard industrial controllers has limited its implementation to higher end hardware. In multivariable input-constrained plants, actuator saturation causes two major problems for control engineers namely windup and directionality.

This research focuses on a Two-stage Multivariable IMC Antiwindup (TMIA) structure for open-loop stable plants which requires minimal computing power and tackles the aforementioned control problems in an intuitive and easy to tune way. The highlight of this IMC-based structure is the solution of two low-order quadratic programs to control both steady-state and transient behaviours of the plant. The TMIA structure is further developed to handle constraints on the input rate in a simple but effective way.

The controller is tested by application to a Quadruple Tank process in both minimum and non-minimum configurations controlled by a PLC. Although the focus of this paper is on computation and not performance, the TMIA structure is found to outperform its IMC counterparts in handling windup and directionality. Comparison of the TMIA controller and Model Predictive Controller is also carried out and shows competing results, hence is a suitable alternative for this class of systems.

Results on computation obtained demonstrate the realizability of the advanced control technique on an off-the-shelf low-end industrial PLC using three different quadratic programming methods. Worst case computation time is in the region of 5ms using the projected fast gradient method, this shows that the controller embedded on the PLC can be applied to much faster processes. Thus the TMIA structure is presented as a competitive alternative for input-constrained multivariable plants in terms of tuning transparency and reduced computations compared to other advanced control techniques such as MPC which are limited by the low computational power offered by standard PLCs.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

I am grateful to the Almighty God for His grace and sustenance through this program. I say a big thank you to my husband, Roy, and my son, Jed, for their constant support and love. I am immensely grateful to my supervisor, Dr. William Heath for putting up with me, for his sound guidance, invaluable insight and understanding. I say thank you to Razak Allie-Oke for his collaboration on this project and technical advice. I will not fail to acknowledge my sponsor, Petroleum Technology Development Fund (PTDF) for all their moral and financial support in this project.

Chapter 1

Introduction

1.1 Background and motivation

All real world control applications involve magnitude or rate constraints on actuators. If these constraints are not accounted for in the controller design, it could lead to poor transient response, closed-loop performance degradation and even instability [3]. At these constraint limits, saturation occurs and there is a mismatch between the requested controller output and the achievable plant input. The significant difference causes an accumulation of error which must be offset in the opposite direction long enough for the controller action to return to normal. This results in transients/overshoots which must decay before the system returns to a linear regime and the controller is said to 'windup'.

In the design of analytical dynamic controllers such as Internal Model Control, a common approach is to first design a linear controller neglecting the constraints which satisfies all the closed-loop performance requirements and then add a saturation compensation scheme to ensure graceful performance degradation of the closed-loop when the constraints become active. These ad hoc methods are usually known as *antiwindup compensation* [3–6].

In multivariable systems, a further problem associated with actuator saturation is called *input directionality*. Unlike SISO systems, the MIMO gain depends on the direction of the input vector [7]. A process exhibits directionality when the saturated controller output yields a system response that is not 'closest' to the system response of the unconstrained controller output [8]. The problem of directionality compensation is calculating a feasible plant input on the basis of a given unconstrained controller output. In [8, 9], an optimal directionality compensation problem is presented as a finite-time horizon, state dependent, constrained quadratic optimization problem. Its objective is to minimize the distance between the output of the unsaturated plant with an ideal controller and the output of the saturated plant with a directionality compensator. This problem is similarly dealt with under the name of *input allocation* with an extensive survey in [10] on control allocation algorithms for linear and non-linear models.

The problems of windup and directionality can be overcome with Model Predictive Control where the constraints are explicitly accounted for and the controller action is the solution to a constrained optimization problem [9]. MPC applications have gained widespread use in industrial process especially in refining and petrochemical industries [11]. However, so-called robust MPC design techniques increase the controller computational and memory requirements and long-horizon MPC is usually deployed on expensive dedicated digital computers.

There is current research into fast online MPC implementation [12–14] on platforms such as Programmable Logic Controllers (PLCs) [15, 16], Programmable Automation Controllers (PACs) [17, 18], FPGAs [19, 20] and other embedded devices [21]. In practice, PLCs offer the least computing capacity in the range of MHz processing power and a few KB of memory to several MB in the higher end PLCs but are still most widely used in industry for control tasks because of their robust operation even in harsh conditions, reliability and ease of maintenance [15]. In [15, 16, 21] the

PLCs used are in the range of several MBs hence there is need to test other online optimization algorithms on platforms with memory in the KB range and processor clock frequencies in the MHz range.

In [22], the authors report a hardware-in-the-loop simulation of an online quadratic optimization program and in [23] an MPC implementation is carried out on a SISO plant both using low end PLCs. An IMC-based online optimizing anti-windup control structure for multivariable plants that simultaneously handles the issues of windup and directionality is proposed in [24]. This Two-stage Multivariable IMC Antiwindup (TMIA) control structure does not require the receding horizon computation of MPC and may serve as an alternative to MPC which is computationally less expensive and more transparent in terms of tuning and robustness.

This is particularly attractive hence forms basis for the development, implementation and testing carried out in this thesis. The algorithm can be easily implemented on a cheap digital industrial platform as is demonstrated in this thesis. Simulation results presented in [24] show that the TMIA controller competes favourably with long horizon MPC (where the number of optimization variables and constraints is a multiple of the prediction horizon [10]) while only requiring the computation capacity of a single horizon MPC.

1.2 Recent trends in Fast Model Predictive Control

Model predictive control is widely used in the process industries where plants have slow time constants and sample times are large. It involves solving a finite horizon constrained optimal control problem for the current state of the plant at each time step [25]. The theory of MPC as well as stability studies is well established in the literature [11, 26–28]. The main strength of MPC is dealing with MIMO and nonlinear systems in a systematic way and major weakness being the high computational requirements

which is a limitation to real time implementation.

Work is being carried out at various research institutes to enable MPC to be used for fast sampled systems, commonly known as 'Fast MPC'. Practical applications of MPC require that storage and computation needs are taken into account in the controller design. Two common methods of implementing the MPC controller are the use of explicit MPC and online MPC. In the former, the control action is pre-computed and stored offline while the latter involves online optimization.

Explicit MPC seeks to move the computational burden of online optimization offline to obtain an explicit control law which is stored on a look up table which is cheap and easy to implement [29–33]. A survey of some explicit methods can be found in [34] and a Multi-parametric toolbox has also been developed in MATLAB for implementing explicit MPC [35]. Explicit MPC offers the advantage of high sampling rates for high speed systems since the optimal solution is pre-computed. Another major advantage is the ability for stability, robustness, performance and closed loop feasibility pre-processing analysis to be carried out especially for safety critical systems [25].

The main disadvantage is the size of the partition and computation grows exponentially with the size of the problem which limits its use to relatively small problems and as such may require approximation of the explicit solution. In [29], the state space partition is represented by a search tree consisting of orthogonal hypercubes where the optimal solution is computed explicitly using quadratic programming only at these vertices, an approximate solution based on this data is then computed for the whole hypercube. A combination of both explicit and online paradigms in [36] is introduced to overcome both individual limitations.

Online MPC has the advantage of being applicable to all problem sizes but limited when the applied to high speed systems required fast sampling. Recent trends have been focused on computation reduction by exploiting the structure of the problem or by warm starting techniques which reduce the number of iterations by starting the

optimization at a good initial point [12,37–41]. Most solutions for fast implementation of linear MPC are based on interior point, active set and fast gradient methods. In [12], the effect of exploiting sparsity structure, warm starting and early termination of the quadratic program is examined.

An online scheme based on Nesterov's fast gradient method [42] is presented in [43] with both cold and warm starting techniques where the same fixed sequence of controls is provided at each time step in the former and in the latter, an initial iterate is obtained based on the solution from the previous time step. The benefits of the warm starting technique is quantified in terms of computational complexity.

There is now need for research to be carried out to produce more efficient algorithms in order to meet the challenge of implementing MPC on embedded platforms with memories in the KB range and processor clock frequencies in the MHz range. Previous works which focus on implementation considerations for MPC on embedded hardware are presented in [15, 44–48] and some impressive results using custom reconfigurable Field Programmable Gate Array (FPGA) architecture for solving QPs have been reported in [49, 50].

Some efficient online solvers for embedded platforms are now readily available such as but not limited to FORCES [38], qpOASES [37] and CVXGEN [51]. In [52] an efficient solver is presented which optimizes linear algebra for implementing linear MPC on various embedded devices such Intel atom (found in netbooks), ARM processor (found in smart phones) and a high end ABB 4MB PLC processor with C-code capability. The solver is compared with the popular FORCES (Fast Optimization for Real-time Control on Embedded Systems) code [38] on the three devices.

In the light of the surrounding literature and ongoing research, the digital platform used in this research which is a Siemens S7-300 series PLC has memory capacity of less than 200KB and about 1.7MHz CPU processing for floating point operations.

1.3 Objective of the Thesis

1. To implement a fast online optimizing controller on a digital platform with memory in the KB range and processor clock frequency in the MHz range.

2. To compare and make recommendations on quadratic program algorithms suitable for fast online optmization on such devices.

3. To highlight considerations that need to be applied to maximize the limited computing capacity of such devices.

4. To compare the controller used with standard Model Predictive Control.

1.4 Thesis Outline

This thesis is organized in 7 chapters.

Chapter one gives a background to the control problem and the motivation for the research. It also discusses the limitations of recent research in this area and why the results in this thesis are relevant.

Chapter two introduces the concept of constrained quadratic optimization and discusses algorithms which will be implemented in the controller for simulations and real time experiments.

Chapter three discusses the problems associated with actuator constraints such as windup and directionality. It also explains the development of an existing IMC-based optimization technique which handles this problem effectively. Further development of the original technique is carried out to handle actuator constraints on both input magnitude and input rate. This is a novel contribution.

Chapter four presents a quadruple tank process as a multivariable process for real time control. The TMIA controller is designed for the model of the process to be

implemented on 2 control platforms. The practical application of the control scheme on the test rig is an original contribution.

Chapter five describes the set up of the PLC, relevant considerations to take into account when implementing on the PLC and the main body of results. It also highlights the limitations of the programming standard used, compares computation results for three QP algorithms and makes recommendation for a suitable algorithm based on th results obtained. The application of the TMIA controller on an industrial PLC using ladder logic programming for quadratic optimization is novel and a valuable result of the thesis.

Chapter six presents results on the performance of the TMIA controller on the PLC compared with classical IMC methods. The TMIA controller implemented on the QuaRC platform is also compared with Model Predictive Control.

Chapter seven contains the main conclusions and recommendations for further work.

1.5 Contributions

The contributions of this research are contained in Chapters 3, 4, 5 and 6.

In **Chapter three** a Two Stage Multivariable IMC Antiwindup controller was modified and extended to incorporate both input magnitude and rate constraints for multivariable processes thus providing a complete simple but effective design. The extension of this optimizing antiwindup technique is novel.

In **Chapter four** the TMIA controller was applied to an experimental Quadruple Tank process and compared with classical IMC methods and Model Predictive control. The TMIA controller was designed for both minimum and non-minimum phase configurations of the experiment. The application of the design technique to a real plant is a novel contribution as only simulation studies were carried out in the literature.

Chapter five is the main focus of the results in this thesis. the TMIA design was coded using a Siemens SIMATIC S7-300 CPU 314C-2 PN/DP PLC. The PLC program was implemented in ladder logic using 3 different algorithms namely interior point, active set and projected fast gradient method for the quadratic programs of the TMIA controller. Extensive results were obtained based on computation requirements and suitability of the 3 algorithms.

Three main novel contributions are presented in this chapter. First the results show that a simple and efficient advanced controller like TMIA controller can be realized on a standard PLC which is an affordable industrial alternative to MPC for input-constrained multivariable processes.

Secondly, three algorithms were coded on the PLC and projected fast gradient method was found to be the most suitable QP algorithm for the PLC implementation due to its least computing requirements. The memory utilization and execution time in the region of a few milliseconds using show that the TMIA controller on the PLC can be applied to much faster processes.

Thirdly, the results show that though difficult, advanced algorithms like quadratic optimization can be coded on a PLC from scratch using ladder logic for ease of understanding and debugging by operators and technicians in industry.

In **Chapter six** results based on performance of the TMIA controller and other IMC techniques are compared. Also results based performance of the TMIA controller using a QuaRC platform are compared against MPC.

An initial draft of this work was presented at the 2014 IEEE Multi-Conference on Systems and Control, Nice, France [53].

1.6 Publications

- A. R. King-Hans, W. P. Heath, and R. Alli-Oke, Two-stage Multivariable IMC Antiwindup (TMIA) control of a quadruple tank process using a PLC, in Proc. IEEE Multiconference on Systems and Control, Nice, 2014, pp. 16811686.

1.7 Summary

This chapter has summarized the background and motivation for this work citing relevant material related to fast online optimization and their implementation on embedded devices. The structure of the thesis and main contributions including published papers are presented.

Chapter 2

Optimization

Optimization according to [54] can be defined as *"the science of determining the 'best' solutions to certain mathematically defined problems, which often are models of physical reality. It involves the study of optimality criteria for problems, the determination of algorithmic methods of solution, the study of the structure of such methods, and computer experimentation with methods both under trial conditions and on real life problems"*.

Optimization problems involve maximizing or minimizing an objective or cost function which is dependent on one or more variables. These variables may or may not be subject to constraints and are hence called *constrained* or *unconstrained optimization* problems respectively. This chapter focuses on a subset of optimization problems known as *convex quadratic programming* and highlights some methods of solution. Convex optimization deals with a problem where the objective function is convex and its constraints define a feasible convex set.

A set $\mathscr{A}$ is convex [55] if for all $a, b \in \mathscr{A}$ and for all λ such that $0 \leq \lambda \leq 1$,

$$\lambda a + (1-\lambda)b \in \mathscr{A} \tag{2.1}$$

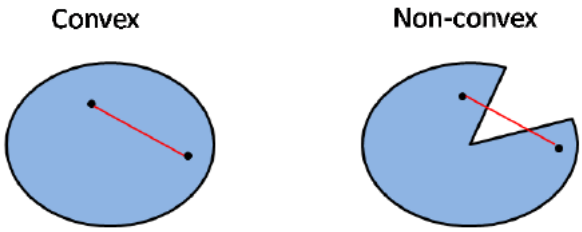


Figure 2.1: Convex and Non-convex set

An objective function $f : \mathscr{A} \mapsto \Re$ defined on a convex set $\mathscr{A}$ is convex if for all $a, b \in \mathscr{A}$ and for all λ such that $0 \leq \lambda \leq 1$,

$$f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda) f(b) \tag{2.2}$$

2.1 Unconstrained optimization

Unconstrained optimization [56, 57] is concerned with minimizing an objective function that depends on real variables without any restriction on the values of these variables. The problem is formulated mathematically as

$$\min_x f(x) \tag{2.3}$$

where $x \in \Re^n$ is a real vector of variables with $n \geq 1$ components and the objective function f: $\Re^n \mapsto \Re$ is a smooth function (second derivatives exist and are continuous).

A point x^* is a global minimizer if $f(x^*) \leq f(x)$ for all $x \in \Re^n$ while a point x^* is a local minimizer (sometimes called a weak local minimizer) if there is a neighbourhood $\mathscr{N}$ of x^* such that $f(x^*) \leq f(x)$ for all $x \in \mathscr{N}$.

Furthermore, a point x^* is said to be a strict (strong) local minimizer if there is a neighbourhood $\mathscr{N}$ of x^* such that $f(x^*) \leq f(x)$ for all $x \in \mathscr{N}$ with $x \neq x^*$. This point is an outright winner in its neighbourhood. When $f(x)$ is convex, any local minimizer

x^* is global minimizer of $f(x)$.

2.1.1 Optimality Conditions

Necessary conditions If $f(x)$ is continuously differentiable in an open neighbourhood of x^*, first and second order necessary conditions for x^* to be a local minimizer are

$$\nabla f(x^*) = 0 \quad \textit{and} \quad \nabla^2 f(x) \geq 0 \tag{2.4}$$

Sufficient conditions If $f(x)$ is continuously differentiable in an open neighbourhood of x^*, first and second order sufficient conditions for x^* to be a strict local minimizer are

$$\nabla f(x^*) = 0 \quad \textit{and} \quad \nabla^2 f(x) > 0 \tag{2.5}$$

Optimization algorithms begin with a starting point x_0, and generate a sequence of iterates $\{x_k\}_{k=0}^{\infty}$ that terminate when no more progress can be made or a solution point has been approximated with sufficient accuracy. The algorithms use information about the function f at x_k and possibly information from earlier iterates $x_0, x_1, \ldots, x_{k-1}$, to find a new iterate x_{k+1} with a lower function value than x_k. Two common methods for moving x_k to the next iterate x_{k+1} are the line search and trust region methods, the former will be discussed briefly.

Line search In the line search strategy, the algorithm chooses the search direction p_k, and moves along this direction for an appropriate distance α_k, which is a positive scalar called the step length in the iteration given by

$$x_{k+1} = x_k + \alpha_k p_k. \tag{2.6}$$

An effective choice of the direction p_k and step length α_k determines the success of the line search method. Most algorithms require p_k to be a *descent direction* ensuring that $f(x_{k+1}) < f(x_k)$. For p_k to be a descent direction, the property $p_k^T \nabla f(x_k) < 0$ (i.e. it makes an angle of strictly less than $\pi/2$ radians with $-\nabla f(x_k)$) must be satisfied to guarantee that $f(x_k)$ can be reduced along this direction.

Search direction

The search direction often has the form

$$p_k = -B_k^{-1} \nabla f(x_k) \tag{2.7}$$

where B_k is a symmetric non-singular matrix. The search direction can be obtained via the following methods

1. Steepest descent: The search direction is the negative gradient where B_k is simply the identity matrix I.

2. Newton's method: This direction is derived from the second-order Taylor series approximation to $f(x_k + p)$ where B_k is the exact Hessian $\nabla^2 f(x_k)$ and $\nabla^2 f(x_k)$ is positive definite.

3. Quasi-Newton method: Here B_k is an approximation to the Hessian that is updated at every iteration by a low-rank formula described in [56].

Newton's method will be utilized in the algorithms described in this chapter.

Step length

The distance to be moved from the current iterate x_k to a new iterate with a lower function value along the chosen search direction p_k can be found by approximately

solving the optimization problem to find a step length α_k:

$$\min_{\alpha_k>0} f(x_k + \alpha_k p_k) \tag{2.8}$$

The *Armijo condition* ensures that the choice of α_k leads to sufficient decrease in the objective function $f(x_k)$, according to the inequality

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^T p_k \tag{2.9}$$

for some constant $c \in (0,1)$, i.e. reduction in $f(x_k)$ should be proportional to both the step length α_k and the directional derivative $\nabla f(x_k)^T p_k$. The sufficient decrease in $f(x_k)$ may not be enough to ensure the algorithms make reasonable progress towards the solution, hence Wolfe and Goldstein conditions described in [56] impose additional requirements to avoid unacceptably short steps, α_k.

Alternatively, a *backtracking* approach can be implemented which uses the sufficient decrease inequality to terminate the line search procedure. Here an initial step length α_0 is chosen and reduced by a contraction factor ρ, such that after a finite number of trials α_k becomes small enough to satisfy the Armijo condition. This approach ensures that the chosen step length α_k is either a fixed value (initial α_0) or short enough to ensure the Armijo condition, but not too short. A backtracking algorithm is described below:

Backtracking Line Search algorithm [56]

Choose $\alpha_0 > 0$, $\rho, c \in (0,1)$; *set* $\alpha \leftarrow \alpha_0$;

repeat until $f(x_k + \alpha p_k) \leq f(x_k) + c_1 \alpha \nabla f(x_k)^T p_k$

set$\alpha \leftarrow \rho \alpha$;

end (**repeat**)

Terminate with $\alpha_k = \alpha$.

*In practice, c_1 is chosen to be quite small, say 10^{-4}.

2.2 Constrained optimization

A constrained optimization [54, 56] problem is formulated as

$$\min_{x \in \Re^n} f(x) \quad subject\ to \quad \begin{cases} c_i(x) = 0, & i \in \mathscr{E} \\ c_i(x) \geq 0, & i \in \mathscr{I} \end{cases} \tag{2.10}$$

where f and c_i are smooth real valued functions on a subset of $\Re^n$ and $\mathscr{E}$ and $\mathscr{I}$ are finite sets of indices. As before f is the objective function, $c_i : i \in \mathscr{E}$ are the equality constraints and $c_i : i \in \mathscr{I}$ are the inequality constraints. Any point that satisfies all the constraints in (2.10) is said to be a feasible point and a set of all such points is referred to as a feasible region. Hence the minimizer x^* of the constrained problem must exist in the feasible region.

The subject of constrained optimization is split into two main parts, *linear constraint programming* and *non-linear programming*. In linear constraint programming, each constraint is a linear function of the form $c_i(x) = a_i x - b_i$. The simplest cases are where the objective function is either linear or quadratic as in the case of *linear programming* or *quadratic programming* respectively. The *Simplex method* developed by Dantzig in the 1940's proved very effective for dealing with linear programming problems. An alternative method called *interior point* was developed by Karmarkar in 1984, others include *active set* and *gradient projection methods*. Most of the ideas behind these alternative methods can be carried over to quadratic programming problems which is the focus of this chapter.

An important concept in constrained optimization is the *Lagrange multiplier* obtained by equating the first order partial derivatives of the Lagrangian function to zero. The Lagrangian function $\mathscr{L}$ is given by

$$\mathscr{L}(x,\lambda) = f(x) - \sum_{i \in \mathscr{E} \cup \mathscr{I}} \lambda c_i(x) \tag{2.11}$$

where λ_i are the Lagrange multipliers and i indicates the index of the constraints. An explanation of how the Lagrangian multiplier determines the sensitivity of the cost function to changes in the constraint is given in [54].

Another important concept, the *active set* $\mathscr{A}(x)$ at any feasible x is the union of the set $\mathscr{E}$ with the indices of the active inequality constraints given by

$$\mathscr{A}(x) = \mathscr{E} \cup \{ i \in \mathscr{I} | c_i(x) = 0 \}. \tag{2.12}$$

2.2.1 Optimality conditions

If x^* is a solution of equation 2.10, there is a Lagrange multiplier vector λ^*, with components $\lambda_i^* : i \in \mathscr{E} \cup \mathscr{I}$, such that the following conditions are satisfied at (x^*, λ^*)

$$\begin{aligned} \nabla_x \mathscr{L}(x^*, \lambda^*) &= 0, \\ c_i(x^*) &= 0, \quad \text{for all } i \in \mathscr{E} \\ c_i(x^*) &\geq 0, \quad \text{for all } i \in \mathscr{I} \\ \lambda_i^* &\geq 0, \quad \text{for all } i \in \mathscr{I} \\ \lambda_i^* c_i(x^*) &\geq 0, \quad \text{for all } i \in \mathscr{E} \cup \mathscr{I} \end{aligned} \tag{2.13}$$

The conditions in (2.13) are often called the *Karush-Kuhn-Tucker conditions* (KKT) [56].

2.2.2 Quadratic programming

A quadratic program [57] is a special case of constrained optimization where the objective function of the optimization problem is quadratic of the form

$$f(x) = c^T x + \frac{1}{2} x^T H x \tag{2.14}$$

where c is a constant vector, H is a constant symmetric $n \times n$ matrix and $x \in \Re^n$ is a set defined by a finite number of equality and inequality constraints. The quadratic program is convex if H is a positive semi-definite matrix. The gradient and Hessian of the objective function are

$$\nabla f(x) = Hx + c \quad \textit{and} \quad \nabla^2 f(x) = H$$

Let the primal (original) quadratic program with inequality constraints be given by

$$\begin{aligned} &\min_{x \in \Re^n} c^T x + \frac{1}{2} x^T H x \\ &\text{subject to} \quad Ax \geq b. \end{aligned} \tag{2.15}$$

We define a corresponding dual problem as

$$\begin{aligned} \max_{\lambda \in \Re^m, w \in \Re^n} \Psi(\lambda, w) &= b^T \lambda + \frac{1}{2} w^T H w \\ \text{subject to } A^T \lambda &\geq Hw + c \\ \lambda &\geq 0 \end{aligned} \tag{2.16}$$

If λ^* is the Lagrange multiplier vector for the primal QP problem, and $w^* = x^*$, then (λ^*, w^*) is the solution of the dual. To ensure that a solution of the primal can be recovered from a solution of the dual, H must be non-singular and $AH^{-1}A^T$ must be positive definite.

Two methods for the solution of convex quadratic programs will be discussed in this chapter, namely interior point and active set methods.

2.2.2.1 Interior point for convex QP

Interior point methods [56, 58–60] generally focus on searching for primal and dual variables that satisfy the KKT conditions of section 2.2.1 and hence solve the primal and dual programs concurrently. Primal and dual variables that are required to be non-negative at the solution are kept strictly positive at each interior point iteration. This means that the iterates will stay interior with respect to this constraints though some variables will approach zero in the limit. Interior point methods have been proven to be advantageous for solving problems that are large and convex.

This section considers a primal-dual interior point approach to convex quadratic programs with inequality constraints as follows:

$$\begin{aligned} \min_x q(x) &= \frac{1}{2}x^T Hx + c^T x \\ \text{subject to} \quad & Ax \geq b. \end{aligned} \tag{2.17}$$

where H is a symmetric positive semi-definite matrix and where the $m \times n$ matrix A and b on the right hand side are defined by

$$A = [a_i]_{i \in \mathscr{I}}, \quad b = [b_i]_{i \in \mathscr{I}}, \quad \mathscr{I} = \{1, 2, \ldots, m\}$$

Obtaining the necessary KKT conditions for equation 2.17, if x^* is the solution of (2.17), there is a Lagrange multiplier vector λ^* such that the following conditions are

satisfied for $(x, \lambda) = (x^*, \lambda^*)$:

$$\begin{aligned} Hx - A^T\lambda + c &= 0 \\ Ax - b &\geq 0 \\ (Ax-b)_i\lambda_i = 0, \quad i &= 1,2,\ldots,m \\ \lambda &\geq 0 \end{aligned}$$

We introduce a slack vector $s \geq 0$, and rewrite the necessary conditions as

$$\begin{aligned} Hx - A^T\lambda + c &= 0 \\ Ax - s - b &= 0 \\ \lambda_i s_i = 0, \quad i &= 1,2,\ldots,m \\ (\lambda, s) &\geq 0 \end{aligned} \tag{2.18}$$

The KKT conditions are both necessary and sufficient because the objective function and feasible region are convex. The solution of the convex quadratic program of (2.17) is obtained by solving (2.18).

We are concerned with related problem where the *complementary condition*, $\lambda_i s_i = 0$ is replaced by the following relation

$$\Lambda S e = \mu e$$

where $\Lambda = \text{diag}(\lambda_i, \lambda_2, \ldots, \lambda_m)$, $S = \text{diag}(s_i, s_2, \ldots, s_m)$, $e = (1,1,\ldots,1)^T$.
Given a current iterate (x, λ, s), that satisfies $(\lambda, s) \geq 0$, the *duality measure* μ is defined by

$$\mu = \frac{1}{m}\sum_{i-1}^{m} \lambda_i s_i = \frac{\lambda^T s}{m} \tag{2.19}$$

As $\mu \to 0$ the primal and dual problems coincide. Rewriting (2.18) we need to find

(x, λ, s) such that

$$F(x,\lambda,s) = \begin{bmatrix} Hx - A^T\lambda + c \\ Ax - s - b \\ \Lambda Se - \mu e \end{bmatrix} = 0, \quad (\lambda, s) \geq 0 \tag{2.20}$$

The constrained system of non-linear equations in (2.20) can be solved iteratively by Newton's method, however, exact solutions for each target value of μ is not required. Rather, μ is adaptively reduced at each iteration aiming in the limit for $\mu = 0$, in which cases the optimality conditions for the QP are recovered.

We apply Newton's method from section 2.1.1 to $F(.)$ to find a search direction $(\Delta x, \Delta\lambda, \Delta s)$ which satisfies

$$J(x,\lambda,s) \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{bmatrix} = -F(x,\lambda,s) \tag{2.21}$$

where J is the Jacobian of F. Expanding (2.21) for a $\sigma\mu$-perturbed system gives

$$\begin{bmatrix} H & -A^T & 0 \\ A & 0 & -I \\ 0 & S & \Lambda \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_b \\ -\Lambda Se + \sigma\mu e \end{bmatrix} \tag{2.22}$$

where the *centering parameter* $\sigma \in [0\ 1]$ is a parameter chosen by the algorithm and

$$r_d = Hx - A^T\lambda + c, \qquad r_b = Ax - s - b.$$

If $\sigma = 1$, equation (2.22) defines a *centering direction* while at the other extreme $\sigma = 0$ is referred to as the *affine scaling direction*. The choice of $\sigma \in [0\ 1]$ provides a trade-off

between moving towards the central path and moving towards optimal solution of the problem ($\mu = 0$).

The next feasible iterate is obtained by setting

$$(x^+, \lambda^+, s^+) = (x, \lambda, s) + \alpha(\Delta x, \Delta\lambda, \Delta s) \tag{2.23}$$

where α is chosen such that $(\lambda^+, s^+) > 0$. By elimination, we restate (2.22) in an augmented form as

$$\begin{bmatrix} H & -A^T \\ A & \Lambda^{-1}S \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_b + (-s + \sigma\mu\Lambda^{-1}e \end{bmatrix} \tag{2.24}$$

Further elimination gives

$$(H + A^T S^{-1}\Lambda A)\Delta x = -r_d + A^T S^{-1}\Lambda[-r_b - s + \sigma\mu\Lambda^{-1}e] \tag{2.25}$$

The form in (2.25) is usually called the *normal form* and can be solved by Cholesky factorization and two back-solves. This is the major computational operation at each iteration of the interior point program. A modification of the basic primal-dual interior point program devised by Mehrotra and usually called the *Predictor-Corrector algorithm* is discussed in detail in [56, 58, 61]. If the convex QP problem has only equality constraints, it may be solved via *Gaussian elimination* and back substitution.

Summary of Interior Point algorithm for Convex QP [56]

Compute a starting point (x_0, λ_0, s_0) with $(\lambda_0, s_0) > 0$;

for $k = 0, 1, 2, \ldots$

Choose $\sigma_k \in [0, 1]$ and solve (2.22) for $(\Delta x^k, \Delta\lambda^k, \Delta s^k)$;

where $\mu_k = (\lambda^k)^T s^k / m$

Set $(x^{k+1}, \lambda^{k+1}, s^{k+1}) = (x^k, \lambda^k, s^k) + \alpha_k(\Delta x^k, \Delta\lambda^k, \Delta s^k)$

choosing α_k so that $(\lambda^{k+1}, s^{k+1}) > 0$

end (for)

2.2.2.2 Active set for convex QP

An *optimal* active set [56] at an optimal point x^* following from equation (2.12) is defined as the indices of the constraints at which equality holds, that is,

$$\mathscr{A}(x^*) = \{i \in \mathscr{E} \cup \mathscr{I} : a_i^T x^* = b_i\}. \tag{2.26}$$

Active set methods generally start by making a guess at the optimal active set, if the guess is incorrect, it repeatedly uses information from the gradient and Lagrange multiplier to drop and add an index from the current estimate of $\mathscr{A}(x^*)$. Active set methods for QPs are of three variants namely primal, dual and primal-dual; the first will be discussed in this section.

For the primal quadratic program,

$$\begin{aligned} \min_x q(x) &= \frac{1}{2}x^T Hx + c^T x \\ \text{subject to } a_i^T x &= b_i, \quad i \in \mathscr{E} \\ a_i^T x &\geq b_i, \quad i \in \mathscr{I}, \end{aligned} \tag{2.27}$$

primal active set methods start by computing a feasible initial iterate x_0 and ensures that all subsequent iterates remain feasible. A step from one iterate to the next is found by solving a quadratic sub-problem on a subset called the *working set* denoted by $\mathscr{W}_k$ at the kth iterate x_k. The working set contains all the equality constraints and some or all of the inequality constraints with the condition that the gradient a_i of the constraints

contained in it must be linearly independent.

Given a working set $\mathscr{W}_k$ at iterate x_k, we determine if x_k minimizes (2.27) subject to the constraints defined by the working set. If it does not, a step p is computed by solving an equality constrained sub-problem where the working set constraints are regarded as equalities and others are temporarily discarded. We define,

$$p = x - x_k, \quad g_k = Hx_k + c$$

and substituting into the objective function $q(x)$ in (2.27), we obtain

$$q(x) = q(x_k + p) = \frac{1}{2} p^T H p + g_k^T p + d$$

where $d = \frac{1}{2} x^T H x + c^T x$ is a constant term. d can be dropped from the objective function without changing the solution to the problem, hence the sub-problem to be solved at the kth iteration is rewritten as

$$\begin{aligned} &\min_p \frac{1}{2} p^T H p + g_k^T p \\ &\text{subject to } a_i^T p = 0 \text{ for all } i \in \mathscr{W}_k \end{aligned} \tag{2.28}$$

with $a_i^T p = b_i - a_i^T x_k$. The following KKT conditions for the sub-problem are solved for (p_k, λ_k):

$$\begin{bmatrix} H & -A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} p_k \\ \lambda_k \end{bmatrix} = \begin{bmatrix} -g_k \\ 0 \end{bmatrix} \tag{2.29}$$

If the optimal p_k from (2.28) is non-zero, it be must be decided how far to move along this direction. If $x_k + p_k$ is feasible with respect to all the constraints, using p_k as the search direction, we set $x_{k+1} = x_k + p_k$. Otherwise we set

$$x_{k+1} = x_k + \alpha_k p_k \tag{2.30}$$

where the step length $\alpha_k \in [0,1]$ is chosen to be the largest value for which all constraints are satisfied. The largest possible value of α_k which achieves feasibility is defined by:

$$\alpha_k := \min\left(1, \min_{i \notin \mathscr{W}_k,\, a_i^T p_k < 0} \frac{b_i - a_i^T x_k}{a_i^T p_k}\right) \tag{2.31}$$

The constraints i for which the minimum in (2.31) are achieved are called the *blocking constraints*. If $\alpha_k = 1$, there are no blocking constraints at this iteration, i.e. no new constraints are active at $x_k + \alpha_k p_k$.

If $\alpha_k < 1$, it means that a step along p_k was blocked by some constraint not in $\mathscr{W}_k$, hence one blocking constraint is added to $\mathscr{W}_k$ to obtain $\mathscr{W}_{k+1}$. The iteration is continued in this manner till $\hat{x}$ is obtained that optimizes the objective function over its current working set $\hat{\mathscr{W}}$ at $p = 0$. $p = 0$ satisfies the optimality conditions in (2.29) for (2.28), hence

$$\sum_{i \in \hat{\mathscr{W}}} a_i \hat{\lambda}_i = g = Hx + c \tag{2.32}$$

for some Lagrange multipliers $\hat{\lambda}_i,\ i \in \hat{\mathscr{W}}$.

The non-negativity of the Lagrange multipliers $\lambda_i \geq 0$ guarantees satisfaction of all the optimality conditions of section 2.2.1, hence $\hat{x}$ is the KKT point for the original problem (2.27).If H is positive semi-definite, $\hat{x}$ is a local minimizer and if H is positive definite, $\hat{x}$ is a strict local minimizer as described in section 2.1.1. Conversely, if $\lambda_i < 0$, optimality conditions are not satisfied and the objective function is decreased by dropping this constraint. The index corresponding to one of the negative multipliers is removed from the working set and a new problem of the form (2.28) is solved. A primal active set algorithm for convex QP is summarized as follows:

Summary of Active Set algorithm for Convex QP [56]

Compute a feasible starting point x_0;

Set $\mathcal{W}_0$ to be a subset of the active constraints at x_0;

for $k = 0, 1, 2, \ldots$

Solve (2.29) to find p_k;

if $p_k = 0$

Compute Lagrange multipliers $\hat{\lambda}_i$ that satisfy (2.32),

set $\hat{\mathcal{W}} = \mathcal{W}_k$;

if $\hat{\lambda}_i \geq 0$ for all $i \in \mathcal{W}_k \cap \mathcal{I}$

STOP with solution $x^* = x_k$;

else

Set $j = \arg\min_{j \in \mathcal{W}_k \cap \mathcal{I}} \hat{\lambda}_j$;

$x_{k+1} = x_k$; $\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k \setminus \{j\}$;

else $(*p_k \neq 0*)$

Compute α_k from (2.31);

$x_{k+1} \leftarrow x_k + \alpha_k p_k$;

if there are blocking constraints obtain $\mathcal{W}_{k+1}$ by adding one of the blocking constraints to $\mathcal{W}_{k+1}$;

else

$\mathcal{W}_{k+1} \leftarrow \mathcal{W}_k$;

end(for)

Comparison of algorithms The major difference between active set and interior point methods for convex quadratic programming is that active set generally requires a large number of steps in which the search direction is relatively inexpensive to compute while interior point methods generally take a smaller number of more costly steps. Active set algorithms are generally most effective for small and medium scale problems while interior point methods are typically used where the problem is large. Active set

methods are in some cases more complicated to implement with respect to factorization updating procedures at each active set iteration while in interior point methods, because the system of linear equations retains the same structure and size throughout the algorithm, standard sparse factorization software can be used.

2.3 Summary

This chapter has discussed a basic background to quadratic optimization. The quadratic programming algorithms discussed in this chapter are key tools required for the contributions in Chapters 4, 5 and 6. Active set and Interior point methods have been discussed and their step by step algorithm summarized.

In particular, the quadratic programming algorithms steps in section 2.2.2.1 and 2.2.2.2 were implemented on a Siemens S7 PLC using ladder logic programming to compute the optimized solution of the TMIA control scheme discussed in section 3.4 of Chapter 3. The results shown in Table 5.1 of Chapter 5 are based on comparison of the computation of the quadratic programming algorithms discussed in this chapter.

Chapter 3

IMC Anti-windup techniques

Input constraints in multivariable systems controlled by analytical dynamic controllers such as PID and Internal Model Controllers give rise to problems such as directionality change and controller windup. This leads to significant performance deterioration of the closed loop system and must be accounted for in the controller design. This chapter discusses the concepts of controller windup and process directionality and introduces the TMIA control structure [24] to simultaneously tackle these problems within an Internal Model Control framework.

3.1 Windup and Directionality

3.1.1 Windup

Windup occurs when the states of the controller are driven by the error when the actuator is in saturation [5]. The inconsistency between the controller output and states of the controller causes the feedback loop to run as open loop. It is a performance degradation exhibited by dynamic controllers with slow or unstable nodes [62]; a special case being integrator windup occurring in PI/PID controllers. The controller is said to

'wind-up' and requires that the error has an opposite sign for a long time before the feedback loop is active again. This results in a significant overshoot before the system returns to the linear region.

The control of systems under actuator constraints fall under two major categories. One approach is the *a priori* design of the control system which inherently satisfies both nominal performance requirements and saturation constraints. A typical example is a Model Predictive Controller (MPC) [26], in which constraints are explicitly accounted for and the controller action is the solution to a constrained optimization problem [9]. The *a posteriori* approach first designs a controller fitted to meet all nominal performance specifications, then adds an anti-windup compensator to minimize the undesirable windup effects that may occur in the event of saturation [3, 63] as in figure 3.1.

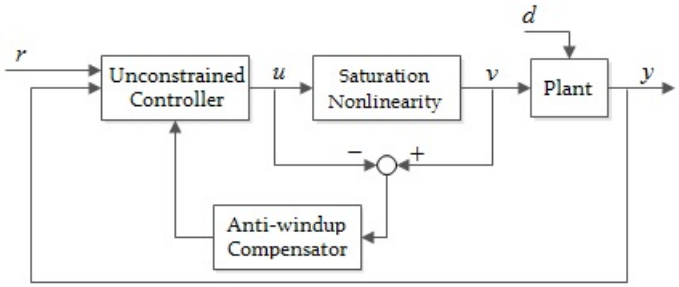


Figure 3.1: Anti-windup compensation

The anti-windup compensator is designed to achieve the following goals:

1. Maintain closed loop stability.

2. Recover linear design specifications when constraints are inactive (i.e. plant response with anti-windup compensation coincides with unconstrained response).

3. Graceful performance degradation during saturation (i.e. plant response with anti-windup compensation is as close as possible to the unconstrained response).

3.1.2 Directionality

The phenomenon of directionality [8, 9] usually occurs in multiple-input multiple-output (MIMO) systems with actuator saturation non-linearities. For the case of a single-input single-output (SISO) system, the boundary of the feasible plant input set is naturally closed and convex and consists of two isolated points such that when the output of the controller is infeasible, only one of those two points that is closest to the output of the unconstrained controller yields an optimal response. This 'clipping' of an unconstrained controller output in SISO plants leads to an optimal feasible plant input. On the other hand, for constrained MIMO systems, the boundary of the feasible plant input set contains an infinite number of points such that when the controller output is infeasible the closest feasible point to the unconstrained controller output in the plant input space may not yield an optimal response. Hence MIMO clipping of an unconstrained controller output may not lead to an optimal feasible plant input.

To ensure a graceful performance degradation of the MIMO system in the presence of input saturation, an artificial non-linearity (NL) in form of directionality compensator [3–6, 9, 64, 65] is inserted in the system as shown in figure 3.2. The directionality compensator calculates a feasible plant input on the basis of a given unconstrained controller output.

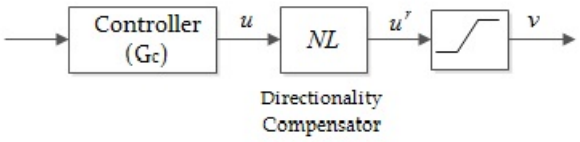


Figure 3.2: Directionality compensation

The artificial non-linearity can be defined as a quadratic optimization problem as

$$u^{r*} = \arg\min_{u^r} ||\mathscr{T}u^r - \mathscr{F}u||^2_W \tag{3.1}$$

subject to

$$u_{\min} \leq u_i \leq u_{\max} \quad i = 1, \ldots, m$$

Feasible plant inputs can be obtained by one some of the following forms of compensators:

- Clipping/limiting [4]: $\mathscr{T} = 1, \quad \mathscr{F} = 1$

  This is equivalent to the saturation function in section 3.3.

- Direction preservation [5, 62]: $\begin{cases} \mathscr{T} = 1, \quad \mathscr{F} = 0 \quad if \quad \|u\|_\infty < u_{\max} \\ \mathscr{T} = \|u\|_\infty, \ \mathscr{F} = 1 \quad if \quad \|u\|_\infty \geq u_{\max} \end{cases}$

  This approach is suggested for plants with ill-conditioned steady state gain matrix. The constrained control action is obtained by scaling down the controller outputs such that u and v has the same direction when saturation occurs. Hence subsequent saturation has no effect since u^r always remains in the linear region.

- Optimization based conditioning techniques [6, 66]: $\mathscr{T} = \mathscr{F} = D_k^{-1}$

  where D_k^{-1} is the non-singular feedthrough matrix of a the classical feedback controller G_c described in section 3.2. When a controller output in infeasible, an online optimization problem is solved to obtain a feasible controller output by calculating a new set-point value closest in the set-point space to the original set-point value.

- Optimal directionality compensation (ODC) [8, 9]: $\mathscr{T} = \mathscr{F} = \mathscr{P}\mathscr{C}$

  where $\mathscr{P}$ is a diagonal matrix whose elements depends on the relative orders of each controlled output and $\mathscr{C}$ is the characteristic (or decoupling) matrix.

In MIMO plants, the clipping and direction preservation compensators may produce different feasible plant inputs which steer the plant response in the wrong direction. Also the optimization based conditioning technique may lead to poor performance since it is based on the controller used and not the plant being controlled. The optimal directionality compensator is most effective and is employed in the design of the TMIA control structure of section 3.4. The nature of the characteristic matrix of the plant determines when it is optimal to use the clipping method or direction preservation. The characteristic matrix over a short horizon primarily determines the direction in which the output response will change and the extent of the effect of input changes on the controlled output response.

A class of systems that do not exhibit process directionality are those with diagonal characteristic matrices. Also noteworthy is the fact that since the ODC optimizes over a short time horizon, the calculated plant input may not be optimal over a long time horizon. For this reason, the TMIA structure of section 3.4 incorporates a steady state optimization to guarantee optimality over a long horizon.

3.2 Internal Model Control (IMC)

A classical IMC structure as introduced in [67–69] is shown in Figure 3.3.

$G(z)$ = discrete plant,

$\tilde{G}(z)$ = approximate plant model,

$d(z)$ = vector of disturbances,

$r(z)$ = setpoint vector,

$y(z)$ = output vector,

$u(z)$ = input vector and

$Q_{IMC}(z)$ = the IMC controller.

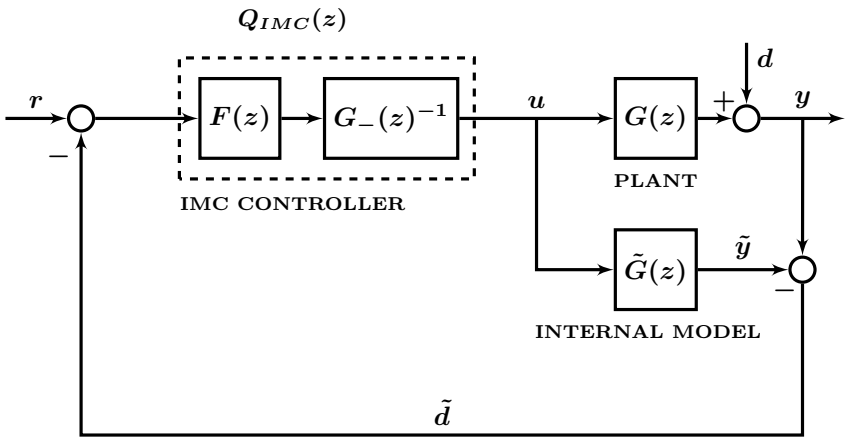


Figure 3.3: Conventional IMC Structure

This IMC structure is equivalent to a classical feedback controller defined as

$$G_c(z) = [I - Q_{IMC}(z)\tilde{G}(z)]^{-1} Q_{IMC}(z) \tag{3.2}$$

Likewise, any feedback loop with arbitrary controller can be converted to IMC structure by defining

$$Q_{IMC}(z) = [I + G_c(z)\tilde{G}(z)]^{-1} G_c(z) \tag{3.3}$$

The feedback signal $\tilde{d}(z)$ from figure 3.3 is zero when no disturbances $d(z)$ affect the system and the model is perfect (i.e. $G(z) = \tilde{G}(z)$), thus it acts as a feedback controller only when necessary.

$$\tilde{d}(z) = d(z) + [G(z) - \tilde{G}(z)]u(z) \tag{3.4}$$

Input and output transfer functions obtained by block diagram manipulation are

$$u(z) = [I + Q_{IMC}(z)(G(z) - \tilde{G}(z))]^{-1} Q_{IMC}(z)(r(z) - d(z)) \tag{3.5}$$

$$y(z) = d(z) + G(z)[I + Q_{IMC}(z)(G(z) - \tilde{G}(z))]^{-1} Q_{IMC}(z)(r(z) - d(z)) \tag{3.6}$$

Assuming $G(z) = \tilde{G}(z)$, (3.5) and (3.6) become

$$\begin{aligned} u(z) &= Q_{IMC}(z)(r(z) - d(z)) \\ y(z) &= d(z) + G(z)Q_{IMC}(z)(r(z) - d(z)) \end{aligned} \tag{3.7}$$

Hence if both the process $G(z)$ and the controller $Q_{IMC}(z)$ are stable, then closed stability of the system is guaranteed [7].

Assuming a perfect controller given by

$$Q_{IMC}(z) = \tilde{G}^{-1}(z) \tag{3.8}$$

Equation (3.8) achieves perfect set point tracking despite disturbances and plant-model mismatch since equation 3.6 becomes

$$y(z) = d(z) + G(z)[I + \tilde{G}^{-1}(z)G(z) - I]^{-1}\tilde{G}^{-1}(z)(r(z) - d(z)) = r(z) \tag{3.9}$$

This perfect controller is not a feasible design because the closed loop may be rendered unstable and $\tilde{G}^{-1}(z)$ may often not be realizable. However, it will serve as a good starting point for the IMC controller design. Though perfect control cannot be achieved for the whole frequency range, it can be achieved in steady state using

$$Q_{IMC}(1) = \tilde{G}(1)^{-1} \tag{3.10}$$

Integral action can thus be achieved by making the steady state controller gain the inverse of the model gain.

3.2.1 IMC Design

Implementation of the perfect controller is not suitable for two main reasons:

1. If $\tilde{G}(z)$ contains time delays, then $\tilde{G}^{-1}(z)$ would involve predictive terms which cannot be evaluated.

2. If $\tilde{G}(z)$ contains zeros outside the complex unit circle, this will translate to unstable poles when inverted thus rendering the overall loop unstable.

The IMC controller is thus designed using the following steps.

3.2.1.1 Step 1: Model Factorization

The plant model $\tilde{G}(z)$ is factorized into an invertible and non-invertible part such that

$$\begin{aligned} \tilde{G}(z) &= \tilde{G}_+(z)\tilde{G}_-(z) \\ \tilde{G}_+(1) &= I \end{aligned} \tag{3.11}$$

where $\tilde{G}_+(z)$ contains time delays and zeros of $\tilde{G}(z)$ outside the complex unit circle and $\tilde{G}_-(z)$ has a stable realizable inverse.

3.2.1.2 Step 2: Robustness Filter Design

Since the plant model is never accurate ($G(z) \neq \tilde{G}(z)$), a low-pass filter $F(z)$ as shown in figure 3.3 is added to improve the robustness of the IMC loop to modelling errors. For a multivariable plant, a diagonal filter is used to make input actions less severe and shape the plant output response. Closed loop dynamics of the system can be directly specified in the IMC design and the required controller is calculated from these specifications and the model of the plant. The IMC controller is given by

$$Q_{IMC}(z) = F(z)\tilde{G}_-(z)^{-1} \tag{3.12}$$

where $F(z)$ which directly expresses a trade off between robustness and performance is given by

$$F(z) = diag\left(\frac{1-a_i}{1-a_i z^{-1}}\right), \qquad 0 \leq a_i \leq 1 \qquad i = 1,2,\ldots,m \tag{3.13}$$

m is the number of manipulated variables.

If $G(z) = \tilde{G}(z)$, the closed loop transfer matrix is $\tilde{G}_+(z)F(z)$ and following from (3.7),

$$y(z) = \tilde{G}_+(z)F(z)(r(z) - d(z)) + d(z) \tag{3.14}$$

Since $F(z)$ forms a factor of the closed loop transfer matrix, improved robustness is being traded with sluggish performance. Adjustment of the filter tuning parameter a_i is very transparent and suitable for manipulating online, this makes IMC a good choice for multivariable control. For integral action, $\tilde{G}_+(1) = F(1) = I$ to ensure zero steady state offset even when $G(z) \neq \tilde{G}(z)$.

3.3 Modified IMC Anti-windup (MIMC)

For a saturating system where the actual plant input $v = sat(u)$, the conventional IMC structure of section 3.2 could lead to performance degradation and instability when constraints are active [7].

If the input signal is constrained such that

$$u_{min} \leq u_i \leq u_{max} \qquad i = 1,2,\ldots,m \tag{3.15}$$

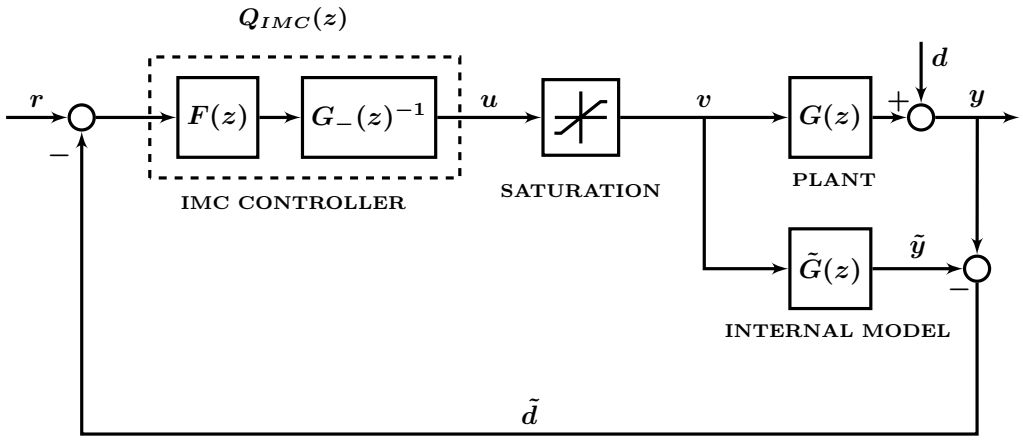


Figure 3.4: IMC Antiwindup Structure

where m is the number of manipulated variables, this can be represented by the saturation function $sat(u)$ defined as

$$sat(u) = \begin{cases} u_{max} & u_i > u_{max} \\ u_i & u_{min} \leq u_i \leq u_{max} \\ u_{min} & u_i < u_{min} \end{cases} \tag{3.16}$$

For the saturating system in figure 3.4, the saturation effect is not fed back directly to the controller, the controller acts only on the error between the reference signal and the output disturbance as shown in the closed loop equation (3.17).

$$u(z) = Q_{IMC}(z)(r(z) - d(z)) \tag{3.17}$$

$$y(z) = G(z)v(z) + d(z) \tag{3.18}$$

A modified IMC structure shown in figure 3.5 is proposed in [4] to deal with the undesirable effects associated with the standard IMC structure during saturation. $Q_{IMC}(z)$

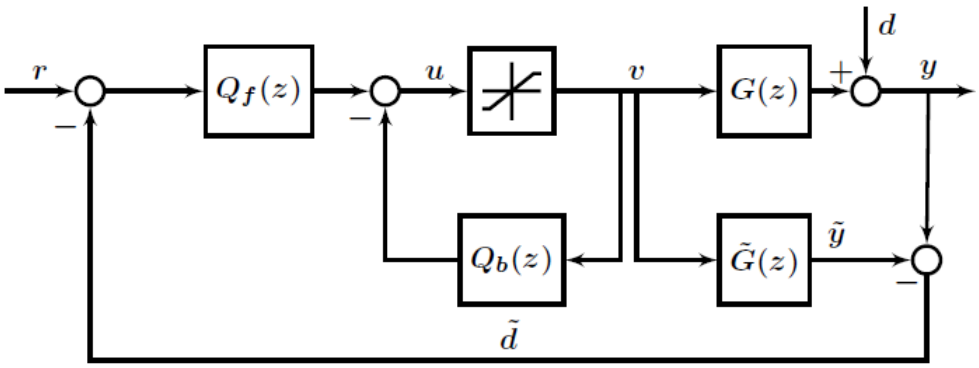


Figure 3.5: Modified IMC Antiwindup Structure

from figure 3.4 assumed to be bi-proper minimum phase stable is factorized such that

$$Q_{IMC} = (I + Q_b(z))^{-1} Q_f(z) \tag{3.19}$$

Assuming no plant-model mismatch, the closed loop equations are given by

$$u(z) = Q_f(z)(r(z) - d(z)) - Q_b(z)v(z) \tag{3.20}$$

$$y(z) = G(z)v(z) + d(z) \tag{3.21}$$

In this case the controller is also directly fed with information on the saturation action as shown in (3.20). In order to preserve output direction, [4] recommends 2 choices of factorization for $Q_f(z)$ and $Q_b(z)$ as:

3.3.1 Factorization 1

$$Q_f(z) = F_A(z)\tilde{G}(z)Q_{IMC}(z) \tag{3.22}$$

$$\begin{aligned} Q_b(z) &= Q_f(z){Q_{IMC}}^{-1}(z) - I \\ &= F_A(z)\tilde{G}(z) - I \end{aligned} \tag{3.23}$$

where $F_A(z)$ is a non-causal filter chosen to be diagonal in order to avoid introducing any change in the output direction and satisfies certain conditions such that $F_A(z)\tilde{G}(z)$ is biproper and $\lim_{z\to\infty}[F_A(z)\tilde{G}(z)] = I$. $Q_b(z)$ is strictly proper and thus implementable (free of algebraic loops). $Q_f(z)$ is minimum phase stable to guarantee internal stability of the closed loop system because if Q_{IMC} is minimum phase and $Q_f(z)$ is non-minimum phase, then $(I + Q_b(z))^{-1}$ must be unstable. When the constraints are inactive (i.e. $v = u$), equations (3.20) and (3.21) revert to the closed loop equations for the implementation of figure 3.3, thus linear performance is recovered.

3.3.2 Factorization 2

$$Q_f(z) = \Lambda Q_{IMC} + (I - \Lambda) Q_{IMC}(\infty) \tag{3.24}$$

$$Q_b(z) = Q_f(z) {Q_{IMC}}^{-1}(z) - I \tag{3.25}$$

where $\Lambda = \lambda I$ is a diagonal weighting matrix and $\lambda \in [0, 1]$. The choice of $\lambda = 1$ results in the classical IMC structure which yields a sluggish performance but stability is guaranteed [5], while the choice of $\lambda = 0$ results in $Q_f(z) = Q_{IMC}(\infty)$ [70], where performance is improved but internal stability is not guaranteed if Q_{IMC} is non-minimum phase. A trade-off between performance and stability is achieved through a suitable choice of λ.

3.3.3 Summary of the Modified IMC algorithm

1. Factorize the plant model $\tilde{G}(z)$ into an invertible and non-invertible part such that

$$\tilde{G}(z) = \tilde{G}_+(z)\tilde{G}_-(z) \quad \textit{and} \quad \tilde{G}_+(1) = I$$

where $\tilde{G}_+(z)$ contains time delays and zeros of $\tilde{G}(z)$ outside the complex unit circle and $\tilde{G}_-(z)$ has a stable realizable inverse.

2. Choose a filter $F(z)$ (diagonal for multivariable plant) such that

$$F(z) = diag\left(\frac{1-a_i}{1-a_i z^{-1}}\right), \qquad 0 \leq a_i \leq 1 \qquad i = 1,2,\ldots,m$$

where m is the number of manipulated variables.

3. Set the IMC controller as

$$Q_{IMC}(z) = F(z)\tilde{G}_-(z)^{-1}$$

with $\tilde{G}_+(1) = F(1) = I$ to ensure zero steady state offset.

4. Choose a factorization of $Q_{IMC}(z)$ such that

$$Q_{IMC} = (I + Q_b(z))^{-1} Q_f(z)$$

$Q_{IMC}(z)$ should be bi-proper and minimum phase stable.

(a) Factorization 1

$$\begin{aligned} Q_f(z) &= F_A(z)\tilde{G}(z)Q_{IMC}(z) \\ Q_b(z) &= Q_f(z){Q_{IMC}}^{-1}(z) - I \\ &= F_A(z)\tilde{G}(z) - I \end{aligned}$$

where $F_A(z)$ is a non-causal diagonal filter, $F_A(z)\tilde{G}(z)$ is biproper and $\lim_{z\to\infty}[F_A(z)\tilde{G}(z)] = I$. $Q_b(z)$ is strictly proper and $Q_f(z)$ is minimum phase stable.

(b) Factorization 2

$$Q_f(z) = \Lambda Q_{IMC} + (I - \Lambda) Q_{IMC}(\infty)$$
$$Q_b(z) = Q_f(z) {Q_{IMC}}^{-1}(z) - I$$

where $\Lambda = \lambda I$ is a diagonal weighting matrix and $\lambda \in [0, 1]$.

3.4 TMIA Control Structure

The Two-stage Multivariable IMC Antiwindup TMIA control structure proposed by [24] for open loop stable plants directly addresses the problems of windup and directionality change which affect multi-variable plants under input constraints such as actuator saturation. The core of this structure is the solution of two low-order quadratic programs at each sample step to control the transient and steady state behaviours of the system. The dynamic QP (QP_1) as shown in figure 3.6 addresses transient behaviour of the plant ensuring that the constrained system response is as close as possible to the unconstrained, while the steady state QP (QP_2) ensures optimal steady state performance. According to [9]:

> "*The characteristic matrix of a process characterizes the sensitivity of the process to input changes over a short time horizon, while the steady-state gain matrix of a process characterizes the sensitivity of the process to input changes over an infinite horizon*".

Hence the dynamic QP is formulated based on the characteristic matrix of the plant while the steady state QP is formulated based on the steady state gain matrix of the plant.

The two QPs have similar structures - convex QPs solved over a single-step horizon subject to the same input constraints. The steady state QP initially has an equality

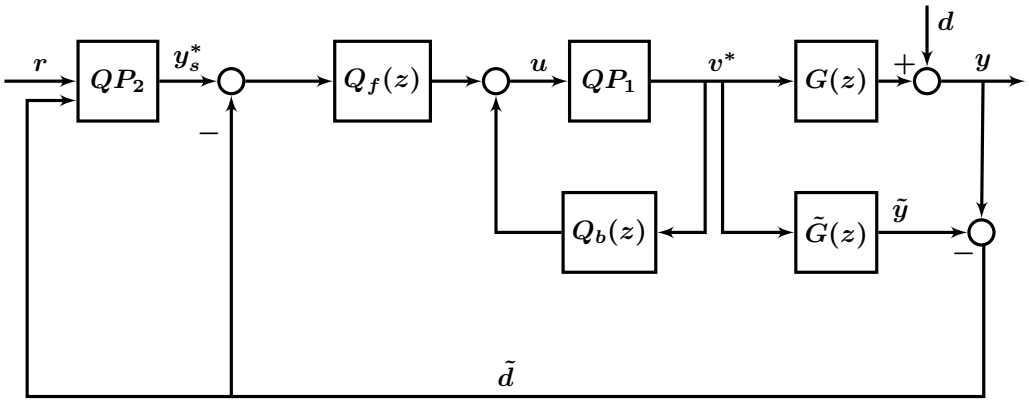


Figure 3.6: TMIA Control Structure

constraint which models the steady state response of the system, this is eliminated such that both QPs solve similar problems. The control objective is to keep the constrained output y as close to the unconstrained output y' as possible. For the dynamic QP, the optimization problem is formulated as:

$$v^* = \arg\min_{v} ||Cv - Cu||^2 W \tag{3.26}$$

subject to the constraints:

$$u_i^{\min} \leq v_i \leq u_i^{\max} \quad i = 1, \ldots, m$$

where m is the number of MVs and v and u are the constrained and unconstrained inputs respectively with W assumed to be a positive definite symmetric matrix that penalizes deviations in the constrained control inputs from the unconstrained and their relative importance. C is the characteristic matrix of the square system defined according to [9] as

$$C = \lim_{z \to \infty} [diag\{z^{r_m}\} G] \tag{3.27}$$

where $r_i = min(r_{i1}, r_{i2}, \ldots, r_{im})$ and r_{ij} is the relative order of output y_i with respect to manipulated input u_j.

For a multivariable linear time-invariant system described by a state space model of the form

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{3.28}$$

where A, B and C are $n \times n$, $n \times m$ and $m \times n$ matrices respectively, the characteristic matrix [8] is given by

$$C = \begin{bmatrix} c_1 A^{r_1 - 1} B \\ \vdots \\ c_m A^{r_m - 1} B \end{bmatrix} \tag{3.29}$$

The steady state performance optimization seeks to minimize the error between feasible steady state target y_s and the reference r within the limitations of the input constraints by obtaining suitable values for y_s and u_s which satisfies:

$$y_s = K_p u_s + \tilde{d} \tag{3.30}$$

where u_s is the value of the control input which causes the controlled variable to attain y_s in steady state. From figure 3.6, $\tilde{d}$ is the disturbance estimate obtained as the difference between the measured plant output y and the model output $\tilde{y}$ and K_p is the steady state gain of the plant ($G(1)$ for discrete systems). The steady state gain K_p is obtained from the state space matrices of the plant in (3.28) as:

$$G(1) = C(I - A)^{-1} B \quad for \quad G(z) \tag{3.31}$$

provided $(I - A)$ is non-singular. The steady state QP is solved to obtain an optimal feasible steady state target y_s^* according to:

$$y_s^* = \arg\min_{u_s, y_s} ||r - y_s||^2 Q_{ss} \tag{3.32}$$

subject to the inequality and equality constraints:

$$u_i^{\min} \leq v_i \leq u_i^{\max} \qquad i = 1, \ldots, m$$

$$\begin{bmatrix} -K_p & I \end{bmatrix} \begin{bmatrix} u_s \\ y_s \end{bmatrix} = \tilde{d}$$

where Q_{ss} is a positive definite symmetric matrix for penalizing deviations in the controlled variables and their relative importance. The equality constraint ensures the steady state requirement of (3.30).

TMIA structure summarily works such that if the set point target is achievable in steady state, the difference between the reference r and the disturbance estimate $\tilde{d}$ is passed to the dynamic QP for the solution of the optimal control input v^*. Otherwise if input constraint violation causes the steady state target to be unachievable, the steady state QP computes a feasible steady state target y_s which is optimally close to reference r . The difference between the feasible steady state target and the current disturbance estimate is then passed to the dynamic QP for computation of v^*.

An equivalent optimization problem to (3.32) can be obtained by eliminating the equality constraints to obtain:

$$u_s^* = \arg\min_{u_s} ||r - \tilde{d} - K_p u_s||^2 Q_{ss} \tag{3.33}$$

subject to the constraints

$$u_i^{\min} \leq v_i \leq u_i^{\max} \quad i = 1, \ldots, m$$

This transformed optimization problem now optimizes only over variable u_s with inputs as the difference between reference r and the disturbance estimate $\tilde{d}$. An optimal solution u_s^* of (3.33) ensures that the optimal y_s^* of the original problem in (3.32) is found via the steady state model of (3.30).

Equations (3.26) and (3.33) can be written in standard QP form as:

$$v^* = \arg\min_{v} \frac{1}{2} v^T H_1 v - v^T \tilde{H}_1 u \tag{3.34}$$

$$subject\ to \quad Lv \leq b$$

$$u_s^* = \arg\min_{u_s} \frac{1}{2} u_s^T H_2 u_s - u_s^T \tilde{H_2}^T (r-d) \tag{3.35}$$

$$subject\ to \quad Lu_s \leq b$$

where $H_1 = \tilde{H_1}^T W \tilde{H_1}$ and $H_2 = \tilde{H_2}^T Q_{ss} \tilde{H_2}$ are symmetric positive definite Hessian matrices which are defined based on the structural properties of the plant such that $\tilde{H_1}$ and $\tilde{H_2}$ are the characteristic matrix and steady state gain matrix of the plant respectively (i.e. $\tilde{H_1} = C$ and $\tilde{H_2} = K_p$). W and Q_{ss} are the respective positive definite symmetric weighting matrices for the dynamic and steady state optimization problems.
L and b in the inequality constraints are defined as:

$$L = \begin{bmatrix} -I_m \\ I_m \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} -u^{min} \\ u^{max} \end{bmatrix} \tag{3.36}$$

where m is the number of MVs and $u^{min} = [u_1^{min}, ...u_m^{min}]^T$ and $u^{max} = [u_1^{max}, ...u_m^{max}]^T$. The solutions v^* and u_s^* to (3.26) and (3.33) are unique if the characteristic matrix C and the steady state gain matrix K_p are non-singular.

3.4.1 TMIA control algorithm

Given an open-loop stable plant $G(z)$ with a nominal model $\tilde{G}(z)$, and feasible optimal solutions v^* and u_s^* satisfying (3.26) and (3.33) respectively, the control law that achieves both optimal transient and steady state performance by the TMIA structure of figure 3.6 is given by:

$$\begin{aligned} \tilde{d} &= y - \tilde{G}(z)v^* \\ u_s^* &= \psi_2(r, \tilde{d}) \\ y_s^* &= \tilde{H}_2 u_s^* + \tilde{d} \\ u &= Q_f(z)(y_s^* - \tilde{d}) - Q_b(z)v^* \\ v^* &= \psi_1(u) \end{aligned} \tag{3.37}$$

where ψ_1 and ψ_2 are the non-linear functions representing the dynamic and steady state QPs from (3.26) and (3.33) respectively.

When the constraints are inactive, the control law reduces to the standard unconstrained IMC control equation such that:

$$\begin{aligned} \tilde{d} &= y - \tilde{G}(z)u \\ u &= (I + Q_b)^{-1} Q_f (r - \tilde{d}) \end{aligned} \tag{3.38}$$

Good steady state performance is ensured by designing Q_{IMC} from section 3.2 such that $Q_{IMC}(1) = G(1)^{-1}$ for $G(z)$.

[24] further states that for systems which have similar characteristic matrix (C) and steady state gain matrix (K_p), the two QPs solve the same optimization problem, hence only the dynamic QP needs to be solved to achieve both optimal transient and steady state responses under control input saturation.

3.5 Extension to Rate Constraints

In this section, the author extends the anti-windup synthesis for input magnitude constrained multivariable processes developed in [24] to rate-constrained inputs. This is a novel addition an existing scheme.

3.5.1 Rate saturation

Rate constraints occur when actuators cannot change faster than a specified value. Compared to magnitude constraints there is a limited amount of literature available on both magnitude and rate constraints [71–73]. For systems subject to only rate constraints the controller development can be carried out in the same way as in Section 3.4 by using using the derivative of the input signal in place of the input signal which is then subject to magnitude constraints. However for systems subject to both input and rate constraints, such an approach would not work [74].

A better approach would be to incorporate the rate constraints into the controller design such that the optimized output always satisfies the constraints and the actuator is never overworked. Since the rate is the derivative of the input it is natural that the modelling framework is in discrete time:

$$u_k = u_{k-1} + \Delta u_k \tag{3.39}$$

where u_k is the input at sample time k and Δu_k is the change between the previous and current input. The block diagram of the input rate is shown in Figure 3.7.

Rate constraints are easily handled by Model Predictive Control, but where MPC is either not feasible or not affordable, it is necessary to provide and alternative simpler but effective controller design to handle this dynamic non-linearity.

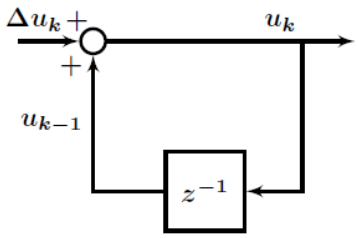


Figure 3.7: Input rate block diagram

3.5.2 TMIA extension

Additional constraints can be added to incorporate the rate non-linearity into the TMIA controller. We redefine the input constraints of the dynamic QP in section 3.4 by adding constraints on the change in input as:

$$u_{min} \leq u_k \leq u_{max}$$

$$\Delta u_{min} \leq \Delta u_k \leq \Delta u_{max}.$$

This can also be written in the form:

$$u_k \leq u_{max}$$

$$\Delta u_k \leq \Delta u_{max}$$

$$-u_k \leq -u_{min}$$

$$-\Delta u_k \leq -\Delta u_{min}.$$

Since equation 3.39 can be re-written as:

$$\Delta u_k = u_k - u_{k-1} \tag{3.40}$$

we substitute equation 3.40 so that the constraint inequalities become:

$$u_k \leq u_{max}$$

$$u_k \leq \Delta u_{max} + u_{k-1}$$

$$-u_k \leq -u_{min}$$

$$-u_k \leq -\Delta u_{min} - u_{k-1}.$$

The final constraints realization is of the form

$$Lu_k \leq b$$

where

$$L = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} u_{max} \\ \Delta u_{max} + u_{k-1} \\ -u_{min} \\ -\Delta u_{min} - u_{k-1} \end{bmatrix} \tag{3.41}$$

For this extension the dynamic optimization cost function section 3.4 remains unchanged and still minimizes over u. However the TMIA controller structure changes slightly to incorporate these new constraints. The QP algorithm needs new information about the rate limits and the past value of the input u_{k-1}. Figure 3.8 shows the modified TMIA controller where QP_1 now has two sets of inputs. A sample delayed input and the current input are fed to the dynamic QP. The dynamic QP uses the modified constraints in Equation 3.41.

A snippet of the modification to the TMIA controller in MATLAB simulation is shown in Figure 3.9.

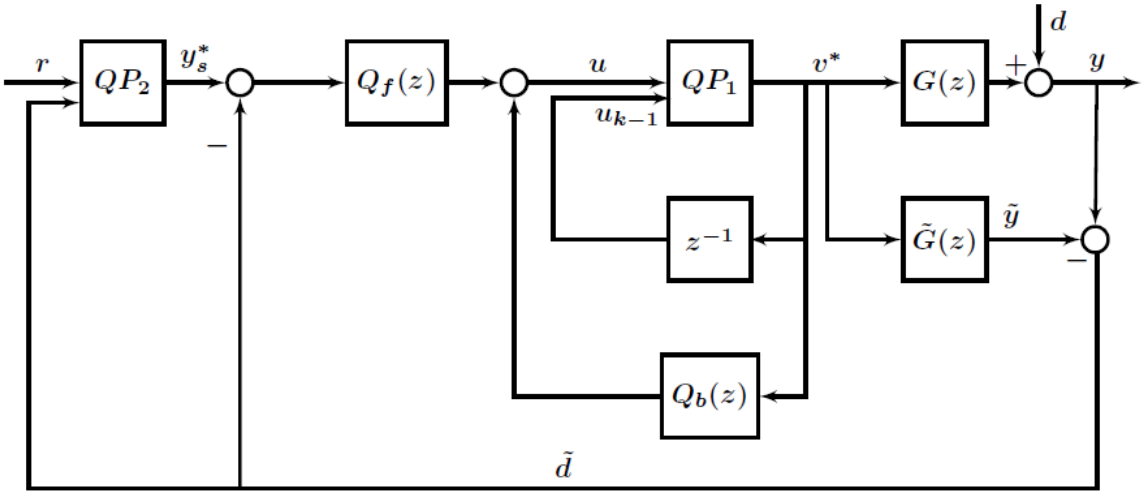


Figure 3.8: Modified TMIA controller

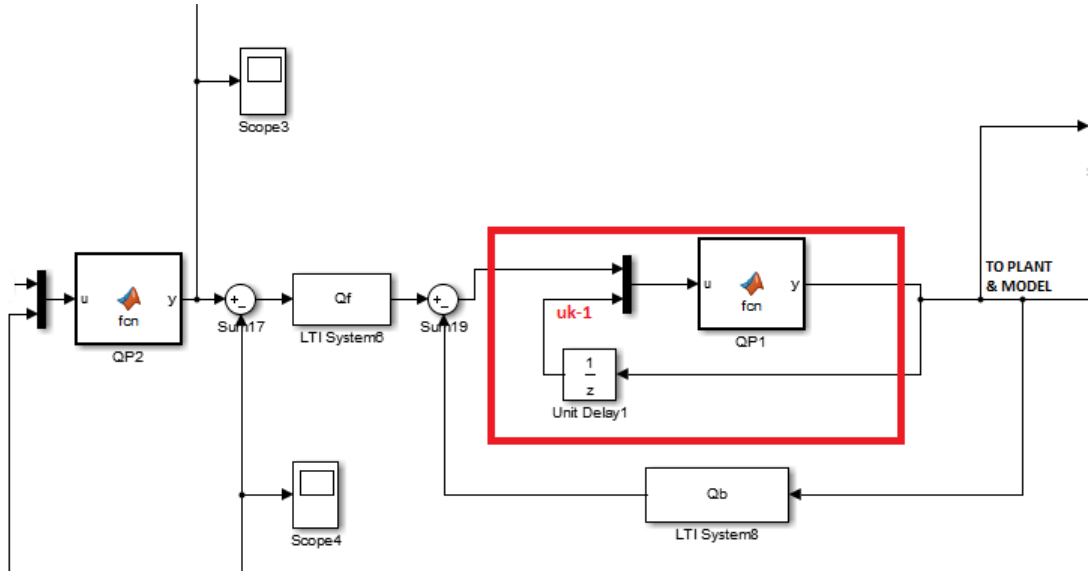


Figure 3.9: Snippet of TMIA controller modification in MATLAB

3.6 Summary

This chapter has discussed anti-windup techniques that can be used to deal with the performance deterioration in plants that are subject to input constraints. A TMIA controller has been presented as superior to classical anti-windup techniques by the inclusion of a quadratic optimization routine which utilizes minimal computing. The

TMIA controller is the basis for the online control of a Quadruple Tank system on a PLC discussed in Chapter 4.

Furthermore, the main contribution to this algorithm is the extension to incorporate rate constraints in a simple but efficient way. The TMIA controller has now been developed to become a complete design that handles both magnitude and rate constraints on the input.

Chapter 4

TMIA control of a Quadruple Tank process

4.1 The Quadruple Tank Process

The Quadruple Tank system is a multivariable laboratory process consisting of 4 interconnected water tanks of uniform cross-sectional area, 2 pumps and a water basin as presented in [1]. The Quadruple Tank system fabricated by Quanser is shown in Figure 4.1 with inputs as voltages to the 2 pumps, and outputs as water levels in the two lower tanks.

The tank pumps are gear pumps composed of a DC motor rated for 12V continuous voltage, hence the inputs are constrained between 0-10V in the controller design. Each of the pumps thrust water vertically to the tanks via two orifices of different sizes provided by Quanser. Through these orifices, a piping connection is made to the tank based on the configuration required. Each tank also has outlet orifices for discharge of water back into the water basin and the user can also vary the areas of discharge using different orifice sizes. The apparatus forms an autonomous and closed recirculating system [2].

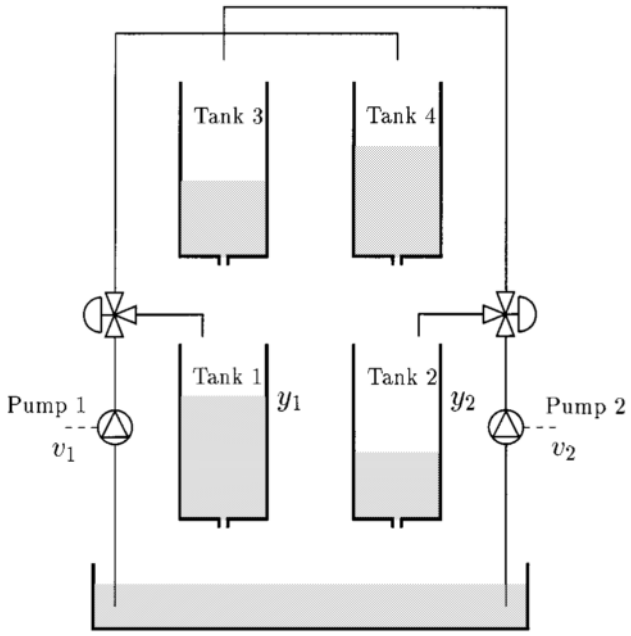


Figure 4.1: Schematic diagram of the Quadruple Tank process [1]

The water levels are measured by pressure sensors located at the bottom of each tank. The outputs of the sensors are passed through a signal conditioning board with a range of 0-5V which corresponds to 0-25cm of water height. The sensors are calibrated according to information provided in the Quanser user manual [2].

The Quadruple tank process has an adjustable zero described in [1] that can be moved along the real axis between the right and left plane s plane (in or out of the unit circle for discrete time), therefore the process can be set up to minimum or non-minimum phase depending on system configuration used. Details of how the location of zeros inside or outside the unit circle can be adjusted by varying the process connections is described in [2]. Results for both minimum phase and non-minimum phase settings of the quadruple tank rig are reported in this thesis. However, since our interest is in the PLC computation, focus is placed on the minimum phase configuration which allows higher closed-loop bandwidth and thus requires faster computation on the PLC.

4.1.1 Tank configurations

Water from the tank is pumped from Pumps 1 and 2 pump through rubber pipes to Out1 and Out2 shown in Figure 4.2. Out1 and Out2 have different diameters. For the minimum phase configuration, the larger flows through Out 1 are pumped to the two bottom tanks 2 and 4, while the smaller flows through Out2 are pumped to the top tanks 1 and 3. In the non-minimum phase case, the larger flows through Out 1 are pumped to the two top tanks 1 and 3, while the smaller flows through Out2 are pumped to the bottom tanks 2 and 4.

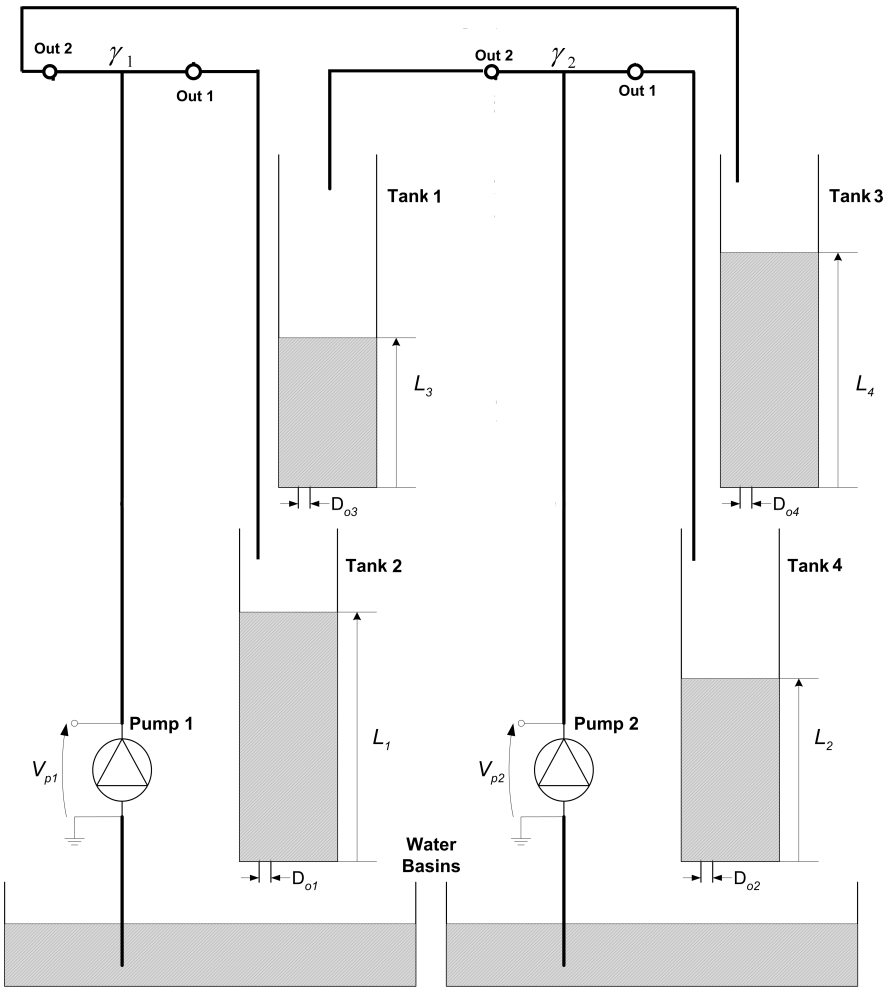


Figure 4.2: Quadruple Tank process connections - minimum phase [2].

4.1.2 Hardware Peripherals

The Quanser quadruple tank system is provided with hardware and QuaRC (Quanser Real-time Control) software that allows real-time interface between MATLAB its sensors and actuators. The complete real time Quadruple Tank process components provided by Quanser are:

1. Quadruple Tanks.
2. QUARC real time control software for MATLAB/SIMULINK.
3. Quanser 8-channel USB Data Acquisition board.
4. Analog Sensors Adapter.
5. 2-channel Linear Voltage Amplifier.

The data acquisition board provides a standard USB connection to a PC for reading/writing from the MATLAB-QUARC software with drivers pre-installed during installation of the software. The data acquisition board carries out digital to analog and analog to digital conversion with configurable ranges. It provides 8 analog inputs and 8 analog output connectors to the quadruple tanks. Figure 4.3 shows a snapshot of the Quadruple Tank system connected to the hardware peripherals and a PLC.

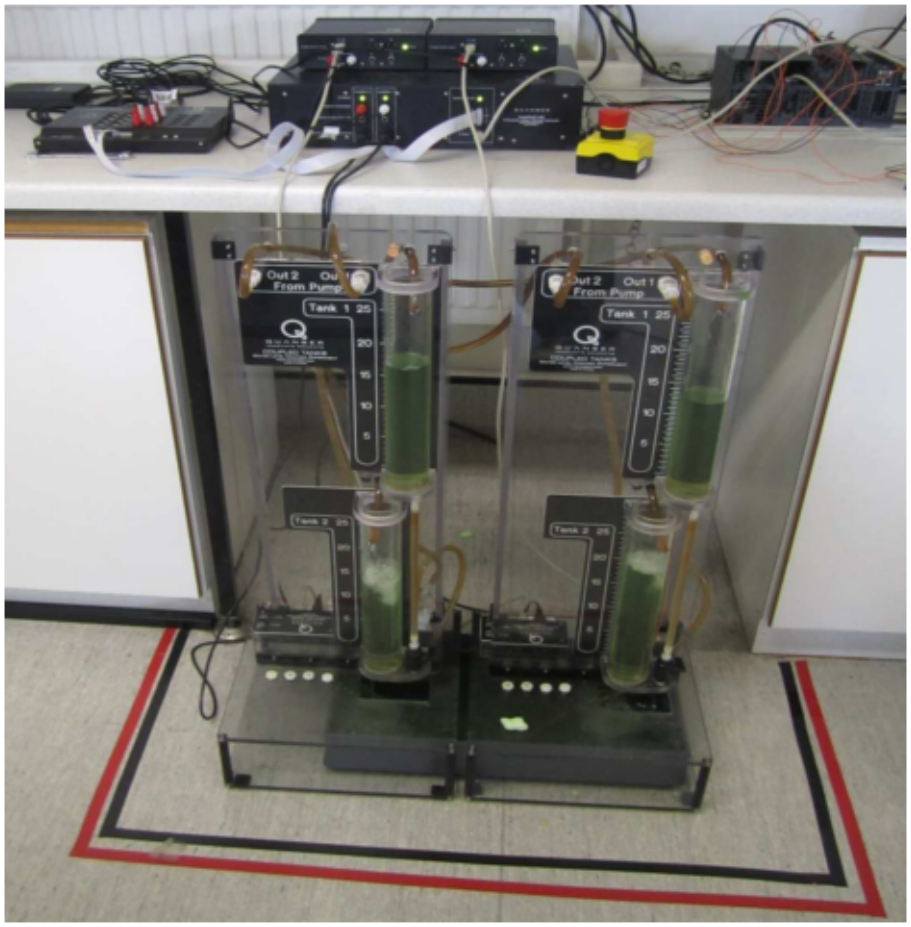


Figure 4.3: Snapshot of the Quadruple Tank process with connections to a PLC and Quanser hardware. Top right corner is a Siemens SIMATIC S7-300 CPU 314C-2 PN/DP PLC. Top left corner is a Quanser Data Acquisition board. Top middle consists of Analog Sensors Adapter (above) and Voltage Amplifier (beneath).

4.2 Control Platforms

Two control platforms are utilized; the QuaRC platform for real-time testing of the control design before deploying to a Siemens PLC which is main focus of our results. When the process is controlled using the QuaRC platform, all hardware peripherals are used, whereas when the process is controlled by the PLC, the Data acquisition board is not required since the inputs and outputs to the PLC are analog signals.

4.2.1 Quanser Real-Time Control (QUARC)

The communication set-up of the QUARC platform for control of the Quadruple Tank process is as shown in Figure 4.4. It comprises of a PC with integrated MATLAB-QUARC software installed, a power amplifier with a gain of 3, an analog sensors adapter and a data acquisition board. Here MATLAB-QUARC controls the process. The control signal range from the controller is 0-4V which is amplified to a 0-12V range for the pumps. Pressure-type level sensors are used for the 4 tanks with outputs as a voltage signal to feedback to the controller, the relationship between the sensor voltage and tank height in cm is linear and is calibrated with a sensitivity of 6.25 cm/V. Electrical connections are set up as described in [2]. Figure 4.5 shows a snaphot of the SIMULINK block diagram for the TMIA control of the plant in real time using the MATLAB-QUARC environment where the 2 quadratic programs (QP_1) and (QP_2) are implemented using embedded MATLAB functions.

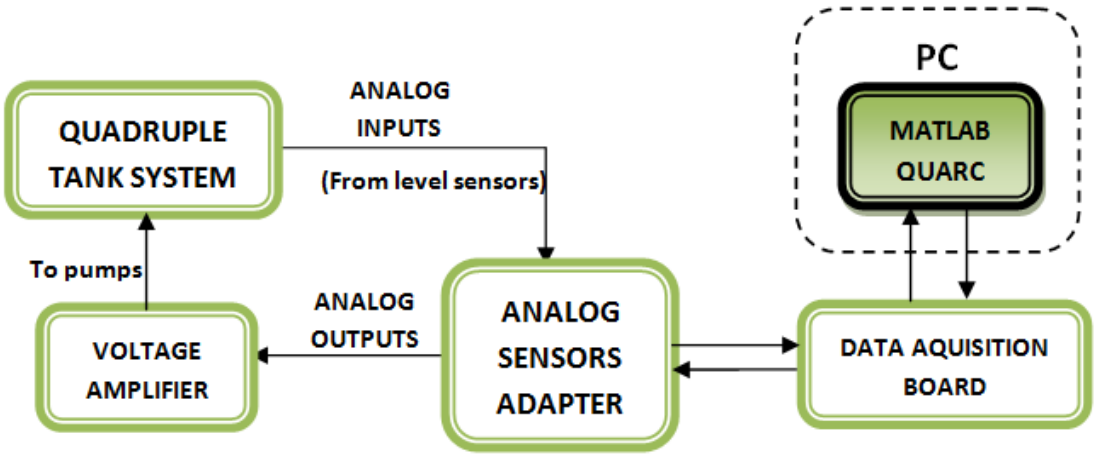


Figure 4.4: Communication setup with MATLAB-QUARC platform.

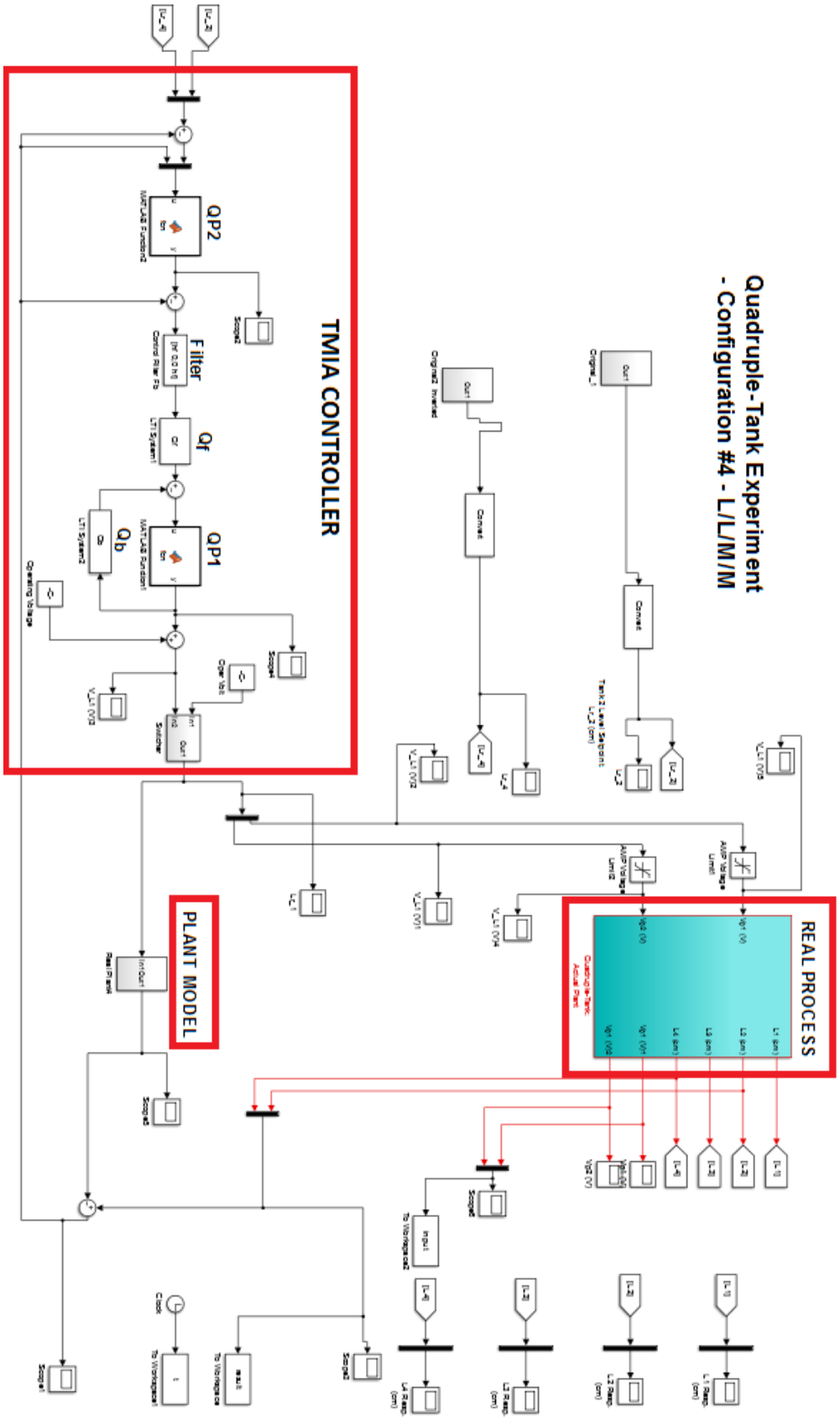


Figure 4.5: SIMULINK block diagram of TMIA control of plant in real time using the QUARC environment

4.2.2 Siemens PLC

With this platform, a Siemens SIMATIC S7-300 CPU 314C-2 PN/DP PLC controls the process. This is discussed in detail in Chapter 5.

4.3 Plant Model

The model of the quadruple tank process derived from first principles is non-linear, the linearised model will be of the form in Equation 4.1 where G_{12} and G_{21} are second order transfer functions [1].

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{4.1}$$

y_1 and y_2 are the levels of the two lower tanks and v_1 and v_2 are the input voltages from pumps 1 and 2 respectively. Due to shared laboratory equipment, the plant model used in this experiment is one that had been previously identified by colleagues, see [83] for details of modelling and verification. The linearized system model was identified with a 1s sampling time using MATLAB System Identification Toolbox from input-output data obtained using a Pseudo Random Binary Sequence (PRBS) excitation signal to obtain a transfer function model of the system at an operating voltage of [7V, 7V] for the pumps. This operating voltage was chosen via experimental testing in open loop to satisfy the conditions that at this voltage none of the bottom tanks overflowed, none of the top tanks were empty and there was still an allowance of a 30% pump range to control the tank levels.

In order to capture the dynamics of the system, the process was first stabilized at the operating voltage. Then for each case one pump is held constant at the operating voltage while a PRBS signal (of small enough magnitude that the voltage is still within

the pump operating range) is applied to the other pump. To obtain G_{11} and G_{21} data, a PRBS signal is applied to Pump 1 while Pump 2 is kept constant while to obtain G_{12} and G_{22} data, the PRBS signal is applied to Pump 2 while Pump 1 is kept constant. The identified and validated model is then discretized with a 1s sampling in order to implement it on the PLC. The minimum phase discrete model of the process is given in Equation 4.2 and the non-minimum phase in Equation 4.6.

4.4 TMIA control design

A Two-stage Multivariable IMC Anti-windup controller is designed for the Quadruple Tank process as described in Chapter 3.

4.4.1 Minimum phase

$$\tilde{G}(z) = \begin{bmatrix} \frac{0.1359z^{-2}}{1-0.9362z^{-1}} & \frac{0.1326z^{-2}-0.01253z^{-3}}{1-0.3906z^{-1}-0.5538z^{-2}} \\ \frac{0.03781z^{-2}-0.02813z^{-3}}{1-1.833z^{-1}+0.8395z^{-2}} & \frac{0.1642z^{-2}}{1-0.9215z^{-1}} \end{bmatrix} \tag{4.2}$$

The system in equation (4.2) is minimum phase stable with poles and zeros inside the unit circle.

To design the TMIA controller for the Quadruple Tank process, first a classical IMC controller is designed according to Section 3.2 factorizing the plant model $\tilde{G}(z)$ is into an invertible and non-invertible part

$$\tilde{G}_{-}(z) = \begin{bmatrix} \frac{0.1359}{1-0.9362z^{-1}} & \frac{0.1326-0.01253z^{-1}}{1-0.3906z^{-1}-0.5538z^{-2}} \\ \frac{0.03781-0.02813z^{-1}}{1-1.833z^{-1}-0.8395z^{-2}} & \frac{0.1642}{1-0.9215z^{-1}} \end{bmatrix} \tag{4.3}$$

and

$$\tilde{G}_{+}(z) = \begin{bmatrix} z^{-2} & 0 \\ 0 & z^{-2} \end{bmatrix} \tag{4.4}$$

Next, a robustness filter following from equation (3.13) is designed as

$$F(z) = \begin{bmatrix} \frac{1-0.65}{1-0.65z^{-1}} & 0 \\ 0 & \frac{1-0.85}{1-0.85z^{-1}} \end{bmatrix} \tag{4.5}$$

The filter $F(z)$ is used to avoid some chattering effects on the control signals [63] as well as shape the closed loop response. The loop shaping procedure involves ensuring high loop-gain at low frequencies for disturbance rejection and good reference tracking (where $S(j\omega) \approx 0$ and $T(j\omega) \approx 1$), a low roll-off rate around cross-over frequency for good damping and stability and a high roll-off rate at high frequencies for insensitivity to sensor noise and high frequency unmodelled plant dynamics (where $S(j\omega) \approx 1$ and $T(j\omega) \approx 0$). For a multivariable system, these specifications can be visualized in the frequency domain using singular value plots. The tuning parameters for $F(z)$ $(a_1 = 0.65, a_2 = 0.85)$ were selected to meet these design criteria and can be seen on the singular value plot response in Figure 4.7.

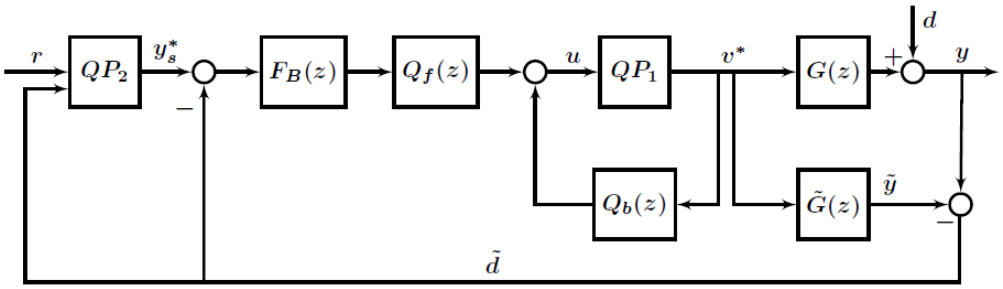


Figure 4.6: TMIA implementation with additional bandwidth filter

$$Q_{IMC}(z) = \tilde{G}_{-}(z)^{-1} F(z)$$

For the Quadruple Tank model, the open loop bandwidth is obtained approximately as $1/\tau_{max}$, where τ_{max} is the maximum time constant of the system. We specify the desired closed loop bandwidth of the system to be twice the open loop bandwidth

corresponding to about 0.2rad/s. In order to further shape the response, an additional bandwidth filter $F_B(z)$ shown in figure 4.6 was added outwith the IMC controller to limit the cut-off frequency of the control signal to this closed loop bandwidth. For a bandwidth filter of 0.2rad/s, the discrete equivalent is obtained as:

$$F_B(z) = \frac{0.1813z^{-1}}{1-0.8187z^{-1}} * I$$

The IMC tuning parameters in equation (4.5), $(a_1 = 0.65, a_2 = 0.85)$ for $F(z)$ along with $F_B(z)$ is such that the frequency of the maximum sensitivity is greater than the closed loop bandwidth ($\bar{\sigma}(S) > 0.2rad/s$). The sensitivity plots in figure 4.7 show that the loop shaping specifications are met. A biproper form of $Q_{IMC}(z)$ is used according to [4] for the design of the MIMC controller. However, for the implementation of the classical IMC controller, a strictly proper $Q_{IMC}(z)$ is used.

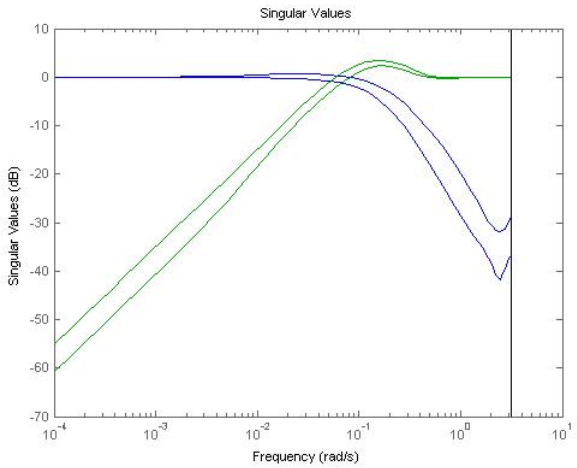


Figure 4.7: Singular value plots

Also the specification for reference tracking and zero steady state error is achieved since the conditions for integral action, $[F_B(z)F(z)G_+(z)]_{z\to 1} = I$ is met.

Next, the MIMC controller is designed by factorizing the biproper $Q_{IMC}(z)$ via

'factorization method 1', to obtain $Q_b(z)$ and $Q_b(f)$ choosing a non-causal filter $F_A(z)$

$$F_A(z) = \begin{bmatrix} \frac{1}{0.1359}z & 0 \\ 0 & \frac{1}{0.1642}z \end{bmatrix}$$

such that $\lim_{z\to\infty}[F_A(z)\tilde{G}(z)] = I$ (notice that $F_A(z)$ matches the top diagonal elements of $\tilde{G}(z)$).

Finally the TMIA controller is designed according to section 3.4. The characteristic and steady state matrices for the Quadruple Tank to be used in the quadratic optimization are obtained by equations (3.29) and (3.31) as

$$C = \begin{bmatrix} 0.1359 & 0.1326 \\ 0.03781 & 0.1642 \end{bmatrix} \quad \text{and} \quad K_p = \begin{bmatrix} 2.13009 & 2.1595 \\ 1.4892 & 2.0917 \end{bmatrix}$$

.

4.4.2 Non-minimum phase

$$\tilde{G}(z) = \begin{bmatrix} \frac{0.09956z^{-2}}{1-0.951z^{-1}} & \frac{0.01118z^{-2}+0.009991z^{-3}}{1-1.7z^{-1}+0.7138z^{-2}} \\ \frac{0.007059z^{-2}+0.006682z^{-3}}{1-1.842z^{-1}+0.8482z^{-2}} & \frac{0.09038z^{-2}}{1-0.9407z^{-1}} \end{bmatrix} \tag{4.6}$$

For the identified plant model of equation (4.6) in non-minimum phase configuration [1, 2], with zero locations at [0.6622; 1.01203; 0.9574; 0.9417], the inverse of the plant model would yield an unstable IMC controller. Hence the model is factorized into an all pass and minimum phase portion using the inner outer factorization method [84–86].

We assume that

$\tilde{G}(z) = C(zI-A)^{-1}B + D := \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \in \mathscr{RH}^{p\times m}(p \geq m)$ is a minimal realization

such that $0 < |\lambda_i(A)| < 1, \forall\, i = 1, 2, \ldots, n$.

Assuming the matrix D has full column rank m, there exists a right coprime factorization $\tilde{G}(z) = N(z)M(z)^{-1}$ such that N is inner ($N^*N = I$) if and only if and $\tilde{G}(z)$ has no zeros on the unit circle. The realization is given as

$$M = \left[\begin{array}{c|c} A+BF & BR^{-1/2} \\ \hline F & R^{-1/2} \end{array}\right],$$

$$N = \left[\begin{array}{c|c} A+BF & BR^{-1/2} \\ \hline C+DF & DR^{-1/2} \end{array}\right]$$

where

$$R_D = D^T D > 0,$$

$$R = R_D + B^T X B$$

$$F = -(R + B^T X B)^{-1}(B^T X A + D^T C)$$

and $X = X^T > 0$ is the unique stabilizing solution of the discrete algebraic Riccati equation

$$(A - BR^{-1}D^T C)^T X (A - BR^{-1}D^T C) - X + C^T (I - DR^{-1}D^T)C$$
$$-(A - BR^{-1}D^T C)^T X B (R + B^T X B)^{-1} B^T X (A - BR^{-1}D^T C) = 0$$

The model $\tilde{G}(z)$ is strictly proper ($D = 0$), therefore in order to correctly apply the factorization method, a small $D = \varepsilon I$ with $\varepsilon = 0.0001$ is added [7]. Thus the model is factorized into an invertible part $M^{-1}(z)$ and a non invertible part $N(z)$. The IMC formulation involves the inversion of the invertible portion of $\tilde{G}(z)$ and integral action is obtained by ensuring that the product of the controller inverse gain and the process model gain is identity. Since the inner-outer factorization of the plant model does not

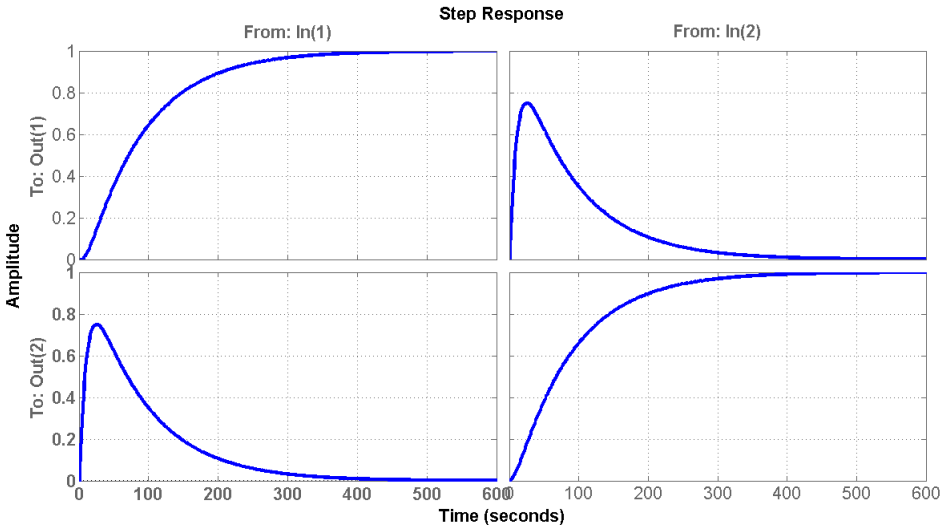
guarantee this, the non-invertible portion is scaled down by pre-multiplying $N(z)$ by $N(1)^{-1}$ to obtain $N(z)N(1)^{-1}$ such that the invertible portion becomes $N(1)M^{-1}(z)$ [87].

Then the classical IMC controller is

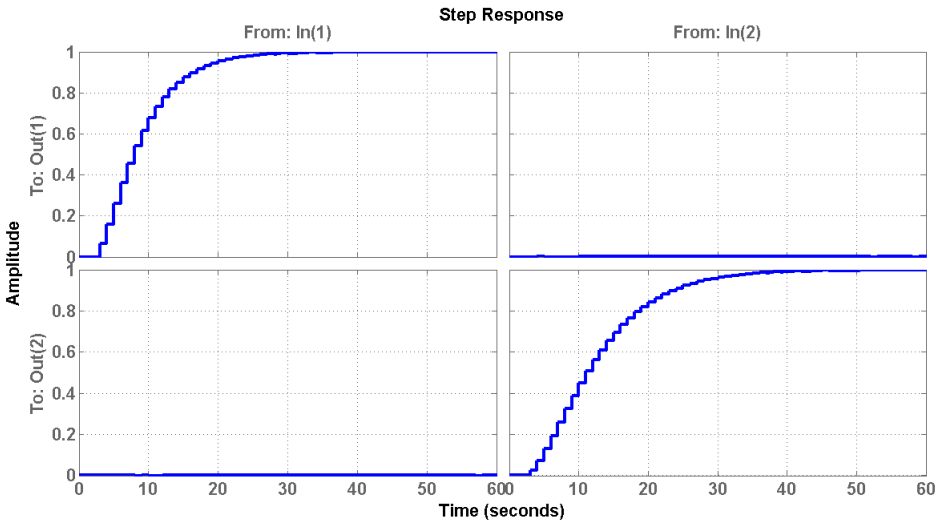
$$Q_{IMC} = M(z)N(1)^{-1}F(z) \tag{4.7}$$

with a diagonal filter $F(z)$ to shape the closed loop response. The closed loop response for the non-minimum phase control is about 10 times slower than that of the minimum phase response as seen in the time scales of Fig. 4.8.

As in the minimum phase case, the TMIA controller is formulated by first obtaining the Modified IMC (MIMC) controller by factorizing Q_{IMC} according to equation (3.19). Using factorization method 1 for the plant in this case can be restrictive because in order to satisfy the conditions, the design of $F_A(z)$ may require that the plant model be diagonal or that the off-diagonal transfer function components have higher relative orders than the diagonal components [88]. For the quadruple tank the dynamics are coupled across input channels and all the transfer function components of $\tilde{G}(z)$ have the same relative orders. Hence $\tilde{G}(z)$ in Equations (3.23) and (3.22) would need to be modified to have a relative order of 2 in the off diagonal components in order to achieve a strictly proper $Q_b(z)$. Also since the plant model is non-minimum phase, $Q_f(z)$ from Equation (3.23) is also non-minimum phase and the condition for internal stability is not met, which makes factorization method 1 unsuitable. Hence we used Factorization method 2 instead. Again the controllers $Q_f(z)$ and $Q_b(z)$ are designed more a slightly modified plant model by increasing the relative orders of the off-diagonal elements of $\tilde{G}(z)$ in order to ensure $Q_f(b)$ is strictly proper and thus implementable. The characteristic and steady state matrices for the Quadruple Tank in



(a) Non-minimum phase



(b) Minimum phase

Figure 4.8: Simulation of closed loop responses for minimum and non-minimum phase.

non-minimum phase are

$$C = \begin{bmatrix} 0.0996 & 0.01118 \\ 0.0071 & 0.0904 \end{bmatrix} \quad \text{and} \quad K_p = \begin{bmatrix} 2.0303 & 1.574 \\ 2.2013 & 1.525 \end{bmatrix}$$

In summary, the design parameters needed for the TMIA implementation are:

1. Factorized $\tilde{G}(z)$ using an appropriate factorization method.

2. a_i used to obtain $F_A(z)$ and hence $Q_{IMC}(z)$.

3. $F_A(z)$ used to obtain Q_b and Q_f with factorization 1 or λ with factorization 2.

4. $F_B(z)$ if required to further shape the response.

5. Weighting matrices W, Q_{ss}.

6. Characteristic and steady state matrices C and K_p.

4.5 Summary

This chapter has discussed the set-up, model and control platforms of a multivariable experimental rig to be controlled. In particular the design of the TMIA controller has successfully been described for two control cases: minimum and non-minimum phase. For the non-minimum case, special care needs to be taken in the factorization and inversion of the plant model so that the controller does not yield and unstable system. The development of the TMIA controller involved first designing the classical IMC anti-windup controller, then the modified IMC controller to be used for comparison in simulation and real-time experiments with the TMIA controller.

Chapter 5

PLC implementation

This chapter discusses authors contribution of implementing the TMIA controller designed in Chapter 4 on a real time experiment. It specifically details the controller implementation via the PLC platform using the quadratic programming methods discussed in Chapter 2. Results from three different quadratic program methods are presented in order to make useful recommendations on the most suitable algorithms for low computing industrial hardware.

5.1 Siemens S7 PLC

The TMIA design for the control of the quadruple tank process was implemented in the Siemens SIMATIC S7-300 CPU 314C-2 PN/DP PLC using Ladder logic. The IEC-61131-3 standard for programmable logic controllers defines five programming languages which are Ladder Logic, Function Block Diagram, Structured Text, Instruction List and Sequential Function Chart.

The following programming languages are supported by the S7-300 PLC [89]:

1. Ladder Logic: This is graphical programming language based on circuit diagram representation of relay logic hardware. Networks are formed with a combination

of circuit elements like normally open and normally closed contacts to form rungs/ladders of code executed from left to right and top to bottom.

2. Function Block Diagram: This a graphical programming language based on boolean algebra. Logic blocks are used to connect input and output variables.

3. Structured Control Language (SCL): this conforms to the IEC-61131-3 standard and is available only as an optional package. It is a high level text-based language similar to PASCAL or C language. It uses high level commands to simplify programming of loops and conditional branches and hence is suitable for complex optimization algorithms.

4. Statement List (STL): This is a low-level programming language which resembles assembly code.

5. Continuous Function Chart (CFC): This is also an optional package for graphical linking of complex functions and parallel control processing.

The motivation for choosing ladder logic is that it is easily understood and widely used by operators in industry. The PLC used consists of an integrated power supply and CPU unit alomg with two I/O modules. The power supply converts 230VAC line voltage to a 24VDC operating voltage to supply its S7-300 rack and load circuits. The I/O modules consist of 24 digital inputs, 16 digital outputs, 5 analog inputs (4 current/voltage and 1 PT100) and 2 analog outputs. The 2 inputs to the PLC in the quadruple tank experiment are the water levels in the two lower tanks, the PLC program calculates the required pump voltage. The 2 PLC outputs are the voltages supplied to the voltage amplifier which feed the pumps. The PLC provides 192KB working memory on an external micro memory card (required for the PLC to run), 4 fast counters (60 KHz) and CPU processing times of 0.59μs per floating point operation.

Figure 5.1: S7-300 CPU 314C-2 PN/DP PLC.

5.1.1 PLC communication setup

The PLC provides two types of communication interfaces. A combined MPI (Multi-Point Interface) and Profibus DP interface via an RS485 connector can be used to link up to a maximum of 127 devices on the network with maximum transmission rate of 12 Mbits/s. With this link a PC is connected to the PLC via USB adapter. The PLC also supports Ethernet communication via an integrated PROFINET (Industrial Ethernet) interface; this is the communication link used in this experiment. This can

also link up to 127 devices (for copper medium) up to a maximum transmission rate of 100 Mbits/s via an RJ45 connector. The communication setup for the PLC is shown in Figure 5.2. The PLC scan time is set to 1s which coincides with the sampling time for the discretized process transfer function.

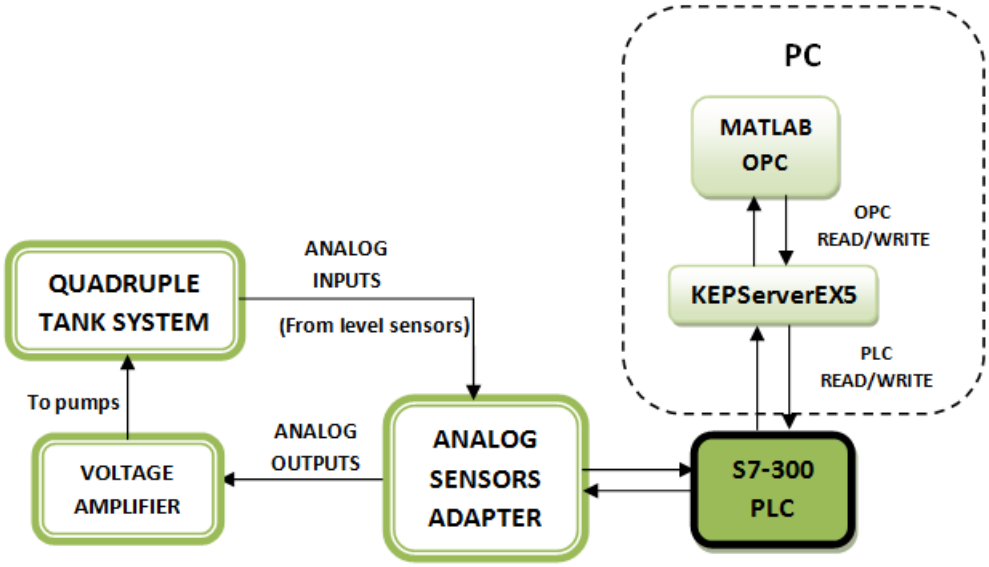


Figure 5.2: PLC communication setup.

5.1.1.1 Program Coding and download using the PC/PG interface

The PLC is programmed using a Programming Device (PG) which in this case is a PC with Siemens Totally Integrated Automation (TIA) Portal version 11 software installed. The PG and PLC are connected via standard twisted pair cables with RJ45 connectors. An Ethernet connection is created in the same network with PLC address 192.168.0.1 and PG address 192.168.0.2. Program code is downloaded from the PG to the PLC and online diagnostics can be monitored during run-time as shown in Figure 5.3.

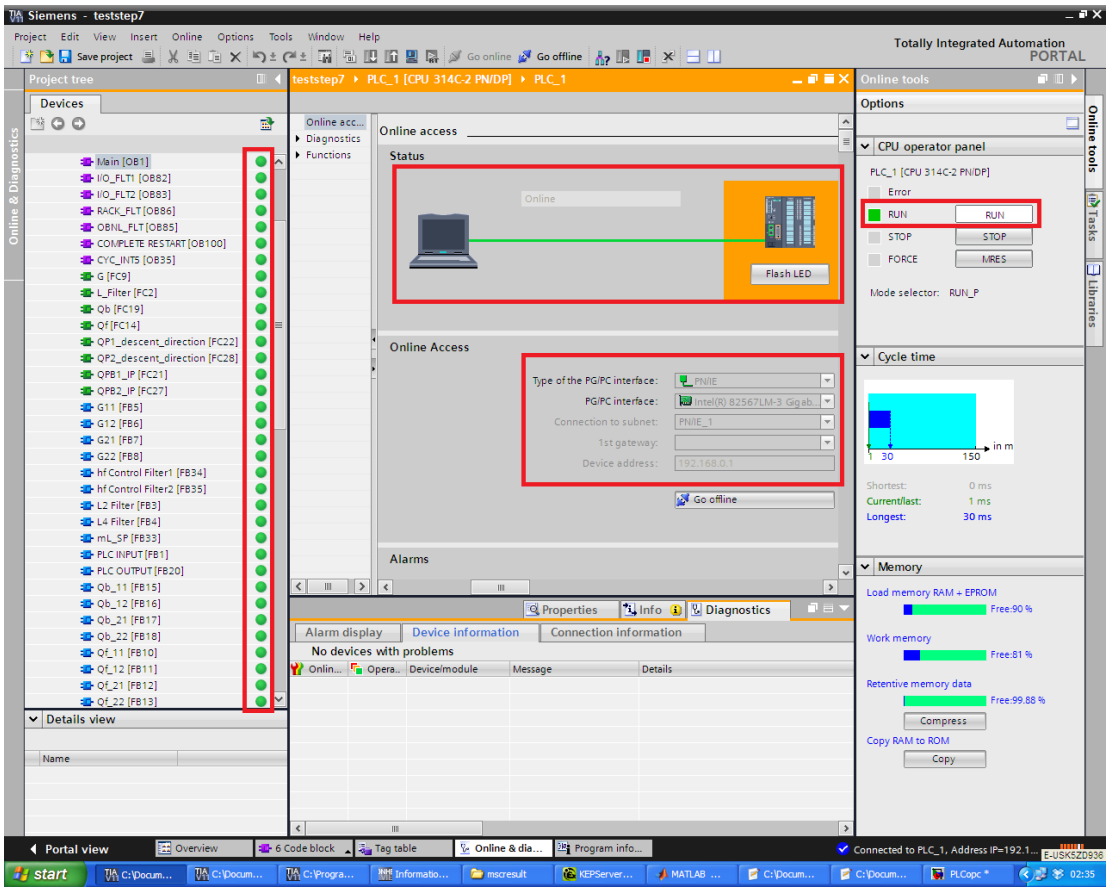


Figure 5.3: Sample Siemens PLC program in run time showing online diagnostics.

5.1.1.2 HMI via MATLAB-OPC communication

The PC serves a dual purpose; not only for program coding and downloading to the PLC but also serves as a Human Machine interface (HM1) for online monitoring and control. Real-time data acquisition is obtained via MATLAB-OPC toolbox and a KEPServerEX5 OPC server. With the TMIA controller running on the PLC, the KEPServerEX5 provides an interface for SIMULINK to read data continuously from PLC memory and to write desired set-points to the PLC. Hence MATLAB is able to communicate with the PLC via the KEPServerEX5 which is useful for trending the process response. Installed on the PC is a Siemens TCP/IP Ethernet driver that works with KEPServerEX5 to pass data between MATLAB-OPC client and Siemens PLC using the TCP/IP Ethernet protocol. The driver communication with the PLC uses a

standard network interface card already installed on the PC. Each parameter that needs to be read from or written to the PLC is configured in the KepserverEX5 input and output channels to be accessed by MATLAB.

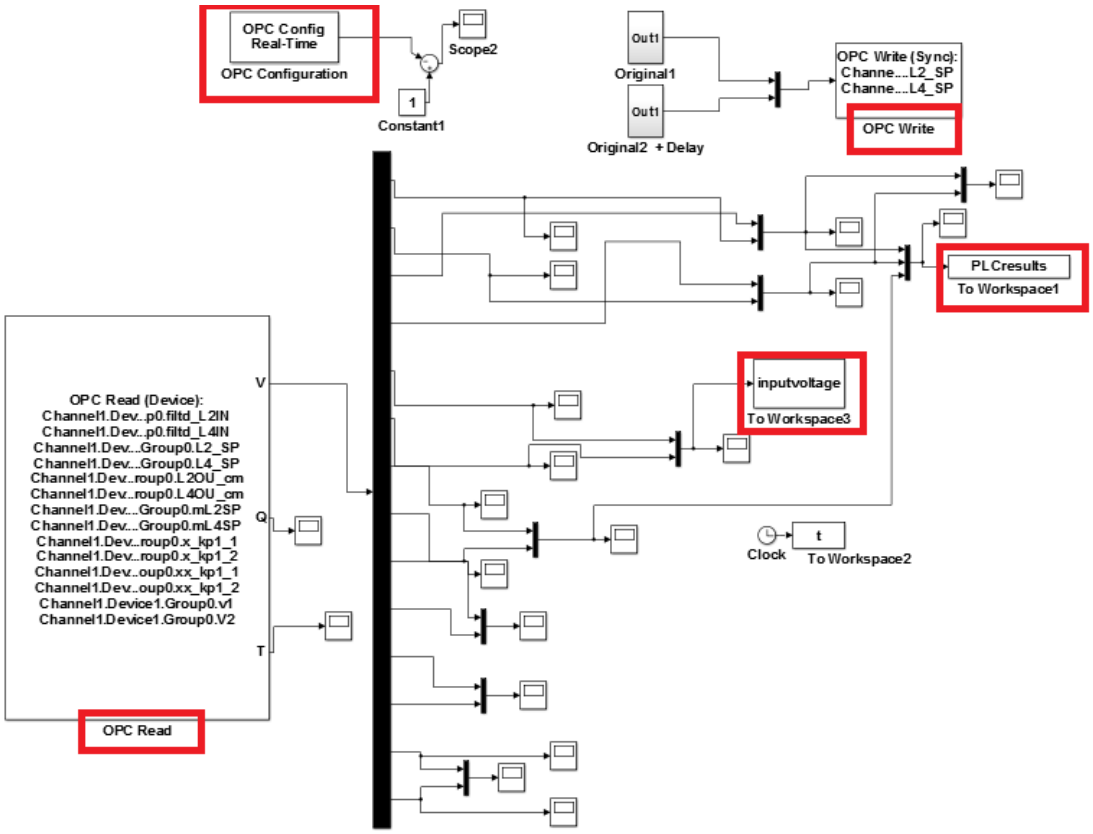


Figure 5.4: Real time MATLAB-OPC communication with PLC via KepserverEX5

A snaphot of the MATLAB-OPC communication with PLC via KepserverEX5 is shown in Figure 5.4 and the read/write communication with the KepserverEX5 channels is shown in Figure 5.5. Plant setpoints can be written directly into the PLC code but this is too cumbersome and inefficient as the program code would need to be re-downloaded to PLC each time a change is made. On an industrial plant this would be expensive since the plant would need to be shut down. Hence, it is more efficient to make these changes via the MATLAB/OPC link. Other controller parameters can be changed dynamically such as filter parameters which control the speed of response.

In industrial applications the PLC would be integrated with a HMI panel e.g. the SIMATIC Panel for machine-level operator control and monitoring of the plant.

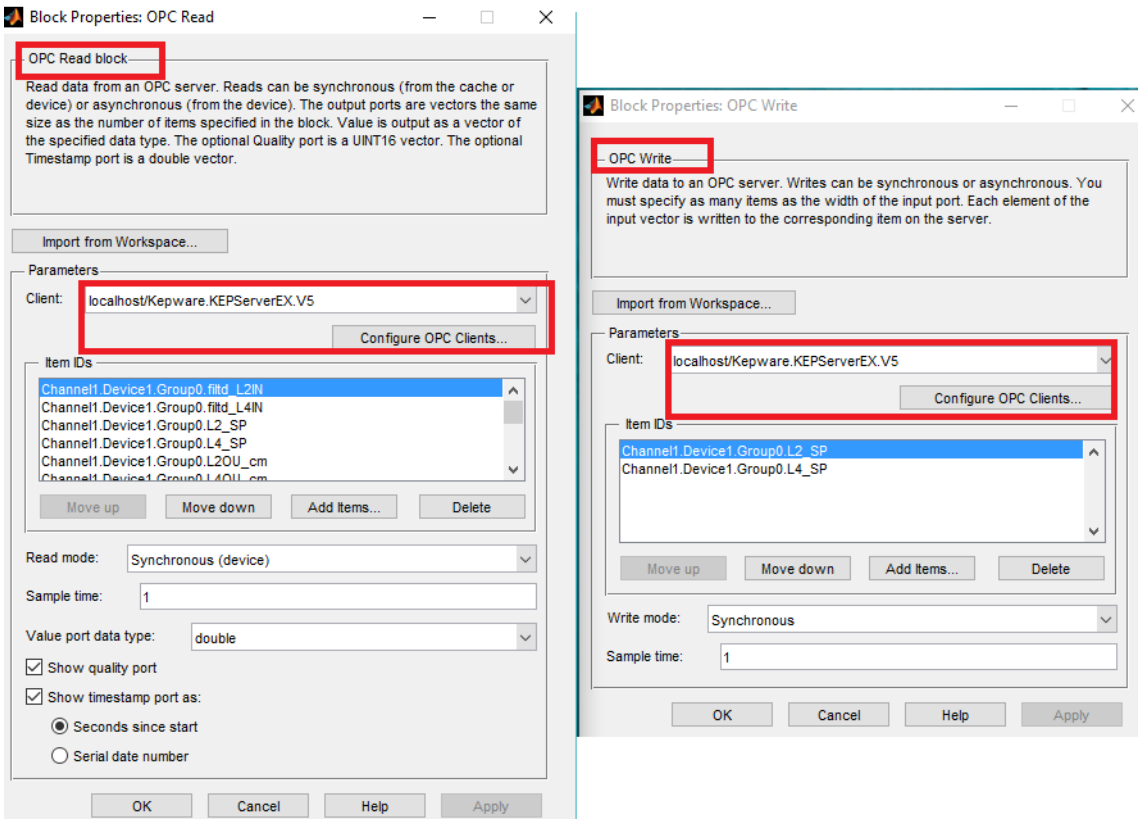


Figure 5.5: OPC Read/Write to KepserverEX5 channels.

5.1.2 Program Structure

The PLC CPU runs two programs: the operating system and the user program. The operating system runs all sequences and functions not associated with any specific control task e.g. calling user programs, error handling and managing memory areas. The user program is created by the user for download to the CPU for a specific control task. The user program is organised in blocks which are self-contained program sections [89]. The main blocks available in the S7-300 PLC are:

- Organizational Blocks (OB): This is the interface between the CPU and user

program that defines the structure of the user program. They are called by the operating system for controlling start-up, cyclic and interrupt execution and error handling.

- Functions (FC): These are user programs for frequently used routines. They are logic blocks without a memory hence any temporary variables used in the function are lost after execution.

- Function blocks (FB): These are logic blocks with a memory. It is assigned a Data Block as its memory such that any temporary variables are lost after execution but static variables are saved in the instance Data Block after execution.

- Data Blocks (DB): These are data areas assigned to a Function Block for storing user data.

5.1.3 PLC code

The PLC code structure for the TMIA controller shown in Figure 5.6 is summarized as follows:

Offline: The design parameters are computed with MATLAB offline for storage in PLC memory. The Hessian matrices, the difference equations for the plant model and controllers Q_b and Q_f etc are prepared beforehand.

Online:

1. OB1 (Main program): This block runs standard PLC tasks and housekeeping functions. It also runs during CPU idle time between OB35 calls. It has the lowest priority of monitored OBs.

2. OB100 (Complete Restart): This is the start up block for warm restart. Run initialization parameters are written here. Design parameters such as Hessian matrices are initialized in this block. This block is called once when the controller is started.

3. OB35 (Cyclic Interrupt): This is a timed loop that executes at fixed intervals. The TMIA closed loop program is written here. The default interval for this block is 200ms however is set to execute every 1s. This block reads the plant reference from the OPC server, calls FB blocks with their associated DBs which contains the plant model and also calls FC blocks which contain the quadratic program. The FC blocks called calculate the optimized plant inputs to be passed to the outputs of the PLC and read by the OPC server. The block runs every second till the experiment time is elapsed.

4. OB82-OB86: These blocks are assigned for handling asynchronous errors such as rack failure, errors like short circuit or removal of an input module etc.

5. FCs (Functions): The FC blocks are used to code the main quadratic program functions and also combine the models of the plant (G_{11}, G_{12}, G_{21} and G_{11}) and the controllers by calling their function blocks.

6. FBs (Function Blocks): The FB blocks along with their corresponding data blocks are used to write the parameters of the discretized plant model $\tilde{G}$, the controllers Q_b and Q_f, PLC inputs and outputs, setpoints and filters.

7. FC105 and FC106: These are in-built scaling blocks native to the S7 PLC used for analog input and output conversions. The inputs to the PLC are of integer type and need to be converted to float type values. The PLC input is scaled using the FC105 block and unscaled using the FC106 block.

5.1.4 Optimization of memory utilization

The PLC program utilizes both memory and temporary variables. The values of the memory variables are stored in the PLC memory until a reset occurs while temporary variables are not stored. MD (Memory Double Word) variables are used in the TMIA program, however the S7-300 CPU 314C-2 PN/DP model provides 256 bytes of memory for the MD format which are 4 bytes long. This results in a total of 64 MD variables

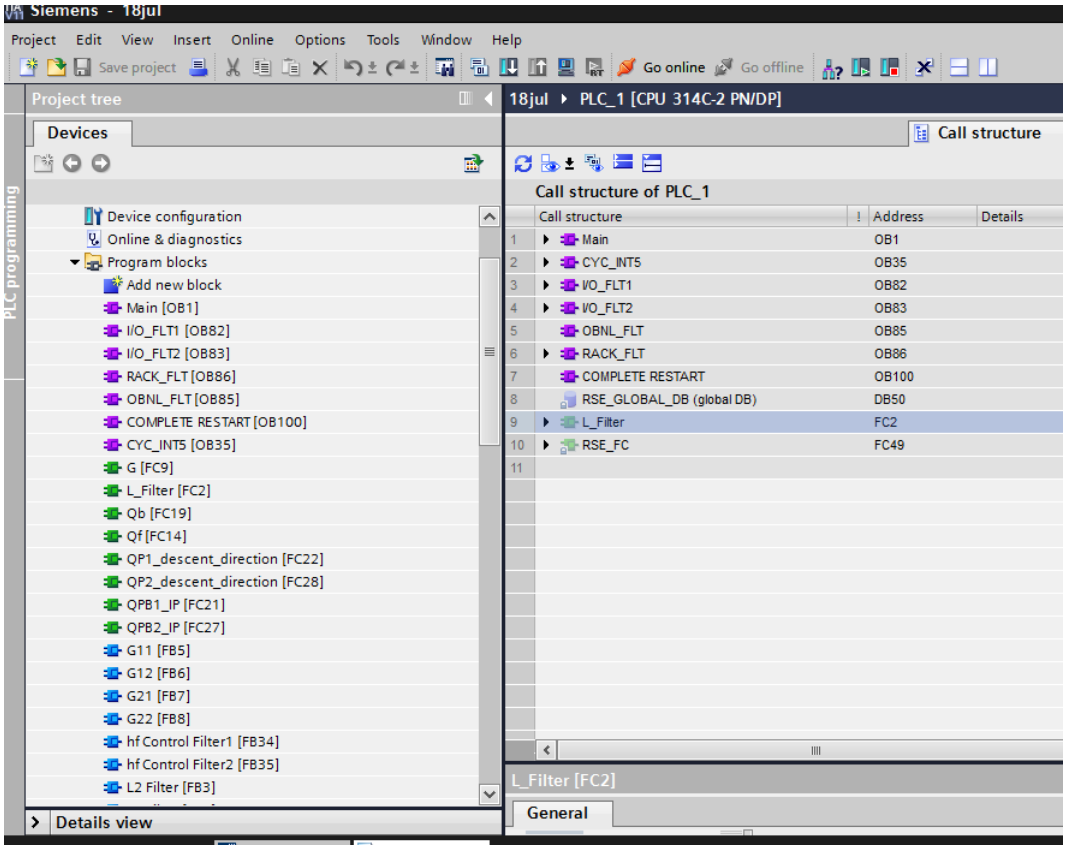


Figure 5.6: PLC code call structure.

available for use. Since this is insufficient for the TMIA program, the problem is overcome by mainly using temporary variables in the main program (OB35) and functions (FC) while only a few variables which need to be pre-loaded to memory such as the QP parameters and variables which need to be read and written to by the OPC server are stored as MD variables. The use of MD variables can also be limited by taking advantage of the symmetry of the Hessian matrices where only one off-diagonal element is stored in memory. The design parameters are calculated offline and stored in memory to reduce online computation so the PLC program focuses on implementing the TMIA closed loop system and solving the quadratic programs.

5.1.5 QP Algorithm implementation

Three different quadratic program algorithms are implemented in the PLC using ladder logic: a basic primal-dual interior point algorithm as outlined in [56, p. 484], active set method for convex QPs outlined in [56, p. 472] and the Projected fast gradient method specified in [90] and also in [43, 91, 92]. The positive definite symmetric weighting matrices, W and Q_{ss} described in Chapter 4 are chosen as identity. The stopping criteria implemented in the PLC for each of the three QP algorithms are given the same tolerance level of 10^{-3} in order to provide an effective comparison. Also the maximum number of iterations for the algorithms was set at 200 to ensure termination within the scan cycle. For each iteration of the active set algorithm, the amount of computation carried out is variable while the computations required for the interior point method is fixed.

5.1.6 Computational considerations

It must be noted that the Siemens PLC does not support matrix multiplication and as such discrete transfer functions are used instead of state space representations. A major computational step in the implementation of the quadratic program of section 4.4.1 is the solution of the linear system of equations which may involve a Cholesky factorization and backward or forward substitutions. The matrix multiplications and factorizations are carried manually in the logic the way one would normally solve them by hand. Complex algorithms are no easy feat to code on a PLC using ladder logic and as such a text-based language like Structured Control Language (S7 SCL) which is better suited for loops and conditional branches could be used. However, the ease of understanding would be lost as graphical user programs are more in demand in industry due to the necessity for debugging and tuning.

5.2 PLC implementation results

A comparison of three algorithms based on computational demand on the PLC is shown in Table 5.1. It can be seen that the projected fast gradient method is most

Item	Active Set	Interior Point	Projected fast gradient method
% of total PLC work memory used by TMIA algorithm	23%	19%	16%
Highest number of QP block iterations per scan cycle	7	37	16
Highest execution time per scan cycle (OB35 block)	6ms	30ms	5ms
QP code size (No. of FLOPs required per QP)	421	186	70
% of used memory consumed by 2 QPs	38.6%	25.5%	10.3%

Table 5.1: Comparison of QP algorithms (minimum phase).

suitable for the PLC implementation as it requires the least amount of memory (16%), PLC coding (70 FLOPs) and computation time (5ms). It should be noted that the results obtained are based on coding in ladder logic which would require several ladder rungs to implement the checking conditions required for *if then else* statements.

From the results table in Table 5.1, the Active set ladder logic algorithm has the largest QP code size due to the number of *if else* checking statements carried out on the working set. An important note on Table 5.1 is that even though the Active set algorithm uses the most PLC code size, execution time remains low with the lowest number of QP iterations (7) since not every ladder rung is implemented during each scan cycle. The implementation of the rungs are dependent on the contents of the active and inactive working sets which changes with each iteration (see ref [56]). There is sufficient reason to believe that the Active set method would yield even more promising results if programmed used a text-based language.

The interior point method used the most computation time of the three algorithms

due to the number of matrix multiplications and back-solves carried out. This is inherent in the formulation of the interior point algorithm and as such we use the results as a worst case scenario. For Interior point, TMIA program utilized only 19% (36.2KB) of the PLC working memory, 10% of its available RAM and 0.12% of retentive memory as shown in Figure 5.7.

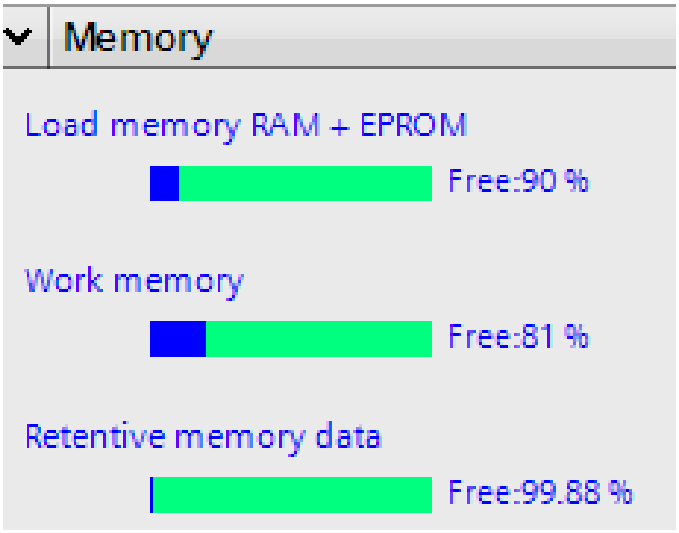


Figure 5.7: PLC Memory usage (TMIA controller using Interior point method)

25% of this total utilized memory is contained in the two quadratic programs of the TMIA control structure. Each interior point algorithm utilized 186 FLOPs including move operations. The highest number of iterations for the quadratic program is 37 as shown in Figure 5.8. The rise and fall in the plots of Figure 5.8 closely follow the pattern of the set point profile 1 used in the experiments in Chapter 6. It can be seen that the number of iterations of the QP and execution time of the cyclic program (OB35) tends to increase as the set-point requested increases and vice versa. This is due to the constraints being active for high set points. From the plots in Figure 5.8 it is seen that there is a direct relationship between the number of iterations and the execution time because the QP called within the OB35 block much reach the optimum solution within the scan cycle. Also the number of FLOPs used clearly impacts the memory utilized.

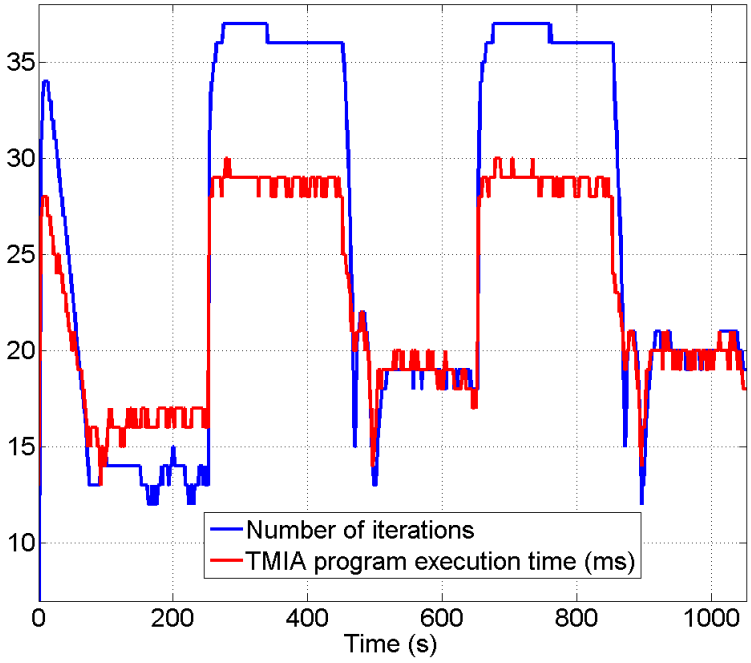


Figure 5.8: PLC code: Number of QP iterations per scan cycle and TMIA program execution time for OB35 block (Interior point algorithm).

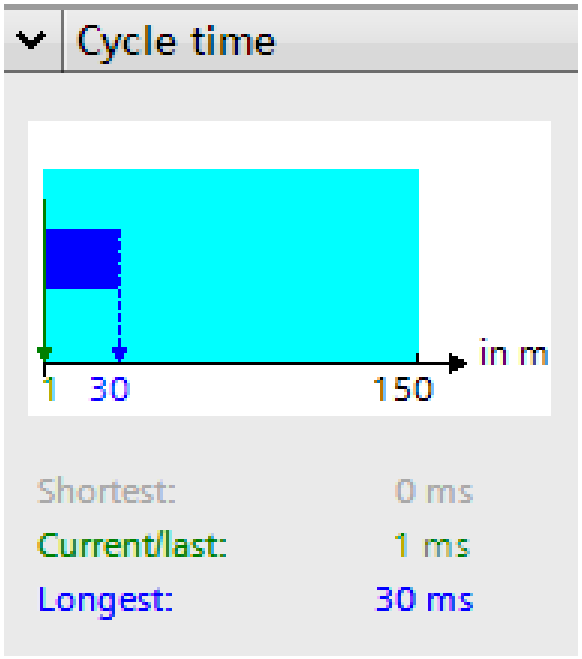


Figure 5.9: PLC online diagnostics: Longest and shortest execution time

The longest cycle time for the execution of the TMIA program obtained from the PLC online diagnostics (Figure 5.9) was 30ms which is far below the scan cycle monitoring rate of 1s. Thus the optimal solutions for the references and control inputs were always obtained during each scan cycle even for the slowest of the three algorithms. These results indicate the the PLC can be used for the control of faster, larger and more complex processes.

5.3 Summary

Three main contributions are presented in this chapter. First the results show that a simple and efficient advanced controller like TMIA controller can be realized on a standard PLC which is an affordable industrial alternative to MPC for input-constrained multivariable processes. This is beneficial in an industrial setting since PLCs provide the stability and reliability which most digital computers do not offer.

Secondly, three algorithms have been coded on the PLC and projected fast gradient method was found to be the most suitable QP algorithm for the PLC implementation due to its least computing requirements. The memory utilization and execution time in the region of a few milliseconds using the TMIA controller on the PLC has shown the technique to be a useful tool for the practical implementation of optimized multivariable control that can be applied to much faster processes.

Thirdly, advanced algorithms like quadratic optimization can be coded on a PLC from scratch using ladder logic for ease of understanding and debugging by operators and technicians in industry.

Chapter 6

Experimental Results - Quadruple Tank Process

This chapter provides experimental results of the implementation of a classical IMC controller (IMC), Modified IMC controller (MIMC), TMIA controller and an MPC controller on a multivariable quadruple tank process. Comparison in this chapter is made based on performance rather than computation as in the case of Chapter 5. The TMIA competes favourably compared to other IMC schemes in terms of handling windup and directionality issues.

The transient and steady state performances of the controllers are compared especially looking at the effects of windup such as overshoot, directionality issues where the controller steers the plant in the wrong direction leading to large steady state error and also multivariable coupling effect.

6.1 Performance

In order to test the efficiency of the TMIA controller, three set-point profiles were applied to the Quadruple Tank to activate the constraints and thus sufficiently excite

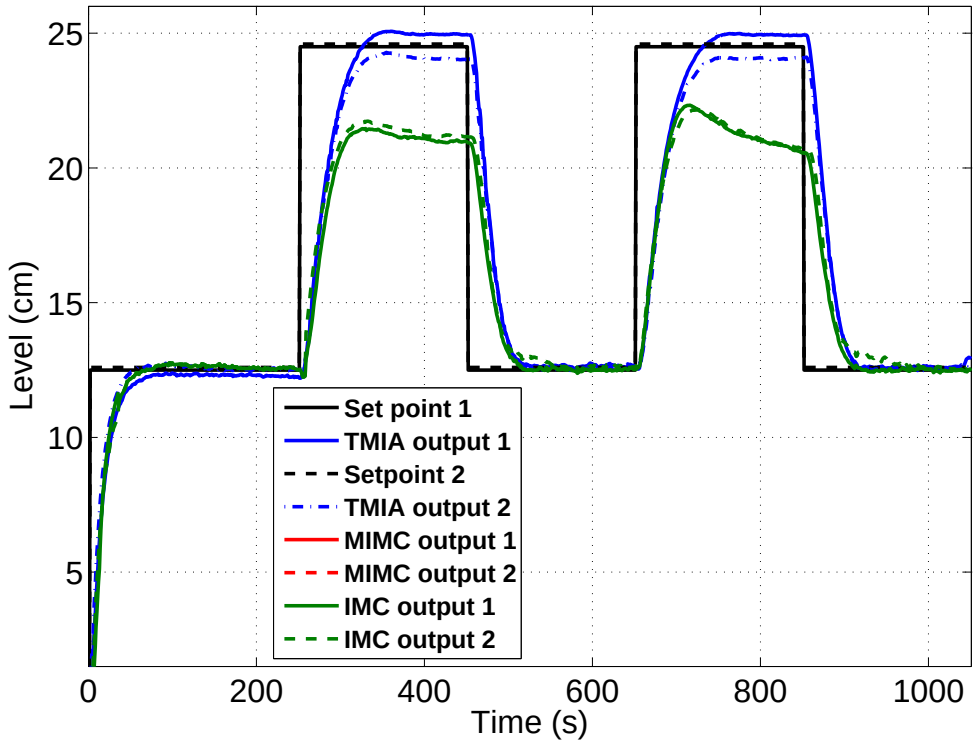
windup and process directionality in the system. Sufficient step magnitudes were chosen to emulate situations where the controllers to be compared would normally yield a degraded performance.

- Profile 1: Level set points of 12cm applied in the same direction with no lag between them.
- Profile 2: Level set points of 4cm applied in the same direction with a 50s lag between them.
- Profile 3: Level set points of 2cm applied in opposite directions with no lag between them.

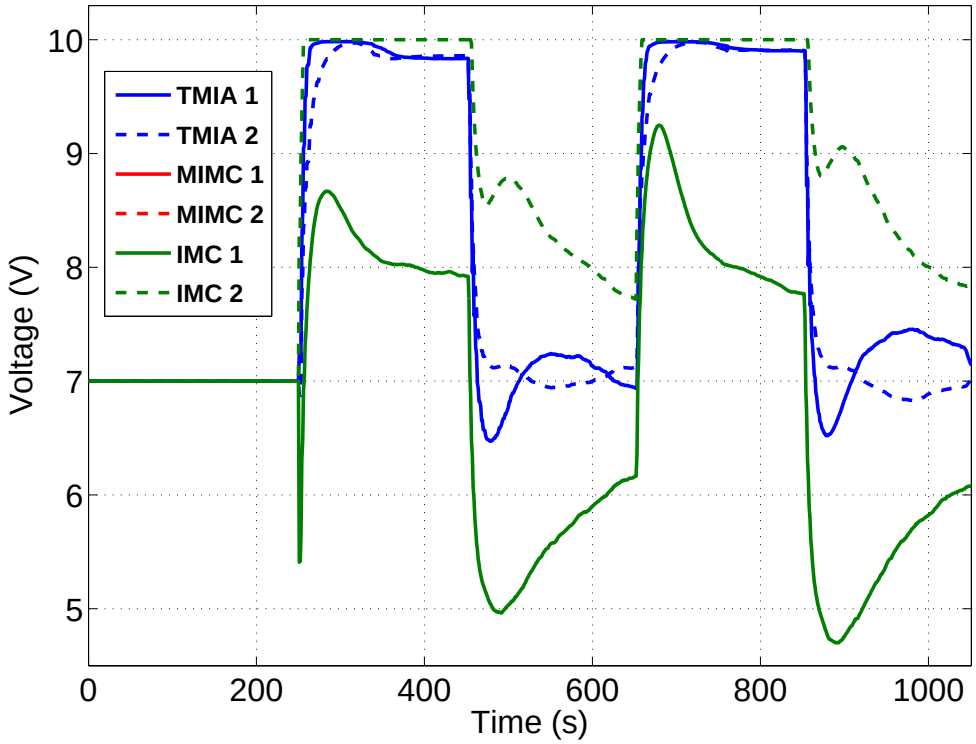
The input voltages to the pumps are constrained between 0 and 10V in the algorithm. The results of the PLC implementation of the TMIA controller are compared with classical IMC and MIMC structures as shown in Figures 6.2 to 6.5. The tank levels were allowed to settle for 250s in open-loop with the application of (7V, 7V) operating voltage before closed-loop control was applied.

6.1.1 Minimum phase

Profile 1: In Figure 6.1 the IMC controller yields a poor performance due to saturation. This response is as expected since the IMC controller takes no account of the saturation effect and as such the output does not track set-point when the constraints are active. The MIMC output in Figure 6.2 yields a better performance where Tank 1 tracks the reference signal but Tank 2 is unable to meet the set-point due to active constraints. The TMIA has the least steady state error as seen in the figures. The TMIA algorithm works by recalculating an optimal realizable reference signal. It recalculates a slightly higher set-point for Tank 1 and a lower one for Tank 2 while keeping the levels as close as possible to the actual reference.

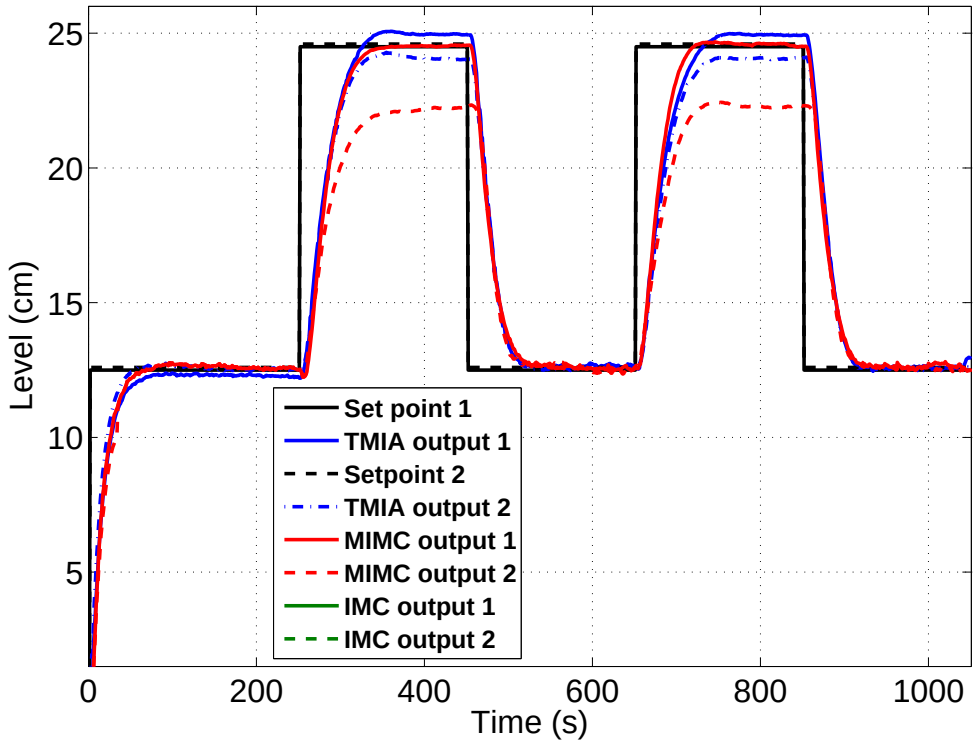


(a) Plant output Profile 1

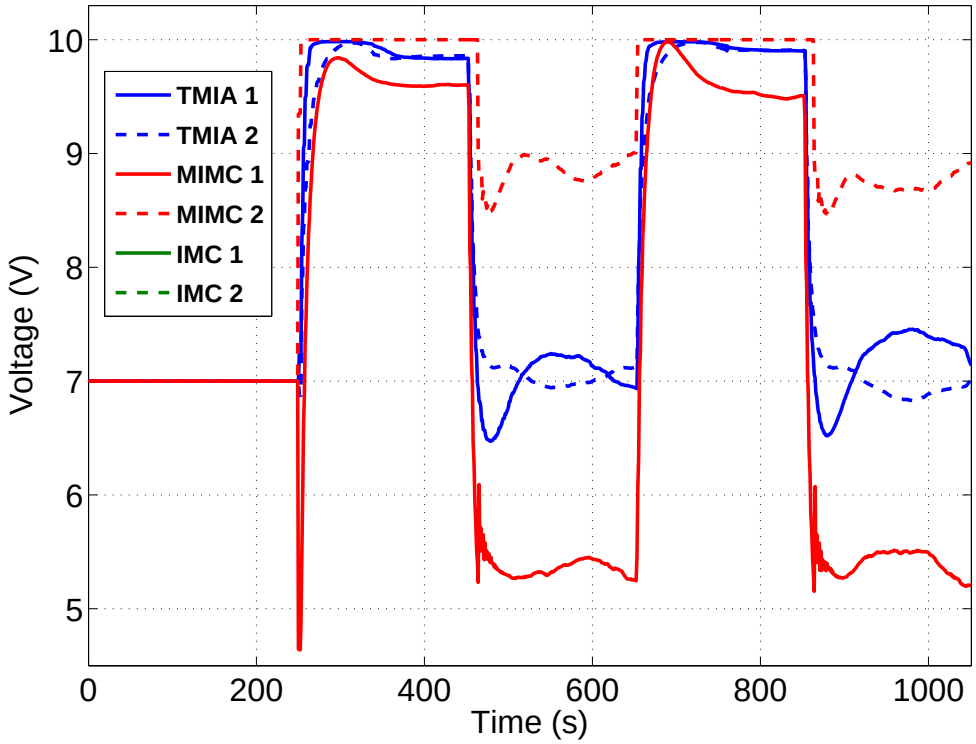


(b) Input voltages Profile 1

Figure 6.1: Controller comparisons for Profile 1 (TMIA vs IMC)



(a) Plant output Profile 1



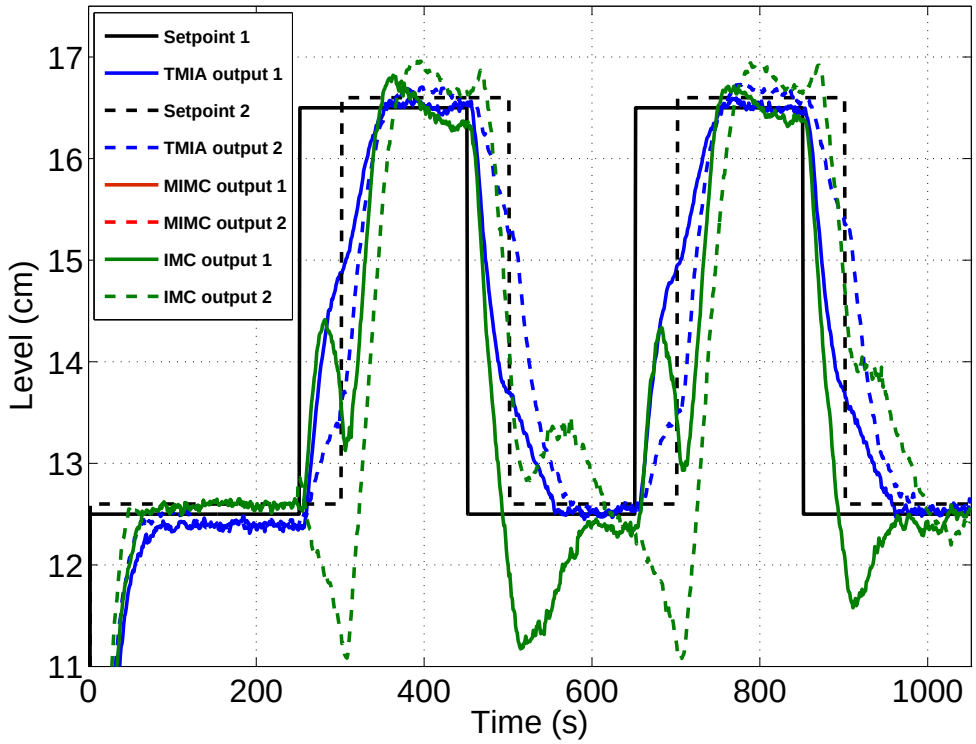
(b) Input voltages Profile 1

Figure 6.2: Controller comparisons for Profile 1 (TMIA vs MIMC)

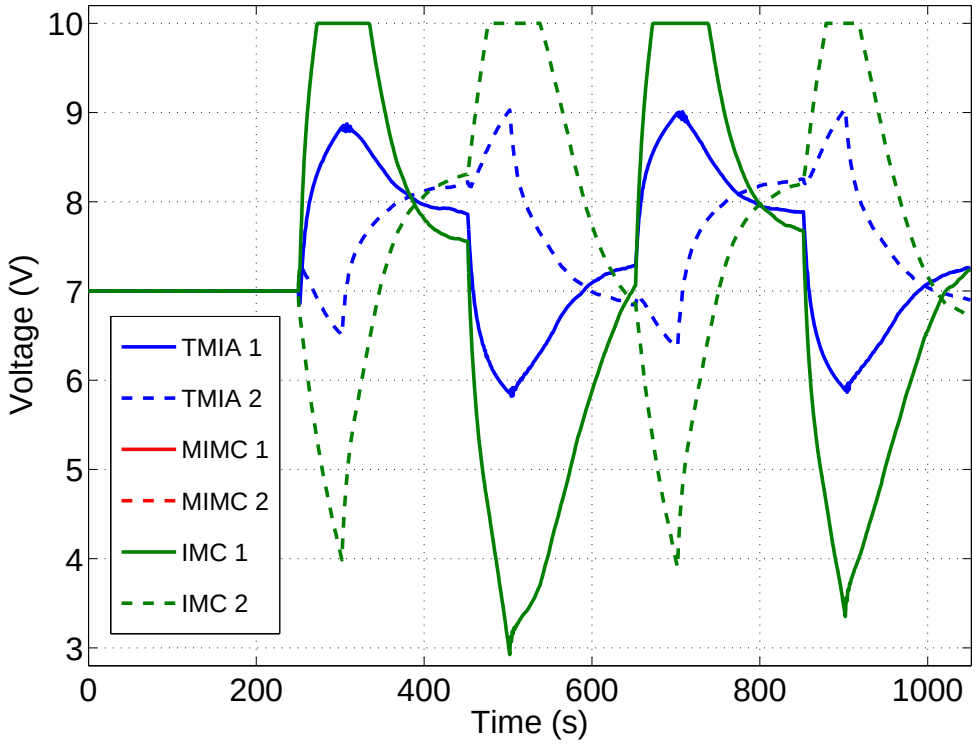
The steady-state QP uses the knowledge of the constraints to calculate a new reference. The weight W in the QP parameters can be adjusted to reflect priority on a particular output in maintaining the set-point. Profile 1 requests set-points close to the limit of the tanks capacity in order to force the TMIA controller to saturation even though for a short time as seen in Figure 6.1b. For a reduced set-point than this value, the TMIA controller achieves perfect set-point tracking and the realizable reference is kept as close to the set-point at all times.

Profile 2: In Figures 6.3 and 6.4, both the IMC and MIMC controllers achieve set point tracking but the effects of saturation are clearly seen in the overshoots/undershoots in the tanks where inputs constraints are active. Also, hints of directionality issues are seen in this time delayed profile and a strong coupling effect is exhibited. The IMC controller in Figures 6.3 yields the least performance with the largest undershoots and coupling effect. The TMIA controller performs better with set point tracking, no overshoots and only a small coupling effect is observed. The inputs for the TMIA control are far away from the constraints as seen in Figure 6.3b. The optimal references and optimal input voltages calculated by the quadratic program keep the actuator away from saturation while ensuring a smoother response.

Profile 3: In Figures 6.5a and 6.6a, both IMC and MIMC controllers are unable to handle directionality problems in the Quadruple Tank system, their control inputs in Figures 6.5b and 6.6b remain saturated for a long time and steer the output of Tank 1 in the wrong direction. The uncompensated IMC controller naturally yields the poorest response with the windup effect lasting for much longer. The TMIA controller handles the problem of directionality by recalculating optimal set-points and control inputs which though much lower, steer the strongly coupled plant in the right direction. The TMIA inputs as seen in Figure 6.5b remain unsaturated.

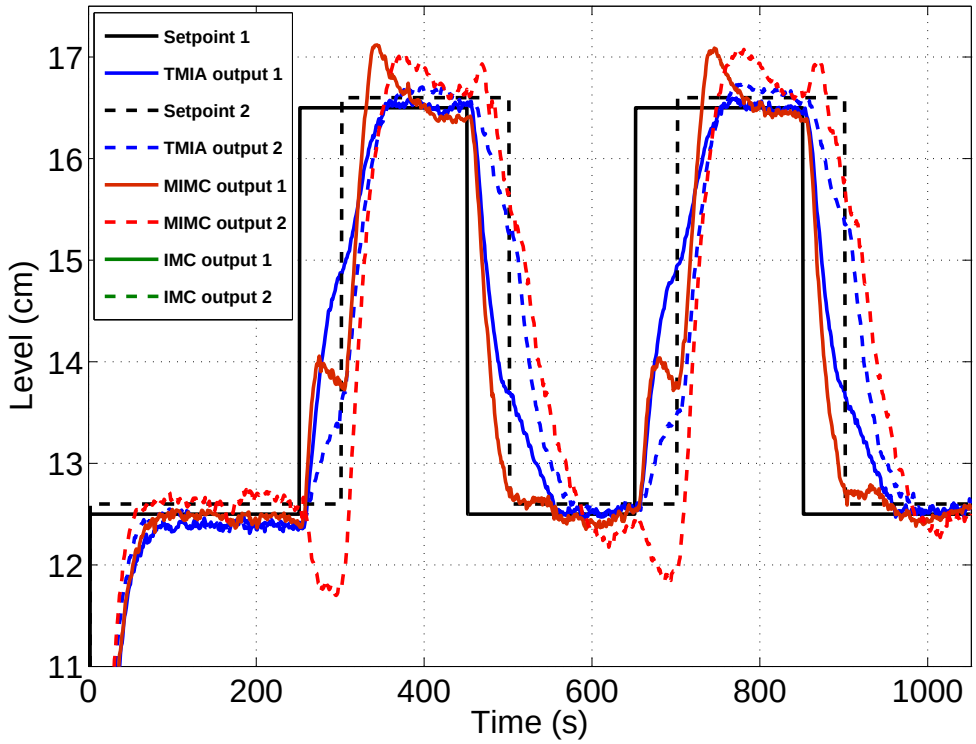


(a) Plant output Profile 2

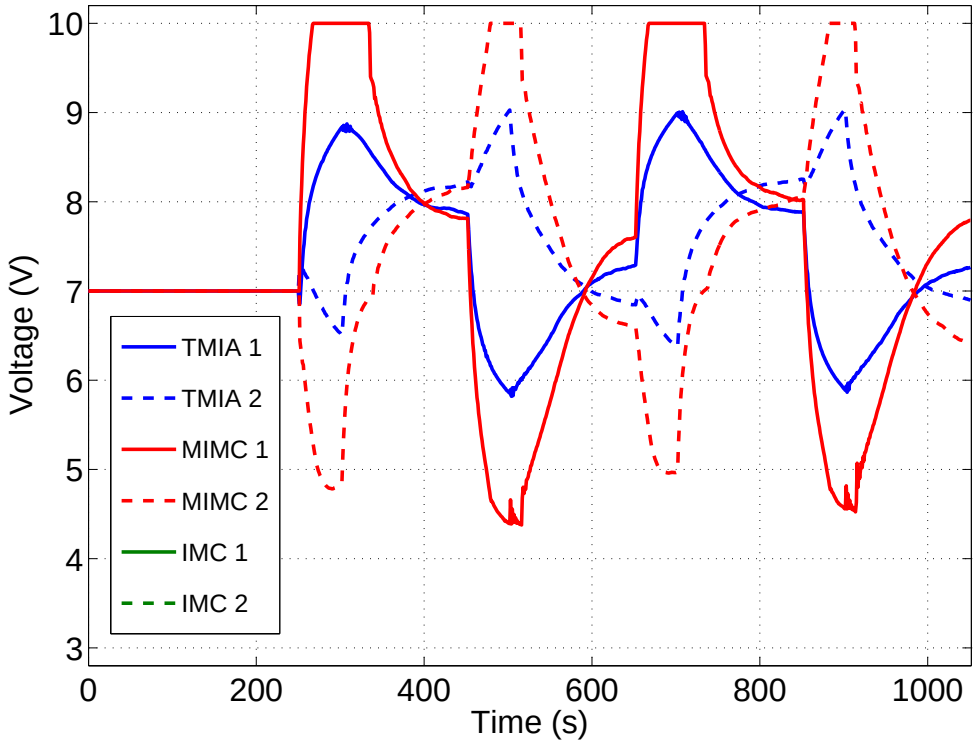


(b) Input voltages Profile 2

Figure 6.3: Controller comparisons for Profile 2 (TMIA vs IMC)

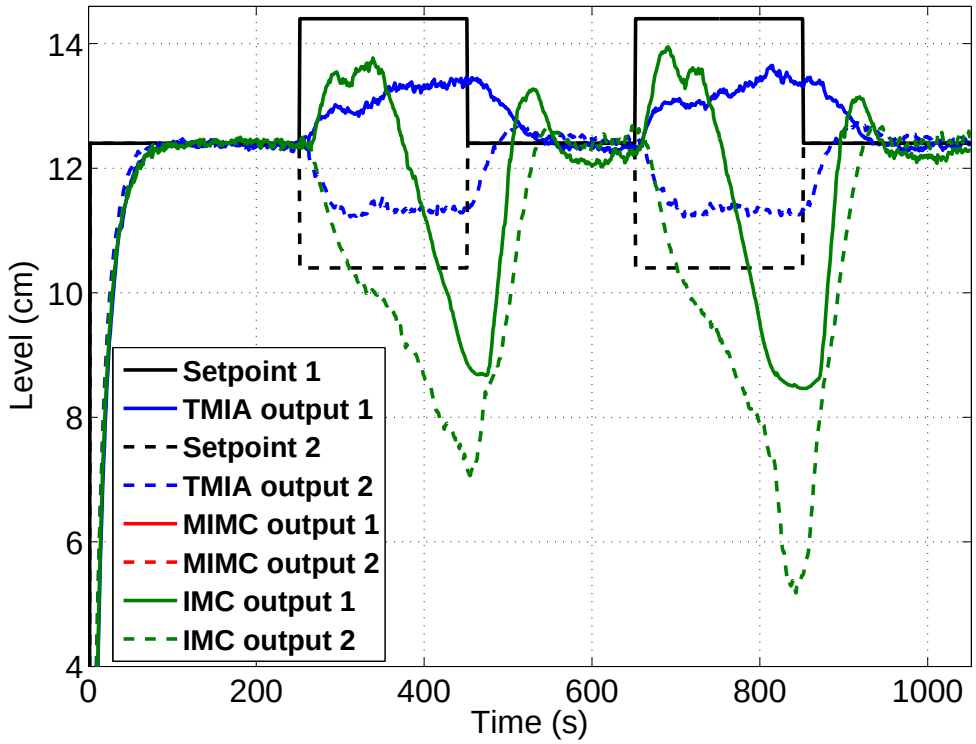


(a) Plant output Profile 2

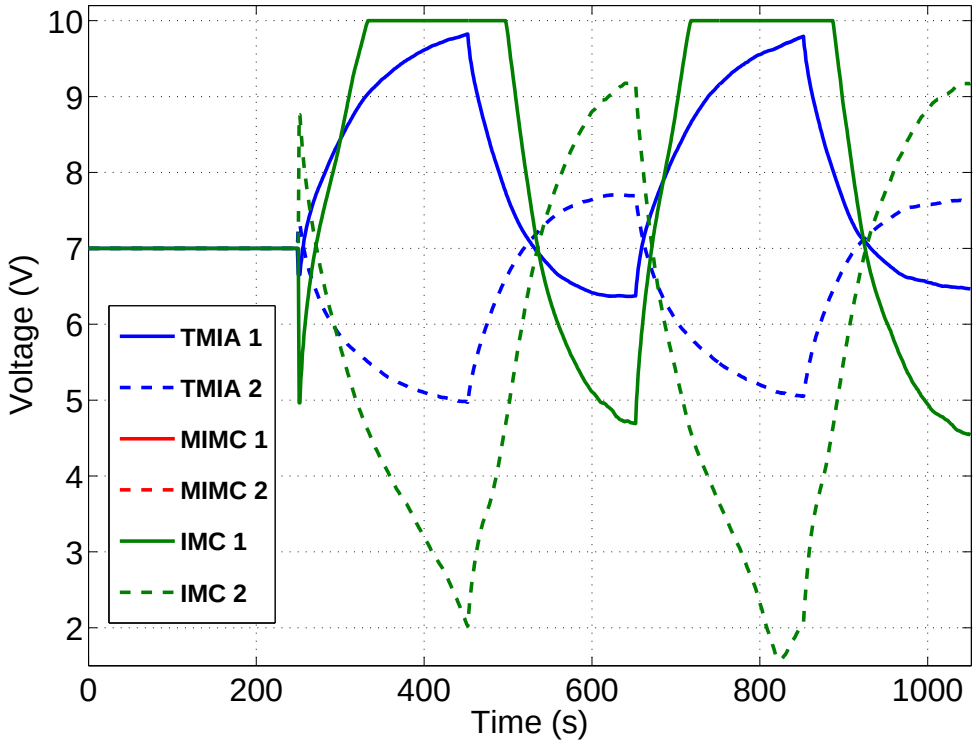


(b) Input voltages Profile 2

Figure 6.4: Controller comparisons for Profile 2 (TMIA vs MIMC)

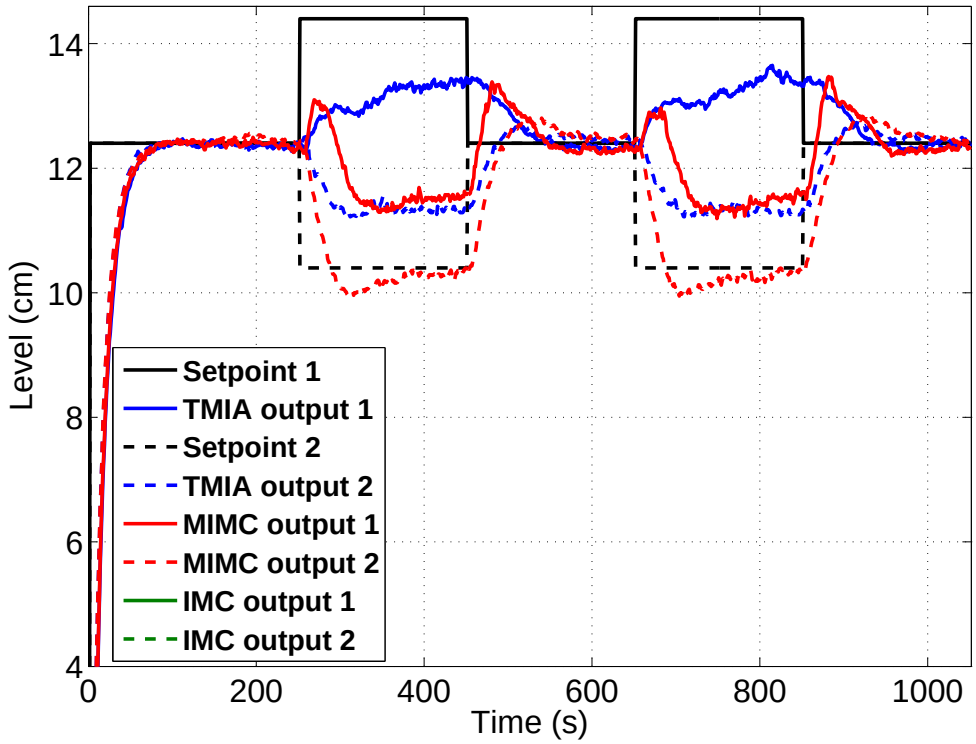


(a) Plant output Profile 3

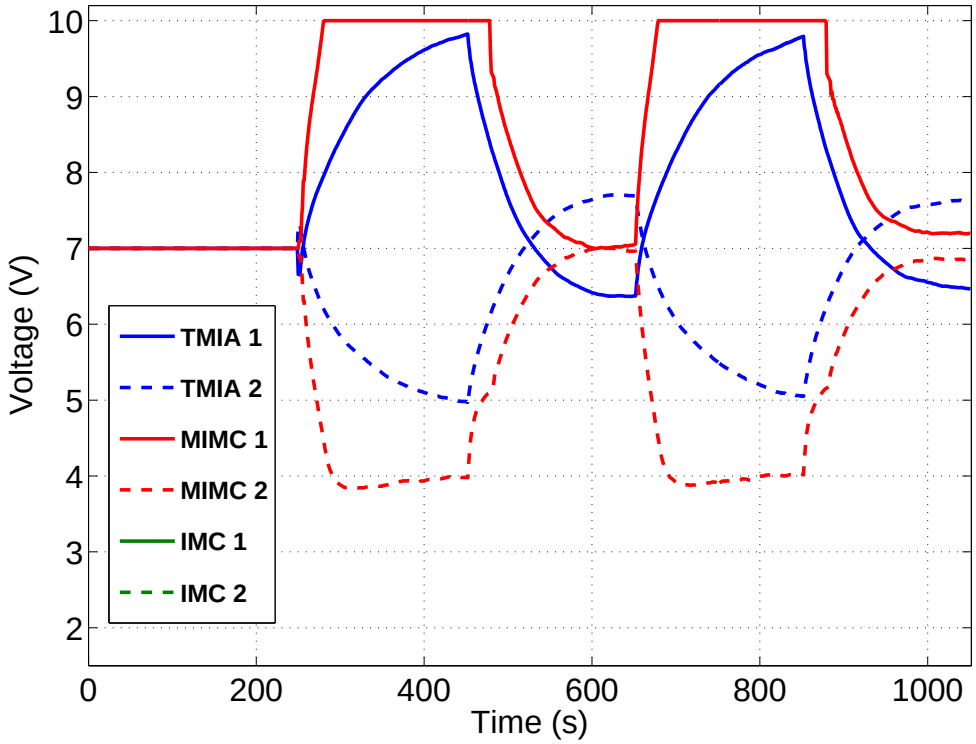


(b) Input voltages Profile 3

Figure 6.5: Controller comparisons for Profile 3 (TMIA vs IMC)



(a) Plant output Profile 3



(b) Input voltages Profile 3

Figure 6.6: Controller comparisons for Profile 3 (TMIA vs MIMC)

6.1.1.1 Minimum phase comparison

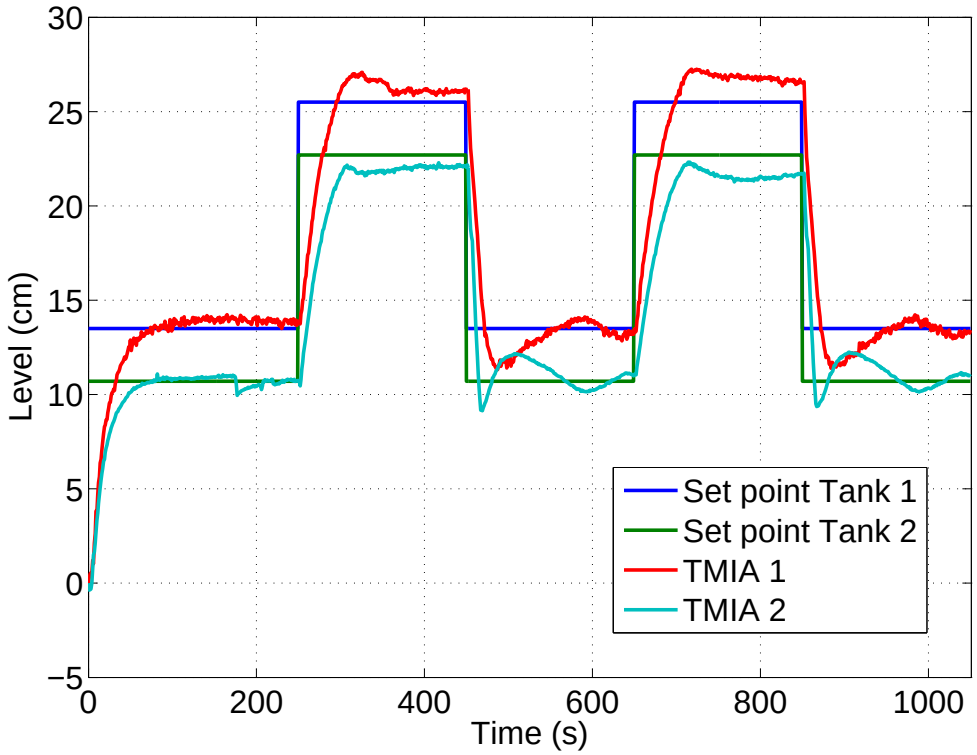
	Steady state error	Overshoot / undershoot	Directionality	Coupling effect
PROFILE 1				
TMIA	Minimum	Small	-	-
MIMC	Large	-	-	-
IMC	Largest	-	-	-
PROFILE 2				
TMIA	Minimum	Minimum	Small	Small
MIMC	Small	Large	Large	Large
IMC	Small	Largest	Largest	Largest
PROFILE 3				
TMIA	Medium	-	-	Small
MIMC	Large	Small	Large	Large
IMC	Largest	Large	Largest	Large

Table 6.1: Performance comparison Profile 1 (minimum phase)

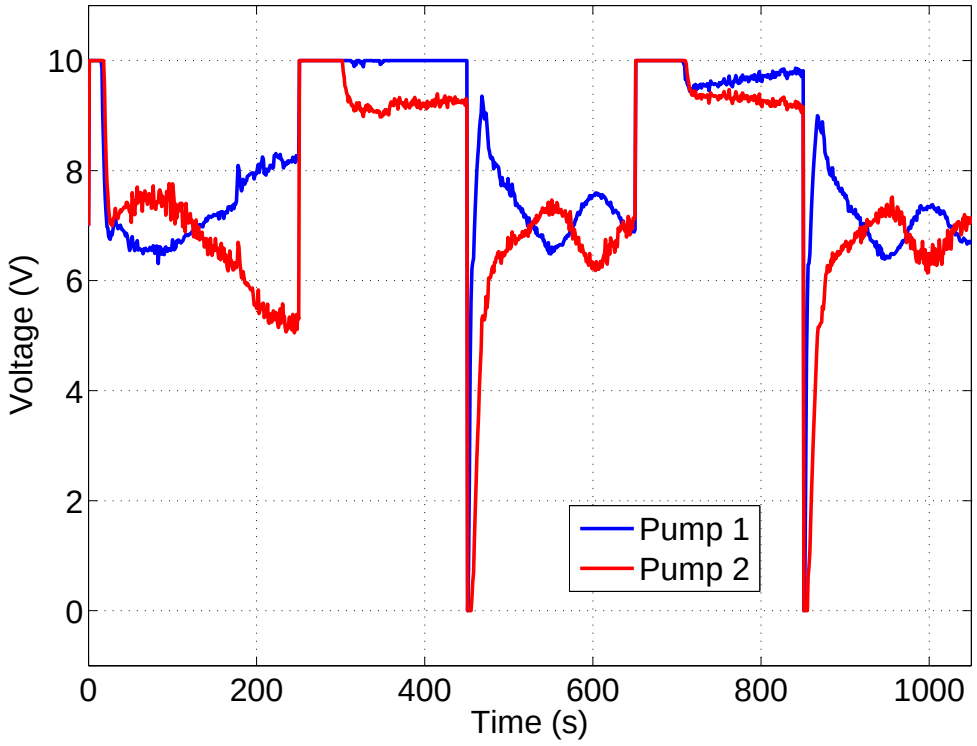
6.1.2 Non-minimum phase

The results for the non-minimum phase implementation for all 3 profiles are reported in this section for TMIA, MIMC and IMC controllers (see figures 6.7 to 6.15). The output responses show that the TMIA controller outperforms the classical IMC and Modified IMC structures in handling the performance degradation associated with controller windup and process directionality during saturation.

The non-minimum phase case is much harder to control [93], is 10 times slower and shows worse coupling effect than the minimum phase case. However it must be noted that for profile 3, the TMIA controller achieves better set point tracking than the minimum phase case (see Figures 6.13 to 6.15). This is because in non-minimum phase configuration, the outlet pipes are adjusted to allow more flow into the two upper tanks. Hence when profile requested set points in opposite directions, the optimum voltage calculated yields more water in the desired tanks.

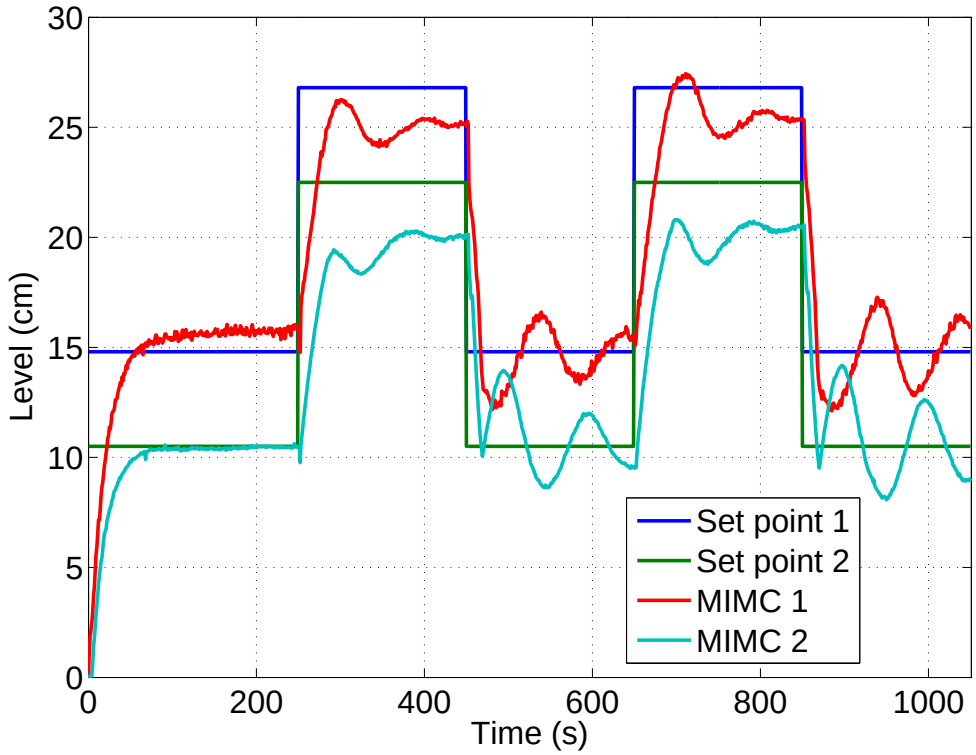


(a) Output responses

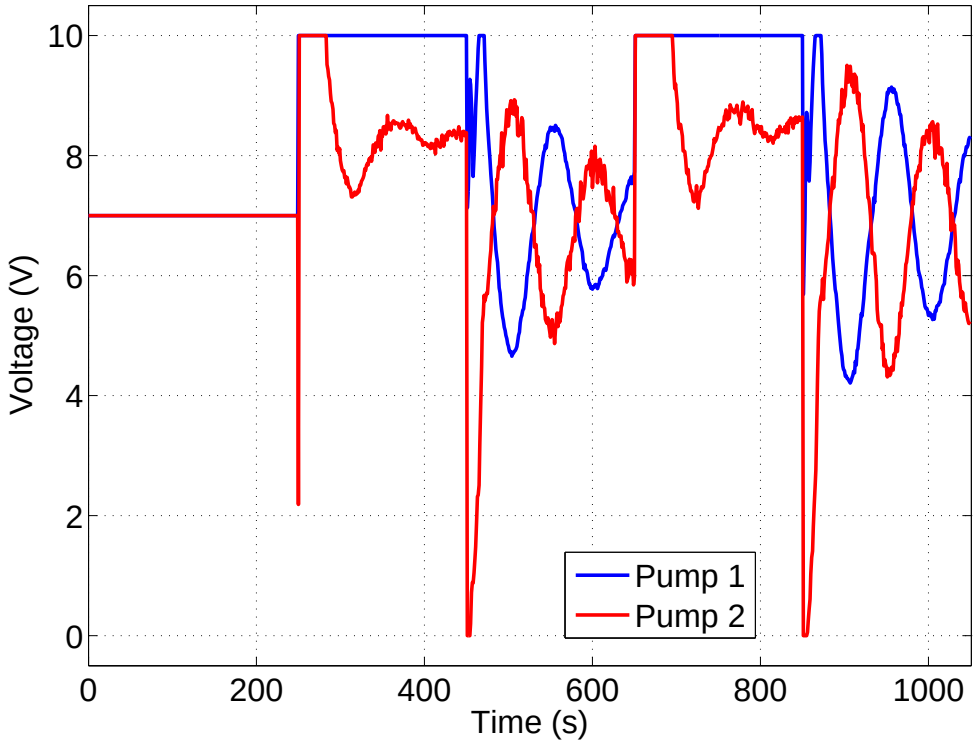


(b) Plant inputs

Figure 6.7: Profile 1: Real-time implementation of TMIA controller for non-minimum phase.

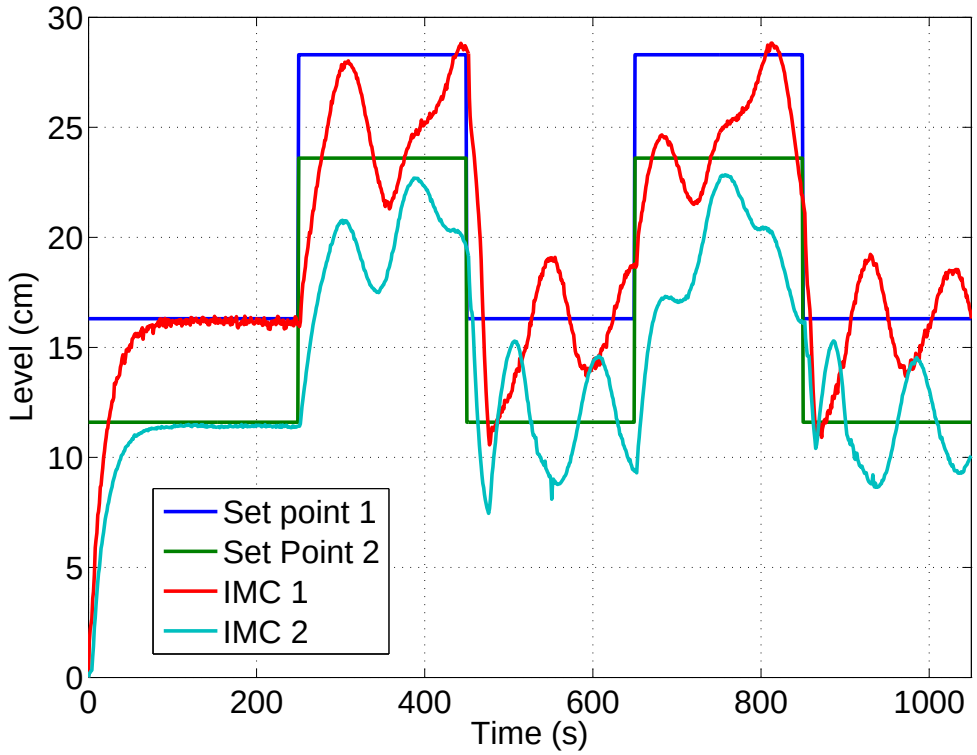


(a) Output responses

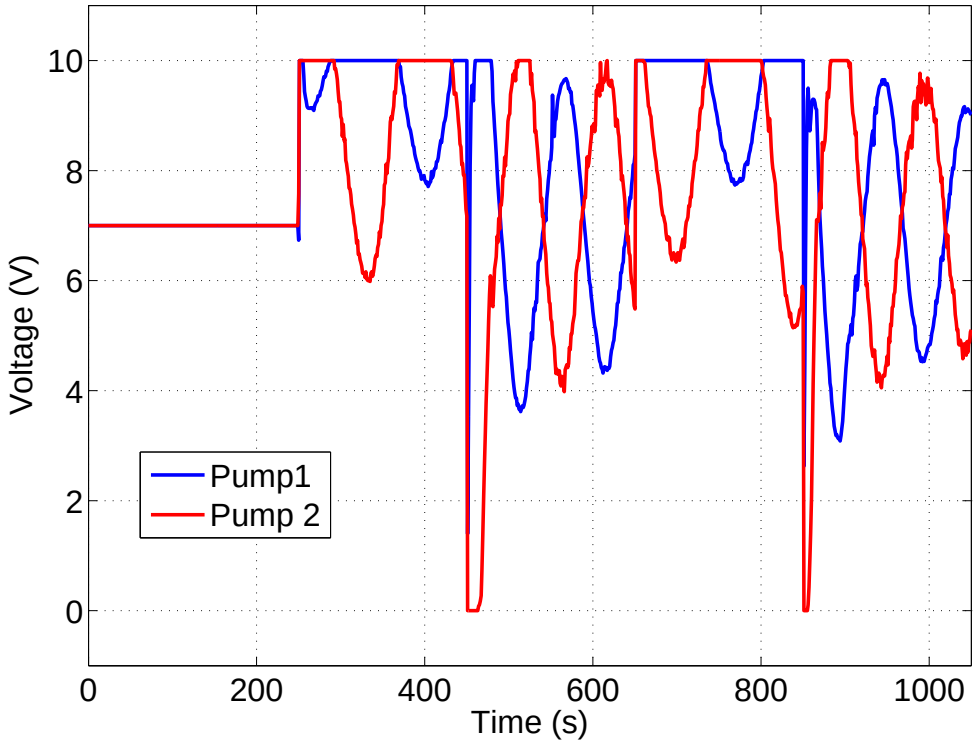


(b) Plant inputs

Figure 6.8: Profile 1: Real-time implementation of MIMC controller for non-minimum phase.

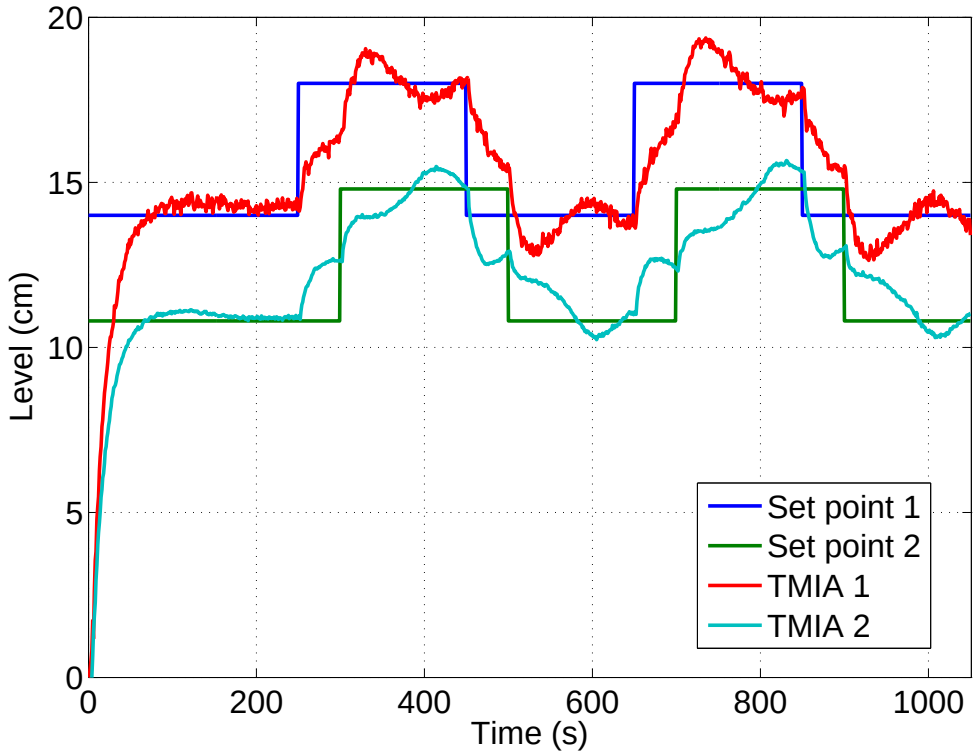


(a) Output responses

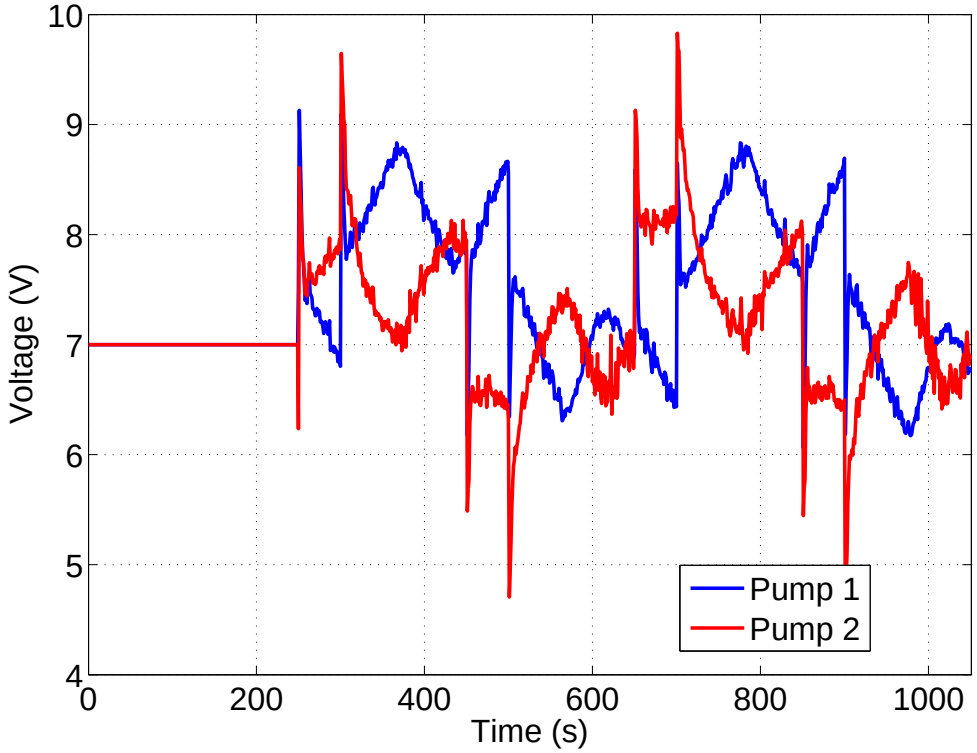


(b) Plant inputs

Figure 6.9: Profile 1: Real-time implementation of IMC controller for non-minimum phase.

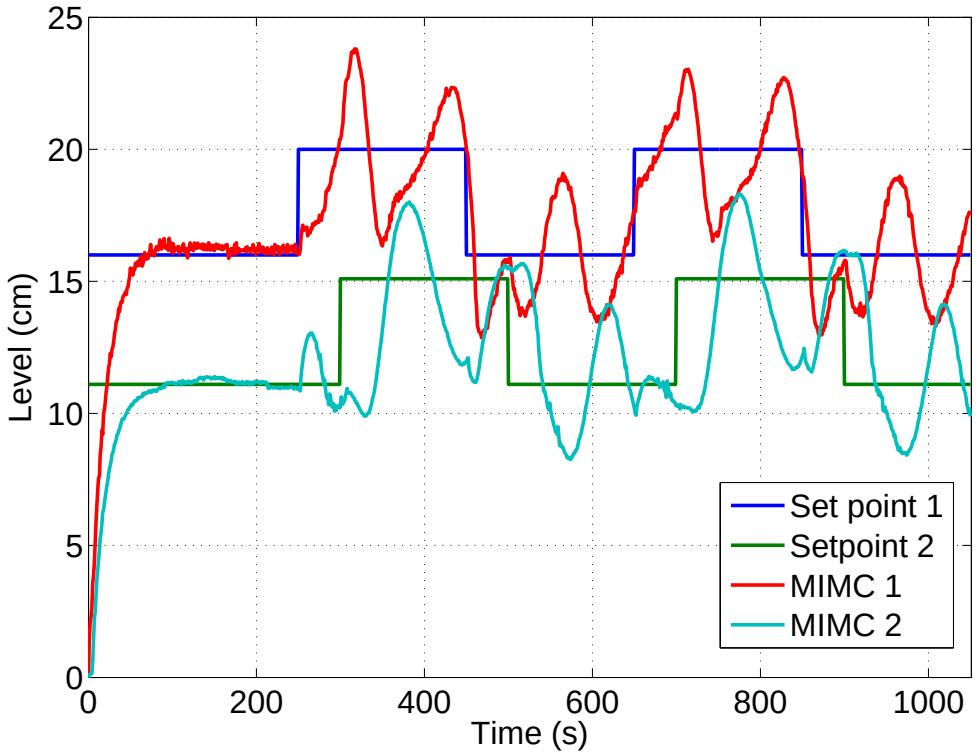


(a) Output responses

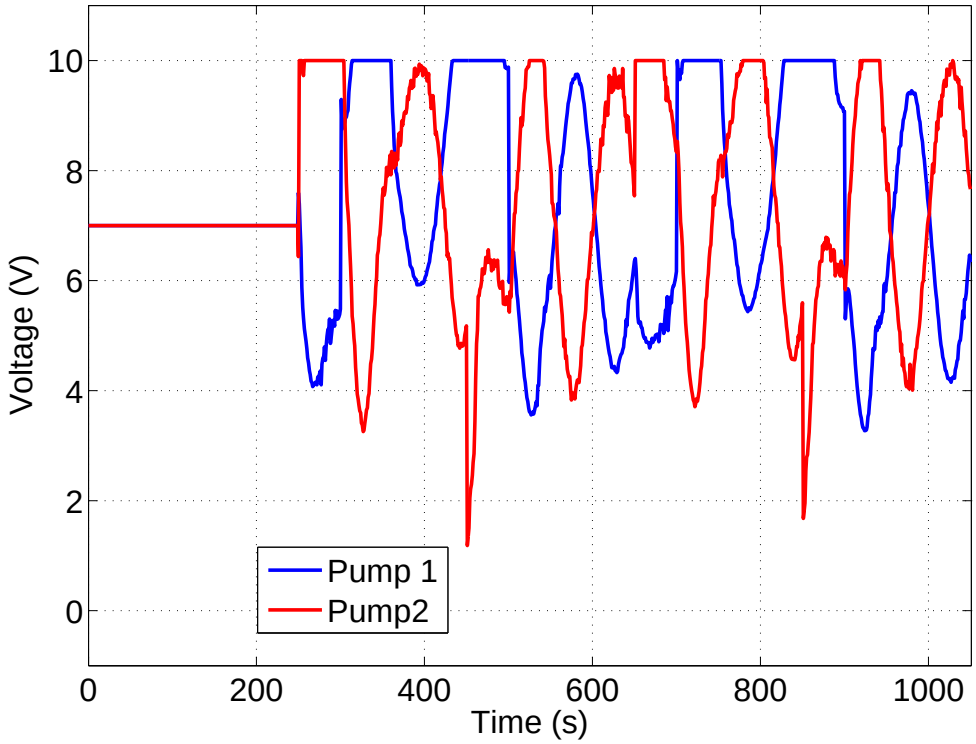


(b) Plant inputs

Figure 6.10: Profile 2: Real-time implementation of TMIA controller for non-minimum phase.

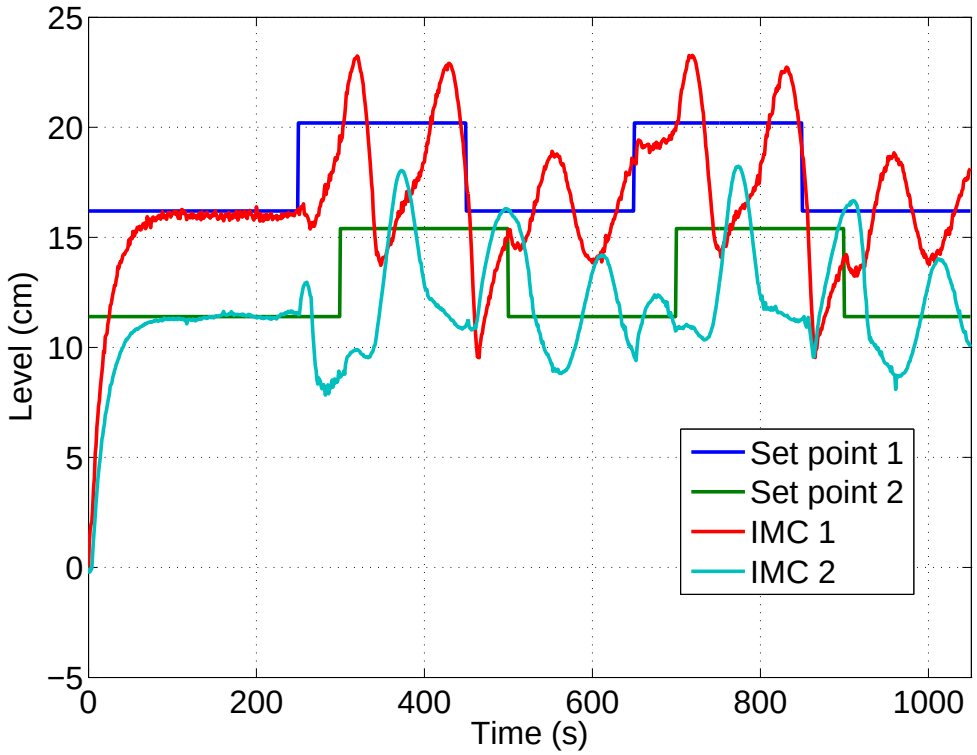


(a) Output responses

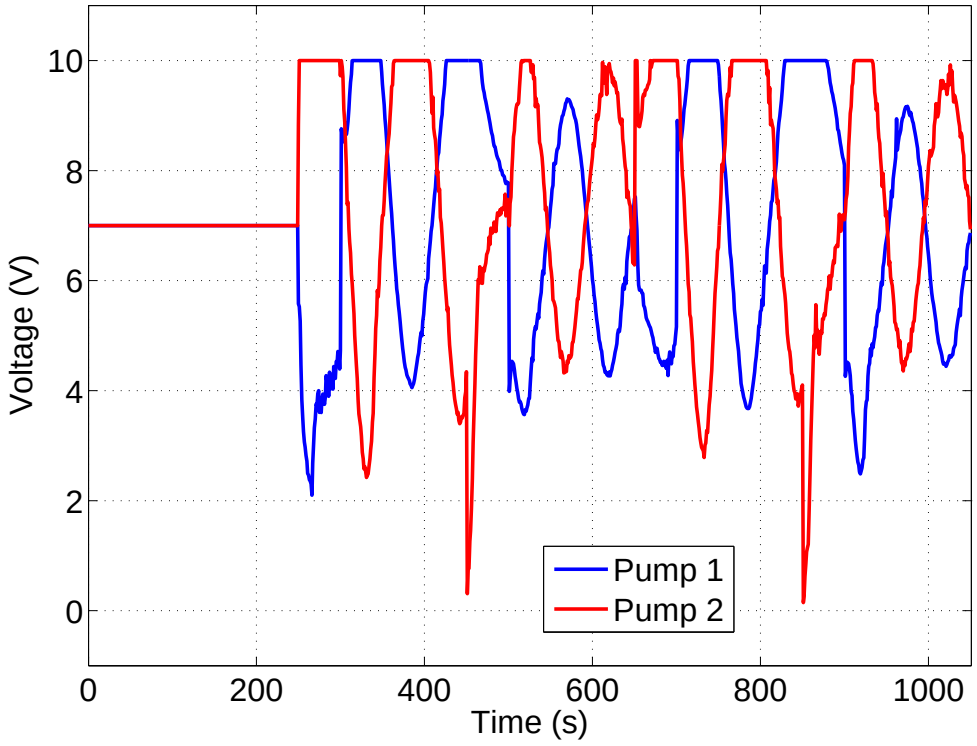


(b) Plant inputs

Figure 6.11: Profile 2: Real-time implementation of MIMC controller for non-minimum phase.

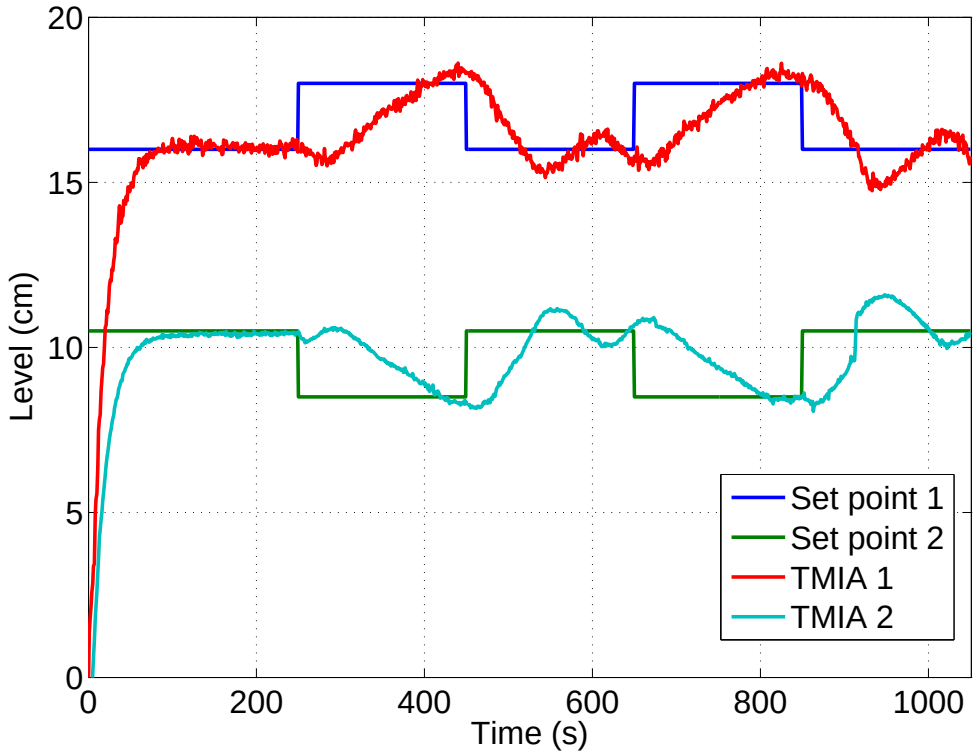


(a) Output responses

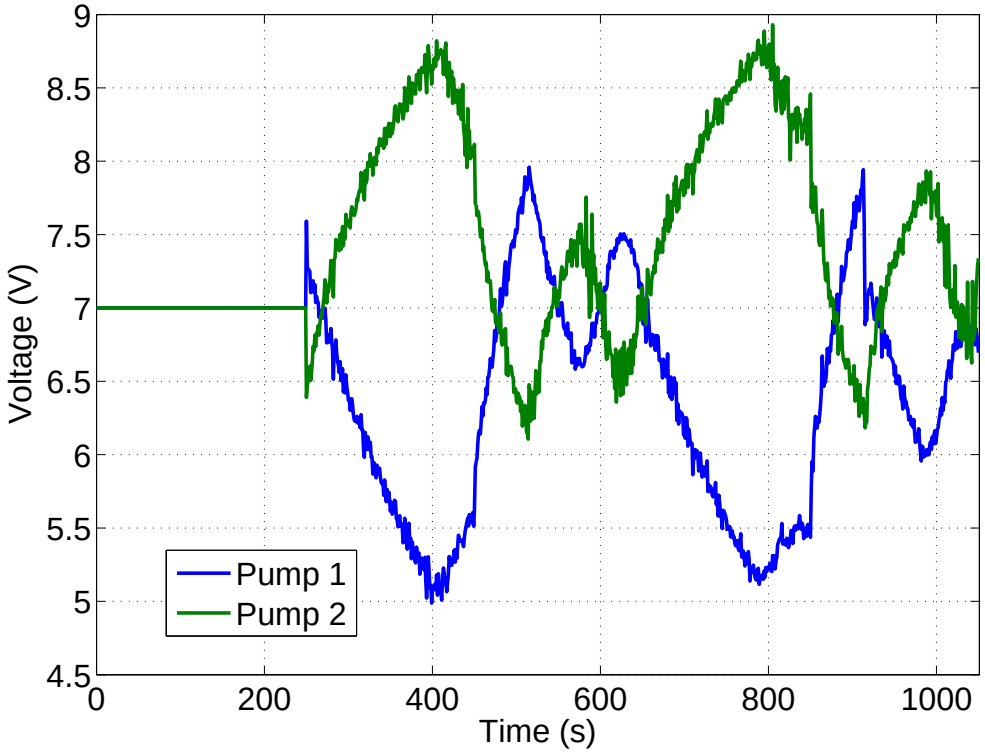


(b) Plant inputs

Figure 6.12: Profile 2: Real-time implementation of IMC controller for non-minimum phase.

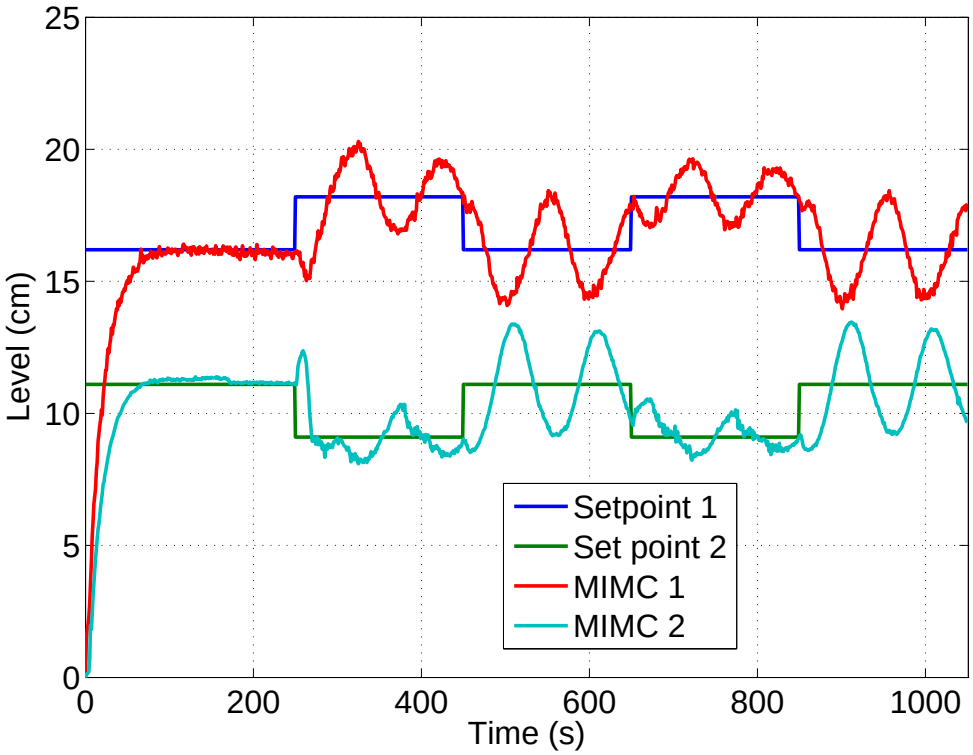


(a) Output responses

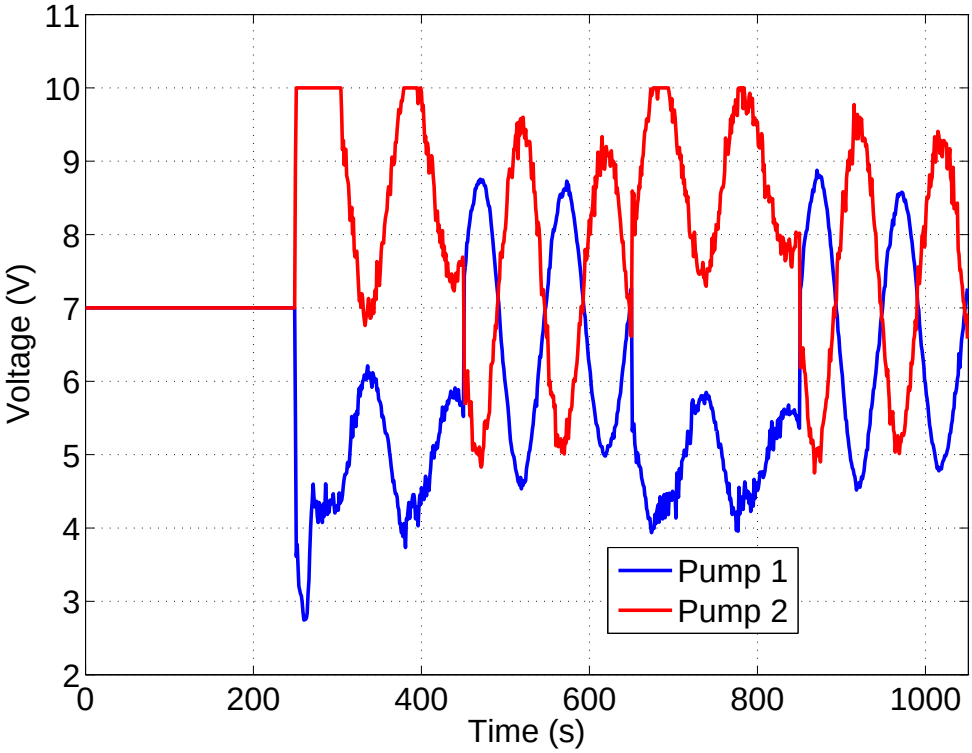


(b) Plant inputs

Figure 6.13: Profile 3: Real-time implementation of TMIA controller for non-minimum phase.

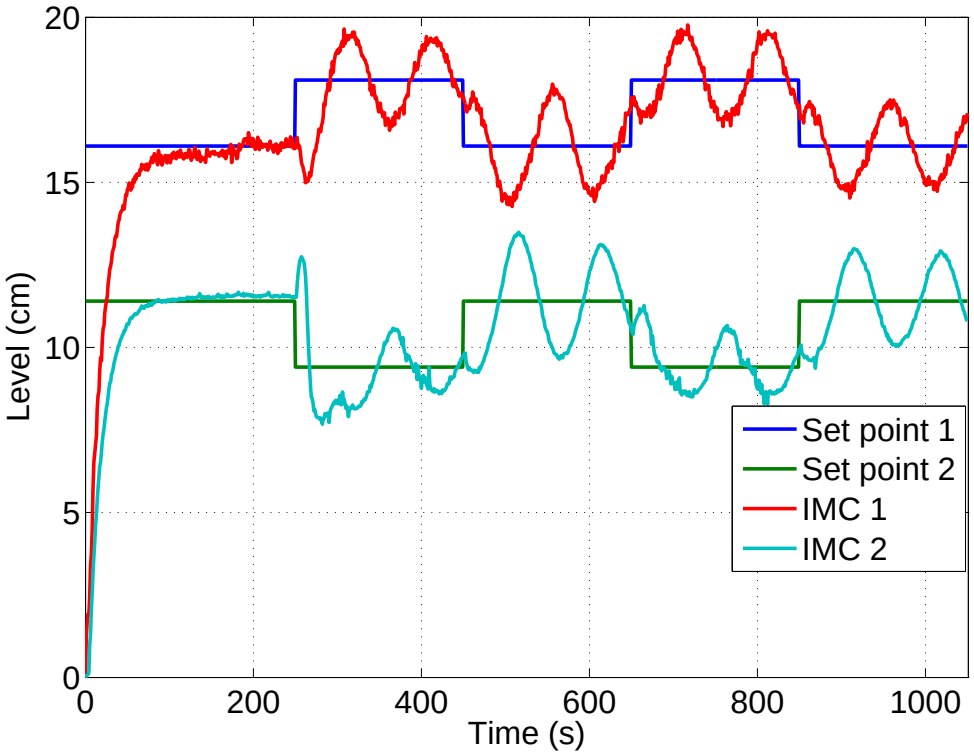


(a) Output responses

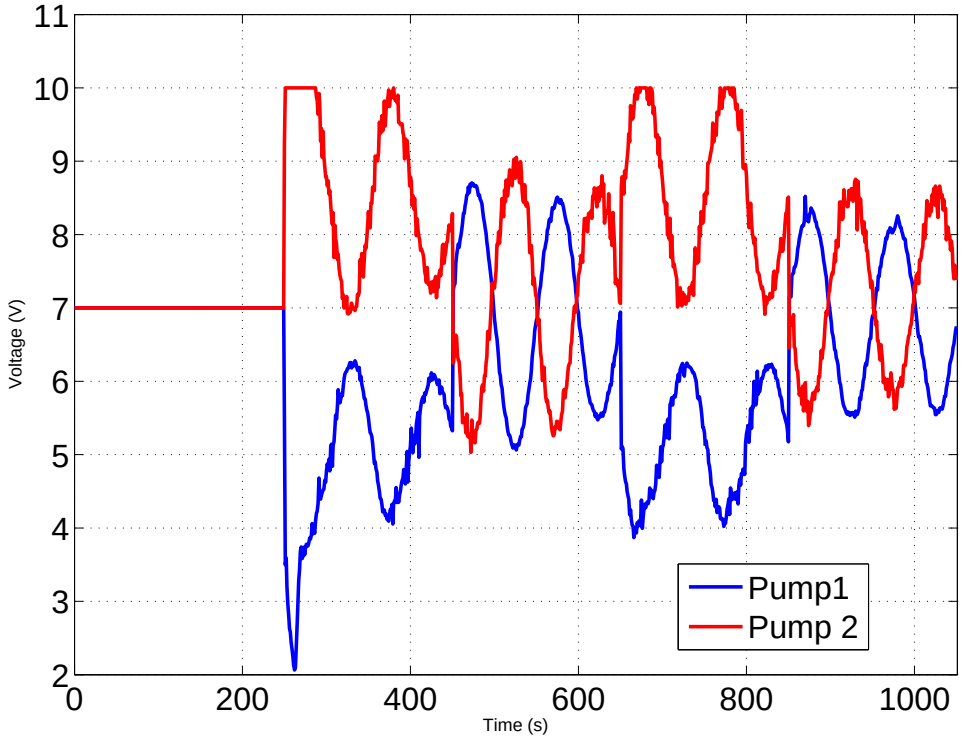


(b) Plant inputs

Figure 6.14: Profile 3: Real-time implementation of MIMC controller for non-minimum phase.



(a) Output responses



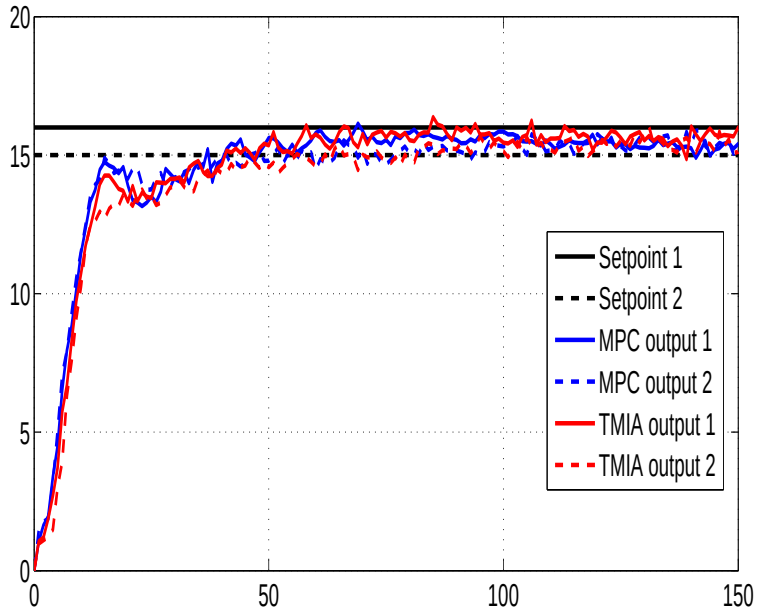
(b) Plant inputs

Figure 6.15: Profile 3: Real-time implementation of IMC controller for non-minimum phase.

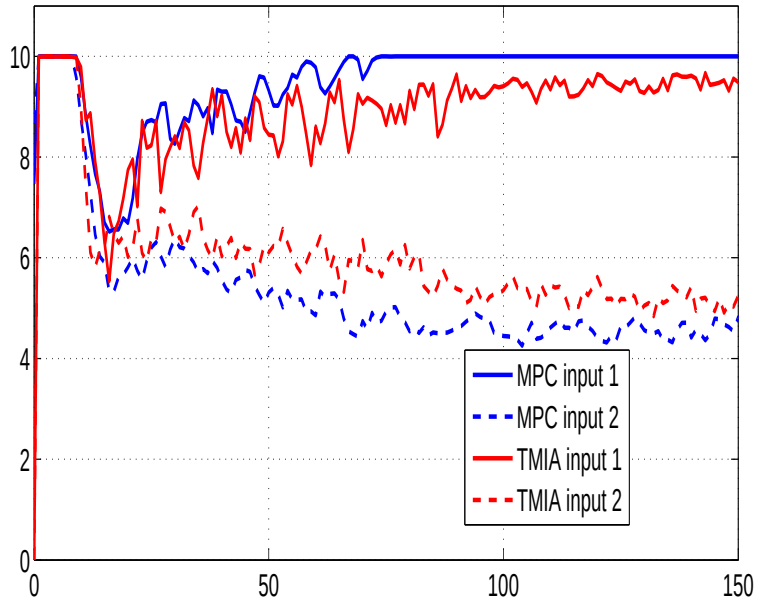
6.2 Comparison with MPC

For a particular Quadratic Dynamic Matrix Control (QDMC) algorithm [94] implementable on the PLC, with multivariable formulation [95], the size of the Hessian matrix and its corresponding matrices increases with the control horizon. This increases the requirements for memory allocation and computation time on the PLC by at least a multiple of the control horizon (in the least case where $P = M$). Hence we compare the performance of the TMIA controller with an QDMC controller on the PC-based Quanser platform. One setback however is that in the QuaRC-MATLAB platform, embedded MATLAB functions used for coding the QPs are limited to 2 x 2 matrix multiplication. Thus multiplication of large sized matrices involved in solving the quadratic program has to be done manually in the program thus increasing coding complexity. For this this reason the MPC implementation on the QUARC platform was limited to a control horizon of 2.

A set point of $[16 \;\; 15]^T$ is requested from zero and inputs are also constrained between 0V to 10V. The input and output weights are set to 1 as in the TMIA algorithm. The TMIA response competes favourably the MPC response with setpoint tracking and similar settling times as shown in Figure 6.16 with control horizon of 2 and prediction horizon of 20 for the minimum phase case and control horizon of 2 and prediction horizon of 20 for the non-minimum phase case in Figure 6.17. With the advantage of reduced computation equivalent to a single horizon MPC, the TMIA controller is the obvious choice for PLC implementation. From the results in Table 5.1, a higher specification of PLC would be required to implement a long-horizon MPC controller with a reasonable control and prediction horizon. Also coding would need to be carried out using a text-based language on a PLC because of the complexity of carrying out matrix multiplication of large sizes.

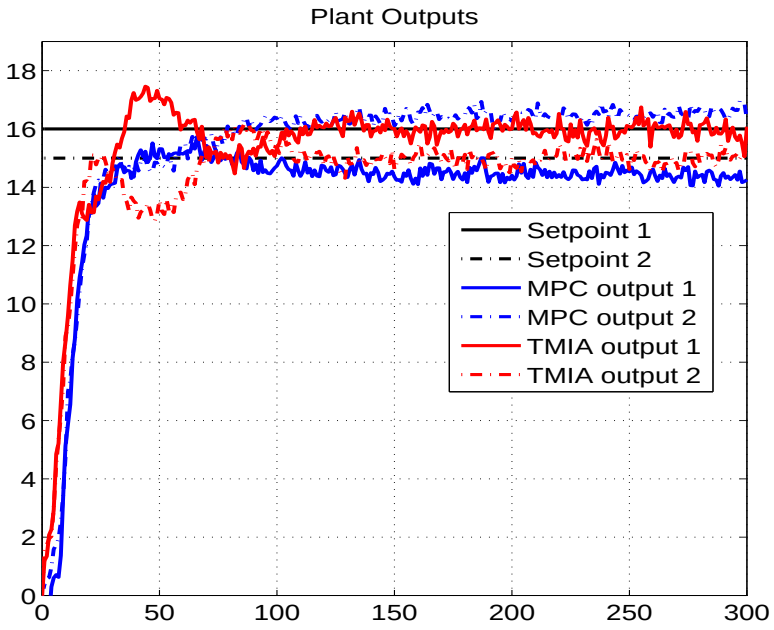


(a) Plant outputs

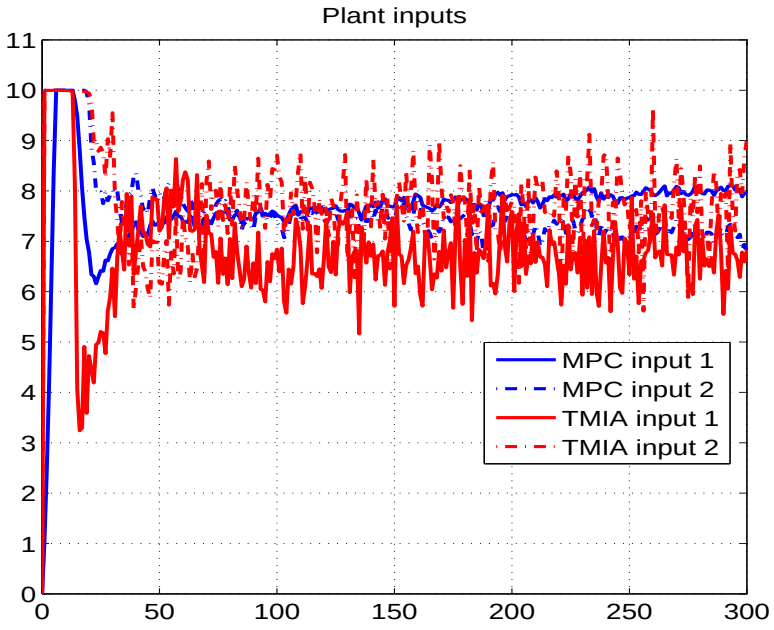


(b) Input voltages

Figure 6.16: Quadruple Tank control in minimum phase with MPC controller ($P = 20$, $M = 2$) and TMIA controller.



(a) Plant outputs



(b) Input voltages

Figure 6.17: Quadruple Tank control in non-minimum phase with MPC controller ($P = 100$, $M = 2$) and TMIA controller.

6.3 Summary

The design of the TMIA controller with the specifications for the control platforms and results have been discussed in this chapter. The TMIA controller was successfully implemented on the Quadruple Tank system via the QUARC platform and the Siemens SIMATIC S7-300 CPU 314C-2 PN/DP PLC and works effectively on both minimum phase and non-minimum phase configurations of the plant. From experimental results, the TMIA structure was found to outperform the classical IMC and Modified IMC controllers in terms of handling windup and directionality when constraints are active. Furthermore, the TMIA controller was also compared with MPC and produced similar performance with long horizon MPC for the minimum phase case.

Chapter 7

Conclusions and Future work

7.1 Conclusions

In conclusion, the objectives of the thesis were achieved. The TMIA controller presented in [24] has been developed to incorporate rate constraints in a simple and effective way. The TMIA controller was also successfully implemented on a multivariable quadruple tank system using a Siemens PLC with limited computing capacity and a QUARC/MATLAB platform. Comparisons were made with other IMC formulations and an MPC controller. Extensive results based on PLC computation were obtained and recommendations for suitable algorithms have been made.

The PLC program was implemented in ladder logic using 3 different algorithms namely interior point, active set and projected fast gradient method for the quadratic programs of the TMIA controller. The results show that a simple and efficient advanced controller like TMIA controller can be realized on a standard PLC which is an affordable industrial alternative to MPC for input-constrained multivariable processes. The projected fast gradient method was found to be the most suitable QP algorithm for the PLC implementation due to its least computing requirements. The memory utilization and execution time in the region of a few milliseconds using show that the TMIA

controller on the PLC can be applied to much faster processes. The results also show that though difficult, advanced algorithms like quadratic optimization can be coded on a PLC from scratch using ladder logic for ease of understanding and debugging by operators and technicians in industry.

7.2 Future directions

7.2.1 PLC coding using Text-based programming

Based on the results in Chapter 5, a limitation in using ladder logic is the difficulty in implementing loops, conditional branches and complex algorithms. The results from the Active Set implementation show low execution times and low number of QP iterations but very high PLC memory utilization for the program code. Hence a further research direction would be to implement the three previously used optimization algorithms on a PLC using a text-based language like Structured Control Language (for S7). The computation requirements will be compared using the same indices and more in order give a clearer view of the algorithm capabilities and also an algorithm recommendation for low computing capacity devices.

7.2.2 Rate constrained TMIA controller for fast systems

There is limited control literature dealing with rate constraints and the literature is mainly focused on input magnitude constraints. The TMIA controller in [24] has been formulated for rate constrained systems in this thesis. Results in this thesis show the application of the TMIA controller to a slow quadruple tank process. Hence a further research direction is to use the rate constrained TMIA controller for implementation on a faster system such as on a nanopositioning stage.

7.2.3 Comparison with MPC

Due to computational limitations on the QuaRC-MATLAB platform, the MPC controller implemented had a limited control horizon. Hence the designed TMIA algorithm with rate constraints on the nano-stage could be embedded on a platform such as an FPGA for comparison with a MPC controller with sufficient control horizon. This would provide a better benchmark to compare computation capabilities of the two algorithms.

Bibliography

[1] K. H. Johansson, "The quadruple-tank process: A multivariable laboratory process with an adjustable zero," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 3, pp. 456–465, 2000.

[2] Quanser, "Coupled tank experiment user manual," 2012. [Online]. Available: http://www.quansershare.com/Home/Search?contentID=139

[3] C. Edwards and I. Postlethwaite, "Anti-windup and bumpless-transfer schemes," *Automatica*, vol. 34, no. 2, pp. 199–210, 1998.

[4] A. Zheng, M. V. Kothare, and M. Morari, "Anti-windup design for internal model control," *International Journal of Control*, vol. 60, no. 5, pp. 1015–1024, 1994.

[5] P. Campo and M. Morari, "Robust control of processes subject to saturation nonlinearities," *Computers and Chemical Engineering*, vol. 14, no. 4, pp. 343–358, 1990.

[6] Y. Peng, D. Vrani, R. Hanus, and S. WELLER SR, "Anti-windup designs for multivariable controllers," *Automatica*, vol. 34, no. 12, pp. 1559–1565, 1998.

[7] M. Morari and E. Zafriou, *Robust process control*. Englewood Cliffs: Prentice-Hall, 1989.

[8] V. Kapila and K. Grigoriadis, *Actuator saturation control*. CRC Press, 2002.

[9] M. Soroush and S. Valluri, "Optimal directionality compensation in processes with input saturation non-linearities," *International Journal of Control*, vol. 72, no. 17, pp. 1555–1564, 1999.

[10] T. A. Johansen and T. I. Fossen, "Control allocation: A survey," *Automatica*, vol. 49, no. 5, pp. 1087–1103, 2013.

[11] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control engineering practice*, vol. 11, no. 7, pp. 733–764, 2003.

[12] Y. Wang and S. Boyd, "Fast model predictive control using online optimization," *Control Systems Technology, IEEE Transactions on*, vol. 18, no. 2, pp. 267–278, 2010.

[13] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit mpc," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 8, pp. 816–830, 2008.

[14] J. L. Jerez, P. J. Goulart, S. Richter, G. Constantinides, E. C. Kerrigan, M. Morari *et al.*, "Embedded online optimization for model predictive control at megahertz rates," *Automatic Control, IEEE Transactions on*, vol. 59, no. 12, pp. 3238–3251, 2014.

[15] B. Huyck, H. J. Ferreau, M. Diehl, J. De Brabanter, J. F. Van Impe, B. De Moor, and F. Logist, "Towards online model predictive control on a programmable logic controller: practical considerations," *Mathematical Problems in Engineering*, vol. 2012, 2012.

[16] B. Huyck, J. De Brabanter, B. De Moor, J. F. Van Impe, and F. Logist, "Online model predictive control of industrial processes using low level control hardware: A pilot-scale distillation column case study," *Control Engineering Practice*, vol. 28, pp. 34–48, 2014.

[17] B. Huyck, J. De Brabanter, B. De Moor, J. Van Impe, and F. Logist, "Model predictive control of a pilot-scale destination column using a programmable automation controller," in *European Control Conference*, 2013, Conference Proceedings, pp. 1053–1058.

[18] National Instruments, "A comparison of PACs to PLCs," White Paper, 2013. [Online]. Available: http://www.ni.com/white-paper/2960/en/

[19] A. G. Wills, G. Knagge, and B. Ninness, "Fast linear model predictive control via custom integrated circuit architecture," *Control Systems Technology, IEEE Transactions on*, vol. 20, no. 1, pp. 59–71, 2012.

[20] A. Wills, A. Mills, and B. Ninness, "FPGA implementation of an interior-point solution for linear model predictive control," in *Preprints of the 18th IFAC World Congress, Milano, Italy*, 2011, pp. 14 527–14 532.

[21] G. Frison, D. Minde Kufoalor, L. Imsland, and J. Jorgensen, "Efficient implementation of solvers for linear model predictive control on embedded devices," in *2014 IEEE Conference on Control Applications (CCA)*. IEEE, 2014, pp. 1954–1959.

[22] A. A. Adegbege and E. Mauro, "Programmable logic controller based embedded quadratic programming for input-constrained internal model control," in *2015 European Control Conference (ECC)*. IEEE, 2015, pp. 2623–2627.

[23] G. Valencia-Palomo and J. Rossiter, "Programmable logic controller implementation of an auto-tuned predictive control based on minimal plant information," *ISA transactions*, vol. 50, no. 1, pp. 92–100, 2011.

[24] A. A. Adegbege and W. P. Heath, "Internal model control design for input constrained multivariable processes," *AIChE Journal*, vol. 57, no. 12, pp. 3459–3472, 2011.

[25] M. N. Zeilinger, "Real-time model predictive control," Ph.D. dissertation, Diss., Eidgenössische Technische Hochschule ETH Zürich, Nr. 19524, 2011.

[26] J. M. Maciejowski, *Predictive control: with constraints*. Pearson education, 2002.

[27] J. B. Rawlings and D. Q. Mayne, *Model predictive control: Theory and design*. Nob Hill Pub., 2009.

[28] M. Morari and J. H Lee, "Model predictive control: past, present and future," *Computers & Chemical Engineering*, vol. 23, no. 4, pp. 667–682, 1999.

[29] T. A. Johansen and A. Grancharova, "Approximate explicit constrained linear model predictive control via orthogonal search tree," *Automatic Control, IEEE Transactions on*, vol. 48, no. 5, pp. 810–815, 2003.

[30] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[31] A. Bemporad, F. Borrelli, M. Morari *et al.*, "Model predictive control based on linear programming˜ the explicit solution," *IEEE Transactions on Automatic Control*, vol. 47, no. 12, pp. 1974–1985, 2002.

[32] A. Bemporad and C. Filippi, "Suboptimal explicit receding horizon control via approximate multiparametric quadratic programming," *Journal of optimization theory and applications*, vol. 117, no. 1, pp. 9–38, 2003.

[33] C. N. Jones, M. Barić, and M. Morari, "Multiparametric linear programming with applications to control," *European Journal of Control*, vol. 13, no. 2, pp. 152–170, 2007.

[34] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear model predictive control*. Springer, 2009, pp. 345–369.

[35] M. Herceg, M. Kvasnica, C. N. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *Control Conference (ECC), 2013 European*. IEEE, 2013, pp. 502–510.

[36] M. N. Zeilinger, C. N. Jones, and M. Morari, "Real-time suboptimal model predictive control using a combination of explicit mpc and online optimization," *Automatic Control, IEEE Transactions on*, vol. 56, no. 7, pp. 1524–1534, 2011.

[37] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit mpc," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 8, pp. 816–830, 2008.

[38] A. Domahidi, A. U. Zgraggen, M. N. Zeilinger, M. Morari, and C. Jones, "Efficient interior point methods for multistage problems arising in receding horizon control," in *Proceedings of the 51st IEEE Conference on Decision and Control*, no. EPFL-CONF-181938, 2012.

[39] E. D. Andersen, C. Roos, and T. Terlaky, "On implementing a primal-dual interior-point method for conic quadratic optimization," *Mathematical Programming*, vol. 95, no. 2, pp. 249–277, 2003.

[40] M. Cannon, W. Liao, and B. Kouvaritakis, "Efficient mpc optimization using pontryagin's minimum principle," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 8, pp. 831–844, 2008.

[41] P. Patrinos, P. Sopasakis, and H. Sarimveis, "A global piecewise smooth newton method for fast large-scale model predictive control," *Automatica*, vol. 47, no. 9, pp. 2016–2022, 2011.

[42] Y. Nesterov, "A method of solving a convex programming problem with convergence rate o (1/k2)," in *Soviet Mathematics Doklady*, vol. 27, no. 2, 1983, pp. 372–376.

[43] S. Richter, C. N. Jones, and M. Morari, "Real-time input-constrained mpc using fast gradient methods," in *Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009.* IEEE, 2009, pp. 7387–7393.

[44] P. Zometa, M. Kogel, T. Faulwasser, and R. Findeisen, "Implementation aspects of model predictive control for embedded systems," in *American Control Conference (ACC), 2012*. IEEE, 2012, pp. 1205–1210.

[45] A. G. Wills, G. Knagge, and B. Ninness, "Fast linear model predictive control via custom integrated circuit architecture," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 1, pp. 59–71, 2012.

[46] J. L. Jerez, K.-V. Ling, G. A. Constantinides, and E. C. Kerrigan, "Model predictive control for deeply pipelined field-programmable gate array implementation: algorithms and circuitry," *Control Theory & Applications, IET*, vol. 6, no. 8, pp. 1029–1041, 2012.

[47] A. Roldao-Lopes, A. Shahzad, G. A. Constantinides, and E. C. Kerrigan, "More flops or more precision? accuracy parameterizable linear equation solvers for model predictive control," in *Field Programmable Custom Computing Machines, 2009. FCCM'09. 17th IEEE Symposium on*. IEEE, 2009, pp. 209–216.

[48] G. Valencia-Palomo and J. Rossiter, "Efficient suboptimal parametric solutions to predictive control for plc applications," *Control Engineering Practice*, vol. 19, no. 7, pp. 732–743, 2011.

[49] M. S. Lau, S. Yue, K. Ling, and J. Maciejowski, "A comparison of interior point and active set methods for fpga implementation of model predictive control," in *Proceedings of the European control conference*, 2009, pp. 157–161.

[50] K.-V. Ling, B. F. Wu, and J. Maciejowski, "Embedded model predictive control (mpc) using a fpga," in *Proc. 17th IFAC World Congress*, 2008, pp. 15 250–15 255.

[51] J. Mattingley and S. Boyd, "Cvxgen: a code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[52] G. Frison, D. Kufualor, L. Imsland, and J. B. Jørgensen, "Efficient implementation of solvers for linear model predictive control on embedded devices," in *Proc. IEEE Multiconference on Systems and Control, Nice*, 2014.

[53] A. R. King-Hans, W. P. Heath, and R. Alli-Oke, "Two-stage Multivariable IMC Antiwindup (TMIA) control of a quadruple tank process using a PLC," in *Proc. IEEE Multiconference on Systems and Control, Nice*, 2014, pp. 1681–1686.

[54] R. Fletcher, *Practical methods of optimization*. John Wiley and Sons, 2013.

[55] N. Young, *An introduction to Hilbert space*. Cambridge university press, 1988.

[56] J. Nocedal and S. Wright, "Numerical optimization, series in operations research and financial engineering," *Springer, New York*, 2006.

[57] P. E. Gill, W. Murray, and M. H. Wright, "Practical optimization," 1981.

[58] M. C. Ferris, O. L. Mangasarian, and S. J. Wright, *Linear programming with MATLAB*. SIAM, 2007.

[59] A. Wills and W. Heath, "Interior-point methods for linear model predictive control," *Report EE03016, University of Newcastle, NSW*, vol. 2308, 2003.

[60] S. J. Wright, *Primal-dual interior-point methods*. Siam, 1997, vol. 54.

[61] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM Journal on optimization*, vol. 2, no. 4, pp. 575–601, 1992.

[62] J. C. Doyle, R. S. Smith, and D. F. Enns, "Control of plants with input saturation nonlinearities," in *American Control Conference, 1987*. IEEE, Conference Proceedings, pp. 1034–1039.

[63] S. Galeani, S. Tarbouriech, M. Turner, and L. Zaccarian, "A tutorial on modern anti-windup design," *European Journal of Control*, vol. 15, no. 3, pp. 418–440, 2009.

[64] T. A. Kendi and F. J. Doyle III, "An anti-windup scheme for multivariable nonlinear systems," *Journal of Process Control*, vol. 7, no. 5, pp. 329–343, 1997.

[65] W. P. Heath and A. G. Wills, "Design of cross-directional controllers with optimal steady state performance, journal = European Journal of Control, volume = 10, number = 1, pages = 15-27, issn = 0947-3580, year = 2004, type = Journal Article."

[66] R. Hanus, M. Kinnaert, and J.-L. Henrotte, "Conditioning technique, a general anti-windup and bumpless transfer method," *Automatica*, vol. 23, no. 6, pp. 729–739, 1987.

[67] C. E. Garcia and M. Morari, "Internal model control. 2. design procedure for multivariable systems," *Industrial and Engineering Chemistry Process Design and Development*, vol. 24, no. 2, pp. 472–484, 1985.

[68] ——, "Internal model control. a unifying review and some new results," *Industrial and Engineering Chemistry Process Design and Development*, vol. 21, no. 2, pp. 308–323, 1982.

[69] ——, "Internal model control. 3. multivariable control law computation and tuning guidelines," *Industrial and Engineering Chemistry Process Design and Development*, vol. 24, no. 2, pp. 484–494, 1985.

[70] G. Goodwin, S. Graebe, and W. Levine, "Internal model control of linear systems with saturating actuators," in *European Control Conference*, 1993, pp. 1072–1077.

[71] S. Gayadeen and S. R. Duncan, "Discrete-time anti-windup compensation for synchrotron electron beam controllers with rate constrained actuators," *Automatica*, vol. 67, pp. 224–232, 2016.

[72] S. Gayadeen and S. Duncan, "Anti-windup compensation for electron beam stabilisation control systems on synchrotrons with rate constrained actuators," in *Control Conference (ECC), 2013 European*. IEEE, 2013, pp. 2752–2757.

[73] T. A. Johansen, T. I. Fossen, and S. P. Berge, "Constrained nonlinear control allocation with singularity avoidance using sequential quadratic programming,"

IEEE Transactions on Control Systems Technology, vol. 12, no. 1, pp. 211–216, 2004.

[74] A. Saberi, A. A. Stoorvogel, and P. Sannuti, *Internal and external stabilization of linear systems with constraints*. Springer Science & Business Media, 2012.

[75] K. J. Astrom and B. Wittenmark, *Computer-Controlled systems: theory and design*. Prentice hall, 1999.

[76] B. W. Bequette, *Process control: modeling, design, and simulation*. Prentice Hall Professional, 2003.

[77] G. H. Golub and C. F. Van Loan, "Matrix computations (johns hopkins studies in mathematical sciences). 3rd," 1996.

[78] K. Houston, *How to think like a mathematician: a companion to undergraduate mathematics*. Cambridge University Press, 2009.

[79] P. Kapasouris, M. Athans, and G. Stein, "Design of feedback control systems for stable plants with saturating actuators," in *Decision and Control, 1988., Proceedings of the 27th IEEE Conference on*. IEEE, Conference Proceedings, pp. 469–479.

[80] L. Wang, *Model Predictive Control System Design and Implementation Using MATLAB*. Springer, 2009.

[81] A. G. Wills and W. P. Heath, "Barrier function based model predictive control," *Automatica*, vol. 40, no. 8, pp. 1415–1422, 2004.

[82] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[83] L. K. Pachemanov, "Internal model control for a quadruple-tank system," Master's thesis, The University of Manchester, 2012.

[84] C.-C. Chu, "On discrete inner-outer and spectral factorizations," in *American Control Conference*. IEEE, 1988, pp. 1699–1700.

[85] V. Ionescu and C. Oara, "Spectral and inner-outer factorizations for discrete-time systems," *IEEE transactions on Automatic Control*, vol. 41, no. 12, pp. 1840–1845, 1996.

[86] C. Oara and A. Varga, "The general inner-outer factorization problem for discrete-time systems," in *Proc. ECC99, Karlsruhe, Germany*, 1999.

[87] E. P. Gatzke, E. S. Meadows, C. Wang, and F. J. Doyle Iii, "Model based control of a four-tank system," *Computers & Chemical Engineering*, vol. 24, no. 2, pp. 1503–1509, 2000.

[88] A. Adegbege, "Constrained internal model control," Ph.D. dissertation, University of Manchester, 2011.

[89] SIMATIC, "Programming with step7." [Online]. Available: https://www.automation.siemens.com/doconweb/pdf/SINUMERIK_SINAMICS_02_2012_E/S7P.pdf?p=1

[90] R. O. Alli-Oke, "Robustness and optimization in anti-windup control," Ph.D. dissertation, University of Manchester, 2014.

[91] J. L. Jerez, P. J. Goulart, S. Richter, G. A. Constantinides, E. C. Kerrigan, and M. Morari, "Embedded predictive control on an FPGA using the fast gradient method," in *European Control Conference (ECC), 2013*. IEEE, 2013, pp. 3614–3620.

[92] S. Richter, S. Mariéthoz, and M. Morari, "High-speed online mpc based on a fast gradient method applied to power converter control," in *American Control Conference (ACC), 2010*. IEEE, 2010, pp. 4737–4743.

[93] J. B. Hoagg and D. S. Bernstein, "Nonminimum-phase zeros," *IEEE Control Systems Magazine*, vol. 27, no. 3, pp. 45–57, 2007.

[94] B. W. Bequette, *Process control: modeling, design, and simulation*. Prentice Hall Professional, 2003.

[95] U.-C. Moon and K. Y. Lee, "Step-response model development for dynamic matrix control of a drum-type boiler–turbine system," *IEEE Transactions on Energy Conversion*, vol. 24, no. 2, pp. 423–430, 2009.