Design of a state-based nonlinear controller

by

Johan van der Merwe

Dissertation

presented in partial fulfillment of the requirements for the degree

Magister Ingeneriae

in

Electric, Electronic and Computer Engineering

in the

Faculty of Engineering

of the

North-West University (Potchefstroom Campus)

Supervisor: PROF. C.P. BODENSTEIN

JUNE 2006

Acknowledgement

To GOD, my father, mother, sponsors and supervisor.

To GOD,

I pledge my best, my everything, to perform and complete every task HE sets me to its full potential.

I thank HIM for HIS guidance and support and intervention, without which this study could not have been.

To my father,

If I can only be half as wise and loving.

I thank him for his advice, humour and encouragement.

To my mother,

Never could a son have asked for better.

I thank her for being the glue that kept it all together.

To my sponsors,

Thank you for financial aid and being approachable whenever I needed technical support.

To my supervisor,

Thank you for coaxing me to achieve my best and keeping me in line.

"True brilliance can only be achieved by DIVINE intervention, otherwise it's just mediocre."

-J.J. van der Merwe

Abstract

A developer of thermofluid simulation software requires algorithms which are used to design and implement PI controllers at some operating points of nonlinear industrial processes.

In general, the algorithm should be applicable to multivariable plant models which may be nonlinear. In some areas there is a hesitancy to use controllers for nonlinear processes which use neural networks or fuzzy logic or a combination thereof. PI controllers are also standard in various SCADA systems.

Since control normally takes place around an operating point, a linearised model is obtained. A controller designed for a particular operating point, may not be suitable for other operating points. Since a multitude of variables are to be controlled in the plant, the problem becomes more acute. In this research, a methodology is derived for the design of multivariable control using PI controllers. The parameters of the controllers depend on the operating point, and are therefore nonlinear. The behaviour is deterministic in a classical control sense around a range of operating points. This should remove concerns of non-deterministic behaviour as attached to neural networks due to the lack of stability tests for them which are industry accepted.

A state-space approach leads to the development of a design methodology, which is then used to implement these algorithms. The P- and PI-controllers will be designed using traditional methods, as well as by an optimal procedure which makes use of a genetic algorithm.

The GA tuning algorithm yields superior performance when compared to other methods.

Opsomming

`n Ontwikkelaar van termodinamiese vloeistof simulasie sagteware benodig algoritmes om PI-tipe beheerders te implementeer by verskeie werkspunte vir sekere nie-linêre industriële prosesse.

Die algoritme behoort in die algemeen toepaslik op multiveranderlike aanleg-modelle te wees wat ook nie-lineêr kan wees. In sekere gebiede is daar huiwering om van nie-lineêre beheerders wat van neurale netwerke, wasige logiese stelsels of 'n kombinasie daarvan gebruik te maak. PI-beheerders word algemeen in verskeie SCADA stelsels geïmplementeer.

Weens die feit dat beheer gewoonlik om `n werkspunt geskied, kan `n gelineariseerde model van die aanleg verkry word. `n Beheerder wat ontwerp is vir `n gegewe werkspunt, is nie noodwendig geskik vir ander werkspunte nie. Aangesien meervoudige veranderlikes in die aanleg beheer moet word, kompliseer dit die problem. In hierdie studie word `n metodologie afgelei vir die ontwerp van multiveranderlike beheerder deur gebruik te maak van PI-beheerders. Die parameters van die beheerders is werkspunt afhanklik, en gevolglik is die beheerders nie-lineêr. Die gedrag is steeds deterministies by `n bestek van werkspunte vanuit `n klassieke oogpunt, wat enige twyfel uitskakel wat gewoonlik gepaard gaan met nie-lineêre beheerders wat gebruik maak van neurale netwerke. Hierdie twyfel is weens die gebrek aan industrie-aanvaarde stabiliteitstoetse.

`n Toestandsveranderlike-gebasseerde metode lei tot die onwikkeling van 'n ontwerpsmetodologie wat gebruik word om die benodigde algoritmes te implementeer. P- en PI-tipe beheerders word ontwerp met tradisionele metodes asook met die gebruik van `n optimeringsprosedure wat gebruik maak van 'n genetiese algoritme.

Die genetiese algoritme wat die beheerder parameters instel, lewer beter resultate as die ander metodes.

Table of contents

Page

List of figures

Page

List of tables

Page

List of abbreviations

CLCP	-	Closed-Loop Characteristic Polynomial
cm	-	centimetres
GA	-	Genetic Algorithm
ISE	-	Integral of the Squared Error
ITSE	-	Integral of Time multiplied by the Square of the Error
LQ	-	Linear Quadratic
LTI	-	Linear Time-Invariant
mA	-	milliAmpere
MIMO	-	Multiple Input Multiple Output
NCCD	-	Non-Compulsive Control Design
NL	-	Nonlinearity
P	-	Proportional
PI	-	Proportional plus Integral
PID	-	Proportional plus Integral plus Derivative
RSSR	-	Remainder Stochastic Sampling with Replacement
RSSWR	-	Remainder Stochastic Sampling Without Replacement
rws	-	Roulette Wheel Selection
SCADA	-	Supervisory Control and Data Acquisition
SISO	-	Single Input Single Output
SSE	-	Steady-State Error
SSPR	-	Stochastic Sampling with Partial Replacement
SSR	-	Stochastic Sampling with Replacement
sus	-	Stochastic Universal Sampling
VSCS	-	Variable Structure Control Systems

List of symbols

The table of symbols below convey their use throughout the study, unless otherwise stated. Experiments for example, may have used some of the symbols differently to their description below, but than then their meaning for that experiment, is supplied there.

All symbols in bold relate to a matrix or vector. An accent next to a bolded symbol refers to its transposed, whereas it refers to the time derivative when used with a scalar function.

A bolded symbol next to a star refers to that symbol being in the modal domain, whereas when the symbol is associated with optimal control theory, the star would imply an optimal vector/matrix.

A 'T' next to a matrix/vector refers to its transposed.

Upper case letters refer to variables in the s-domain (were relevant), and lower case letters refer to variables in the time domain (where relevant), except for the control vector **U(t)**.

Where applicable, SI-units are used.

Name	Description
$x_1,...,x_n$	A set of quantities like state values
t	Time (s)
t_0	Start time
t_f	Final time
$\mathbf{x}$	State vector
$\dot{\mathbf{x}}$	Derivative of state vector
$\mathbf{B}$	Control-Actuator dynamics
$\mathbf{b}$	Control-Actuator dynamics
$\mathbf{C}$	System output-measurement dynamics
$\mathbf{A}$	Dynamic behaviour of system
$\mathbf{Y}$	System output vector
$\mathbf{R}$	Reference vector
$\mathbf{U}$	Control Vector
$\mathbf{G}^{c}_{ab}$	Element of the Controller matrix a - ...to output b - …from input
$\mathbf{G}^{P}_{ab}$	Element of the plant matrix
$\mathbf{G}^{c}$	Controller matrix

Name	Description
$\mathbf{G}^P$	Plant matrix
G_P	Plant transfer function
$\mathbf{G}$	$\mathbf{G}^c\mathbf{G}^P$
$\mathbf{E}$	Error vector
$\mathbf{M}$	Manipulated variable matrix (Controller output)
$\mathbf{K}$	Controller matrix
$\mathbf{k}$	Controller matrix
$\mathbf{V}$	Measurable noise vector
$\mathbf{W}$	Immeasurable noise vector
$\mathbf{N}^V$	Measurable noise-plant dynamics
$\mathbf{N}^W$	Immeasurable noise-plant dynamics
$\mathbf{A}_d$	Desired system dynamics
$\mathbf{G}_f$	Feedforward controller matrix
$\mathbf{H}$	Feedback dynamics
$\mathbf{I}$	Identity matrix
$\mathbf{T}^{-1}$	Mode analyzer
$\mathbf{T}$	Mode synthesizer
$\mathbf{f}$	Function vector
$\mathbf{g}$	Function vector
$\mathbf{h}$	Function vector
$\mathbf{r}$	Reference vector
$\mathbf{a}$	Dynamic behaviour of system
$\mathbf{u}$	Control vector
$\mathbf{e}$	Error vector
$\mathbf{y}$	System output vector
$\mathbf{x}_0$	Initial state vector
n	System order
q	Number of modes
m	Number of measurements
P	Forward path gain
p_i	I^{th} pole position with $i = 1,2,..., n$
K_P	Proportional gain constant
K_I	Integrator gain constant
K_D	Differentiator gain constant
k_i	I^{th} state controller
V	Lyapunov function
$\mathcal{D}$	Defined region
$\mathbf{x}_{ref}$	Reference vector

Name	Description
$\mathbf{x}_{actual}$	State output vector
R	Resistor value (Ω)
C	Capacitor value (F)
V_c	Voltage across the capacitor (V)
V_R	Voltage across the resistor (V)
V(t)	Time varying voltage (V)
i(t)	Time varying current (A)
J	Performance index
H	Real symmetric positive semi-definite $n \times n$ matrix
R	Real symmetric positive definite $n \times n$ matrix
Q	Real symmetric positive semi-definite $n \times n$ matrix
$F(x_i)$	Fitness value where x_i is the position in the ordered population of the i^{th} individual – with regards to a Genetic Algorithm
N	Number of individuals in your population – with regards to a Genetic Algorithm
N_{VAR}	Number of variables per chromosome – with regards to a Genetic Algorithm
GGAP	Generation Gap – with regards to a Genetic Algorithm
P	Parent – with regards to a Genetic Algorithm
O	Offspring – with regards to a Genetic Algorithm
M	Mutated chromosome – with regards to a Genetic Algorithm
α	Scaling factor

Table 1: Symbol declaration

1 Introduction

Firstly, a foundation will be laid for the research. The problem statement as well as proposed solution will then be supplied. Hereafter, the specific experiments that were performed to illustrate and prove the concepts developed during the research will be given, followed by the methodology behind the route taken. The remainder of the thesis will then be outlined.

1.1 Background

The world of control theory is divided into two types, *modern control* and *classic control.* The main difference is that modern control often designs solutions in the time domain, and the model is used exclusively used in some way to design a controller for that specific instance. The classic approach designs solutions in the s-domain. Exceptions do occur.

The type of system that is controlled is either linear, or nonlinear. By implication, modern control is used for control of nonlinear systems, or systems that are harder to control, due to its order or nature.

Classic control, again, tends toward linear systems, unless the nonlinear system is linearised, provided it can be, which then makes it amenable for classic control techniques.

In either case, the complexity of the controller for the system, itself being either linear or nonlinear, varies from the basic single-variable, single-loop feedback control system to more complex multivariable control systems.

If the multivariable case is considered, when using the classic approach, scalar variables and transfer functions are upgraded to vector variables and matrix transfer functions when designing a multivariable control system. This leads to mathematical complications because of the requirements for the matrix algebra.

In short, each plant output is affected by more than one control, which was needed to satisfy the matrix algebra requirements. This implies that conventional design methods presented in chapter 8 and chapter 9 of [1], are not directly applicable. For this theory to apply directly, a solution is to decouple the system, i.e. to design cross controllers with the idea that they cause each input to affect one, and only one output. This, however, is only possible if the cross coupling effect is weak relative to the desirable control performance, if not, the system has to be treated as an entity.

This will be discussed in more detail and better illustrated in chapter 2.

Thus, considering the tools available at the time, the complexity of the system that can be handled is rather limited. Keep in mind that this was in the approach in the pre-computer era. A more powerful approach is the state-space approach, which can be applied in both modern and classic control, on linear- and nonlinear systems. This is due to its mathematical formulation of the problem [1: chapter10].

This approach is very attractive to use, given today's technology and tools at hand.

1.2 Problem statement

My sponsor developed thermofluid simulation software that makes use of PI-type control. They require an algorithm be developed for the design and implementation of PI controllers which are operating point dependant. The focus being processes which are ***multi- input- multi-output*** (MIMO), primarily nonlinear, systems.

The purpose of this study was to derive a methodology for the design of PI-type controllers for MIMO, nonlinear systems, and to simulate the controlled system's response in SIMULINK®.

1.3 Proposed solution

A state-space approach is used to design a multivariable PI controller which is operating point dependant. As a consequence of this investigation, design rules must be formulated, enabling the control engineer to chart the best path for the design of the controller which is required for a specific situation.

Thus, the different areas of control will be investigated, see figure 2. The result of this investigation will be briefly discussed in chapter 2.

A nonlinear system, represented by state variable equations, will be linearised so that conventional, existing control techniques can be applied to it. Then, by using a linear control law in the architecture shown in figure 1, the control law's performance at controlling the MIMO, *non linear* system will be illustrated. In fact, a linear-designed, linear controller will be

used to control a non linear model of the system. This will be illustrated by simulating the system in SIMULINK®.

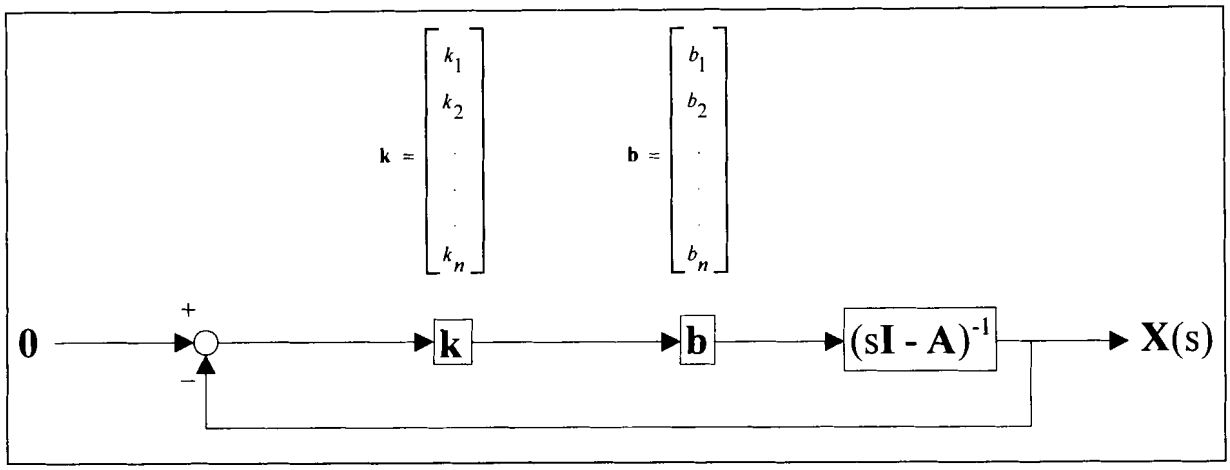

Figure 1: Feedback control using a series design principle [1]

The figure will be discussed in detail in chapter 2.

Strategy 1: The multivariable controller **k**, will be in the form of a pure proportional controller for a MIMO system. Different variations on this topic will be presented.

Strategy 2: The controller, **k**, will be made a traditional PI controller. Different variations will also be illustrated.

In these two scenarios the values for the controller parameters will be exactly calculated using the *linearised* version of the system, and then implemented and simulated on the model of the *nonlinear* system.

The third strategy is set apart from the rest, in that the nonlinear model will be used to obtain the locating of the controller constant values, and not the linearised version.

Strategy 3: The MIMO nonlinear system will be controlled using an optimised PI-controller. The design will be done using the nonlinear system itself, by means of a genetic algorithm to locate and optimise the P and I constants.

The strategies refer to the manner in which the controller parameters are chosen and implemented across the operating range.

The reason:

- It allows the comparison of a proportional controller (P-controller), to that of a PI-controller for the cases where the controller is linear (set point independent) and nonlinear (set point dependent)
- Examine the effect of optimising the PI-controller with a genetic algorithm

1.4 Specific problems

-Experiment 1: MIMO water level control for two interconnected tanks

-Experiment 2: MIMO System Identification

-Experiment 3: Pressure and level control of a nonlinear plant

1.5 Methodology

Control Theory is a constant changing field, adapting to new tools and technologies as it evolves.

Before computers, the ability to perform iterative and numerical operations was very limited, and the success of the controller was based on being able to exactly/analytically determine the solution. Since the advent of computers, it has become less necessary to get as close to the answer as possible analytically, since computers are able to perform iterative operations fast enough to still find a good enough solution within reasonable time with the initial solution domain not being very small. However, there are instances, especially when it comes to nonlinear systems, when the calculations require all the power computers have to offer and still take too long to be viable for solution-finding, needing either super computing, or a method exact enough to lessen the work to be done via iteration on the computer.

This is especially true in the case of some nonlinear systems, where using a method like a genetic algorithm to find the optimal solution takes too long, reinforcing the necessity to use an exact approach.

Computers have thus, for the most part, enabled numerical calculations including iterative methods to come into their own, providing the ability to solve much more complex problems, quicker.

Thus, what has become apparent is that the focus of design now rests heavily on computational resources.

It is important, however, to get basics right. If this is done, the correct foundation has been laid, and more can be done using the available computational resources, rather than just enough, given limited design time. One could say that better progress can be made from an improved starting point. One could argue, that a global minimum, is a global minimum, but how fast you get there, matters.

For this reason, the study has been structured the way it has: first determining the appropriate and applicable control techniques and approaches, then moving through the different control strategies for the test scenarios; Focussing on getting the basics right, and then expanding the idea until the *necessary* solution has been found, whilst still keeping in mind current and possible future industry norms.

A very important consequence of the study is the set of design rules developed in chapter 3, which will enable the follower thereof to design a controller which is *best*, given the system parameters, system constraints and performance criteria. This will eliminate non-compulsive control design (NCCD); the more complex the controller, the more complex the design which will consequent in greater cost to client. The design should only be as complex as it has to be.

The experiments have been chosen to validate the developed design methodology.

Keeping in tune with getting the basics right, one feels that the ideal way to control a nonlinear system is to design a nonlinear controller. A move in this direction has been made in this study by including the design of a multivariable PI controller which is set point dependant, and thus, nonlinear, as the set points vary with time.

1.6 Notes

The use of symbols is discussed in the preceding section entitled "List of symbols".

An exact solution, refers to a method followed that directly computes the answer for a given set of operating parameters, the answer was not achieved via iterative method, as is the case with GA's.

Figure titles with references imply that the figure was derived primarily from that source.

Equations not numbered are those which have already been defined elsewhere in the study. For this reason, especially equations used in the experiments and examples are not numbered.

1.7 Research overview

The research comprise of the following chapters:

Chapter 2: The results of the investigation into the different areas of control plus some additional information, which will then form the basis of this study.

Chapter 3: The details of the method used, as well as how it was applied for this research. The conclusion is the development of the design methodology.

Chapter 4: The methodology generated in the previous chapter is implemented on three experiments.

Chapter 5: The final conclusion and recommendations are stated also including the accomplishments.

Chapter 6: The references.

Chapter 7: The appendix.

2 Control: Techniques, approaches and related research

Various areas of control, applicable to this study, are highlighted as well as which techniques, approaches and design principles are used.

2.1 Modelling engineering systems

Modelling and control go hand-in-hand. Reason being, the solution to an engineering problem starts with a thorough understanding and description thereof [2].

The modelling process used to obtain a state-space model improves understanding. The model itself serves as an integral part of the description. Once a model (a mathematical equation for the process) has been constructed, it can potentially be controlled. According to Christian Schmid, control engineering can be described as follows:

"Control engineering deals with the task of affecting a temporally changing process in such a way that the process behaves in a given way. Such tasks are not only found in technology, but also in daily life in very large number. For example the ambient temperature in a room must be held between given limits, despite temporal changes due to sun exposure and other influences. The grip arm of a robot must move along the edge of a workpiece or be led as fast as possible from one point to another in order to grip a workpiece. The same applies to the grip arm of a crane, which is to carry bricks to a certain place on the building site.

In all of these cases, a manipulated variable must be selected in such a way that the given goal is achieved." [3]

2.2 Control approaches

As stated earlier in 1.1 Background, there exist two primary approaches to control, namely modern control and classic control. The following discussion will have bearing on figure 2 on the next page.

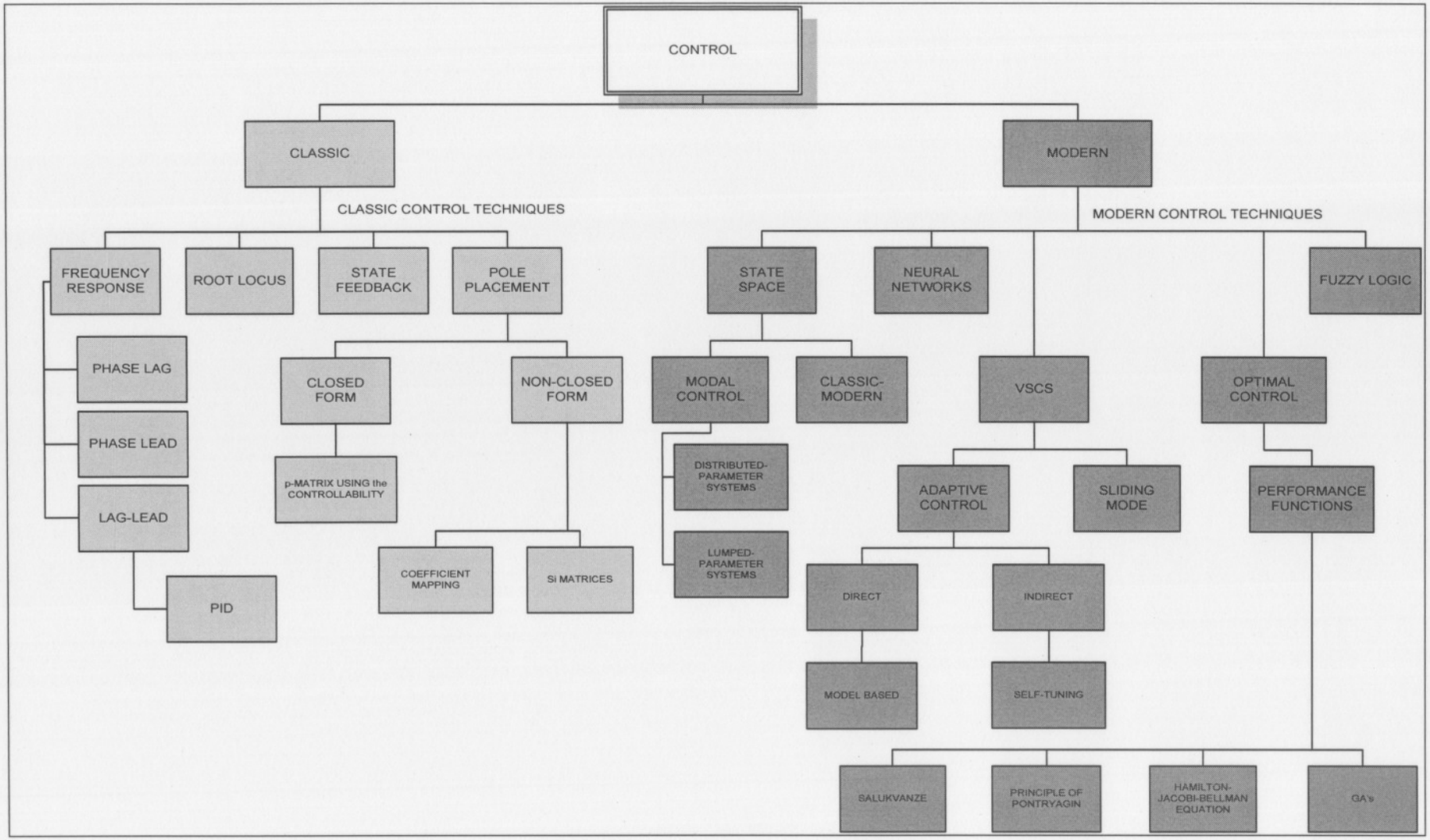

Figure 2: Control schematic

Before starting the discussion, it is important to understand what is meant by the term 'multivariable control system', given the fact that it is the focus of this study.

A single- input- single output (siso) system, is one that has one reference value, i.e. one input, and one controlled value, i.e. one output, with respect to the controller responsible for manipulating the input to produce the controlled output. Otherwise stated, the controller has one input, the error between the single reference value and the controlled output value of the object to be controlled, referred to as the control object or plant. The controller then has one output, the manipulated variable which is applied to the control object. This then produces a controlled output from the plant. See figure 3.

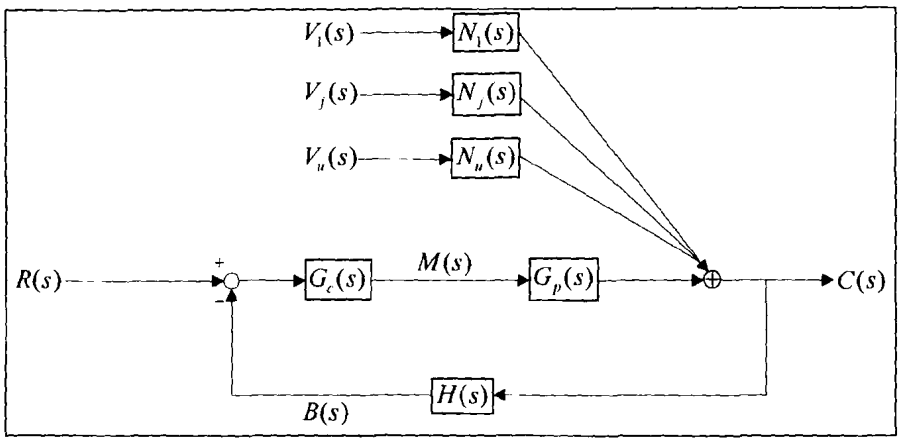

Figure 3: SISO feedback control system [1]

As can be seen, it is a single-variable, single-loop control system.

A multi- input- multi- output (mimo) system, is one which has more than one reference value, i.e. a vector input, and more than one controlled value, i.e. a vector output. The controller will be a control matrix. Figure 4 illustrating the concept for a two-variable system.

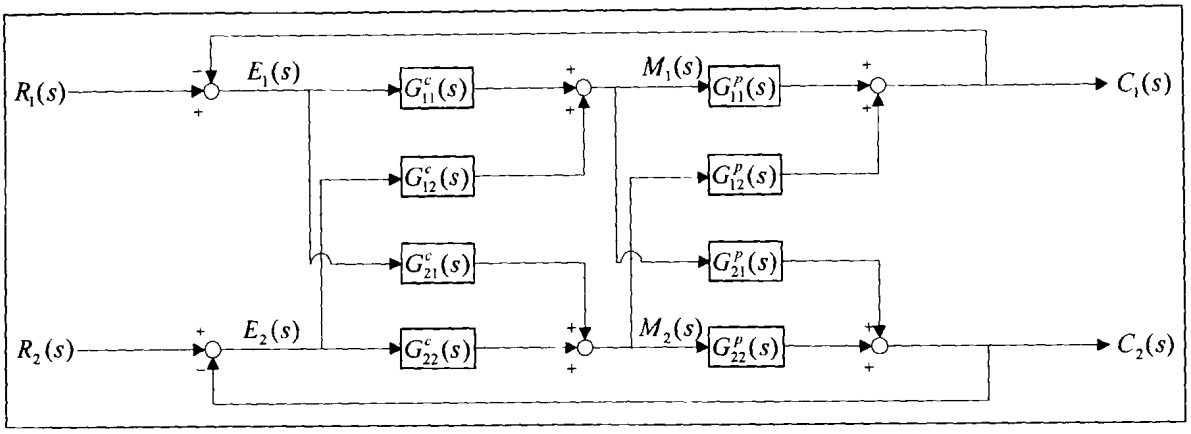

Figure 4: MIMO feedback control system [1]

Here it can be seen, that one has a multivariable, multi-loop (at least one loop for every input-output pair) control system.

Hybrid versions like single- input- multi- output and multi- input- single- output systems do exist, but are not of interest for this study.

A very powerful way to represent a system, is by its state variables. A formal definition for the *state of a system* is given in Definition 2.1.

Definition 2.1

The *state of a system* is a set of quantities $x_1(t), x_2(t), ..., x_n(t)$ which if known at $t = t_0$ are determined for $t \leq t_0$ by specifying the inputs to the system for $t \leq t_0$. [4: p16]

Systems are classified by being linear or nonlinear and time-invariant or time-varying.
Variations on this theme are represented below in terms of state variables.

A nonlinear, time-varying system,

$$\dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t), \mathbf{u}(t), t)$$

A nonlinear, time-invariant system,

$$\dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t), \mathbf{u}(t))$$

A linear, time-varying system,

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t),$$

A linear, time-invariant system,

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t),$$

In using a state-space representation of a MIMO system, it is important to note the dimensionality of the different matrices used, and to ensure that they are consequent (meaning resulting?) and adhere to the laws of matrix algebra.

2.2.1 Classic control

The thread found throughout classic control, is the use of transfer functions in the s-plane, making it amenable to be used on linear-type systems, or nonlinear systems that have been linearised around operating points. The concept of linearising systems will be discussed in section 2.7 to follow.

The most common control design techniques found in this approach are state feedback-, root locus- and the frequency response methods, as well as pole placement techniques.

As part of the frequency response methods, one finds phase-lag, phase-lead and lag-lead compensation [5].

The standard controller used throughout the industry today in SCADA systems is the three-term or PID controller, which is a special lag-lead compensator. A PI controller, which is the one used throughout this study, is the same as the PID controller, but with its derivative constant set to zero.

Aiding in the design of a multivariable controller using PI, the necessary information pertaining to state feedback control, pole placement and PI controllers will be discussed below.

- **State feedback control**

The basic RC network shown in figure 5 will be used to explain the concept of the 'states' of a system for state feedback control. A formal definition of the *state of a system* can be found in Definition 2.1 in section 2.2.

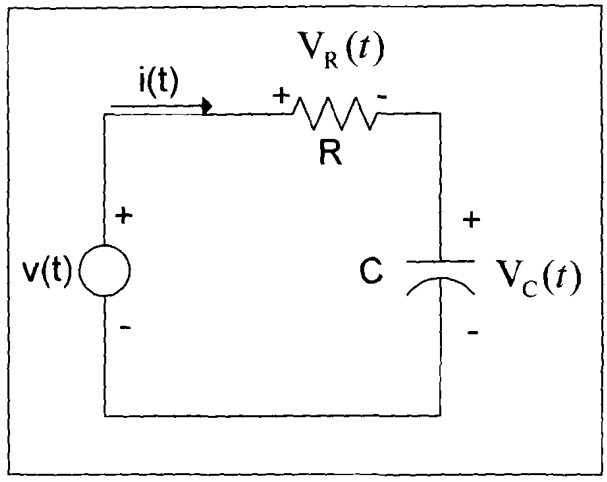

Figure 5: RC-circuit

In figure 5, V(t) is a voltage source, i(t) the current that flows through the network, $V_R(t)$ and $V_C(t)$ the voltages across the resistor and capacitor respectively with R being the resistor, and C the capacitor.

Given that the components are linear, according to Ohm's law and Kirchoff's voltage law [3], the governing network equation for the RC-network of figure 5 is given by,

$$V(t) = i(t)R + V_C(t) \qquad (1)$$

With, i, in equation (1) being the time dependant current flowing through the circuit.

$$i = C\frac{dV(t)}{dt} \qquad (2)$$

If equation (2) is substituted into Equation (1), it reduces to

$$\frac{dVc}{dt} = -\frac{1}{R.C}.Vc + \frac{1}{R.C}V(t) \qquad (3)$$

This can then be presented in the common form of a *state variable equation* for a *linear* system

$$\frac{d}{dt}\mathbf{x} = \mathbf{A}\mathbf{x} + \mathbf{B}u \qquad (4)$$

whereby, in this case, $\mathbf{A} = -\frac{1}{RC}$, $\mathbf{B} = \frac{1}{RC}$, and the only state, **x**, is the voltage across the capacitor, $V_C(t)$.

The number of states in a system is the same as the number of elements with the ability to store the specific energy of interest. This is why there is only one state in the network of figure 5, the capacitor being the only element able to store, in this case, electric energy. The number of states also corresponds to the order of the open-loop system. Thus, a network with two elements that can store the energy of interest is a second order system. For later reference, the number of states of the open-loop system, is also the number of modes, q, of the open-loop system.

> *Note*
> This does not imply that the closed loop controlled system is necessarily an n^{th} order system, n being the number of states. This is because the controller may or may not affect the order of the closed loop system, depending on the controller that is used.

In this example, x was just a single variable. As soon as there are more than one state, **x** from equation (4), becomes a vector, as illustrated.

As an example, a SISO second order system is used.
Let the plant's open-loop transfer function be represented as the product of the states' transfer functions. That is, the plant transfer function

$$G_p(s) = \frac{P}{s(s+1)(s+2)} \qquad (5)$$

with P referring to forward path gain; see the figure. $X_1(s)$, $X_2(s)$ and $X_3(s)$ represent the states $x_1(t), x_2(t)$ and $x_3(t)$ in the s-domain.

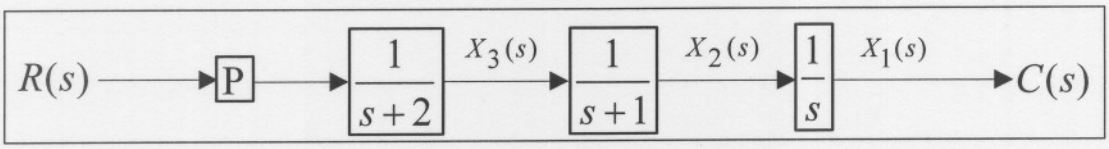

Figure 6: Open-loop system

Assuming each state is measurable, each state can be fed back through a controller as illustrated in the figure below.

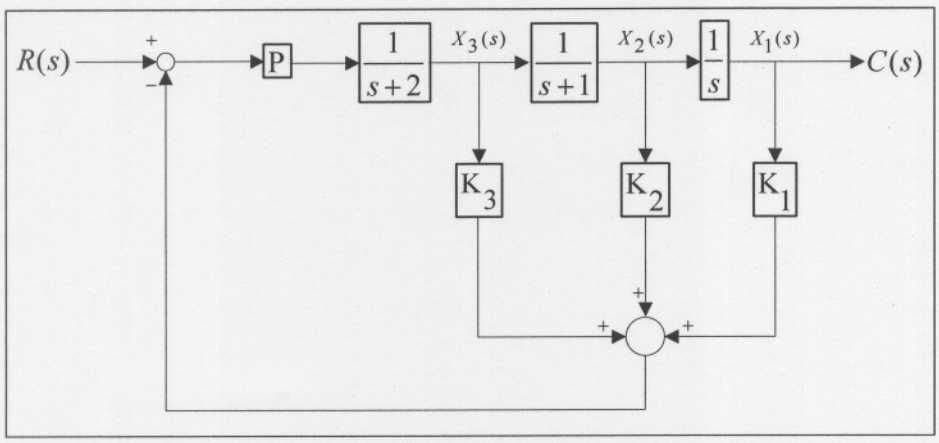

Figure 7: State-feedback control system

Resulting in the following diagram :

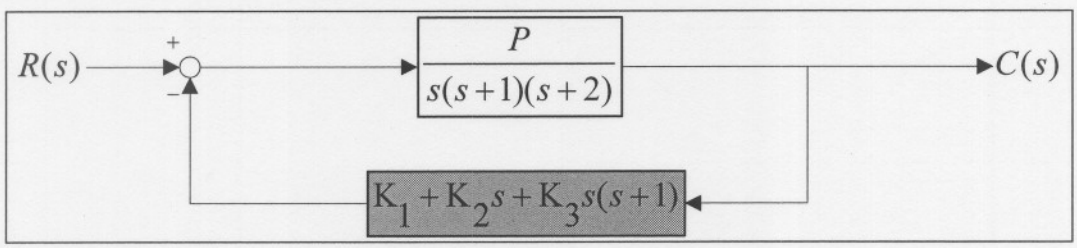

Figure 8: Single variable feedback control

Note:

The shaded block constitutes a state-feedback controller using transfer functions in the s-plane for a SISO system.

Point of fact for state feedback control: All states have to be measurable.

That which cannot be measured cannot be controlled. Thus, if a state is not directly measurable, its value has to estimated using state-estimation techniques. Some commonly used techniques will be mentioned later in section 2.3 on *design principles.*

In the case of this study, **state-feedback control**, as described above, is when transfer functions in the s-plane are used and the **output of *each state*** is fed back to the input. Intuitively this seems to be a better approach because the important parameters are controlled individually.
Feedback control is when the **output of the *controlled system*** is fed back to the input to give an error between the reference value and actual system output value, which is then manipulated by the controller to achieve a zero error [5]. The way it manipulates the error to give the desired response must adhere to certain performance criteria.

With **multivariable feedback control** it works the same, only now vector variables and matrix transfer functions are considered, and thus matrix calculus is used.
Thus, the outputs, after being referenced by the inputs, are fed back to the *matrix controller.* Refer to figure 9 below.

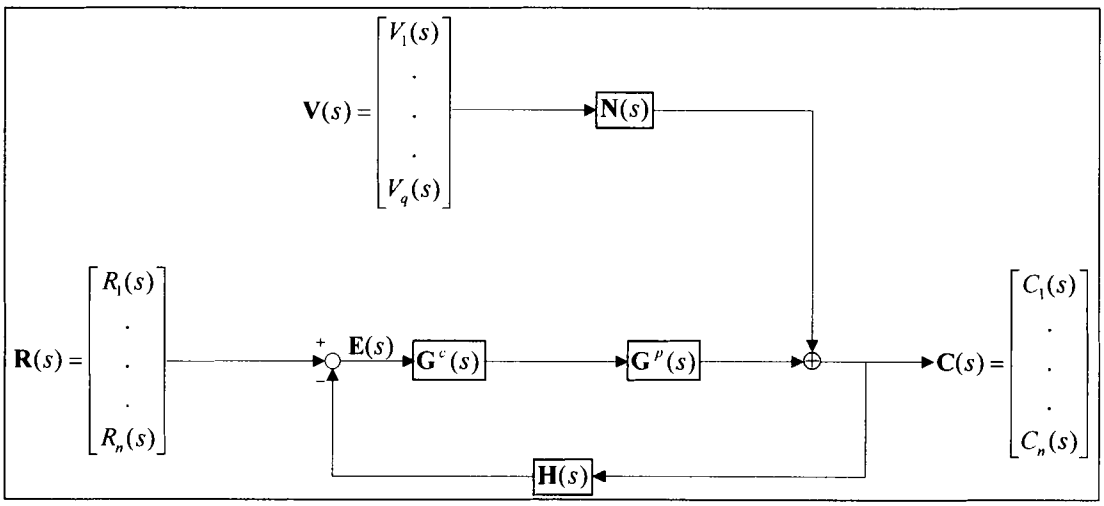

Figure 9: Multivariable control system

According to [1: pp.413-414] the closed loop characteristic equation of the block diagram in figure 9 is found to be

$$\det(\mathbf{I} + \mathbf{G(s)}) = 0 \qquad (6)$$

$\mathbf{G(s)} = \mathbf{G}_c\mathbf{G}_p$, and $\mathbf{I}$ is the identity matrix. $\mathbf{G}_c$ and $\mathbf{G}_p$ represent the controller matrix, and plant transfer matrix respectively.

In the case of a two- input, two- output linear plant as illustrated earlier by figure 4,

$$\mathbf{G}_c = \begin{bmatrix} G_{11}^c & G_{12}^c \\ G_{21}^c & G_{22}^c \end{bmatrix} \quad (7)$$

$$\mathbf{G}_p = \begin{bmatrix} G_{11}^p & G_{12}^p \\ G_{21}^p & G_{22}^p \end{bmatrix} \quad (8)$$

$$\mathbf{G(s)} = \begin{bmatrix} \mathrm{G}_{11}(s) & \mathrm{G}_{12}(s) \\ \mathrm{G}_{21}(s) & \mathrm{G}_{22}(s) \end{bmatrix} \quad (9)$$

By equation 6, the closed-loop characteristic equation will be given by

$$[1+G_{11}(s)][1+G_{22}(s)]-G_{12}(s)G_{21}(s)=0 \quad (10)$$

where, by matrix multiplication

$$G_{11}(s) = G_{11}^c(s)G_{11}^p(s) + G_{21}^c(s)G_{12}^p(s) \quad (11)$$

$$G_{12}(s) = G_{12}^c(s)G_{11}^p(s) + G_{22}^c(s)G_{12}^p(s) \quad (12)$$

$$G_{21}(s) = G_{11}^c(s)G_{21}^p(s) + G_{21}^c(s)G_{22}^p(s) \quad (13)$$

$$G_{22}(s) = G_{12}^c(s)G_{21}^p(s) + G_{22}^c(s)G_{22}^p(s) \quad (14)$$

Ideally, one would have the system be decoupled, i.e. one input affects only one output. This, however, is not always possible.

Using the above example to illustrate this, it can be seen from the set of equations above, and the system of figure 4, that two controllers influence the same output. For the system to be decoupled,

a way of achieving this is by making the off-diagonal elements, $G_{12}(s), G_{21}(s)$ of the matrix , $\mathbf{G}(s)$, zero.

To achieve this, cross-controllers, $G_{12}^{c}(s)$, and $G_{21}^{c}(s)$ are fixed to adhere to the following conditions:

$$G_{12}^{c}(s) = -\frac{G_{12}^{p}(s)}{G_{11}^{p}(s)} G_{22}^{c}(s) \qquad (15)$$

$$G_{21}^{c}(s) = -\frac{G_{21}^{p}(s)}{G_{22}^{p}(s)} G_{11}^{c}(s) \qquad (16)$$

When these conditions hold, 10 becomes:

$$\left[1+\frac{\det(\mathbf{G}^{p}(s))}{G_{22}^{p}(s)} G_{11}^{c}(s)\right]\left[1+\frac{\det(\mathbf{G}^{p}(s))}{G_{11}^{p}(s)} G_{22}^{c}(s)\right] = 0 \qquad (17)$$

There is a catch to this technique: the cross-coupling effect must be small, i.e. the effect of the off-diagonal elements of $\mathbf{G}(s)$ on the system relative to the desired performance must be negligible. If it is, the above-mentioned technique of decoupling can be used. If the cross-coupling is not small and has a dominant effect, this technique cannot be used and the system must be seen as a single entity [1].

- **Pole placement**

It is important to note that pole placement is more a technique to determine controller parameter values once a controller has been designed than a control technique itself, like the aforementioned techniques.

Pole placement techniques further include methods that are in closed form, suitable for direct machine computation, and those that are not, as well as adaptive control methods.

Also known as pole assignment, here, the desired location of the controlled system's poles is known. Knowing this gives the ability to formulate a characteristic equation.

Let's say the desired pole locations are wanted to be at p_1 and p_2. A possible resulting second order equation could have been

$$(s+p_1)(s+p_2)=0 \qquad (18)$$

which expands to

$$s^2+s(p_1+p_2)+p_1p_2=0 \qquad (19)$$

The form in equation (19) is useful because it enables one to compare coefficients with the characteristic equation of the controlled system developed in (6), if it can be written in the same form as (19). The result of which can be used to determine the controller constants.

An alternative to the equation in (19) is to take the desired damping and period of the system, and use the equations of Table 5.6 form [7: p.252], or Table 5.7 [7: p.256].

- **Lag-lead control (PID control)**

The PID controller is a special type of lag-lead controller. It can be described in terms of Laplace variables as follows:

$$G_c(s)=K_p+\frac{K_I}{s}+K_Ds \qquad (20)$$

A SIMULINK® representation of a PID controller is given below.

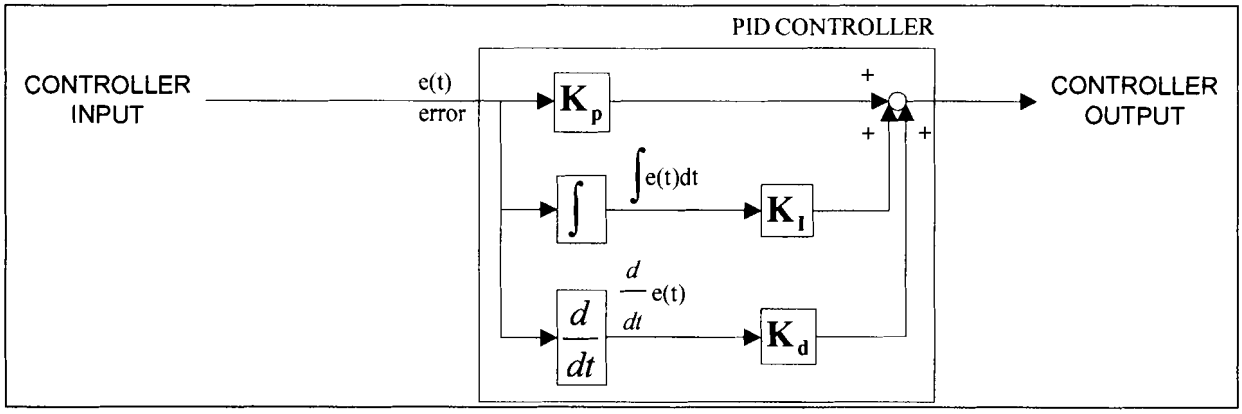

Figure 10: PID controller

It comprises of three parts, hence its other name: three-term controller.

Those parts are the proportional gain path, K_p, the integral path, K_I and the derivative path, K_D. Each part then performs its namesakes operation on the error it receives.
For different variations of PID controller, the corresponding constant can be made zero, as is the case of example a PI controller where the K_D constant is made zero.

2.2.2 Modern control

Here, controllers are designed by primarily making use of equations and representations in the time domain.
The main approaches are:

- State-space, which include modal control techniques
- Neural networks
- Fuzzy logic
- Neural-Fuzzy hybrid forms
- Variable structure control systems (VSCS), which include sliding mode control and adaptive control
- Optimal control

Modal control further includes control of lumped-parameter objects and distributed-parameter systems.
The different applicable approaches to the study will now briefly be discussed. Please refer to figure 2 for an illustration of the discussion below.

- **The state space approach**

In this approach, the problem is formulated using state variables[1]. This enables greater design flexibility to be maintained.

The motivation for using state variables is [4]:

- The differential equations are ideally suited for digital or analogue solution

[1] See Definition 2.1 by section 2.2

- It provides a unified framework for the study of linear and nonlinear systems
- It's invaluable in theoretical investigations
- The concept of state has strong physical motivation

To bridge the gap between classic control techniques and modern control using a state-space approach, an example is given.
The true value of the state space becomes evident out of the discussion on modal control.

Example

See figure 1.

Consider a linear system, described by the following state-variable equation,

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}, \qquad (21)$$

with

$$\mathbf{y} = \mathbf{C}\mathbf{x} \qquad (22)$$

Assume zero reference and zero initial conditions.

Further, assume that the inverses of both **B** and **C** exist, and that there is no direct transmission from **u** to **y**. Let the desired plant behaviour be described by $\mathbf{A}_d$. A vector feedback control law in the form

$\mathbf{u} = -\mathbf{K}\mathbf{y}$, will produce the desired closed loop plant dynamics, i.e. $\frac{\mathbf{dx(t)}}{\mathrm{dt}} = \mathbf{A}_d\mathbf{x}$.

By using 22 and substituting $\mathbf{u} = -\mathbf{K}\mathbf{C}\mathbf{x}$ into 21, one gets

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{A}\mathbf{x}(t) - \mathbf{B}\mathbf{K}\mathbf{C}\mathbf{x} = \mathbf{A}_\mathrm{d} \qquad (23)$$

The equation for the plant to be controlled, control object, then becomes

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t), \tag{24}$$

where **x** is an n-state vector, **A** is a constant matrix, **b** is a n-vector, and the states are directly measurable, implying that **C=I** or **y=x**.

The control law can be represented by

$$\mathbf{u}(t) = -\mathbf{k}'\mathbf{x}(t) \tag{25}$$

where $k' = [k_1, k_2, k_3, ..., k_n]$

The resulting closed loop system can thus be described by

$$\frac{d\mathbf{x}(t)}{dt} = (\mathbf{A} - \mathbf{b}\mathbf{k}')\mathbf{x}(t), \tag{26}$$

from which the characteristic equation of the closed loop system is given by

$$\Delta_c(s) = \det(\mathbf{s}\mathbf{I} - \mathbf{A} + \mathbf{b}\mathbf{k}') \tag{27}$$

This is according to [1: p.423].

Modal control:

To illustrate where a modal controller is used, see the illustration below.

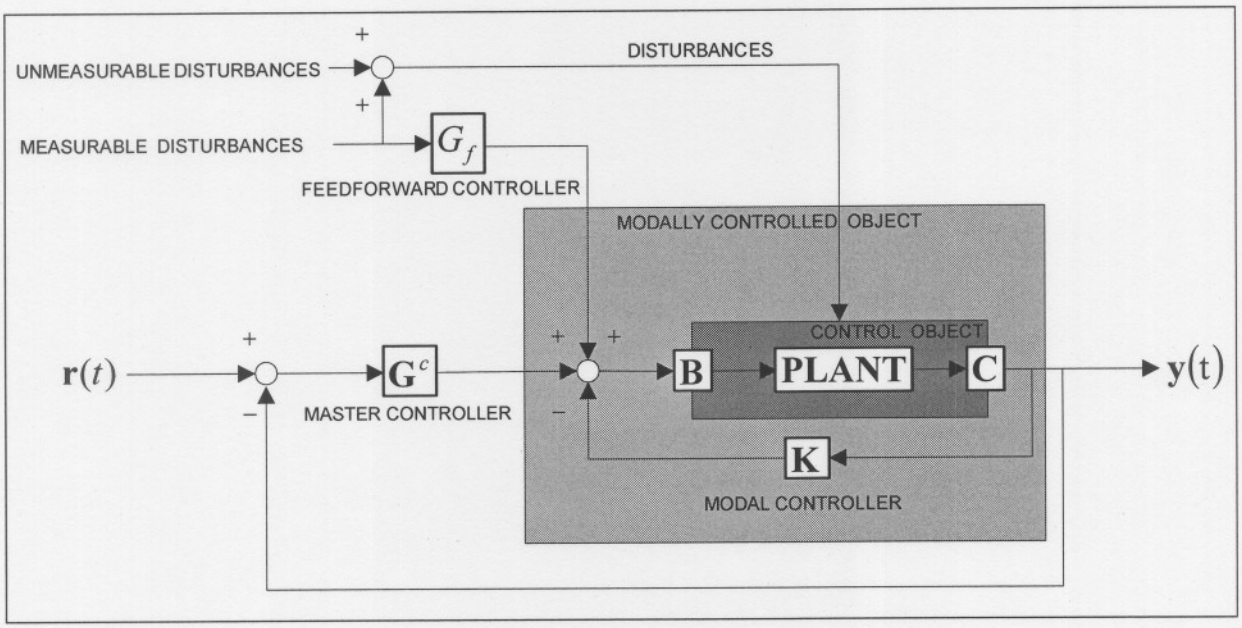

Figure 11: Modal control as a subsystem of a cascade control system [1]

Two types of modal control exist, m*odal control of lumped-parameter objects,* and m*odal control of distributed-parameter systems*. The first is of relevance to this study, whereas the latter is not. For an understanding of modal control of distributed-parameter systems a discussion in [1: p446 section 4-6] is given. The brief discussion below is with regard to the modal control of lumped-parameter objects.

The modal controller can be seen as an inner-loop controller, used in conjunction with an outer-loop, master controller. The controller itself can be a P, PI or PID controller, given its widespread industrial use in SCADA systems.

With this architecture, the modal controller can be used to alter the system's dynamics, i.e. its eigen values, and the master controller ensures that any desired equilibrium state is reached.

According to [1: pp.431-432], the overall scheme as illustrated above will improve dynamic system response in terms of response time and its ability to follow a reference signal.
To now, it was assumed that a complete measurement of the state-vector was possible. It often occurs that one cannot measure all the states and that more than one, but less than n, manipulated variables can be applied to the control object. If there are m number of measurements, and r

number of controls, with the order of the control object being n, a situation occurs were m and/or r is less than n. In the ideal case where a complete state-vector measurement is possible with as many controls as manipulated variables,

$$m = n$$
$$r = n$$

In the non-ideal case, it causes the controller matrix $\mathbf{K}$ to become rectangular.

In order to address this problem, a modal domain state vector $\mathbf{x}^*$ is generated via a "mode-analyzer" $\mathbf{T}^{-1}$ from the measured state-vector $\mathbf{x}$.
Once in the modal domain, a modal domain control vector, $\mathbf{u}^* = -\mathbf{K}^*\mathbf{x}^*$, is produced and then transformed back to the normal state-space control vector, u_C, by "mode synthesizer" $\mathbf{T}$.

A design for a multivariable control system in the modal domain was first proposed by Rosenbrock [8] and then later extended to include linear distributed parameter systems by M.A. Murray-Lasso, L.A. Gould, and F.M. Schlaefer [9,10,11].

The process above is illustrated in the figure below.

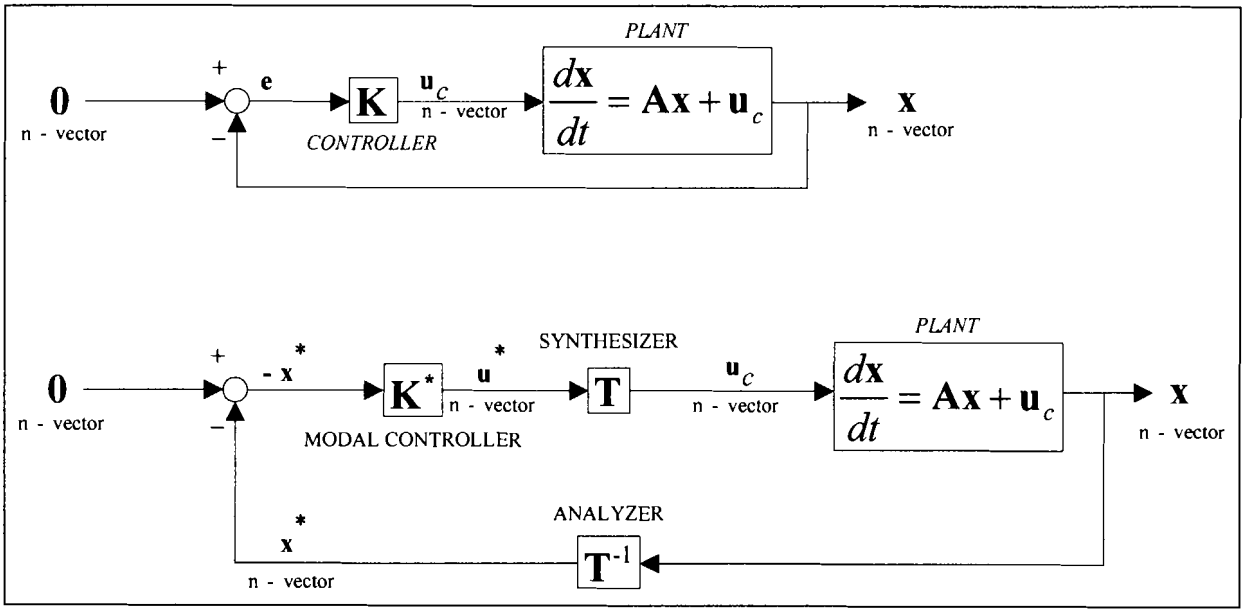

Figure 12: Ideal modal control system [1]

The number of controls and measured states available, determines the number of modes, q, of the control object that can actually be controlled.
Then

$$q = m \text{ if } m \leq r, \text{ or}$$
$$q = r \text{ if } r \leq m.$$

Examples of how this is handled can be seen in example 10-7 [1: p.442] and example 10-8 [1: p.445].

The important aspects of the modal control of lumped-parameter objects can be summarized by the next diagram.

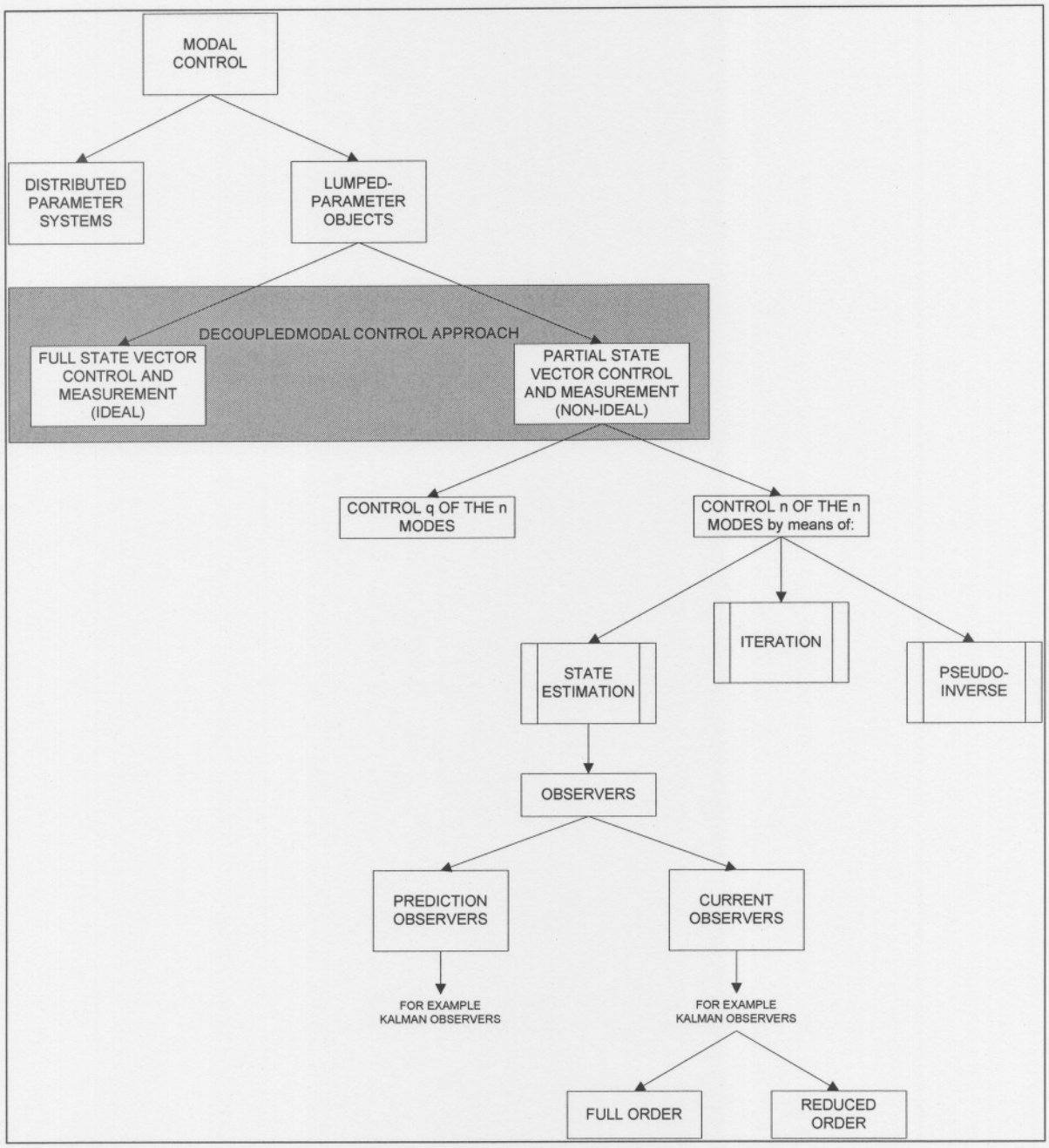

Figure 13: Schematic breakdown of modal control

- **Variable structure control systems**

These systems are a class of systems where the control law is adapted dynamically during the control process according to some rule set. This rule set is dependant on the current state of the system [13: p.1].

Examples of such variable structure control systems are *sliding mode control* and *adaptive control methods.*

Sliding mode control uses a sliding variable $s(t)$, which is selected such that it has a relative degree of one with respect to the control. Its purpose is to ensure the dynamics of the system remain in the sliding mode, $s = 0$. The control acts on the first derivative of the sliding variable, $\dot{s}$ to keep the system's states such that $s = 0$. [13, 14]

Adaptive control can be divided into two main groups, direct and indirect adaptive control. Direct methods (the controller parameters are updated directly), include model reference adaptive controllers and gain scheduling controllers. The control parameters can be updated directly because one has a predetermined reference model of the plant with the desired response built in. Indirect methods include self-tuning regulators. The methods are indirect because the plant parameters are first determined recursively, before updating the controller parameters using some fixed transformation method. [12]. Adaptive control structures are based on heuristic methods. If its arguments could be based on a theoretical framework rather than heuristic methods, the result would be a stochastic controller [12: p.10]. The figures below illustrate the different schemes.

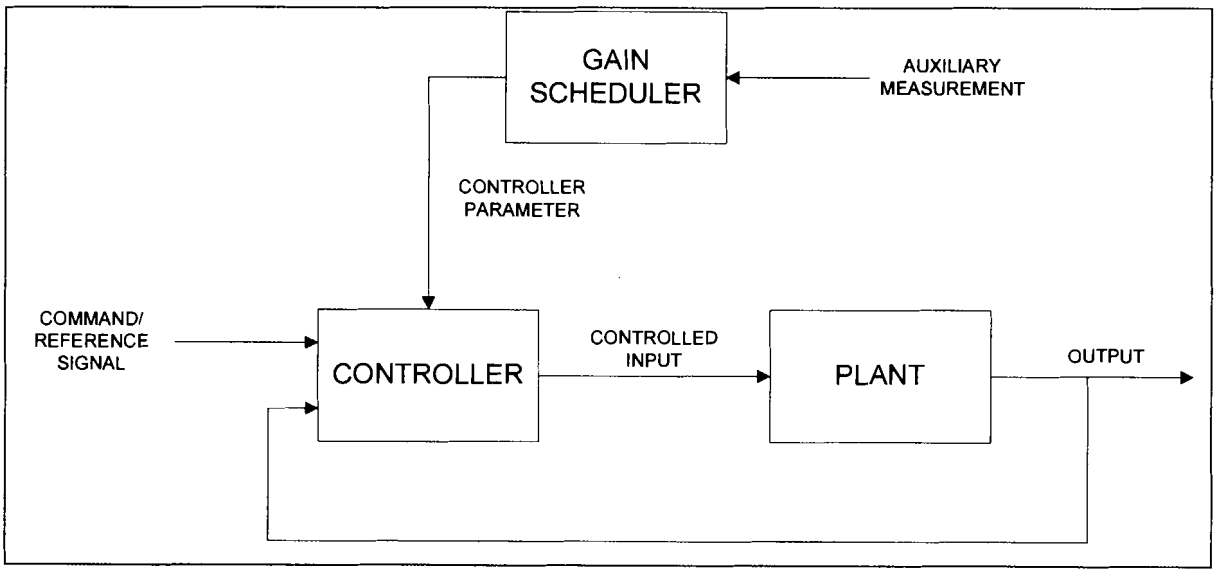

Figure 14: Gain scheduling [12]

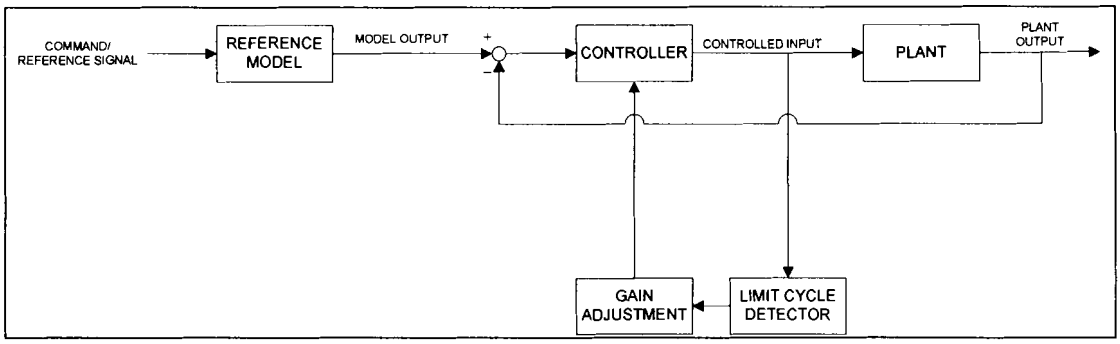

Figure 15: Model reference adaptive control – series scheme [12]

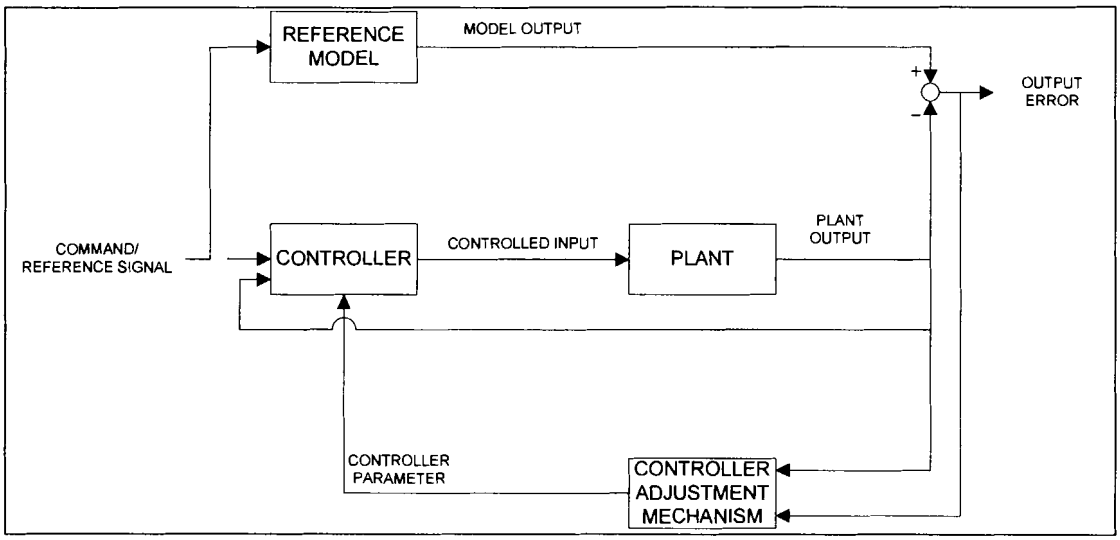

Figure 16: Model reference adaptive control – parallel scheme [12]

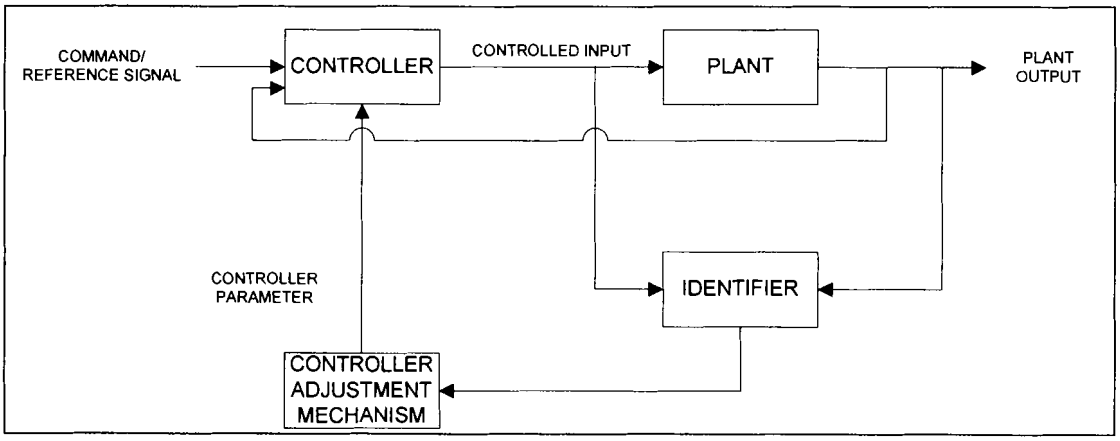

Figure 17: Self-tuning controller [12]

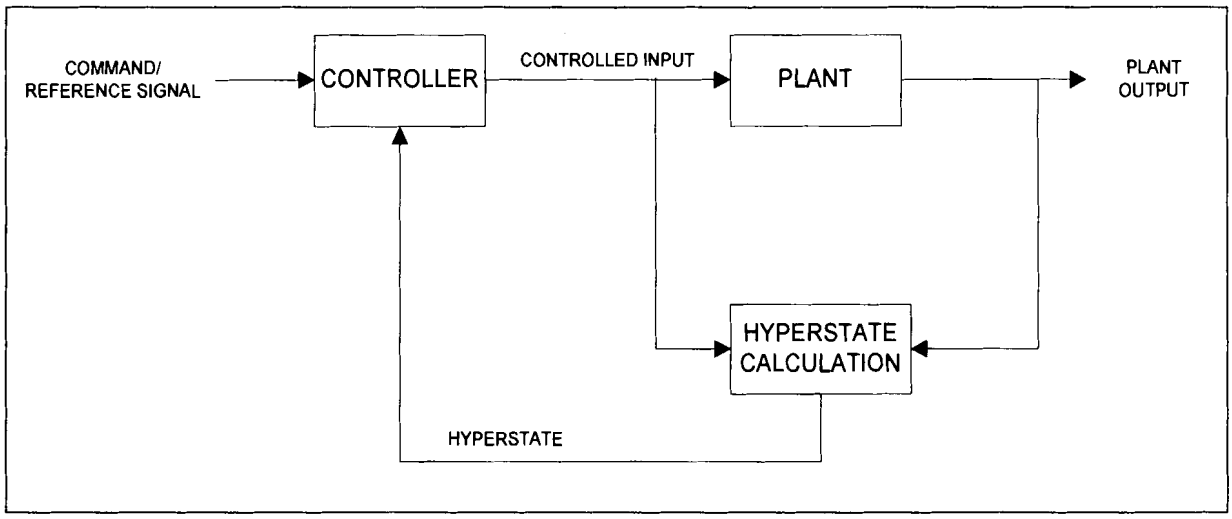

Figure 18: Stochastic controller [12]

An *optimal controller*, is also an example of a VSCS if it is state dependant, i.e. it dynamically changes depending on the current states of the system. However, what sets it apart is the fact that it is mainly time-dependant rather than state-dependant and more important, it is designed specifically to minimise or maximise some performance index. For this reason it can be seen as a separate control approach, as depicted in figure 2 control schematic.

- **Optimal control**

It is necessary to first define what is meant by an optimal control problem.

Before one can define the optimal control problem three things are necessary for its formulation according to Kirk [4: p.4]:

- A mathematical description (model) of the process to be controlled
- A statement of the physical constraints
- Specification of the performance criterion.

Further, one needs the following:

Definition 2.2

A history of control input values during the interval $[t_0, t_f]$ is denoted by $\mathbf{u}$ and is called a *control history,* or simply a *control.* [4: p.6]

Definition 2.3

A history of state values in the interval $[t_0, t_f]$ is called a *state trajectory* and is denoted by $\mathbf{x}$. [4: p.6]

Definition 2.4

A control history which satisfies the control constraints during the entire time interval $[t_0, t_f]$ is called an *admissible control.* [4: p.7]

Definition 2.5

A state trajectory which satisfies the state variable constraints during the entire time interval $[t_0, t_f]$ is called an *admissible trajectory.* [4: p.8]

Finally, one can define the optimal control problem as:
The admissible control, $\mathbf{u}^*$ which causes the system $\dot{\mathbf{x}}(t) = \mathbf{a}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{t})$ to follow an admissible trajectory, $\mathbf{x}^*$ that minimises/maximises the performance measure $J = h(\mathbf{x}(t_f), t_f) + \int_0^{t_f} g(\mathbf{x}(t), \mathbf{u}(t), \mathbf{t})dt$ (general form), denoted by J^*.

Where $\mathbf{u}^*$ and $\mathbf{x}^*$ denotes the optimal control and optimal trajectory respectively.
The form chosen for J, yields terms like linear quadratic (LQ) optimal control. The different forms really depend on the designer.

The optimal control, $\mathbf{u}^*$, *is only as good as the performance function.*

The goal is to use a performance- or cost function that, when minimised/maximised, yields the best solution for the given problem. The best solution being the $\mathbf{u}^*$ that causes the system to follow $\mathbf{x}^*$ and that also best tracks the reference signal whilst adhering to the physical constraints thereof.

An attempt toward optimal control was made using a GA in this study. Given its dependence on a performance function, some examples are briefly mentioned.

One can create this performance function, or use an existing form. Some basic LQ

forms are briefly discussed next [4: pp. 30-34]:

Minimum-time problems

These problems involve the transfer of a system from an arbitrary initial state $\mathbf{x}(t_0) = \mathbf{x}_0$ to a target state in minimum time.

The performance measure to be minimised is then

$$J = t_f - t_0 = \int_0^f dt \qquad (28)$$

Terminal control problems

Here the desired final sate of the system is to be as close as possible to the desired final state.

The performance index then becomes

$$J = \sum_{i=1}^{n} [x_i(t_f) - r_i(t_f)]^2 \qquad (29)$$

where n is the number of states and $i = 1,2,3,...,n$

In terms of matrix notation, useful for the handling of multivariable systems, this can be written as

$$J = [\mathbf{x}(t_f) - \mathbf{r}(t_f)]^T [\mathbf{x}(t_f) - \mathbf{r}(t_f)] \qquad (30)$$

This is often written as

$$J = \left\| \mathbf{x}(t_f) - \mathbf{r}(t_f) \right\|^2 \qquad (31)$$

Which is called the *norm* of the vector $[\mathbf{x}(t_f) - \mathbf{r}(t_f)]$.

What is often done is to insert a real symmetric positive semi-definite $n \times n$ weighting matrix $\mathbf{H}$.[2] The purpose of this matrix is to adjust the contribution each state has to the performance index.

[2] "A real symmetric matrix $\mathbf{H}$ is positive semi-definite (or non-negative definite) it for all vectors $\mathbf{z}$, $\mathbf{z}^T\mathbf{Hz} \geq 0$. In other words, there are some vectors for which $\mathbf{Hz} = \mathbf{0}$, and for all other $\mathbf{z}$, $\mathbf{z}^T\mathbf{Hz} > 0$" [4: p.31]

This is useful when a certain state is of greater importance than others. The equation above is then written as

$$J = [\mathbf{x}(t_f) - \mathbf{r}(t_f)]^T \mathbf{H}[\mathbf{x}(t_f) - \mathbf{r}(t_f)] \qquad (32)$$

Or

$$J = \left\|\mathbf{x}(t_f) - \mathbf{r}(t_f)\right\|_{\mathbf{H}}^2 \qquad (33)$$

It can be seen that if all the states are to contribute equally to the performance index, $\mathbf{H}$ is the $n \times n$ identity matrix.

Minimum control effort problems

This requires the transfer of a system from an arbitrary initial state $\mathbf{x}(t_0) = \mathbf{x}_0$ to a target state with a minimum expenditure of control effort.

The performance index then becomes

$$J = \int_0^{t_f} [\mathbf{u}^{\mathrm{T}}(t)\mathbf{R}\mathbf{u}(t)]dt = \int_0^{t_f} \left\|\mathbf{u}(t)\right\|_{\mathbf{R}}^2 dt \qquad (34)$$

$\mathbf{R}$ being a real symmetric positive definite weighting matrix[3].

Tracking problems

Involve maintaining the system state $\mathbf{x}(t)$ as close as possible to a desired state during the interval $[t_0, t_f]$.

An obvious choice for the performance measure is then

$$J = \int_0^{t_f} \left\|\mathbf{x}(t_f) - \mathbf{r}(t_f)\right\|_{\mathbf{Q}(t)}^2 dt \qquad (35)$$

[3] "A real symmetric matrix $\mathbf{R}$ is positive definite if $\mathbf{z}^{\mathrm{T}}\mathbf{R}\mathbf{z} \geq 0$ for all $\mathbf{z} \neq \mathbf{0}$". [4: p.33]

The weighting matrix **Q** serves the same function where it is used, and has the same properties, as **H** mentioned earlier.

If it also important to expend as little as possible control effort J becomes

$$J = \int_0^f [\|\mathbf{x}(t) - \mathbf{r}(t)\|^2_{\mathbf{Q}(t)} + \|\mathbf{u}(t)\|^2_{\mathbf{R}(t)}]dt \qquad (36)$$

If it further becomes very important for the states to be equal to their desired values at the final time t_f, the performance measure can be written as follows,

$$J = \|\mathbf{x}(t_f) - \mathbf{r}(t_f)\|^2_{\mathbf{H}} + \int_0^f [\|\mathbf{x}(t) - \mathbf{r}(t)\|^2_{\mathbf{Q}(t)} + \|\mathbf{u}(t)\|^2_{\mathbf{R}(t)}]dt \qquad (37)$$

which, as can be seen, is the same as the general form given earlier except for the use of the weighting matrices.

Regulator problems

A regulator problem is identical to that of a tracking problem, but the desired state values $\mathbf{r}(t) = \mathbf{0}$ for all $t \in [t_0, t_f]$.

As stated, the selection of a performance function is paramount.

In this study, the performance measure is to be minimised.

Minimising the performance measure implies that one seeks to find J^*, where $J^* \leq h(\mathbf{x}(t_f), t_f) + \int_0^f g(\mathbf{x}(t), \mathbf{u}(t), \mathbf{t})dt$ for all admissible controls causing admissible state trajectories. In other words, one is searching for the global or absolute minimum of J.

There are many ways to determine this absolute minimum. One possible, but totally inefficient and not always possible way is to literally try every combination of admissible control causing an

admissible trajectory, and then calculate the value of the performance measure. The control that yields the lowest value of J is the optimal control.

The best way, but also a considerably more difficult one, is to use deterministic methods. These methods are especially hard to code because they are not straightforward, and depend heavily upon designer initiative and his/her knowledge of the system.

A few such proposed methods are:

- The method proposed by Salukvanze, based on a Lyapunov function [4: p.452]. This method does not provide a direct algorithm to compute the control law, but is flexible enough for time-varying linear systems and/or systems with constraints. Some other methods which directly compute the control law are those referred to in [12, 21, 22, 23] on [1: pp.458-459].
- The minimum/maximum principle of Pontryagin [4: p.53]
- Dynamic programming developed by R.E. Bellman making use of the so called Hamilton-Jacobi-Bellman equation [4: p.53]

Another method is the use of a genetic algorithm to determine the global minimum of J. This is, however, not a deterministic approach, but a probabilistic one.
A great advantage that it has above the aforementioned methods is its ease to code, as will be seen where it is used in this study.

The **genetic algorithm** is a stochastic global search method that mimics the metaphor of natural biological evolution. [19]
One therefore encounters natural biological terminology such as chromosomes, genotypes and phenotypes.

Here, the chromosome is composed of some alphabet, in such a way that the chromosome values (genotypes) are mapped uniquely onto a decision variable domain called the phenotypic domain.

To better understand the terms as applied in the algorithm, consider an individual, represented by the binary string

1100110110

This is the binary encoded *chromosome* of the individual. By using bs2rv[4], this string represents the real value of

267.8397

which is its *genotypic* value. Here, it was chosen that the individual's genotypic value was to be encoded using a 10 bit precision, any apt precision resolution could have been used. By decoding the chromosome representation to its genotypic value, it has been mapped onto the *phenotypic domain.*

Though the genotypic value is the representation that has meaning in the problem domain, the search process for the fittest individuals operates on the encoding of the decision variable, i.e. its chromosome representation. This holds for when then chromosome is real-valued.

Once in the phenotypic domain, the specific *individual's performance* can be evaluated by determining its *fitness.* This is based on its *objective function* value. Once the objective function values for the population have been determined, they are *ranked* accordingly. In the case of this study, the fittest individual would be the one that **minimises** the performance index[5]. The fittest individuals then stand a better chance of being selected for breeding, depending on the selection method used. *Offspring* are then created by making use of mutation and cross-over techniques which will be discussed later. The *generation gap* is the difference between the number of individuals in the original population, and the number of offspring created.
The next generation's population is then formed by recombining the original population's fittest individuals with the offspring. These fittest individuals that survive to the next generation depend on the *recombination* method used.

The basic idea now known, it is necessary to be familiar with some of the processes involved. These processes include the method used to determine objective function value, the method used to rank the population, the method used to select individuals for breeding, the breeding methods, and the recombination of the original population's surviving individuals and the offspring.

[4] bs2rv is a function part of the GA Toolbox. Please see the appendix for the associated code
[5] Throughout the study either the ITSE or ISE performance index/objective function was used.

Objective function value

This is nothing other than the performance index. In this research the ITSE and ISE performance indices [7: pp.248-249] are used.

The error of the system over the simulation period $[t_0, t_f]$ is fed to the objective/cost function, the result of which is the objective value for that specific individual.
The objective function for the ISE criteria is

$$ISE = \int_0^f e^2(t)dt \qquad (38)$$

and for ITSE is

$$ITSE = \int_0^f te^2(t)dt \qquad (39)$$

where t is the simulation time.

Ranking

The term *fitness value* refers to the ranked objective function value. As stated, the individual's are ranked for minimisation.
Two types of ranking can be used, linear-, and nonlinear ranking. Only linear ranking is used during this study.

The fitness value assigned to each individual, is calculated using the function $F(x_i)$, where MAX is the *selective pressure*; the bias toward the fittest individual.
Generally, the selective pressure varies between 1.1 and 2. In this study $MAX = 2$, causing the fitness value assigned to the fittest individual to be 2, and zero to be assigned to the least fit individual. $F(x_i)$ is directly calculated by:

$$F(x_i) = 2 - MAX + 2(MAX - 1)\frac{x_i - 1}{N_{ind} - 1} \qquad (40)$$

where x_i is the position in the ordered(ascending) population of individual i.

Individual selection

The selection of individuals for breeding of a population was done in one of two ways; via stochastic sampling with replacement (SSR)[6] or stochastic universal sampling[7].

Stochastic sampling with replacement (SSR) is a basic roulette wheel selection method. It finds the real-valued *sum* of either the individuals' expected selection probabilities, or the fitness values over all the individuals in the current population. This sum is then used to represent a circle. Each individual then contributes to a portion of this circle, coherent with its contribution to the value of sum, i.e. the fitter the individual or the greater its probability of selection, the greater part of the circle it represents relative to the other individuals. This is illustrated below in figure 19.

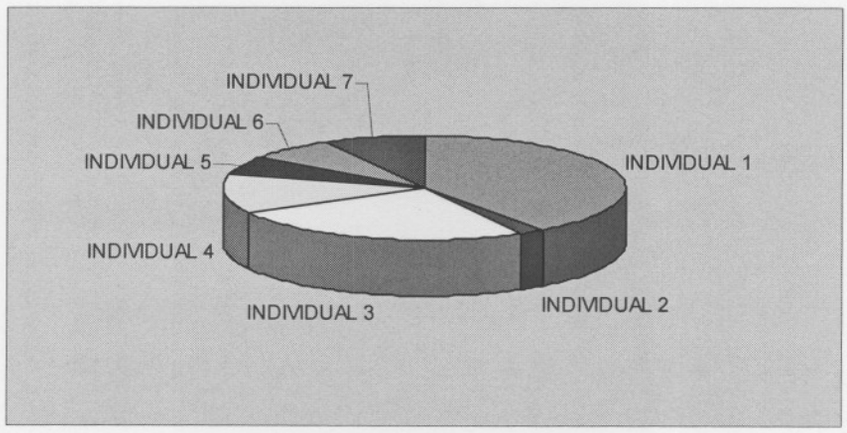

Figure 19: Roulette wheel mechanism [19]

To select an individual, a random number between zero and the sum is selected. The individual whose segment spans the random number is selected. This is then repeated as many times as the number of offspring to be created. It can therefore occur that a specific individual is selected more than once for breeding, or may even fill an entire group of offspring. A way of addressing this is by decreasing the segment of an individual, once chosen. This is known as stochastic sampling with partial replacement (SSPR). The segment can eventually become zero, but not negative.

Two additional methods worth mentioning, but not of interest to this study are remainder stochastic sampling with replacement (RSSR), and remainder stochastic sampling without replacement (RSSWR).

[6]The 'rws' function of the GA Toolbox
[7]The 'sus' function of the GA Toolbox

Stochastic universal sampling is a single phase sampling algorithm with minimum spread and zero bias. It makes use of N equally spaced pointers, where this number is equal to the number of offspring to be created. The population is randomly shuffled and a pointer, ptr, is generated between zero, and the *sum* mentioned earlier, divided by the number of selections. N individuals are then chosen by generating N pointers spaced by 1, $ptr, ptr+1,..., ptr+N-1$, and then selecting the individuals whose fitness span the positions of the pointers.

Breeding methods

Once individuals have been selected for breeding, they can be altered to produce offspring. These methods include:

Crossover techniques

The techniques are discussed in detail in [19].

Single-point crossover is illustrated below.

If two selected binary-encoded chromosomes are considered, for example

$$P_1 = 10010\mathbf{110}$$
$$P_2 = 10111\mathbf{000}'$$

the resulting offspring for a single-point crossover at $i = 5$ will be

$$O_1 = 10010\mathbf{000}$$
$$O_2 = 10111\mathbf{110}$$

In addition, *multi-point crossover* techniques exist where bits cross over at different intervals. An example of this would be using P_1 and P_2 from above the bits cross over from location i = 2,3,4 and i = 7,8,9 between P_1 and P_2. This is illustrated below where $k_i \in \{k_1, k_2\}$, with $k_1 \in \{1,2,3\}$ and $k_2 \in \{6,7,8\}$. k_i are the crossover points. Thus one has

$$P_1 = 1\mathbf{001}01\mathbf{101}0$$
$$P_2 = 1\mathbf{011}10\mathbf{000}1$$

which produce

$$O_1 = 1\mathbf{011}010\mathbf{000}$$
$$O_2 = 1\mathbf{001}101\mathbf{011}$$

Uniform crossover is characterised by the use of a mask, M_1, the length of the chromosome. This mask determines which parent supplies which bit/contribution for/to the offspring.

Example

$$P_1 = 1011000111$$
$$P_2 = 0001111000$$
$$M_1 = 0011001100$$
$$O_1 = 0011110100$$
$$O_2 = 1001001011$$

It can be seen that O_1 is formed by taking P_1's bits if the corresponding mask bit is a 1, and P_2's bits if the corresponding mask bit is a 0. O_2 is formed in the same way but by using the inverse of the mask.

Other forms of crossover operators also exist, like *shuffle* and *reduced surrogate* operators, but are not of importance in this study.

In the event where the chromosome is real-valued, a recombination method is used. *Intermediate recombination* is when offspring are created around and between the values of the parents. The rule which govern this process is

$$O_1 = P_1 * \alpha(P_2 - P_1) \qquad (41)$$

where P_1 and P_2 are the parent chromosomes, and α is a scaling factor chosen uniformly at random over some interval. This interval is usually $[-0.25,1.25]$.

With the making of each offspring, pair of parents comprising gene1 and gene 2, a different α is used. These rules enable the production of new variable within a slightly larger hypercube than the original parents, but within the range of α. The figure below illustrates this.

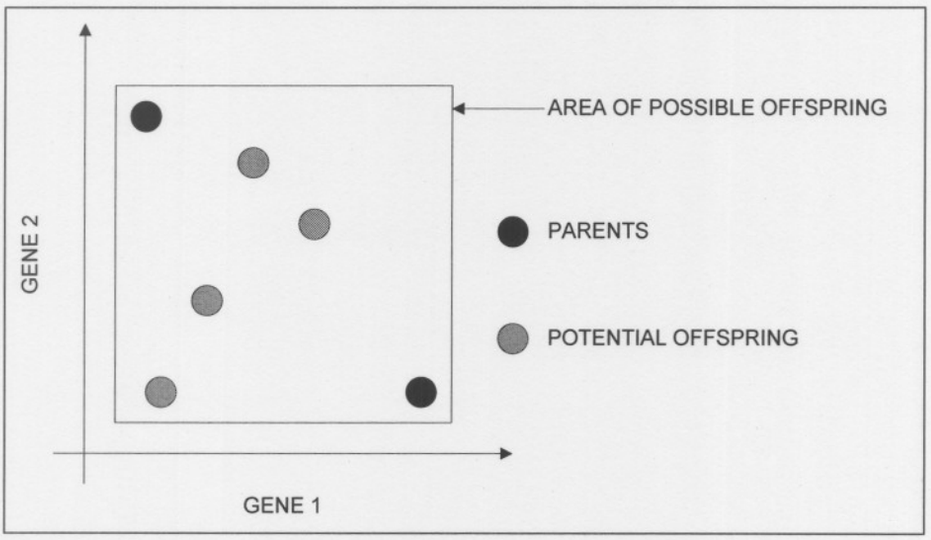

Figure 20: Geometric effect of intermediate recombination [19]

Another recombination method is *line recombination*. It works in the same fashion as intermediate recombination, but the value of α remains constant for each offspring produced. Figure 21 illustrates this effect.

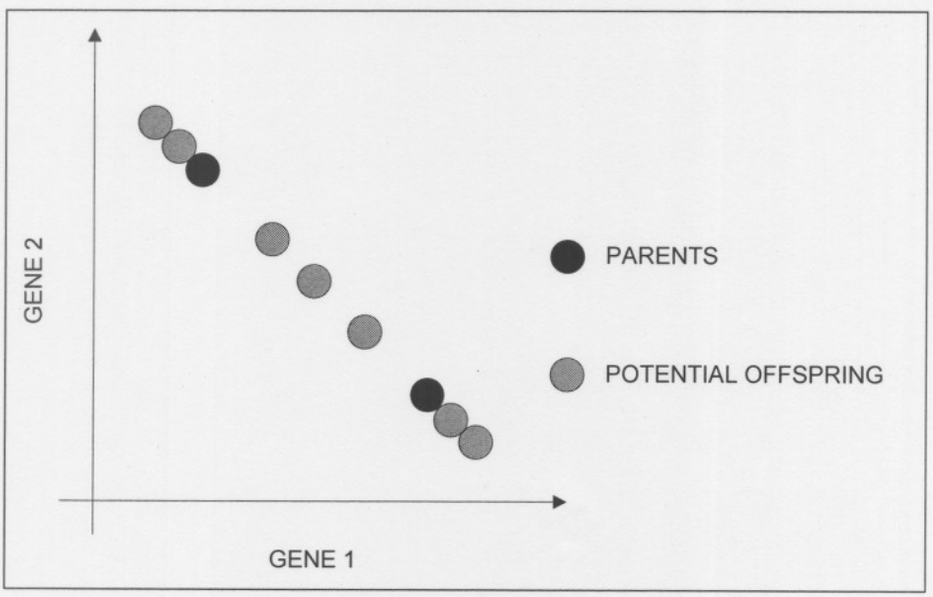

Figure 21: Geometric effect of line recombination [19]

Mutation techniques

Mutation can be applied to already-formed offspring, or on a specific parent to form an offspring for that parent. In the case of a binary-encoded chromosome, mutation would imply that a zero becomes a one, i.e.

$$00\mathbf{0}1100010 -> 00\mathbf{1}1100010$$

The above example depicts a mutation of the original string (left) at location 3, to form the mutated string (right). Mutation can occur at more than one location.

In the event of a real-valued chromosome, each variable in the individual's chromosome is altered/mutated with a given probability[8].

Reinsertion

After forming the offspring, the surviving members of the original population have to be reinserted into the new population containing the offspring. If a generation gap does not exist, or a progressively shrinking population is desired this is not necessary.

It is assumed that a progressively expanding population is not wanted either. The norm is for the next population to become progressively smaller until the surviving member is the solution to the stated problem, or to hold the population the same size, and just go through a number of generations until the performance criteria is met, or the maximum number of generations has been reached.

Reinsertion is thus of interest for the last-mentioned case.

There are two ways of inserting offspring into the parent-population: The offspring can replace parents uniformly at random. This forms a new population comprising of the offspring and parents who were not necessarily the fittest. Another way is to replace the least fit parents with the offspring. This method is an elitist strategy, because it ensures that the fittest individuals propagate from generation to generation, and depending on the selection method, will be used to breed offspring.

[8] GA Toolbox function 'mutbga' illustrates the mutation for a real-valued chromosome

Genetic Algorithms differ from traditional search and optimisation methods in the following ways:

- GA's search a population of points in parallel, not a single point;
- GA's do not require derivative information or auxiliary knowledge because only the objective function value and corresponding fitness values influence the search for a solution;
- GA's use probabilistic transition rules, instead of deterministic ones;
- GA's work on the encoding of the parameter set, i.e. the chromosome, rather than the parameter set itself. In the case where the chromosome is encoded using real-values, the preceding statement still holds [19: pp. 1-5].

2.3 Design principles

The difference between a design technique and a design principle is that the design principle directly affects where in the system the controller is placed, i.e. it determines the topography and signal flow path of the controlled system. The design technique is the manner in which the controller is designed. Section 2.2 discussed the techniques extensively. The different topographies for a controlled system are endless, and left up to the designer's discretion; some tried and proven structures do exist though.

The principle generally used throughout this study is the **series/cascade compensation** principle. An example of this is shown in figure 1, and figure 3. For other structures for MIMO systems, see Mesarovic, M.D. [20].

For basic design principles, see figure 22: 'Schematic of Basic Design Principle', are illustrated below using SISO, linear systems.

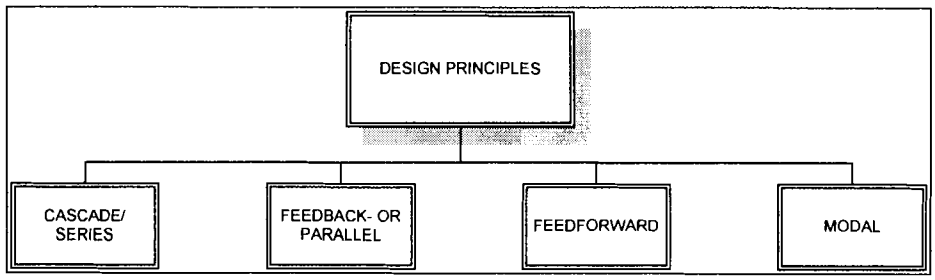

Figure 22: Schematic of basic design principle

- Feedback-/parallel-/minor loop compensation

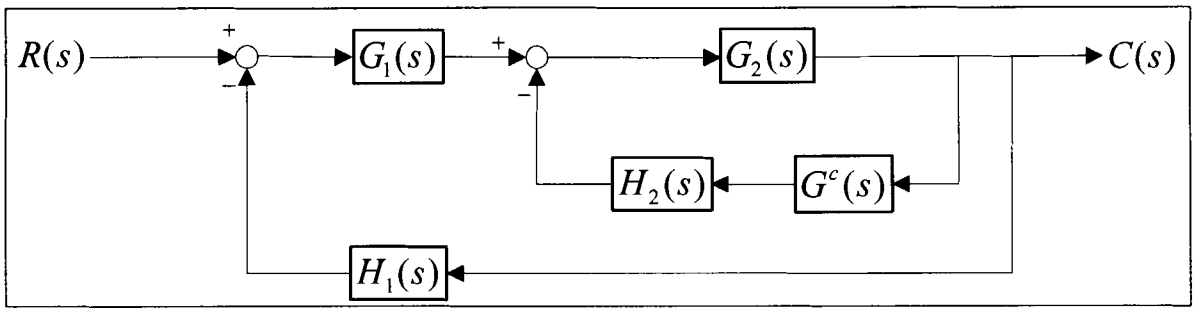

Figure 23: Feedback, parallel-loop compensation [5]

$G_1(s)$ and $G_2(s)$ make up the plant to be controlled. There are instances where it is more feasible to control a subsystem, than the entire plant. Here, $G_2(s)$ is such a subsystem. $G^C(s)$ is the controller, and $H_1(s)$ and $H_2(s)$ the feedback dynamics of the system.

- Feedforward compensation

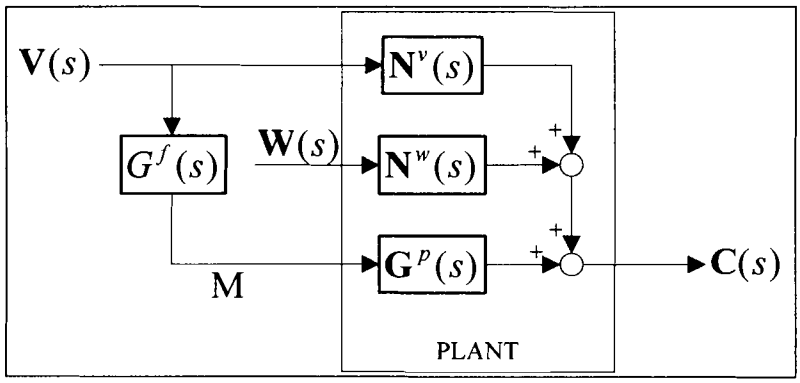

Figure 24: Feedforward compensation [1]

- Combined feedback-feedforward compensation

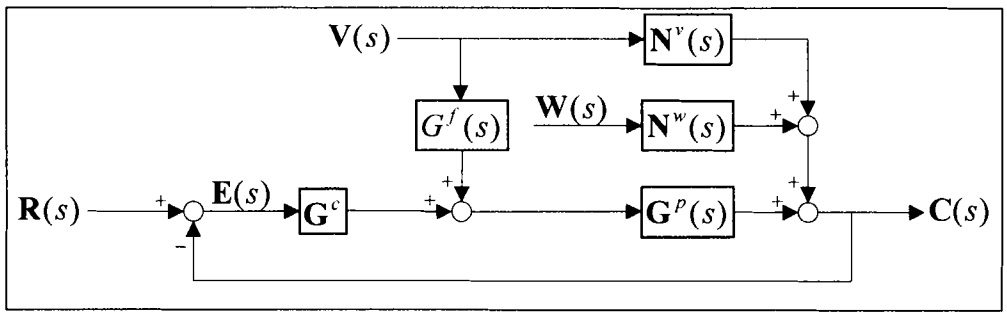

Figure 25: Hybrid feedback- feedforward compensation [1]

- Modal control as a subsystem of a cascade control system [1]

  Modal control is a control technique, but when incorporated as in Figure 11, one can see how it affects the closed loop's topography and why it can be seen as a design principle as well.

2.4 Control basics

Given our prior knowledge of control systems to this point, it is good to add a few basics.

A control system must be designed so that the controlled variables follow the path of the command/desired variables, irrespective of the command/desired variables' path. In addition to this, the system must be resistant if not completely impervious to disturbances. One could therefore say that the controlled system must meet the following requirements:

- The closed system must be at least stable
- Disturbances/noise must be completely rejected, or have a small influence on the controlled variable of interest
- The reference signal/model should be tracked as quickly and precisely as possible
- If possible, the controlled system must have a low sensitivity toward parameter changes. These changes could be due to error creeping into the process, which then has to be rectified, or on the other hand be due to component degradation, which needs to be managed, but cannot be helped [21].

If some sort of a *performance index* is added as to just how well a closed-loop system *must* operate, optimal control theory applies.

In general, the sequence of events in a control system is:

- Get a measurement of the variable which is to be controlled. In the case of state-based control, this will be a measurement of the state one wishes to control. This may be done directly, or may have to be estimated using state estimation techniques.
- Calculate the error between the desired response and actual plant output yielding the control error.

- Process the control error using the controller such that the manipulated variable, i.e. the control input to the plant, reduces or totally removes the control error [22].

Both open-loop and closed-loop control schemes are used in the industry depending on the type of system to be controlled. A feedforward controller is used in the open-loop system due to the absence of feedback, whereas other design principles for controllers are used for closed-loop systems.

As illustrated in figure 11 and figure 25, both feedforward- and feedback controllers can be used simultaneously.

An important difference between the open-loop and closed-loop systems is that closed-loop systems can counteract all disturbances/noise, but can theoretically become unstable if the control input grows to too large values. Open-loop systems on the other hand can only reject/counteract disturbances it was designed to, but can never become unstable, as long as the plant/control object is stable [22].

The controller, $\mathbf{k}$, irrespective of the class of system it will operate on, can be either linear or nonlinear.

An example of a linear control law is

$$\mathbf{u}(t) = -\mathbf{k}\mathbf{x}(t) \tag{42}$$

and an example of a nonlinear control law is

$$\mathbf{u}(t) = -\mathbf{k}(\mathrm{t})\mathbf{x}(t) \tag{43}$$

2.5 *State space*

In addition to what is already known about the states of the system, it is also important to know that the state vector $\mathbf{x}(t)$ at time t, is a set of variables which completely describes the system. A few additional comments are:

- Knowing $\mathbf{x}(t)$ at t and the future inputs of the system, is enough to determine the future response/behaviour of the system;
- A memory mechanism is not needed because the states themselves serve as the memory mechanism
- Initial conditions are state values
- The states often determine the energy in the system
- States are not unique, but are a very convenient and powerful way to fully describe the system (see also the discussion by 2.2.2).
- The length of the states, i.e. number of states used to describe the system, are not unique because unnecessary states can be included, making it important to choose states well.

When choosing the states, keep in mind that the state must be measurable either directly or indirectly, and not be redundant [23].

2.6 *Stability*

In the *linearization* section to follow, a better description will be supplied, but for now it is necessary to know that using linear models to represent systems is convenient due to existing classic control techniques making use of Laplace variables. Plainly put, there exists a lot of math to draw upon to do your controller design if your system is linear. This is also why nonlinear systems are locally linearised, if possible.

Once one has described your system in terms of Laplace variables, being it a single- or multivariable system, classic techniques can be used to design a controller, with a system transfer function and characteristic equation readily formulated.

Having your system described this way, enables it to be easily tested for stability using standard, industry-accepted methods. Examples of ensuring stability are by Eigen value checking, either by using the characteristic equation and determining its root locations, or by determining the Eigen values of the matrix representing the closed loop dynamics. Both of which were used in this study.

An example of such a standard, industry-accepted stability test is the Routh-Hurwitz stability criterion which is the '... necessary and sufficient criterion for the stability of linear systems.' according to Dorf [7: p.259]. This criterion is useful when one has a characteristic equation and want to determine the stability without actually determining the roots of the equation itself. Of course given the ability of machine computation, you could have just computed the pole locations itself; on the left side of the s-plane indicating stability, or on the imaginary axis (jw-axis) indicating marginal stability [7: p.294].

If your system was nonlinear you wouldn't have Laplace variables, eliminating classic design techniques. There are ways of using modern control techniques and still apply classic design methods, as will be shown in this study.

The main difficulty with modern control, applying itself to nonlinear representations of systems, is that there exists no standard, industry-accepted criterion for determining the controlled system's stability. There are some criterions which can be used, like special Lyapunov functions and the methods proposed during the proceedings of the 34th Conference on Decision and Control in New Orleans [25]. The problem is that they do not work for **all** cases, the Lyapunov functions only work if certain sector conditions are satisfied [24] and the methods proposed during the Conference on Decision and Control, only work for a class of nonlinear systems.

In the linear case a stability test like the Routh-Hurwitz criterion, is valid for **all** cases. Granted, the Routh-Hurwitz criterion fails to detect the instability when the jw-axis roots repeat [7: p.300].

2.7 Linearization

In order for the system's behaviour to be determined in a classic control sense, the system must be linear, or at least linear around an operating point (locally linear). This then enables the use of Laplace variables and stability tests, mentioned earlier, to be performed.

The following discussion will elucidate the concept of linearity, linear functions and linearization. The reference for the discussion is a document presented on the World Wide Web (www) by Nayden Kambouchev, translated by Karen E. Willcox [26], which is based on the work of John van de Vegte [27].

An issue to be addressed is what is meant by a linear function. This definition depends on context and the environment. For the purposes of this study, a system is linear, or locally linear, when the derivative exists, and can be determined as to apply the techniques described in [28: p.347-349] These techniques also use partial derivatives with regard to the different states, i.e. $\frac{\partial f_n}{\partial x_n}$, where f_n is the n^{th} differential equation of the system, and n is the number of states. x_n is the n^{th} state.

If the (partial) derivative if the complete system described by state variable equation does not exist for all points found within the dataset of the function, $f_n(x(t),u(t),t)$ describing the system, but does exist around certain operating points, the system is said to be *locally linearised* around those points.

To illustrate the concept of linear and nonlinear, examples of linear- and nonlinear functions for one- and two variable systems are supplied in the representations below. Noting the trend, this concept is extended to hyperspace structures for the multivariable case, not shown.

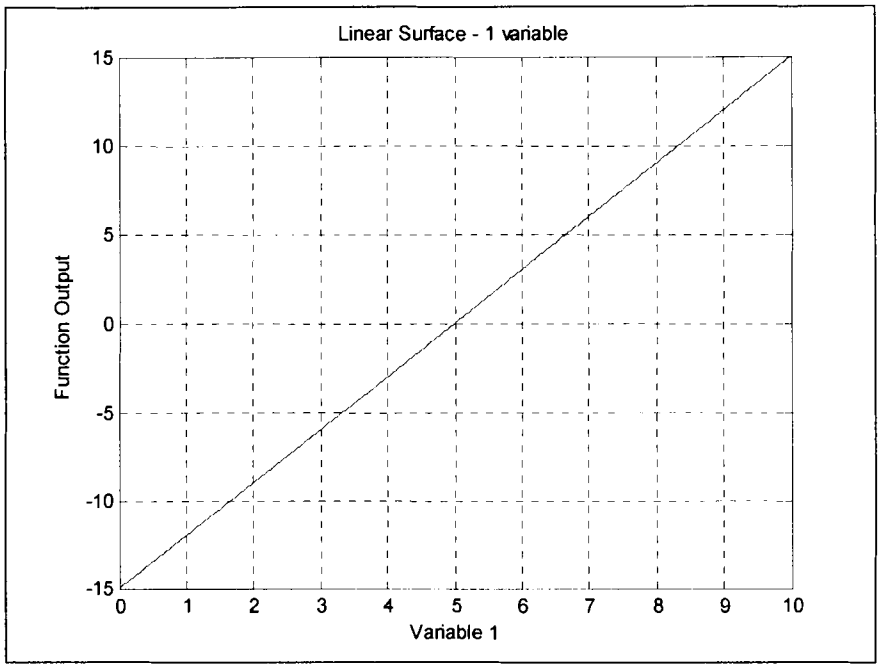

Figure 26: Geometric representation of a single variable linear function

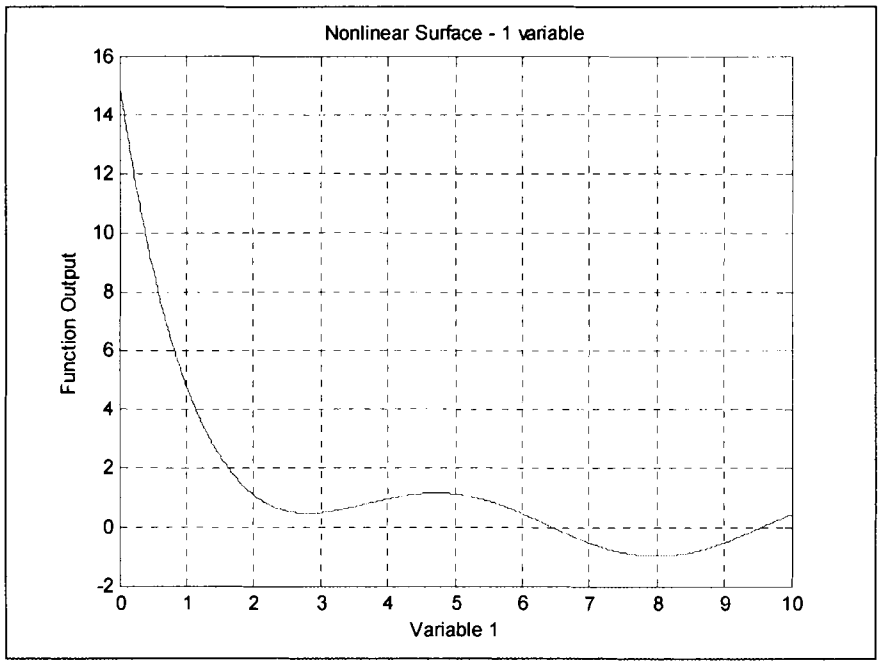

Figure 27: Geometric representation of a single variable nonlinear function

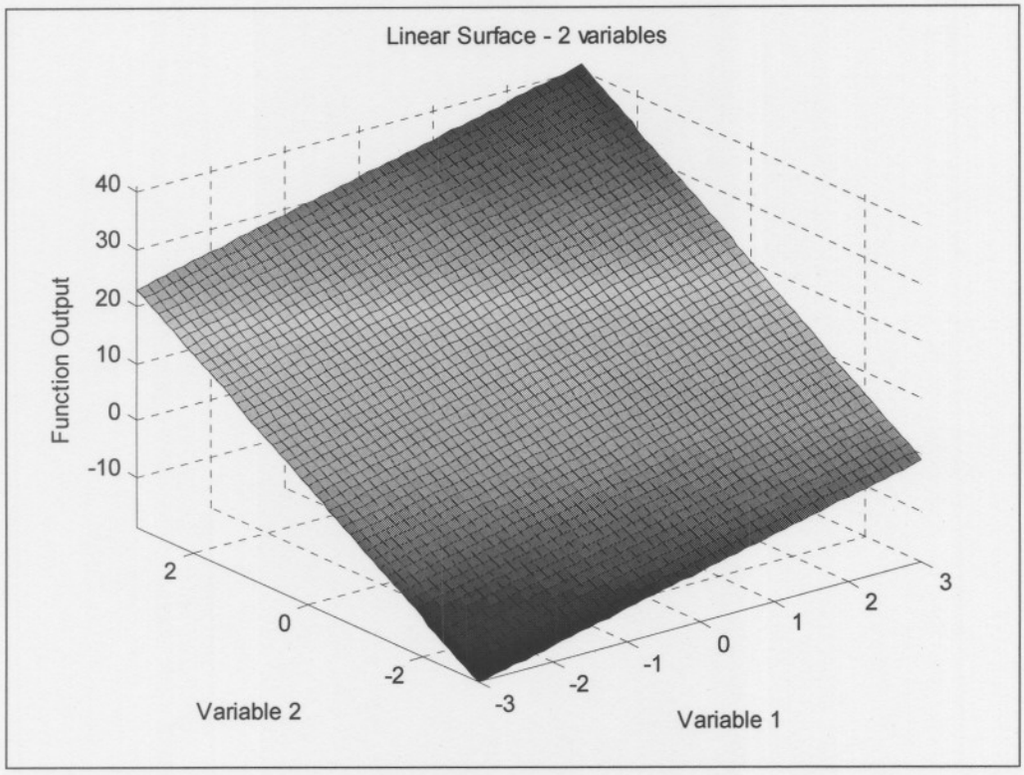

Figure 28: Geometric representation of a two- variable linear function

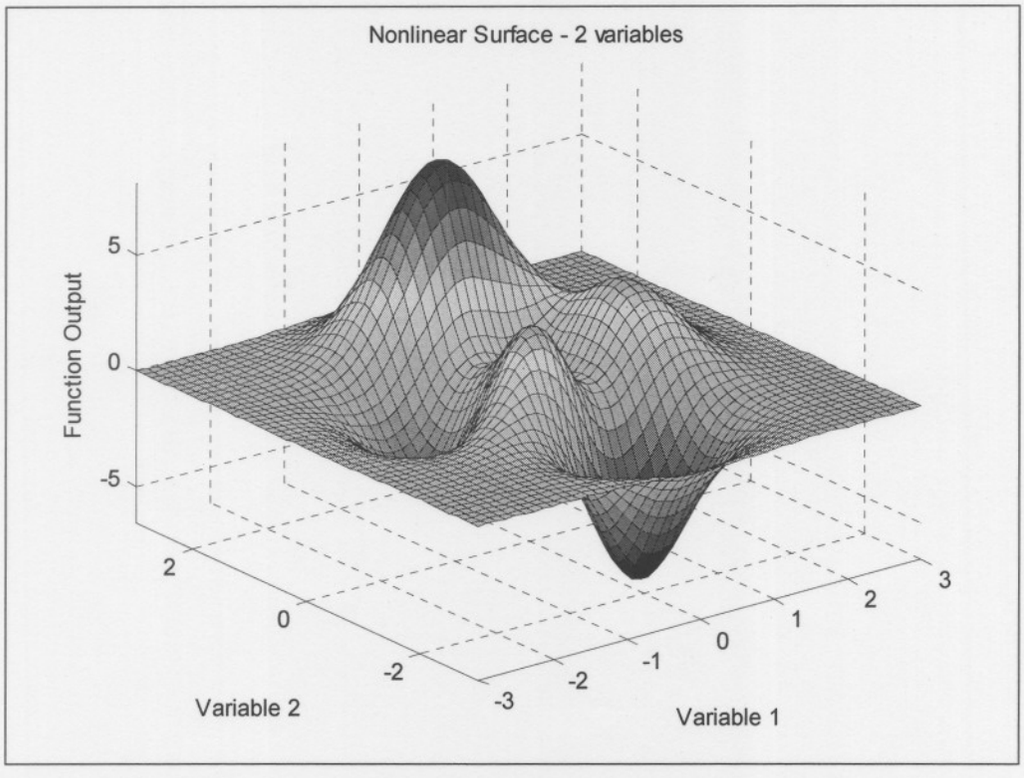

Figure 29: Geometric representation of a two- variable nonlinear function

The superposition principle, also known as the linearity principle, states that if one has a function that maps x_1 to y_1, and x_2 to y_2, then this function is linear if it also maps $ax_1 + bx_2$ to $ay_1 + by_2$. Where a and b are constants.

The Laplace transform can be proven to be linear using the *linearity theorem* stating that if $L[x_1(t)] = X_1(s)$ and $L[x_2(t)] = X_2(s)$, then $L[ax_1(t) + bx_2(t)] = aX_1(s) + bX_2(s)$.

For the interested reader, the online module presents an explanation, examples and exercises on how a function can be linearised [29] using a Taylor series expansion. For the purposes of this study it is enough to be familiar with the techniques described in [28].

From a modelling aspect, assuming that the system can be linearised or at least locally linearised, a linearised model is sometimes inadequate in describing the real-life system. Instances where this is the case is when saturation, hysteresis and/or friction occur [30].

What is often done to circumvent this problem without using a pure nonlinear model, is to add a static nonlinearity (NL) to the linearised model. This nonlinearity can then be transformed into uncertainty, causing one to have an uncertain linear system model which more accurately describes the real-life system. See the figure below using an LTI system to illustrate the idea.

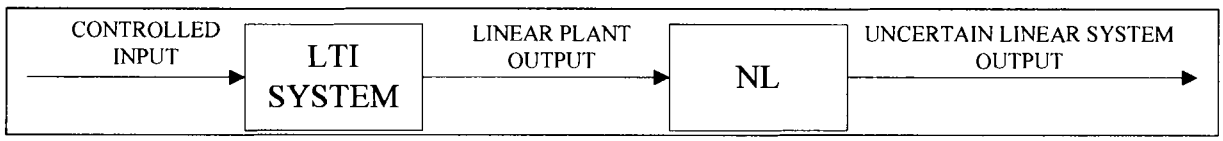

Figure 30: Wiener system model [31]

Depending on where this nonlinearity is added, different known systems follow, i.e. the Wiener model (Figure 30) using a static output NL, a Hammerstein system using a static input NL, and discrete-time bilinear models which add the NL to the state-space equations [31].

The most attractive reasons for using linear models are listed below [32]:

- Linear models are easier to compute, visualize and understand
- Their behaviour in time is deterministic

- The analysis of linear theory is complete
- Linear differential equations are a lot easier to solve

2.8 Lyapunov

The stability of a system for its zero solution can be guaranteed using a Lyapunov function. The nature of this function illustrates the system being either stable, or only marginally stable. The function is in terms of the states of the system and can readily be applied in the *time* domain. This makes it very amenable for use in optimal control theory.

A Lyapunov function, is a scalar function in the form

$$V(x_1, x_2, ..., x_n) \qquad (44)$$

In other words, it's a function in terms of the states of the system which yields a number as result. The function is defined over a region D.

Further, V is continuous, positive definite and greater than 0 for all states $\{x_1, x_2, ..., x_n\} \neq 0$. V has continuous first order derivatives at every point within $\mathcal{D}$. The derivative of V, with respect to the system $\frac{d}{dt}x_n = f_n(x_1, x_2, ..., x_n)$, in other words $V'(x_1, x_2, ..., x_n) = \left(\frac{d}{dt}V(x_1, x_2, ..., x_n)\right)_{x_n}$, is defined as the dot product $V' \equiv \nabla V.f$

The existence of a Lyapunov function, for which $V'(x_1, x_2, ..., x_n) \leq 0$ on region $\mathcal{D}$ which includes the origin, will guarantee the stability of the zero solution for $\frac{d}{dt}x_n = f_n(x_1, x_2, ..., x_n)$.

The existence of a Lyapunov function for which $V'(x_1, x_2, ..., x_n)$ is negative definite on region $\mathcal{D}$ containing the origin only guarantees the asymptotical stability of the zero-solution of the system.

The example below illustrates this.

Example

Given the system

$$\frac{dx_1}{dt} = x_2$$
$$\frac{dx_2}{dt} = -x_1 - 2x_2$$

a possible Lyapunov function is

$$V(x_1, x_2) = \frac{{x_1}^2 + {x_2}^2}{2}$$

Thus, $V'(x_1, x_2) = x_1 x_2 + x_2(-x_1 - 2x_2) = -2{x_2}^2$. As can be seen, this function is non-increasing at every location within the region $\mathcal{D}$ which includes the origin, and therefore the system

$$\frac{dx_1}{dt} = x_2$$
$$\frac{dx_2}{dt} = -x_1 - 2x_2$$

will be stable for the zero solution [33].

When basing an optimal control theorem on a Lyapunov function it is instructive and flexible enough to be applied to linear time-varying systems and systems with constraints. [1: p.452] The Salukvanze method for optimal control is such an approach. Takahashi [1: pp. 452-455] discusses this approach in detail and supplies an example as well.

Salukvanze's approach also makes use of the maximum principle of Pontryagin, which is a necessary but not sufficient condition to be satisfied during optimal control theory [1: p.629].

3 MIMO, state space, nonlinear control theory

Using the basis supplied in the preceding chapter, a methodology is derived for the multivariable control of MIMO, systems which may be nonlinear, using PI controllers.

3.1 *Introduction*

All the elements of control theory relevant to this study now noted, a multivariable, nonlinear system represented with state-variable equations will be linearised, a multivariable controller using PI controllers developed, and the response of the closed loop system simulated using MATLAB® and SIMULINK®.

3.2 *Design methodology*

Knowing the advantages of using states to represent a system, first order differential equations are used to model a multivariable system, yielding the so called state-variable equations.

The state-space model of the system has then got to be,

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{u}) \\ \mathbf{y} &= \mathbf{h}(\mathbf{x},\mathbf{u}) \end{aligned} \qquad (45)$$

where

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_n \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ . \\ . \\ . \\ u_m \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_p \end{bmatrix},$$

$$\mathbf{f}(\mathbf{x},\mathbf{u}) = \begin{bmatrix} f_1(x_1, x_2, ..., x_n, u_1, u_2, ..., u_m) \\ f_2(x_1, x_2, ..., x_n, u_1, u_2, ..., u_m) \\ . \\ . \\ . \\ f_n(x_1, x_2, ..., x_n, u_1, u_2, ..., u_m) \end{bmatrix}$$

$$\mathbf{h}(\mathbf{x},\mathbf{u}) = \begin{bmatrix} h_1(x_1, x_2, ..., x_n, u_1, u_2, ..., u_m) \\ h_2(x_1, x_2, ..., x_n, u_1, u_2, ..., u_m) \\ \vdots \\ h_p(x_1, x_2, ..., x_n, u_1, u_2, ..., u_m) \end{bmatrix}.$$

n is the number of states, m the number of controls and p the number of measured outputs.

Equation (44) contains all the states of interest.

Considering the system is controlled around a range of operating points, this system is then linearised using the techniques described by Ljung [28] to yield the state-space matrices

$$\mathbf{A} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \qquad (46)$$

$$\mathbf{b} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_n} \\ \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_n} \end{bmatrix} \qquad (47)$$

Each element of $\mathbf{A}$ must be in terms of the states only, and $\mathbf{b}$ in terms of only the controls.

Keeping in mind, that one seeks to design a nonlinear controller for the multivariable system, the next step is to take into consideration the physical constraints of the system, and to determine the admissible controls and admissible trajectories, i.e. the operating range of the system and controllers.

The purpose of this is basically to have separate controllers for each operating point in a similar fashion as a multivariable function of n variables has different values for different combinations of variables; the function value is the specific controller parameters, and the n variables the states of the system. The controller parameters change with respect to the operating points as the function values would change with respect to its variables.

To enable this, a matrix is generated with all the relevant controller parameters with regard to the operating points. The system controller matrix $\mathbf{K}$, developed using the linearised model above, thus varies depending on the operating point, and is therefore nonlinear. The stability over the range of operating points can be determined in a classic control sense, eliminating suspicions of non-deterministic behaviour often associated with nonlinear control. In this respect, $\mathbf{K}$ can be seen as an adaptive controller using gain- or PI -scheduling.
To illustrate how the admissible trajectories are determined is for example; a water tank has a minimum water head of zero, and a maximum water head of 14 centimetres, its allowable values will have to be within these bounds. However, the values the states *can* be, is not necessarily its operating range. Other than the desired state values, the system's operating range (all admissible trajectories) is also influenced by the physical make-up of the system and the allowable values of the controls.

One can imagine that for large operating ranges this is not an efficient way of determining the operating range of the system, but if the combination of states chosen to determine operating range is chosen smartly, it becomes efficient. The speed of machine computation aids in this method.

In order to calculate $\mathbf{K}$, one has to first consider the design principle. There was decided to use the series/cascade principle from figure 1.

For this design principle to work, the reference vector, i.e. operating points/desired state values, must be made zero. Obviously, the user will set a reference input which is not necessarily zero. Thus, from a design point-of-view it has to be controlled around zero ($\mathbf{R} = \mathbf{0}$) irrespective the user's desired operating points. Yet, the system itself must be controlled around the user's inputs. Consequently, the simulation of the system in MATLAB® must be adjusted in such a way as to enable this.

The illustration below shows how this can be achieved.

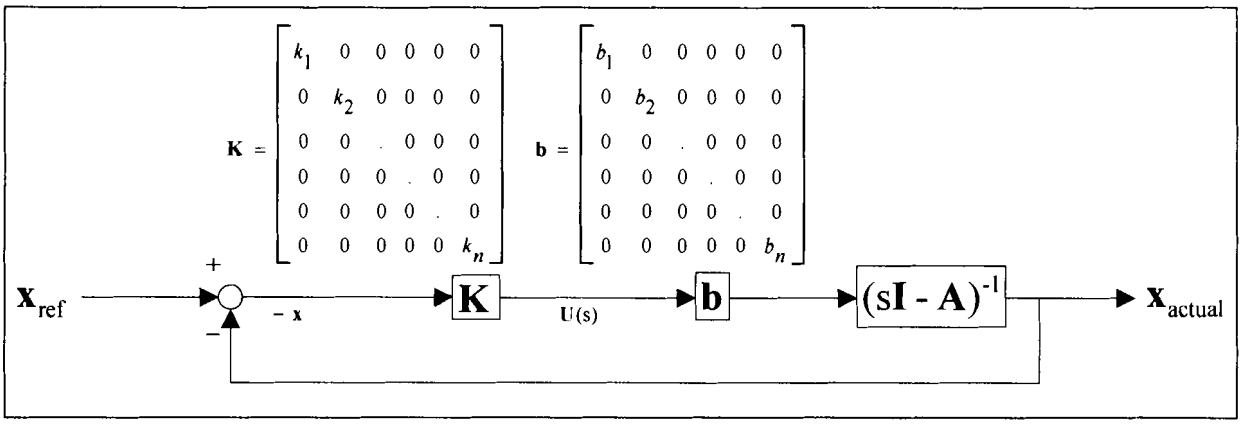

Figure 31: Feedback control

$\mathbf{I}$ is the $n \times n$ identity matrix.

The block diagram of the controlled system now established, together with matrix $\mathbf{A}$, $\mathbf{b}$, and the admissible trajectories, the closed-loop characteristic polynomial can be derived, and controller parameter values generated.

The modified block diagram (Figure 31) is then used with the system's state-variable equations and a model of the controlled system created in SIMULINK®.

The derivation of the closed-loop characteristic polynomial now follows:
According to [1: p. 422] the first step is to define the controller, $\mathbf{k}$, as the column vector

$$\mathbf{k} = \begin{bmatrix} k_1 \\ k_2 \\ \cdot \\ \cdot \\ \cdot \\ k_n \end{bmatrix} \qquad (48)$$

$\mathbf{k}' = [k_1, k_2, ..., k_n]$.

The scalar controlling input is thus given by

$$u(t) = -\mathbf{k}'\mathbf{x}(t) \tag{49}$$

Having derived matrices **A** and **b** earlier, the system can be represented as

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t) \tag{50}$$

By substituting the scalar controlling input into the equation directly above, one gets

$$\frac{d\mathbf{x}(t)}{dt} = (\mathbf{A} - \mathbf{b}\mathbf{k}')\mathbf{x}(t) \tag{51}$$

and the closed-loop characteristic polynomial can then be described as

$$CLCP = \Delta_c(s) = \det(s\mathbf{I} - \mathbf{A} + \mathbf{b}\mathbf{k}') \tag{52}$$

A is an $n \times n$ matrix. For the Δ_c to be calculated, the dimensionality of *CLCP* must be consequent. With $\mathbf{k}'$ already defined as an $1 \times n$, it follows that

$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{bmatrix} \tag{53}$$

$$\mathbf{b}\mathbf{k}' = \begin{bmatrix} b_1k_1 & b_1k_2 & \cdot & \cdot & \cdot & b_1k_n \\ b_2k_1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ b_nk_1 & b_nk_2 & \cdot & \cdot & \cdot & b_nk_n \end{bmatrix} \tag{54}$$

The problem is, when controlling a multivariable system very close to its limit, it tends to become *singular*. A brief discussion will now follow on why this is of importance. The discussion starts with the definition of *singular*.

Definition 3.1

'An $(n \times n)$ matrix $\mathbf{A}$ is non-singular if the only solution to $\mathbf{Ax} = \mathbf{0}$ is $\mathbf{x} = \mathbf{0}$. Furthermore, $\mathbf{A}$ is said to be singular if $\mathbf{A}$ is not non-singular.' [34: p. 76].

Definition 3.2

'A set of m-dimensional vectors $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_p\}$ is said to be *linearly independent* if the only solution to the vector equation

$$a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + ... + a_p\mathbf{v}_p = \mathbf{0}$$

is $a_1 = 0$, $a_2 = 0, ... a_p = 0$. The set of vectors is said to be *linearly dependent* if it is not linearly independent. That is, the set is linearly dependant I we can find a solution to $a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + ... + a_p\mathbf{v}_p = \mathbf{0}$ where not all the a_i are zero.' [34: p. 72]

Theorem 3.1

'The $(n \times n)$ matrix $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_n]$ is non-singular if and only if $\{\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_n\}$ is a linearly independent set.' [34: p. 76]

Theorem 3.2

'Let $\mathbf{A}$ be an $(n \times n)$ matrix. The equation $\mathbf{Ax} = \mathbf{b}$ has a unique solution for every $(n \times 1)$ column vector $\mathbf{b}$ if and only if $\mathbf{A}$ is non-singular.' [34 p. 77]

Theorem 3.3

'Let $\mathbf{A}$ be an $(n \times n)$ matrix. Then $\mathbf{A}$ is singular if and only if $\det(\mathbf{A}) = 0$' [34: p.236]

From this, $\mathbf{bk}' = [\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_n]$ will be non-singular if and only if $\{\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_n\}$ is a linearly independent set. Alternatively, $\mathbf{bk}'$ will be non-singular if the only solution to $(\mathbf{bk}')\mathbf{x} = \mathbf{0}$ is $\mathbf{x} = \mathbf{0}$ or the determinant of $\mathbf{bk}'$ is not equal to zero.

For our system, by theorem 3.2, if the resulting matrix that follows from the bracketed term of equation (51) is singular, i.e. $(\mathbf{A}-\mathbf{bk}')$ is singular, there either does not exist a solution for equation (51), or the solution is not unique [35]. This creates a problem when attempting to simulate the system response.

Otherwise stated, if the bracketed term of $CLCP$ of equation (52) is singular, the $CLCP = 0$.

The abovementioned singularity tends to occur when $\mathbf{bk}'$ looks like equation (54). In order to simplify matters, the system controller matrix $\mathbf{K}$ is chosen to be

$$\mathbf{K} = \begin{bmatrix} k_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & k_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & . & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 \\ 0 & 0 & 0 & 0 & 0 & k_n \end{bmatrix} \tag{55}$$

This induces

$$\mathbf{b} = \begin{bmatrix} b_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & b_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & . & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 \\ 0 & 0 & 0 & 0 & 0 & b_n \end{bmatrix} \tag{56}$$

which then results in

$$\mathbf{bK} = \begin{bmatrix} b_1k_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & b_2k_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & . & 0 & 0 & 0 \\ 0 & 0 & 0 & . & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 \\ 0 & 0 & 0 & 0 & 0 & b_nk_n \end{bmatrix} \tag{57}$$

This is in concord with the required dimensionality of $CLCP$.

Equation (52) then becomes

$$CLCP = \Delta_c(s) = \det(\mathbf{sI} - \mathbf{A} + \mathbf{bK}) \qquad (58)$$

A further motivation for choosing **K** as indicated, is when looking at the control law used to control the system

$$\mathbf{u}(t) = -\mathbf{K}\mathbf{x}(t) \qquad (59)$$

The result is that each controller controls each state independently, i.e.

$$\mathbf{U}(s) = \begin{bmatrix} u_1 \\ u_2 \\ \cdot \\ \cdot \\ \cdot \\ u_n \end{bmatrix} = \begin{bmatrix} -k_1 x_1 \\ -k_2 x_2 \\ \cdot \\ \cdot \\ \cdot \\ -k_n x_n \end{bmatrix} \qquad (60)$$

which, if possible, is ideal.

Remember, because the system is designed as if controlled around zero, **x**(t) from equation (59) translates into deviation from the **desired** value in the controlled system.

Though the control law has the form of (59), it is implemented as illustrated in figure 31.

Now that one has an expression for Δ_c in terms of the controller parameters, (58), one can use pole placement techniques to solve this equation for the chosen admissible trajectories.

The operating point-dependant controller parameters now known, a function is generated which basically maps the operating point, i.e. specific combination of state values, to the corresponding controller parameters. The way this is done in MATLAB® is by using a lookup table or –matrix, which functions in exactly the same way as a multivariable nonlinear function would.

When a combination of states occur that was not used to build the lookup matrix during the design, the surrounding controller parameters in the matrix is used to interpolate a relevant controller parameter value.

The controller design is now complete; $\mathbf{K}$, comprising of a set point dependant controller k_n for each desired reference input, resulting in control signal $\mathbf{U}(s)$ as illustrated above.

The nonlinear controller $\mathbf{K}$, is then implemented in the model of the system in SIMULINK® using both MATLAB® and SIMULINK®. This model is of the **original system represented by state space equation**. The results from the simulation of the controlled system response include graphs of the state values, and graphs of the control signal output.

3.3 Additional design notes

The way the admissible controls and –trajectories were determined in the study for the *proportional controller* matrix $\mathbf{K}$, was by finding the combination of states physically achievable, then determining the necessary controls needed to have the system follow these trajectories, and if these controls fell within the allowable range, it was considered an admissible trajectory. The controls causing the admissible trajectories were selected as admissible controls.

For the *PID controller* matrix $\mathbf{K}$, the physically achievable states were taken as the admissible trajectories, and the corresponding controls as the admissible controls.

By only using desired state values known not to require controls larger than their limit, one effectively has two measures that guards against the control $\mathbf{U}$ becoming infinitely large.

1. As long as initial assumptions hold for the maximum deviation, the bounds of the control will never be exceeded.

   Consider the control law used

$$\mathbf{U}(t) = -\mathbf{K}\mathbf{x}(t) \qquad (61)$$

By knowing the limit of the controlling input $\mathbf{U}(t)$, and designing for a maximum deviation $\mathbf{x}(t)$, the ability exists to discard trajectories which will require a control greater than the allowed value.

Unfortunately, what occurs is that this maximum deviation does hold around an operating point, but as soon as the operating point is changed by the user, **this value is exceeded**.

To illustrate this consider a system in which the absolute maximum deviation around the operating point is 0.5 centimetres, and a tank is set to be controlled around half-full, 7 centimetres. For an operating point of 7, the corresponding controller is selected and the control bounds never exceeded given the deviation remains as per design. Even if the system was controlled exactly on 7 cm, implying a zero error/deviation, if the operating point was now changed to 10 cm, the error at this instant would be 3cm, six times larger than the design. $\mathbf{K}$ will not be six times less for the new set point, because it was designed assuming a deviation of 0.5, causing the much larger error combined with the new controller parameters, to most probably generate a control much larger than it was supposed to be.

One possible solution is to design the maximum deviation equal to the largest jump from one set point to the next. This is unwise because it would set the threshold for the possible controllers' parameters much less than necessary according to the control law, causing many trajectories to be discarded. This will decrease the operating range of the system.

That is the reason why the control $\mathbf{U}$, should be determined according to mechanical en budgetary constraints and the deviation preferably be made as small as realistically possible.

The problem due to changing set points which cause the control to be higher than the designed value, can be compensated for by means of a saturation circuit.

2. Saturation Circuit

   Nothing other than a clipping circuit, a saturation circuit holds the control $\mathbf{U}$ equal to its maximum designed value should it exceed it.

From the discussion above, it is clear that a saturation circuit is always required. For this reason and to remain true to the controlled system's real-life response, a saturation circuit was included in the simulation.

It is also the reason why, in the case of the PI controller matrix, *all* the admissible trajectories and corresponding controllers were taken, unlike the case of the proportional controller matrix.

When using an integrator in the control system, a situation can arise referred to as *reset windup*. Essentially, the integrator needs time to recover from fast changing errors, or sudden large errors. To explain this, consider a positive error. The integrator starts an integrating action in that (positive) direction to compensate (See figure 32). Overshoot then occurs, causing the error to change sign. The direction of the control cannot change immediately if the result of the previously positive integrating action dominates. This causes prolonged overshoot until the integrating action in the negative direction is such that the control can become negative. The same is true for a negative error. The figure below illustrates the phenomenon assuming an initial desired value of zero, which is then changed to one [36].

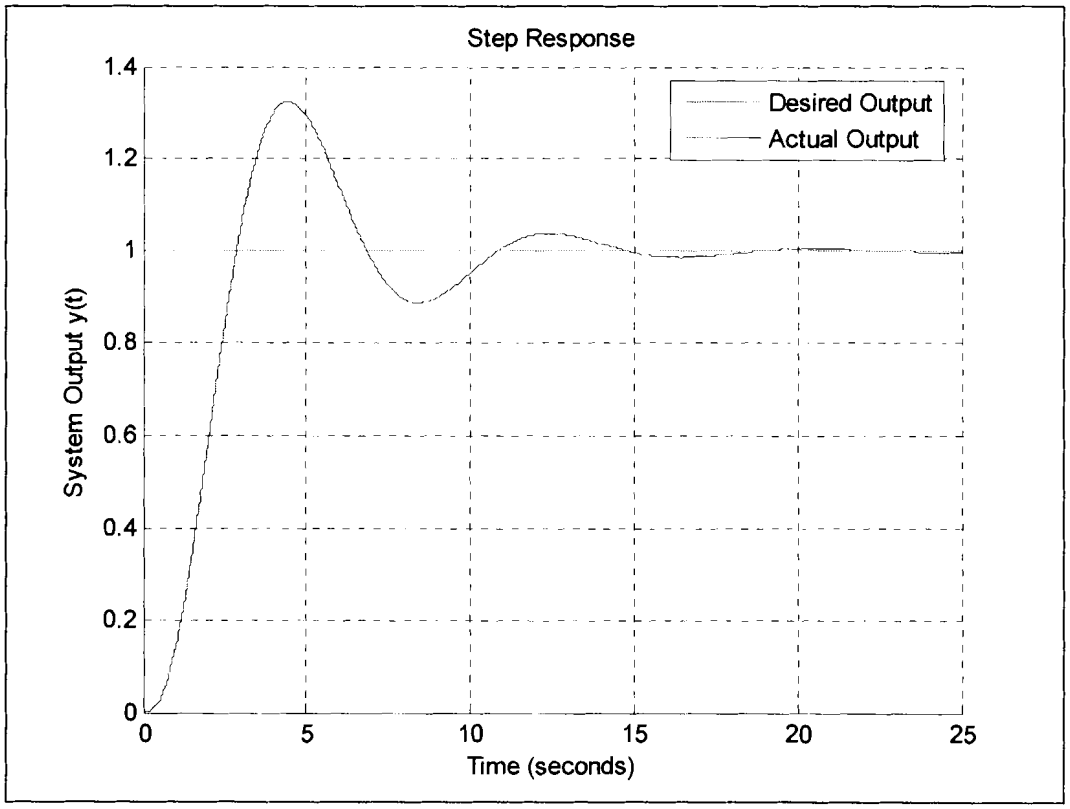

Figure 32: Reset windup

If reset windup does occur, a possible method to speed up the integrator's response and eliminate the overshoot, is by clearing the integrator's previously accumulated value, i.e. its 'memory'/to-date contribution, to zero. This anti-reset windup code is included in the simulation, and results in the next chapter will also illustrate the occurrence of reset windup as well as the effect of implementing the anti-reset windup algorithm. The figure below shows the algorithm as implemented in SIMULINK®.

3.4 Design methodology summary

The design methodology can be reduced to the following steps:

1 Model system using state-space equations;

2 Linearise state-space model around operating points;

3 Determine admissible trajectories, i.e. physically achievable combinations of the state variables;

4 Decide on design principle to be followed

5 Validate design principle and derive block diagram of the controlled system;

6 Derive closed-loop characteristic polynomial in terms of controller parameters;

7 Calculate controller parameter values using the linearised model of the system – apply pole placement techniques

8 Generate lookup table/-matrix containing controller parameter values and corresponding operating points;

9 Specify lookup table/ -matrix's interpolation method;

10 Implement controller in simulation of controlled system. The simulation must be set up using the original system model.

4 Design methodology application - experiments

Three experiments are set up in MATLAB® and SIMULINK®. The methodology developed in the previous chapter is then applied to each of these experiments and the systems controlled around various operating points. The results of the controlled systems are then supplied, intended to validate the design methodology.

4.1 Introduction

To enhance the readability and understanding of the results, the discussion of the results is presented directly after the relevant figures.

Three experiments are performed to validate and verify the developed methodology of the previous chapter.

The first experiment is a showcase of the application of the design methodology, focussing especially on the different controller types. The different types of controllers that are used are P and PI controllers for both linear and nonlinear cases, as well as an optimal linear PI controller.

The second experiment illustrates its versatility by illustrating a situation when the state-variable equations are absent due to it being a black box model.

In conclusion, experiment 3 focuses more on the derivation of a state-variable model for the case where the state-variable equations are absent, but can be determined by deriving a physical model of the system. Consequently, less is focused on the behaviour of different types of controllers, as this aspect of the design methodology was already covered in the first experiment.

4.2 Experiment 1: MIMO water level control for two interconnected tanks

4.2.1 Experiment description

The figure below illustrates the plant (second order system) that requires the liquid level control system.

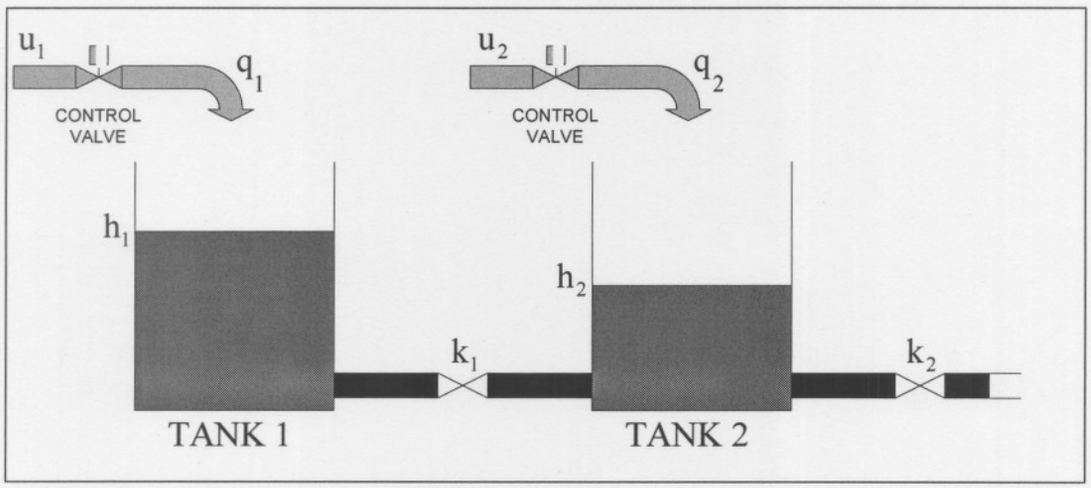

Figure 33: Liquid level control system [14]

The liquid in each tank is to be controlled at a specified height within the system's operating range indicated by h_1 and h_2. This will be achieved by controlling each tank's inflow independently, indicated on the figure by q_1 and q_2. These inflows, only allow flow in one direction, i.e. **into** the respective tank. In turn, these inflows are controlled by the respective control valve's position. The position of each control valve depends on the value of the controls, u_1 and u_2.

The controller is designed using the required flow values to maintain the correct water heads, i.e. **the design assumes that the controller outputs are q_1 and q_2.** However, from the figure this is not the case; the outputs should be u_1 and u_2. Due to this being an academic investigation, **assume** that the actual controller outputs are the inflows into the respective tanks, i.e. $\mathbf{u_1 = q_1}$ and $\mathbf{u_2 = q_2}$.

For implementation in a real-life SCADA system, the controller outputs, u_1 and u_2, will be scaled to the values the actuator needs to actuate the required inflows. In the figure, the actuator is a control valve.

k_1 and k_2 are the pipe admittance coefficients.

For the controller design and controlled system simulation, the setup from M.K. Khan and S.K. Spurgeon in the article: 'ROBUST MIMO water level control in interconnected twin-tanks using second order sliding mode control' [14], was used. According to this, the areas for both tanks were taken to be same, i.e.

$$A_1 = A_2 = A$$

Further, the area and the pipe admittance coefficients were determined via calculation and experimental setup to be

$$A = 155.44cm^2$$
$$k_1 = 23.45$$
$$k_2 = 17.62$$

4.2.2 Controller design

According to the derived design methodology in the previous chapter, the first step is to determine the system's state-variable equations and represent the system in the correct state-space form. Using equations 24a and 24b from [14], we have

$$\dot{h}_1 = -\frac{k_1}{A_1}\sqrt{|h_1 - h_2|}sign(h_1 - h_2) + \frac{1}{A_1}q_1$$
$$\dot{h}_2 = \frac{k_1}{A_2}\sqrt{|h_1 - h_2|}sign(h_1 - h_2) - \frac{k_2}{A_2}\sqrt{h_2} + \frac{1}{A_2}q_2$$

By substituting the known values, becomes these state-variable equations become

$$\dot{h}_1 = -0.1509\sqrt{|h_1 - h_2|}sign(h_1 - h_2) + 0.0064q_1$$
$$\dot{h}_2 = 0.1509\sqrt{|h_1 - h_2|}sign(h_1 - h_2) - 0.1134\sqrt{h_2} + 0.0064q_2$$

which are then taken and linearised around the operating points yielding

$$\mathbf{A} = \begin{bmatrix} \frac{-0.1509}{2}(h_1 - h_2)^{-\frac{1}{2}} signum(h_1 - h_2) & \frac{0.1509}{2}(h_1 - h_2)^{-\frac{1}{2}} signum(h_1 - h_2) \\ \frac{0.1509}{2}(h_1 - h_2)^{-\frac{1}{2}} signum(h_1 - h_2) & \left(\frac{-0.1509}{2} * (h_1 - h_2)^{-\frac{1}{2}} signum(h_1 - h_2)\right) - \frac{0.1134}{2} h_2^{-\frac{1}{2}} \end{bmatrix}$$

$$\mathbf{b} = \begin{bmatrix} 0.0064 & 0 \\ 0 & 0.0064 \end{bmatrix}$$

The states to be controlled are the water heads of tank 1, and tank 2, h_1 and h_2.

In order to perform the next step, i.e. determine the acceptable state values, the system constraints have to be specified.

It was decided to have the state values of tank 1 and tank 2 vary between 0 cm and 14 cm.

$$0 \le h_1 \le 14$$
$$0 \le h_2 \le 14$$

Further, the inflows cannot be negative, thus

$$q_1 \ge 0$$
$$q_2 \ge 0$$

To determine whether an inflow is negative, the inflows' governing equations

$$q_1 = k_2\sqrt{h_2 - BP} - k_1\sqrt{|h_1 - h_2|}signum(h_1 - h_2)$$
$$q_2 = k_2\sqrt{h_2 - BP} - q_2$$

are used. BP, is a variable used to simulate backpressure at the outflow of tank 2. It functions in a similar manner as the effect a fictitious third tank would have on the system.

A value of

$$q_1 = 330$$
$$q_2 = 150$$

was chosen as the maximum values for the inflows to *illustrate the concept of a limited control.*

The admissible trajectories were determined using MATLAB® and the system constraints as described above.

The controlled system was designed in accordance with Figure 31, yielding.

$$\mathbf{K} = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}$$

and the closed-loop characteristic polynomial of

$$CLCP = \Delta_c(s) = \det(s\mathbf{I} - \mathbf{A} + \mathbf{bK})$$

with

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

For the desired system behaviour, the poles were arbitrarily chosen to be $p_1 = -1$ and $p_2 = -3$. The desired characteristic equation must then be

$$CLCP_{desired} = (s + p_1)(s + p_2) = s^2 + 4s + 3$$

Using MATLAB®, the controller parameters for specific set points are determined across the operating range of the system.

The results are loaded into the lookup table, specifying linear interpolation between set points not exactly calculated during the design.

The design now complete, **K** is implemented in the simulation of the plant.

4.2.3 Test scenarios

Given the control strategies as described in chapter 1 page 17, we would like to control the system for the following scenarios:

a) The height of one tank is kept the same, while changing the other. This is done for both tanks;

b) Both tanks are controlled at the same height;

c) Both tanks' heights are changed simultaneously to different values from one another;

d) A constant error is introduced to each tank at a given time. This error will be in the form of a leak which occurred in one tank. The other tank will have some addition to its volume, like a small ball bearing which is accidentally dropped in.

*To exact this, a **reference signal** was generated which takes the system through various operating points such to include all the **scenarios** as listed above.*

The reference signal was constructed using the values in the underlying table:

Time (seconds)	X1	X2
0-40	4	3
40-70	5	4
70-100	8	6
100-130	12	8
130-160	12	9
160-190	11	8
190-220	12	8
220-250	13	13

Time (seconds)	X1	X2
250-270	3	2
270-1400	14	13

Table 2: Reference values and transition times – experiment 1

4.2.4 Results

The three control strategies listed in chapter 1 page 16, are now implemented on the different test scenarios.

All the results to follow were obtained using the **design parameters** as listed below:

Position of pole 1: -1;

Position of pole 2: -3;

Maximum inflow tank 1: 330;

Maximum inflow tank 1: 150;

Maximum constant error value tank 1: -0.7 (example a leak);

Maximum constant error value tank 2: 1 (example a ball bearing dropped in);

The **simulation parameters** for **all** results below are:

Simulation time in seconds $t_0 = 0$;

Simulation time in seconds $t_f = 1400$;

Reset windup threshold tank 1: 0.9;

Reset windup threshold tank 2: 20;

Time error introduced to tank 1: 470 seconds;

Time error introduced to tank 1: 570 seconds;

Note:

Figures where the legend is omitted, is to improve visibility of the results. In all omitted cases, the red graph represents the signal which is not limited (no saturation), the blue graph the signal which is limited (saturation) at some point, i.e. at the maximum flow values, and the magenta graph the reference signal

Mention has to be made that the period just after the introduction of the last error, to the end of the simulation was used to calculate the steady-state error as it appears in the legend of the figures.

The caption of "Same-height response", refers to test scenario (a), as supplied above.

The operating range of the system is illustrated in the figure. The designation of X1 refers to the height of tank 1, and X2 the height of tank 2 in centimetres.

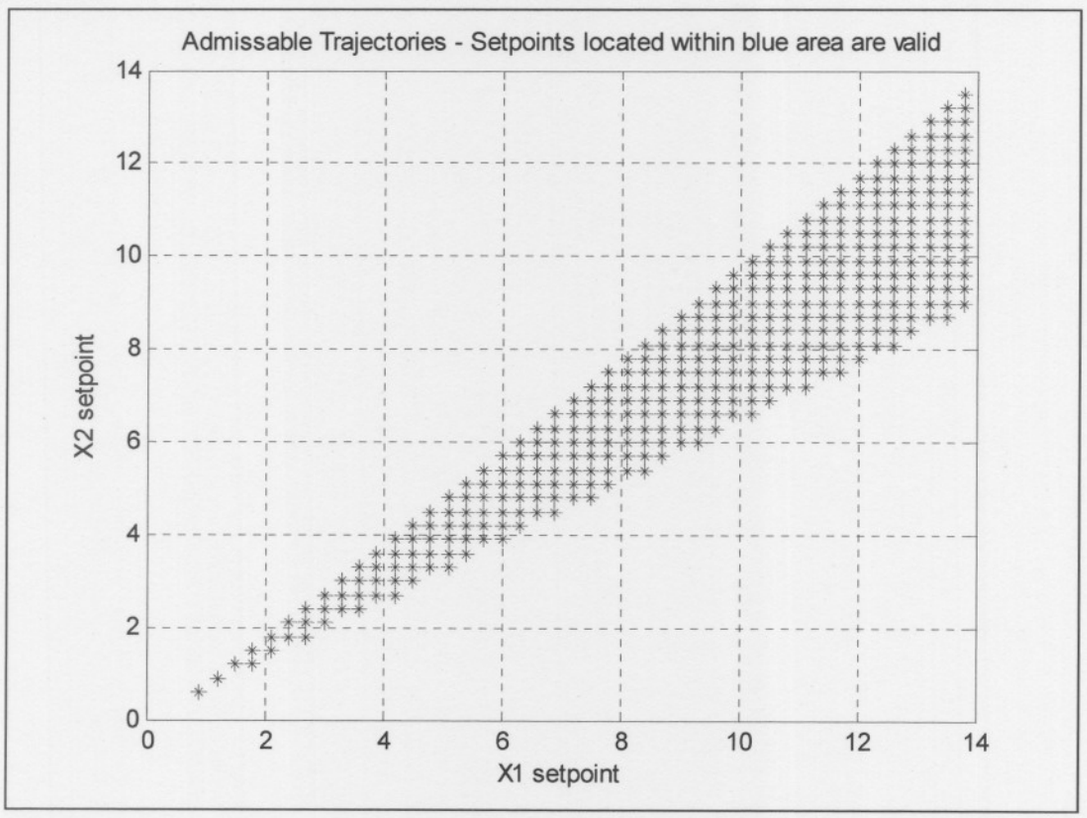

Figure 34: Experiment 1 - operating range

The two figures below, indicate how the P-controller's parameters vary for the different sets of operating points. This information was used to build each tank's controller lookup table. Leaving us with a set point dependant controller for each tank k_1 and k_2, and thus nonlinear controller

$$\mathbf{K} = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}$$

In this case both k_1 and k_2 are proportional gains, in the second control strategy $k_1 = k_{P1} + \frac{k_{I1}}{s}$ and $k_2 = k_{P2} + \frac{k_{I2}}{s}$. In the third control strategy using the genetic algorithm, a single set of proportional and integral constants were tuned for the entire operating range using $k_1 = k_{P1}e_1(t) + k_{I1}\int_0^t e_1(t)dt$ and $k_2 = k_{P2}e_2(t) + k_{I2}\int_0^t e_2(t)dt$, $e(t)$ representing the error at time t. In the case of the GA, the reason the constants were not tuned for each set point in the illustrated (figure 33) operating range is due to the expense of computation.

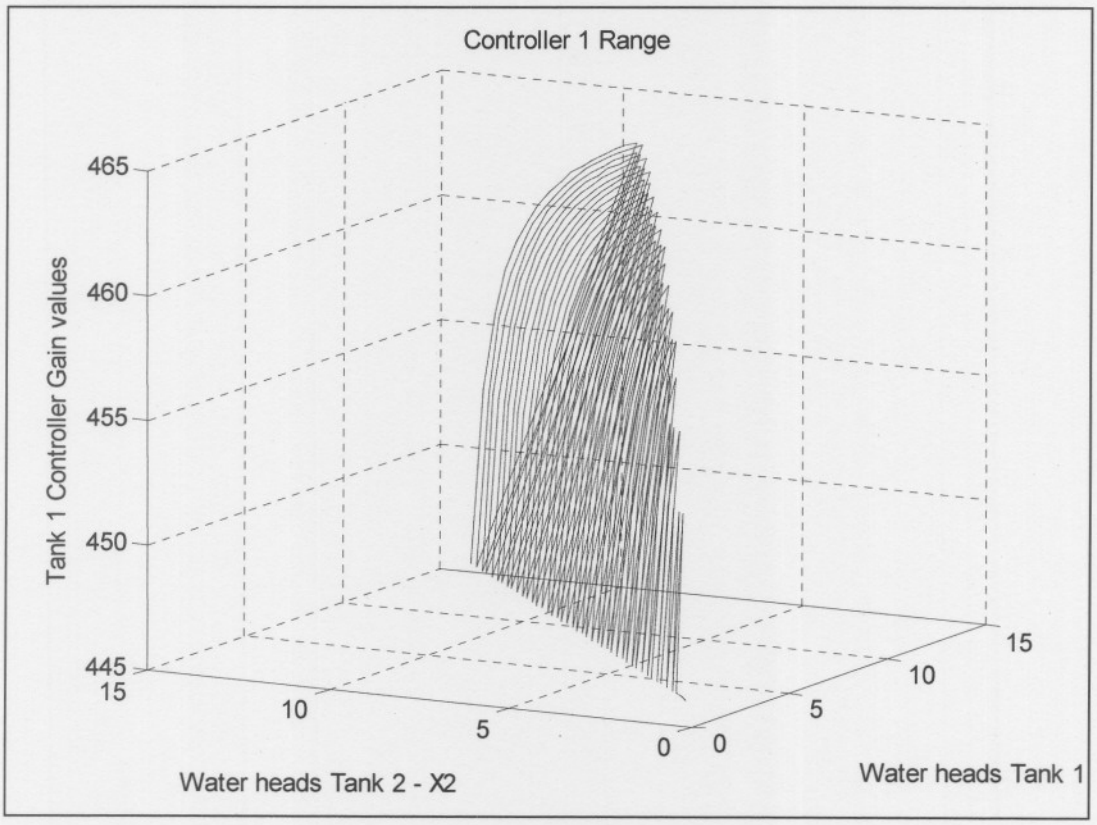

Figure 35: Tank 1 controller parameter range

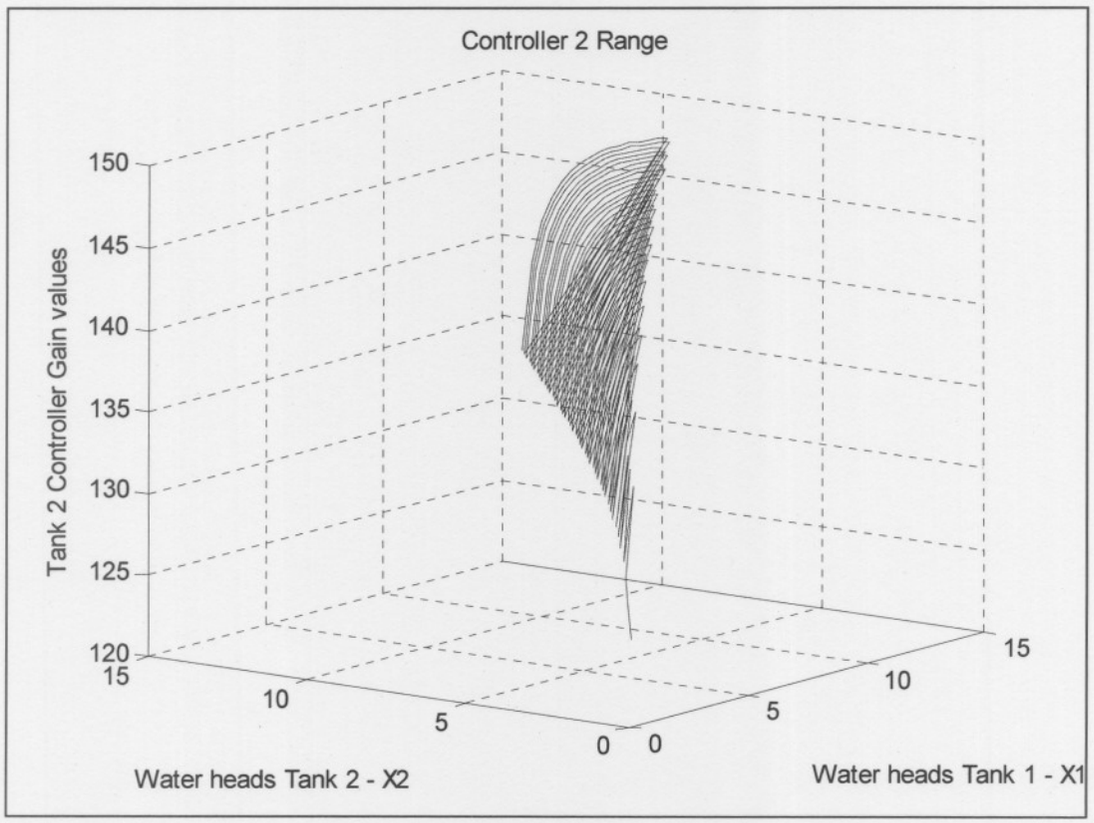

Figure 36: Tank 2 controller parameter range

Before the nonlinear system's response to the reference signal is given, the design will be **validated** using the linearised system. The controller is to control tank 1 at 12 cm and tank 2 at 10 cm. Over the simulation period, a leak is introduced at 470 seconds to tank 1, and an eraser dropped in at 490 seconds into tank 2. Because the response will be shown using the linearised system, the system should be controlled exactly, and noise rejected completely, for the desired values if the design functions properly.

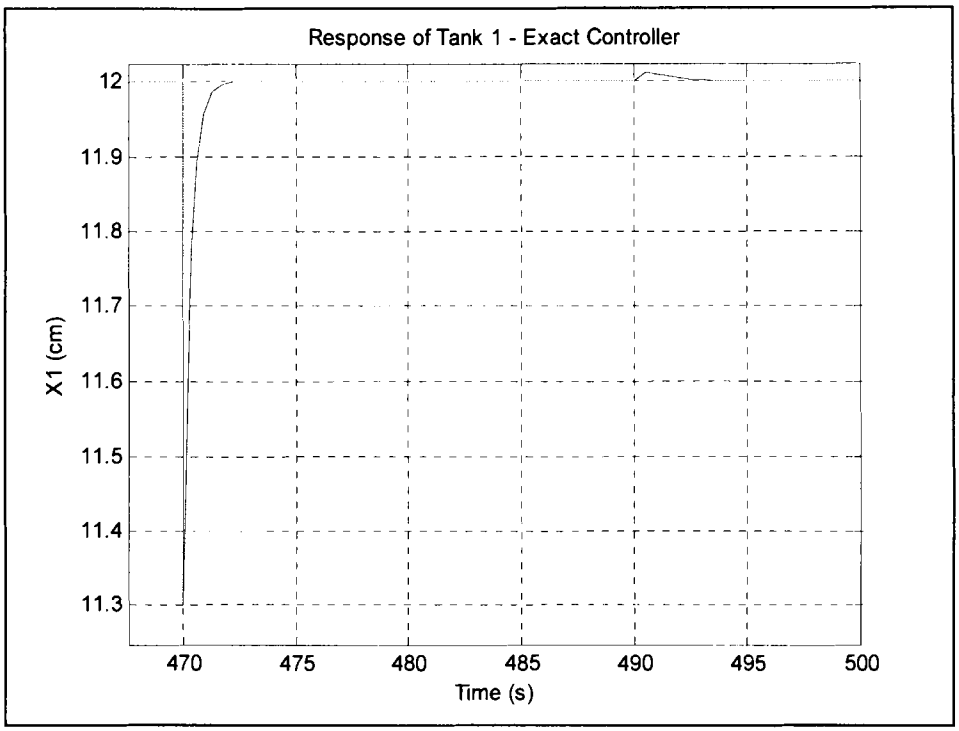

Figure 37: Experiment 1 - tank 1 response – linearised system

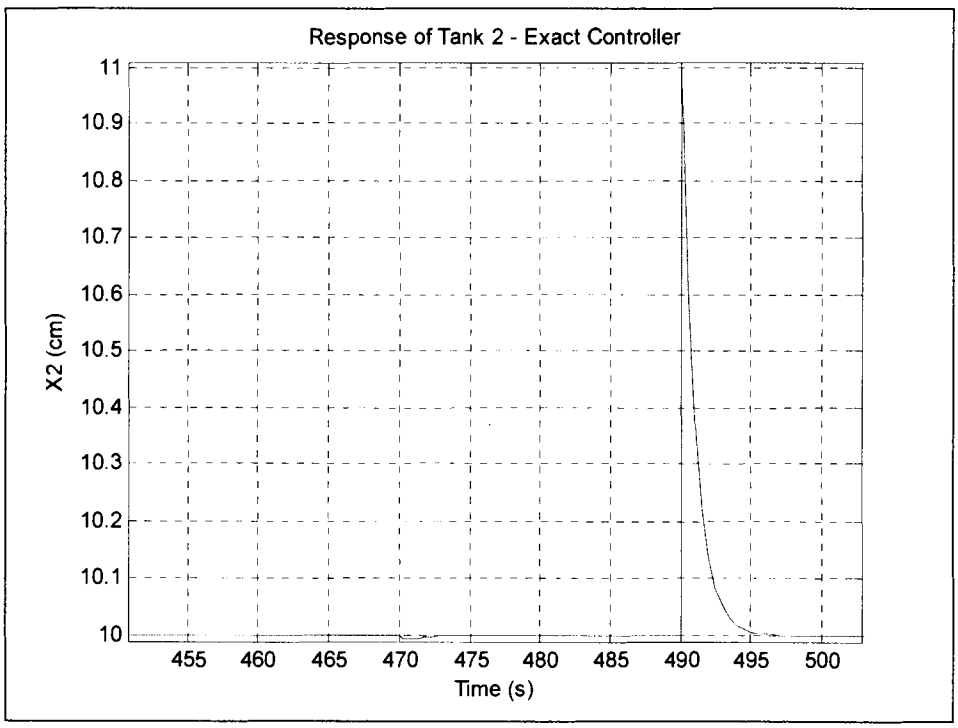

Figure 38: Experiment 1 - tank 2 response – linearised system

The uncontrolled system had open-loop poles at -0.0082 and -0.1164. After applying the controller they were at -1, and -3. The accompanying control signal used is illustrated below.

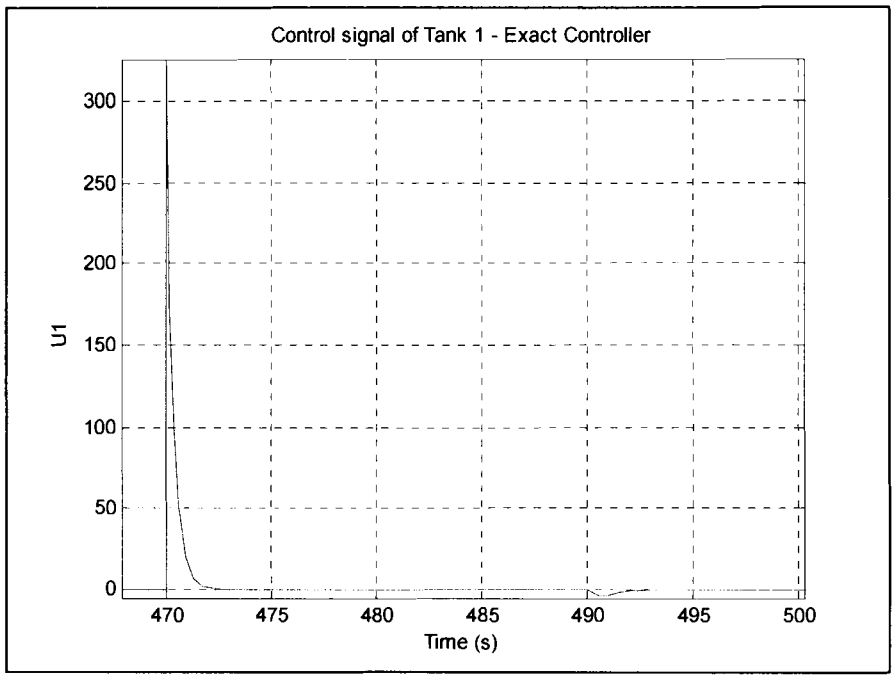

Figure 39: Experiment 1 - tank 1 control – linearised system

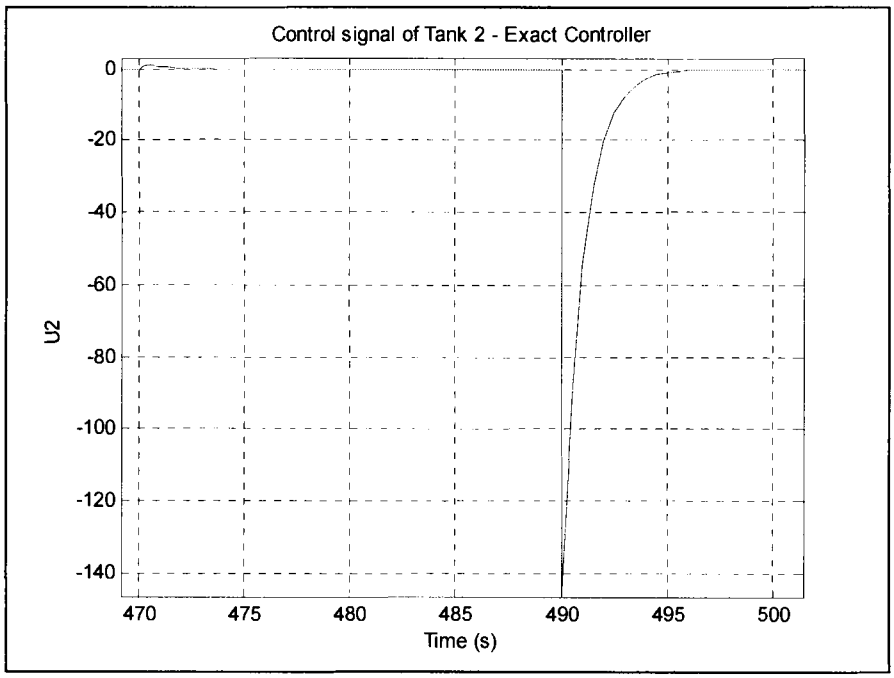

Figure 40: Experiment 1 - tank 2 control – linearised system

As stated, the controllers are determined for each combination of X1 and X2 in figure 34, and if an operating point should be chosen not in this set, controller parameters are estimated. The figure below illustrates the difference between the controller values should it have been calculated exactly versus the estimated values for the current case.

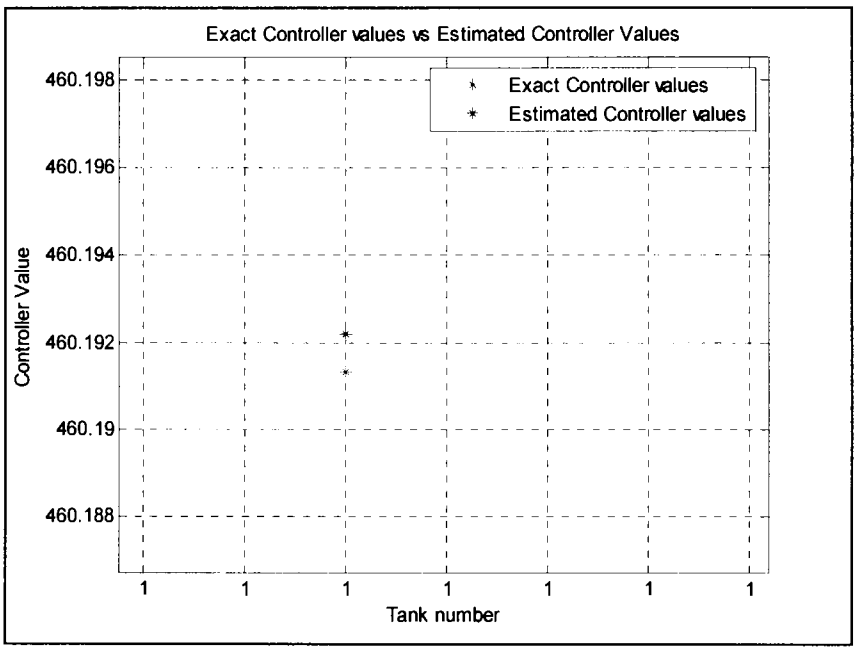

Figure 41: Controller estimation tank 1

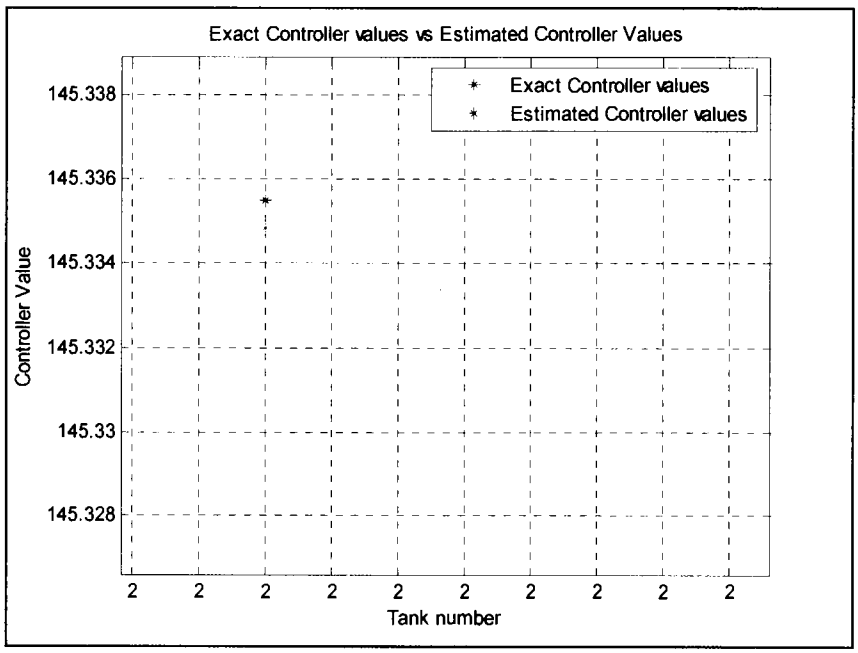

Figure 42: Controller estimation tank 2

Consequently the response is also almost identical. If X1=12 cm and X2=11 cm, the response of the controlled system using the interpolated controller values from figure 35 and 36, yields closed loop poles at X=-1.0002 and X2=0.0002 instead of
-1 and -3; this is acceptable.

Satisfied the controller works on the linearised system around operating points, it can now be implemented in the nonlinear model.

P-controller

To implement it as a linear controller, the average, minimum and maximum controller values from tank 1 (figure 35) and tank 2 (figure 36) are taken respectively, and by the supplied formula, combined to yield a single controller for each tank across the operating range. This formula can readily be adjusted until the desired response is achieved.

Here the formula was chosen to be

$$k_l = 0.77k_{l\_\max} + 0.13k_{l\_\min} + 0.1k_{l\_average}$$

$l = 1,2$ for tank1 and tank 2.

Thus, the controller parameters are

$$k_1 = 460.4025$$
$$k_2 = 145.4914$$

The response for the generated signal is illustrated below for the same simulation parameters as mentioned above. The leak in tank 1 is introduced at 470 seconds and the ball bearing dropped in tank 2 at 570 seconds. Both tanks have an initial value of zero.
SSE refers to the steady-state error.

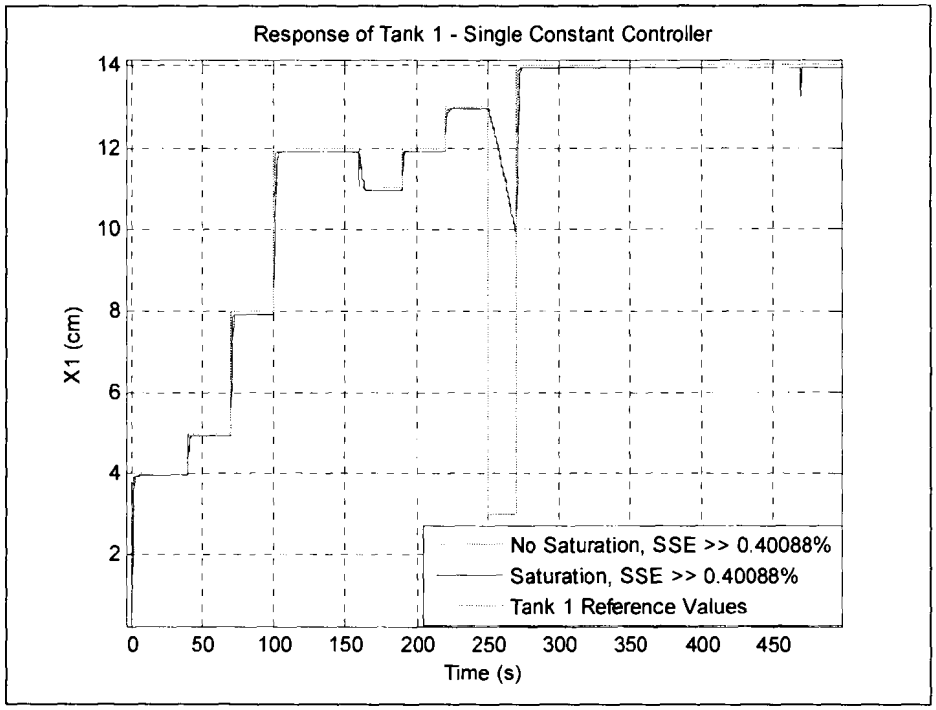

Figure 43: Experiment 1 – tank 1 response P-controller, linear

It can be seen how the system responds too slowly to reach its reference value over the period of 250 seconds to 270 seconds.

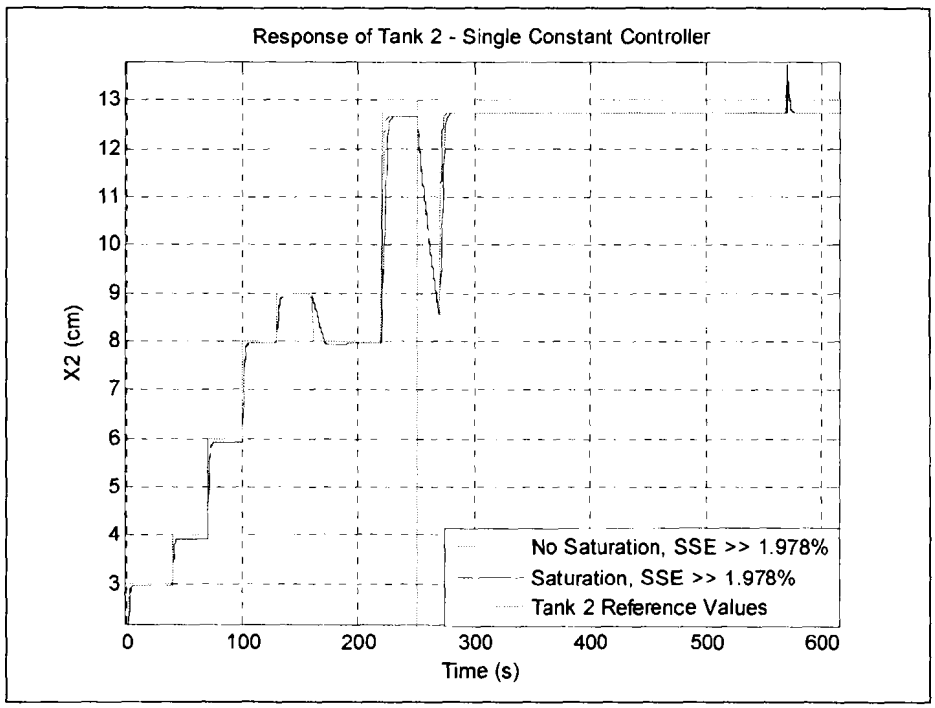

Figure 44: Experiment 1 – tank 2 response P-controller, linear

Figures 45 and 46 illustrate the varying control signal for each tank. Remembering that in this case the inflows and output of the controller is the same thing, one notices the control never falls below zero, and the saturated flow is limited to its maximum amount. The red indicates what the required flow should have been. Referring back to figures 43 and 44, you won't see a significant difference in performance between the saturated and unsaturated response. This complies with NCCD in that the same or very similar performance with smaller control values is achieved. This is advantages in that physically a larger control, in this case would coincide with more water flow requiring a larger water pump. A smaller, cheaper water pump can thus do the necessary actuating without compromising performance.

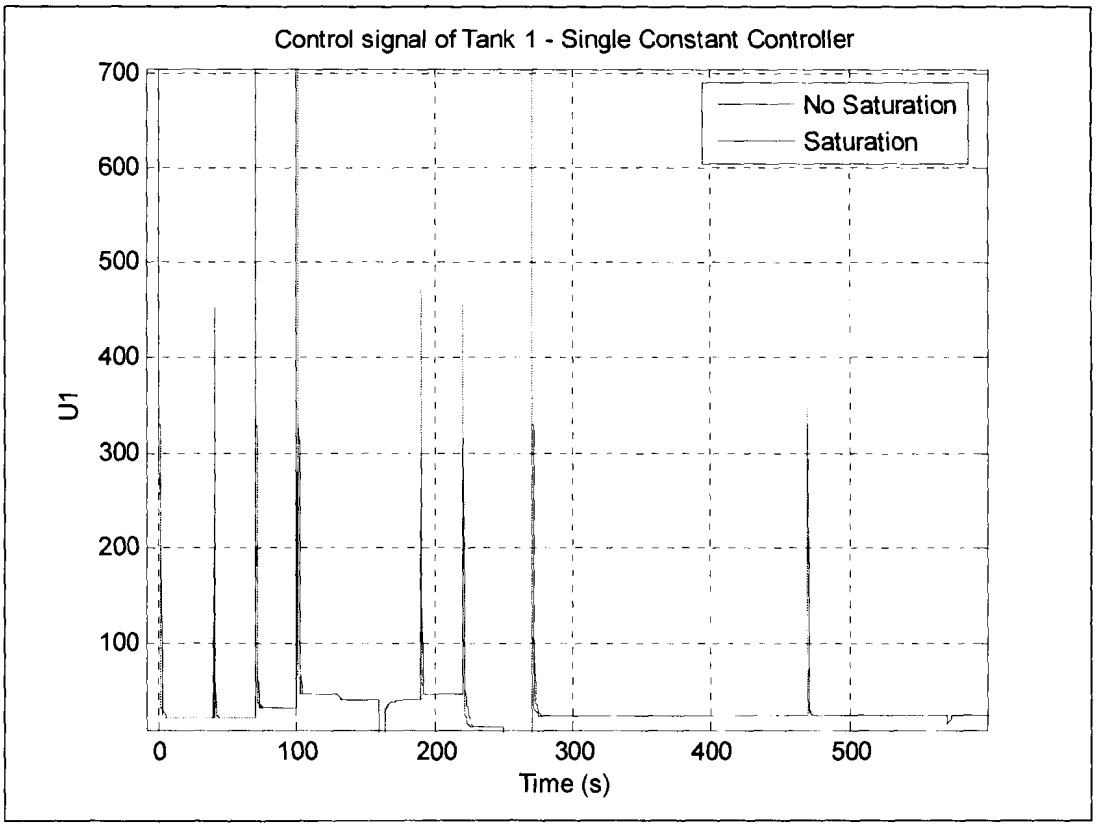

Figure 45: Experiment 1 – tank 1 control signal for P-controller, linear

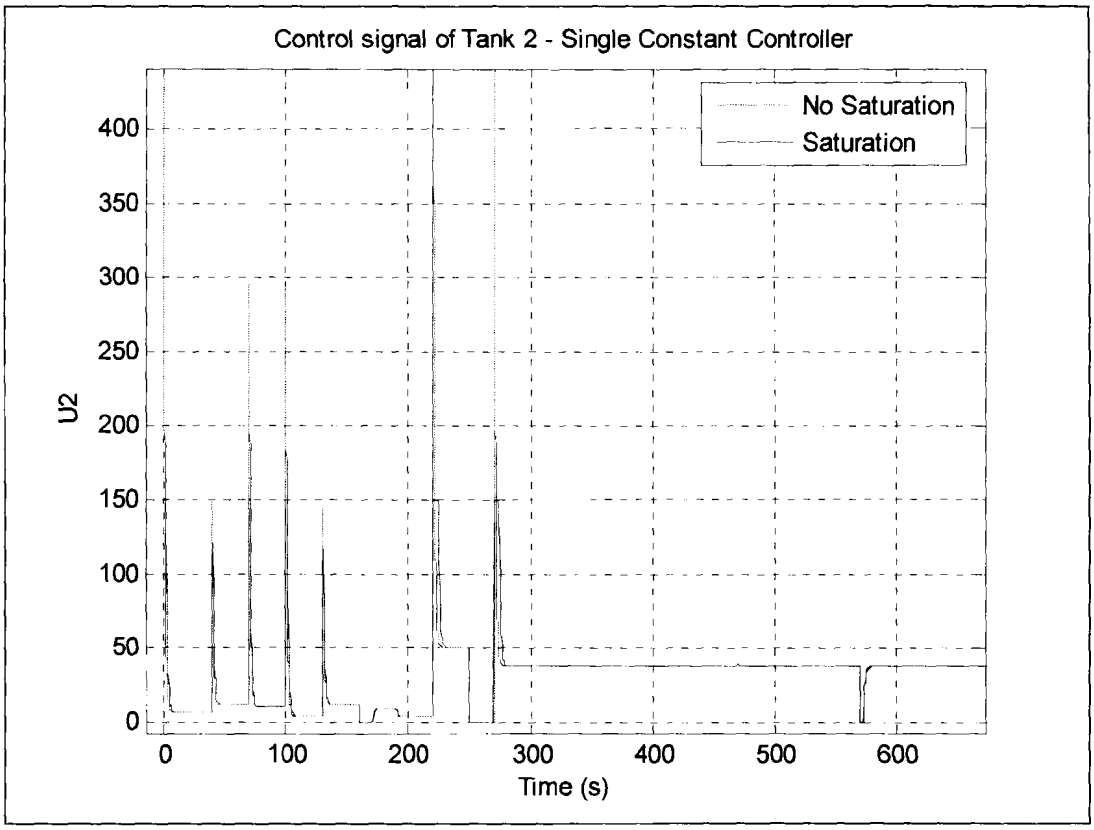

Figure 46: Experiment 1 – tank 2 control signal for P-controller, linear

The saturation legend refers the control signal being limited to its maximum value, whereas the red is the non-limited control signal.

Notice the values of the control signals for both tanks never dip below zero. This is because the inflow can only stop, never become negative, unless a way is found to pump out the water as well.

From this, the response of the system for differing heights are apparent. Figures 43 and 44 also indicate that the introduced errors have also been fully rejected.
The figures below take a closer look at the response when the two tanks have the same heights. Refer to table 2.

Notice that the tank which is to remain at the same reference point experiences very little to no change in its height indicating the controller doing well with the cross-coupling effect of the system as well.

The controls of the nonlinear controller were not supplied as in this case they are very similar to that of the linear case.

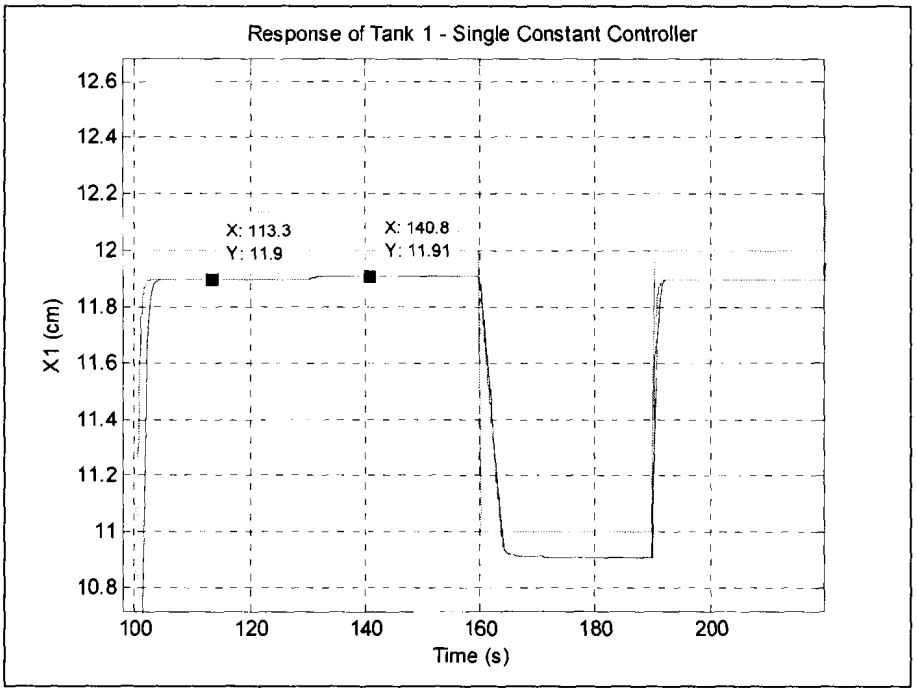

Figure 47: Experiment 1 – tank 1 same-height response, P-controller, linear

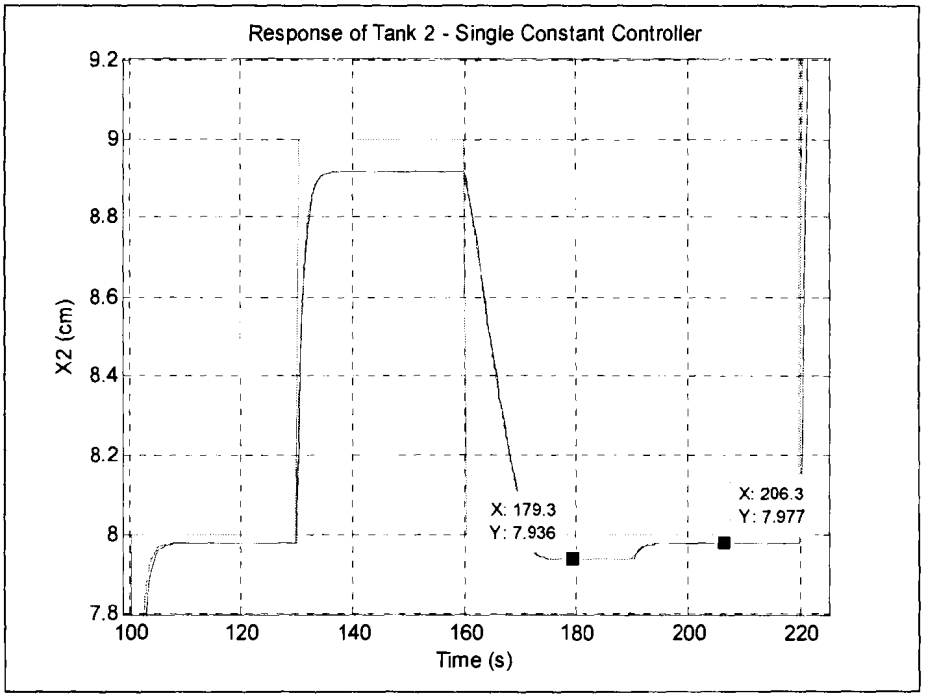

Figure 48: Experiment 1 – tank 2 same-height response, P-controller, linear

Figures 49 to 52 illustrate the same as above, but for the nonlinear proportional controller.

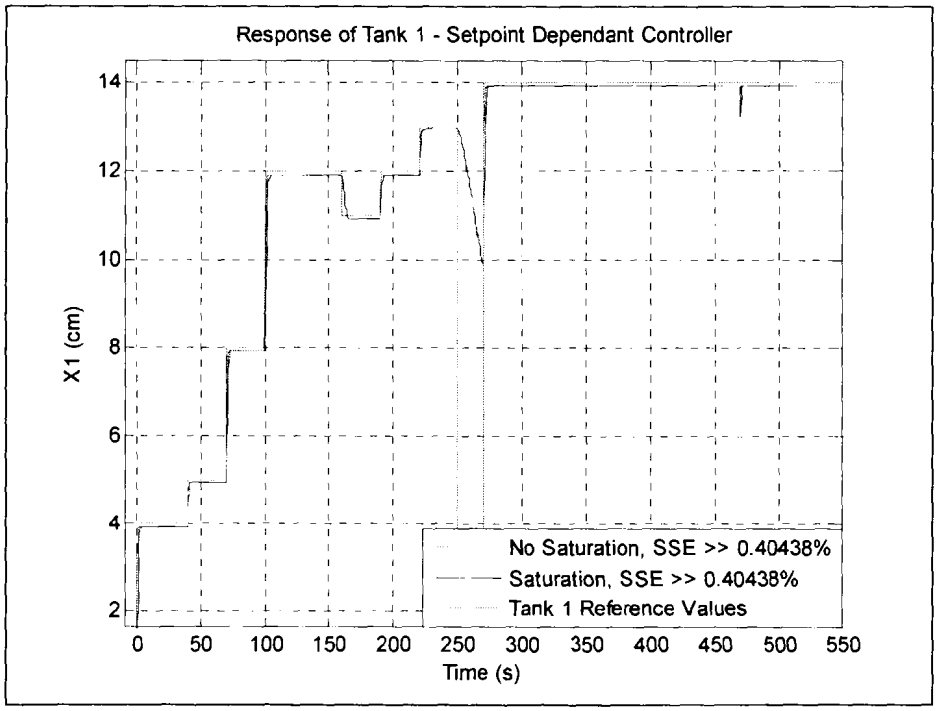

Figure 49: Experiment 1 – tank 1 response P-controller, nonlinear

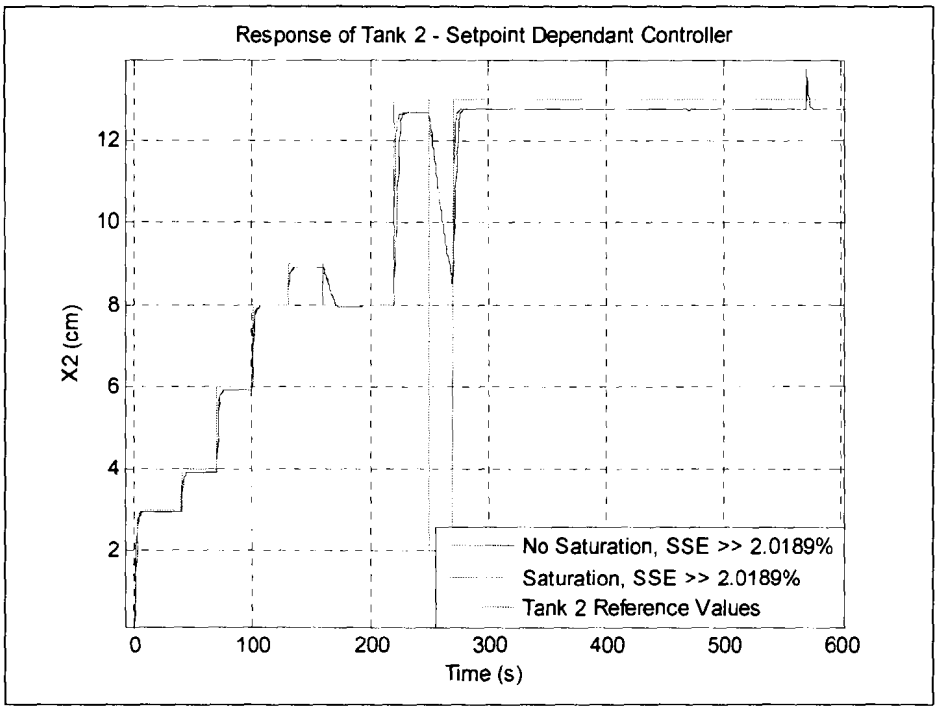

Figure 50: Experiment 1 – tank 2 response P-controller, nonlinear

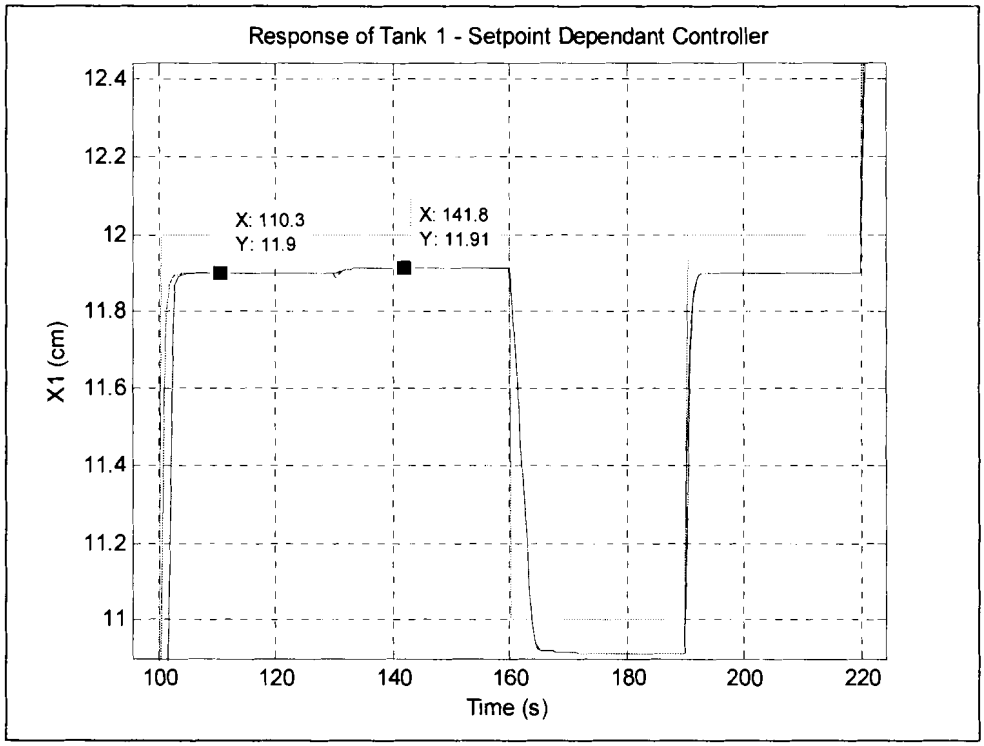

Figure 51: Experiment 1 – tank 1 same-height response, P-controller, nonlinear

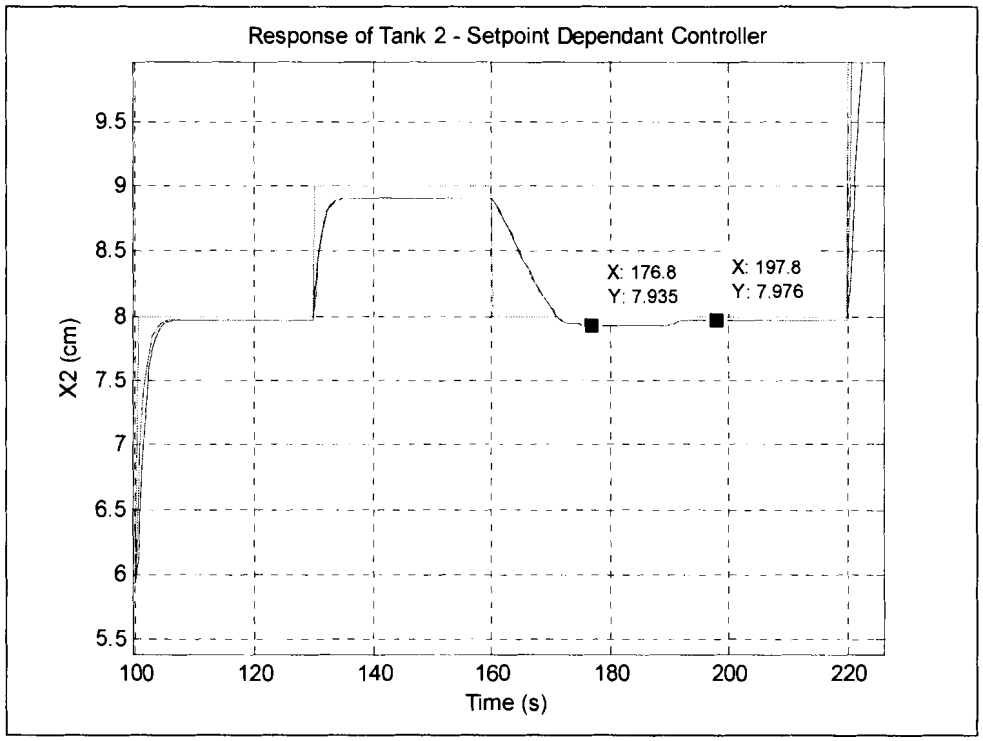

Figure 52: Experiment 1 – tank 2 same-height response, P-controller, nonlinear

From figures 50 and 51 one notices the definite steady-state error and slight cross-coupling effect when changing the one tank's height, but keeping the other one the same.

PI-controller

Due to the fact that one has a second order system, there were 2 equations with 2 variables to solve in the previous control strategy. There are now 4 variables and 2 equations. Considering the controller parameters, i.e. the P- and I-constants for every operating point come in pairs, one can arbitrarily choose the I-constants and exactly calculate its matching P-constant.

An educated guess for the I-constant was made as follows:

The time constant for each pole was determined as $timeconst = \frac{1}{|p_l|}$ with $l = 1,2$.

The shortest time constant, t_{short} was assigned to tank 1 and the longest, t_{long} to tank 2, and each multiplied by a factor. After trial and error, the I-constants which yielded the best system response across the operating range was:

Tank 1: $K_{I\_T1} = t_{long} * 2$;

Tank 2: $K_{I\_T2} = t_{short} * 3$;

Solving the P-constants for each set point, the lookup table is generated.

For the **linear case**, the controller parameter values were chosen using the same formula as the P-Controller; determined separately for the P- and I-constants for both tanks 1 and 2, yielding

$$k_1 = 267.7874 + \frac{2}{s}$$

$$k_2 = 266.8118 + \frac{1}{s}$$

The accompanying response for the test scenarios are visible in the figures to follow.

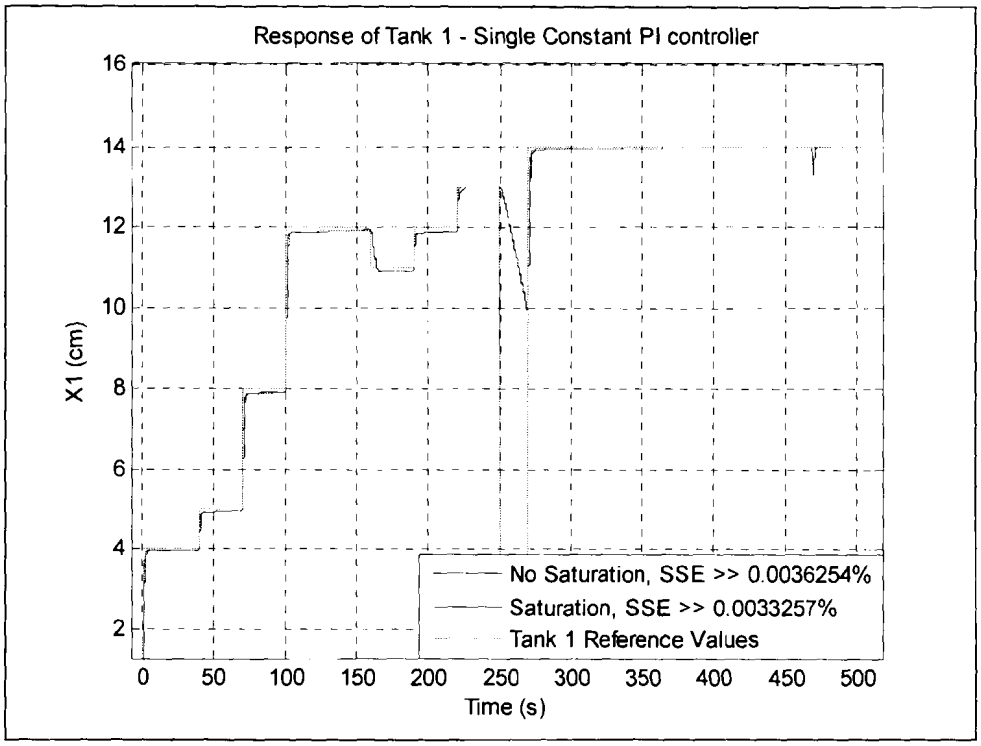

Figure 53: Experiment 1 – tank 1 response PI-controller, linear

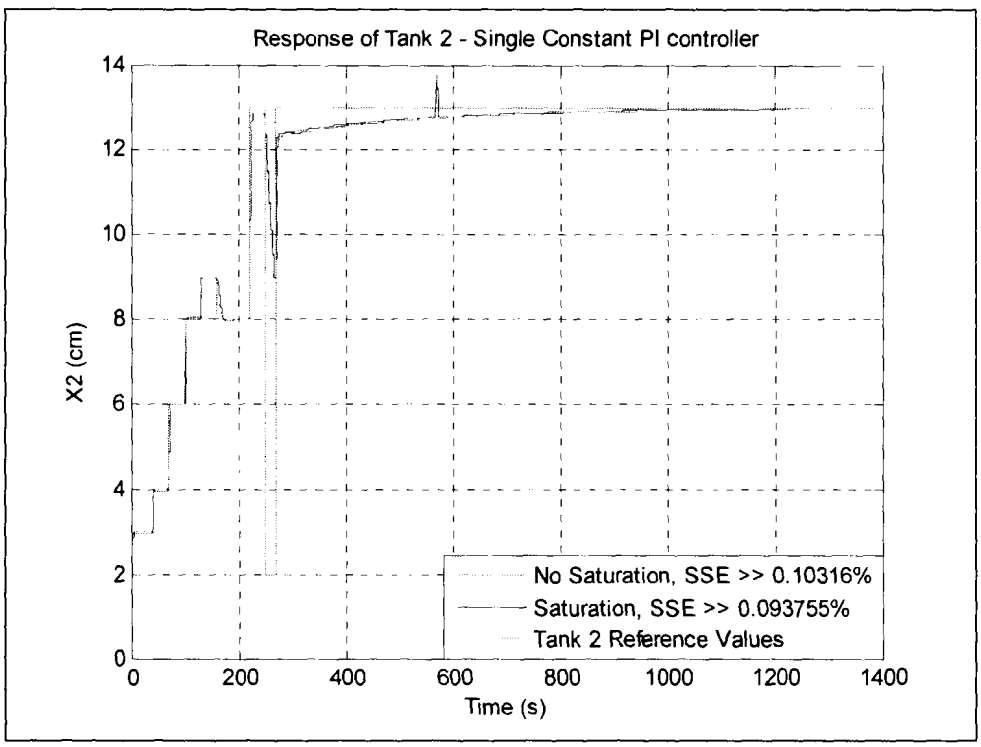

Figure 54: Experiment 1 – tank 2 response PI-controller, linear

For all practical purposes a zero steady-state error is achieved for both tanks' reference values. When comparing this response with that of the P-controller, the distinct difference is the zero steady-state error which is achieved.

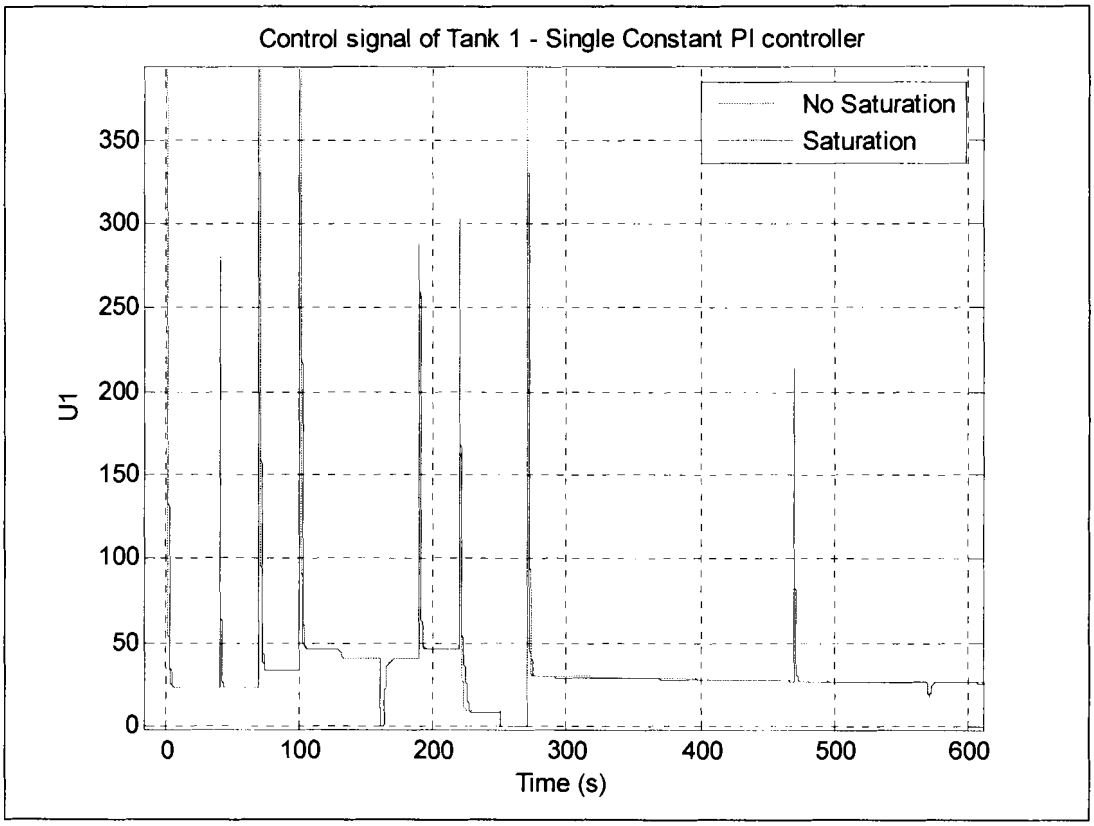

Figure 55: Experiment 1 – tank 1 control signal for PI-controller, linear

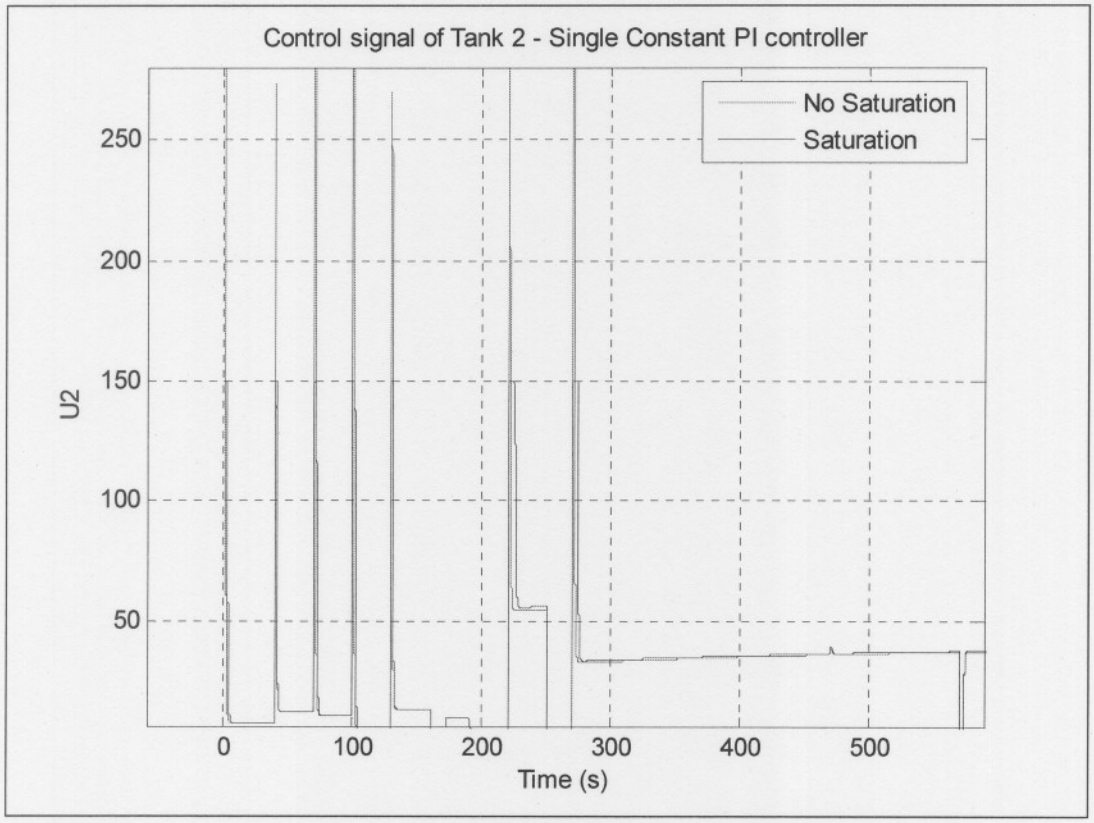

Figure 56: Experiment 1 – tank 2 control signal for PI-controller, linear

When comparing the average control values for the reference signal of tank 1 and 2 of the linear P-controller with that of the linear PI-controller, the overall average is slightly less for the PI-controller, see table 3.

Controller type	U1 (Tank 1)	U2 (Tank 2)
P	30.1278	34.47
PI	27.0641	36.4694

Table 3: Average control values for the linear controllers

The figures to follow focus on the noise rejection ability of this PI-controller.

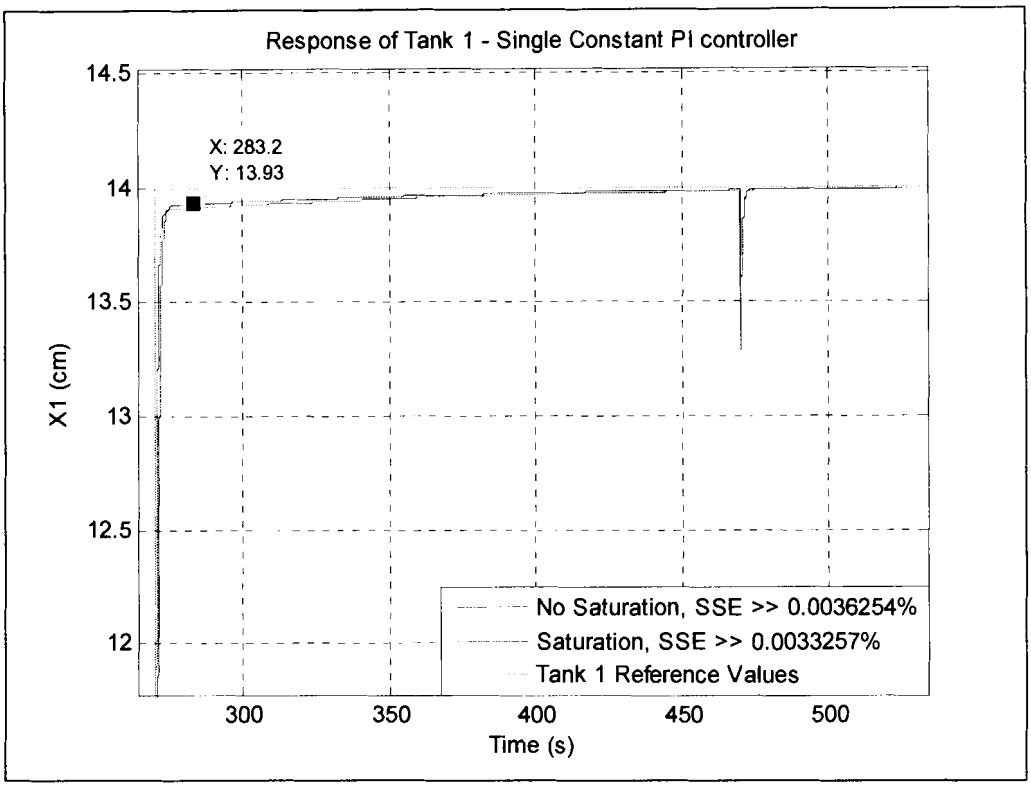

Figure 57: Experiment 1 – tank 2 specific transient response, PI-controller, linear

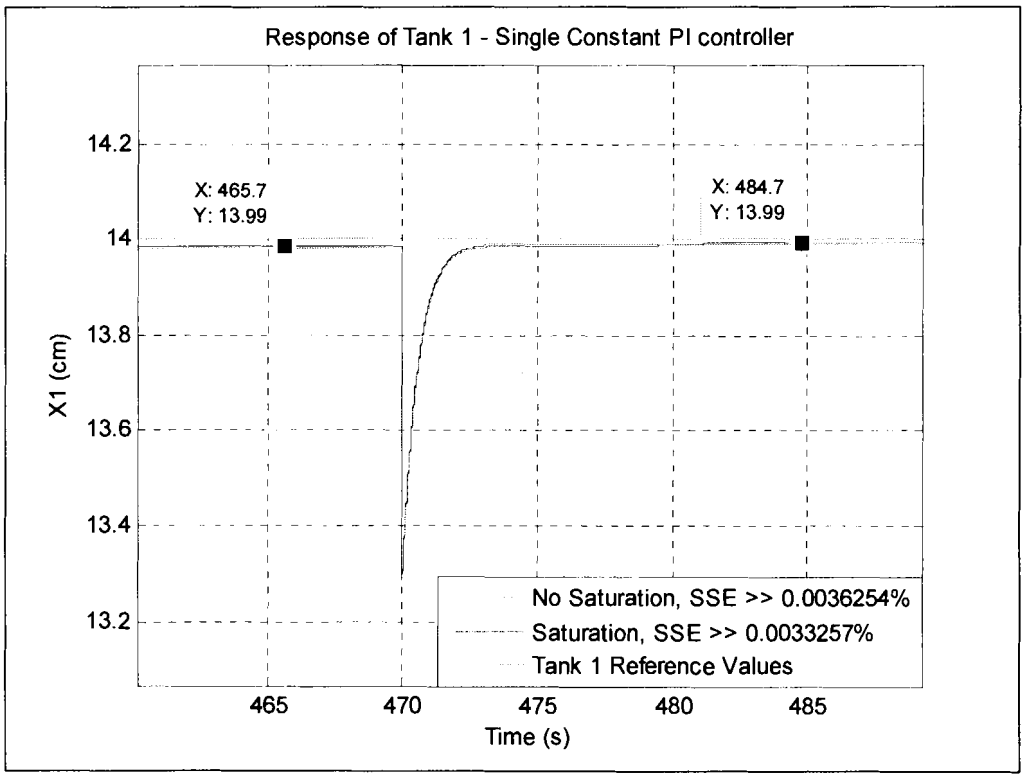

Figure 58: Experiment 1 – tank 1 noise rejection using PI-controller, linear

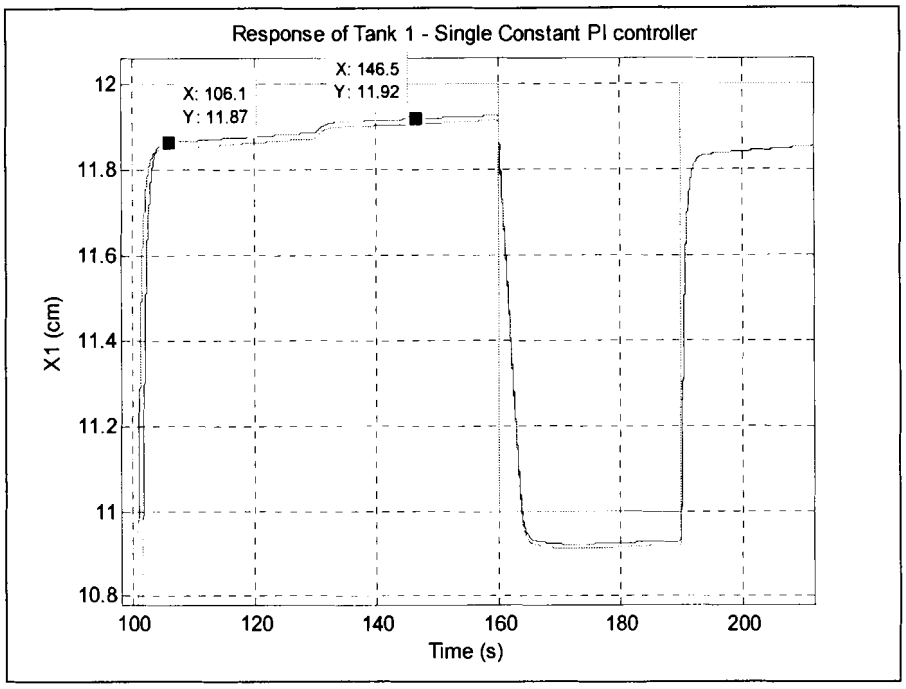

Figure 59: Experiment 1 – tank 1 same-height response, PI-controller, linear

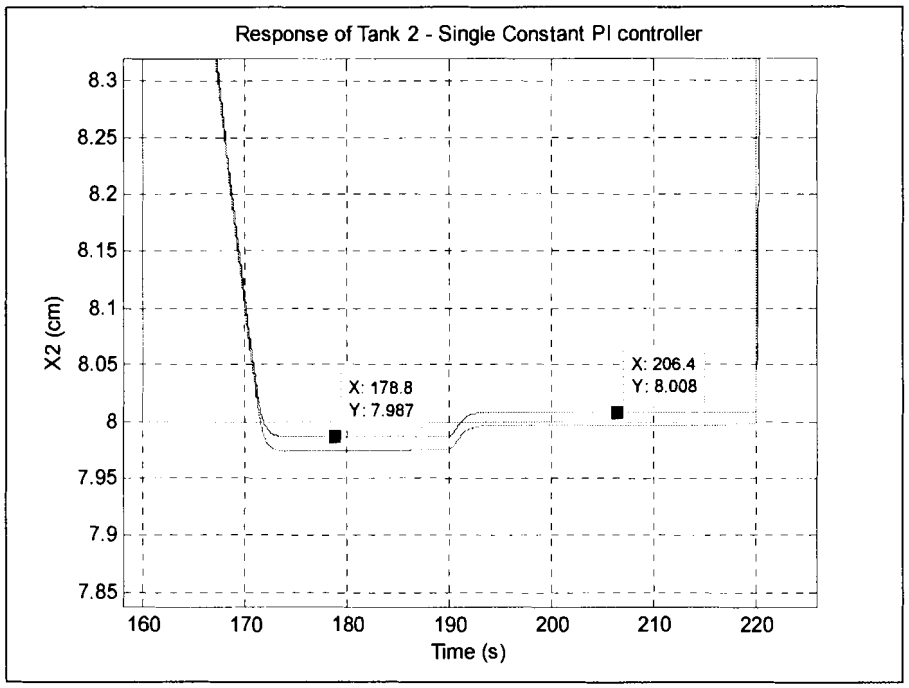

Figure 60: Experiment 1 – tank 2 same-height response, PI-controller, linear

Figures 59 and 60 clearly show the response tends toward a zero steady-state error with a slight visible cross-coupling effect when trying to hold the one tank at a constant height whilst changing the other.

Accordingly the same responses are now supplied when using the nonlinear PI-controller.

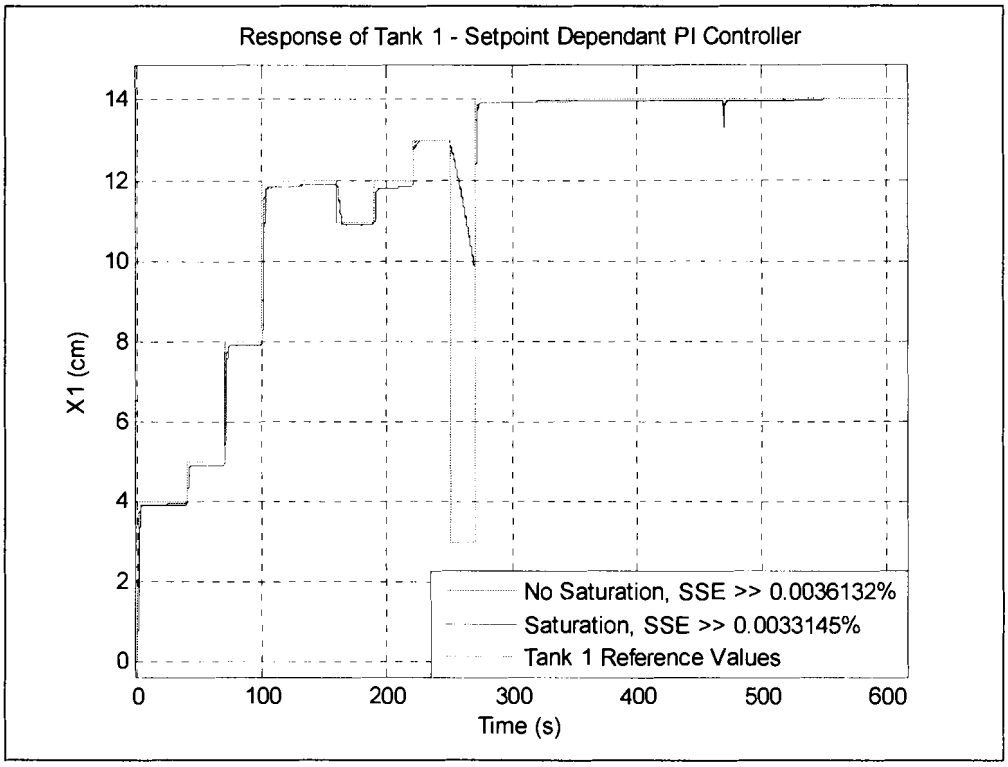

Figure 61: Experiment 1 – tank 1 Response PI-controller, nonlinear

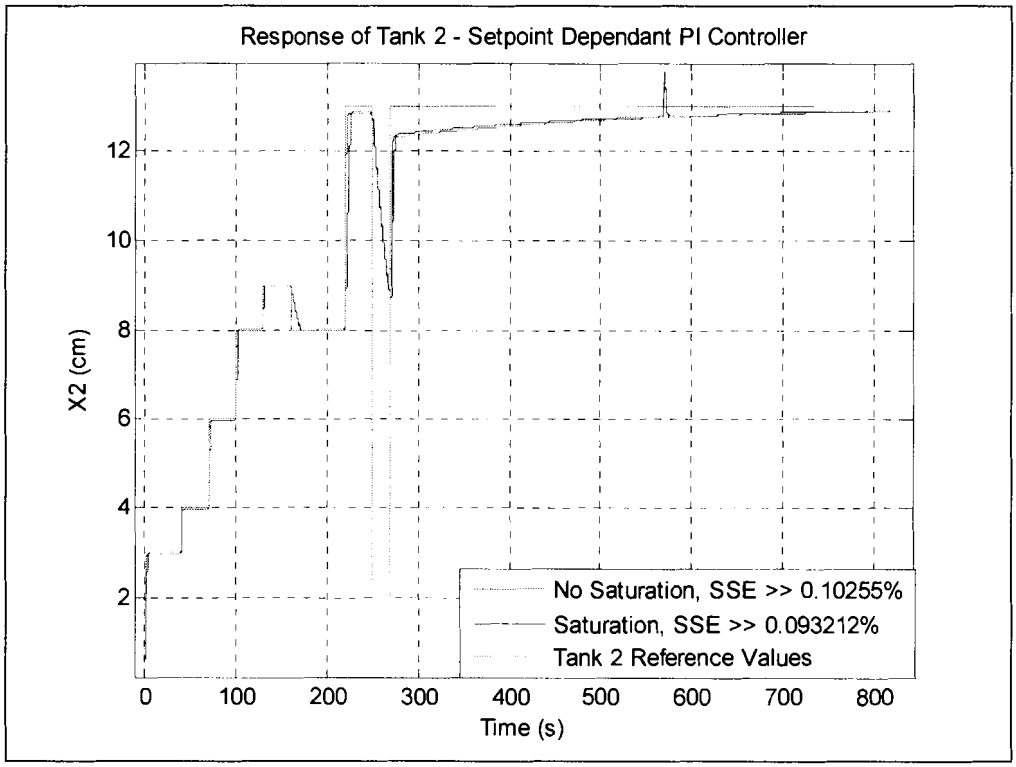

Figure 62: Experiment 1 – tank 2 response PI-controller, nonlinear

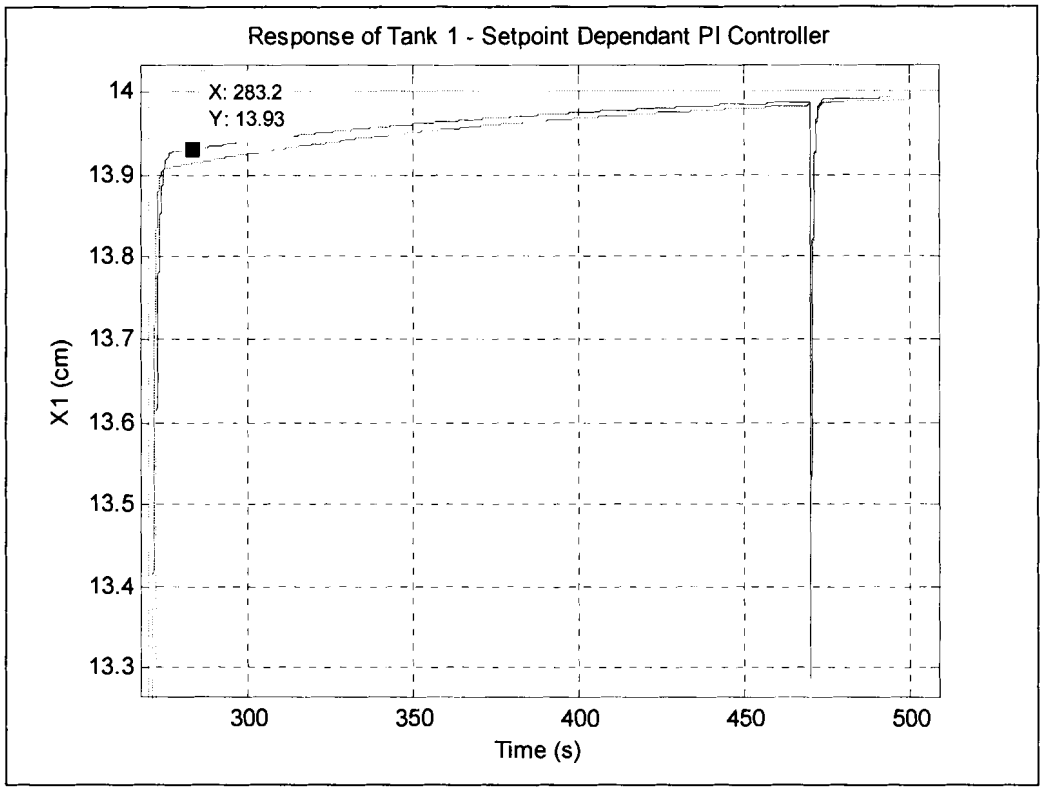

Figure 63: Experiment 1 – tank 1 specific transient response, PI-controller, nonlinear

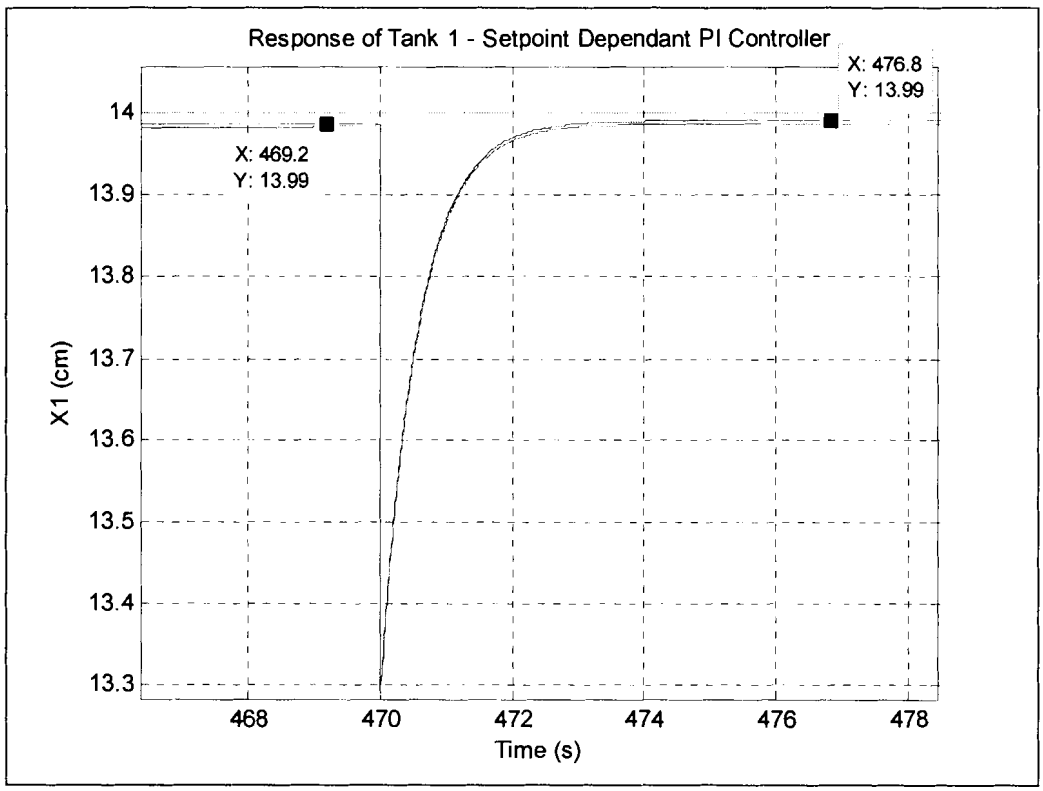

Figure 64: Experiment 1 – tank 1 noise rejection using PI-controller, nonlinear

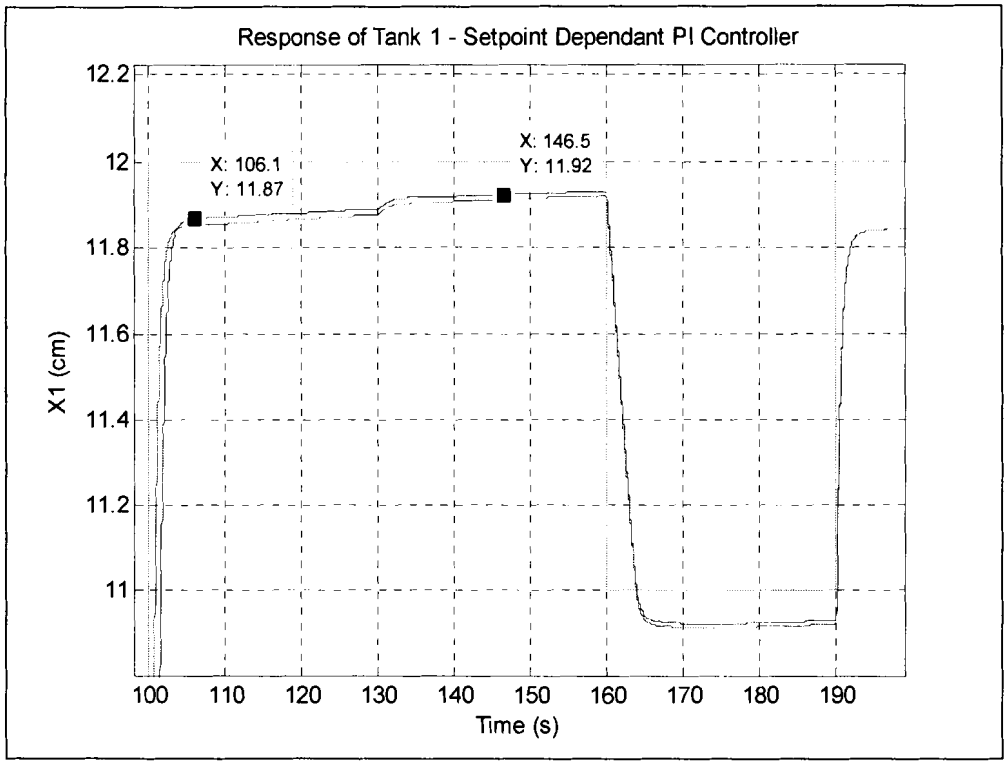

Figure 65: Experiment 1 – tank 1 same-height response, PI-controller, nonlinear

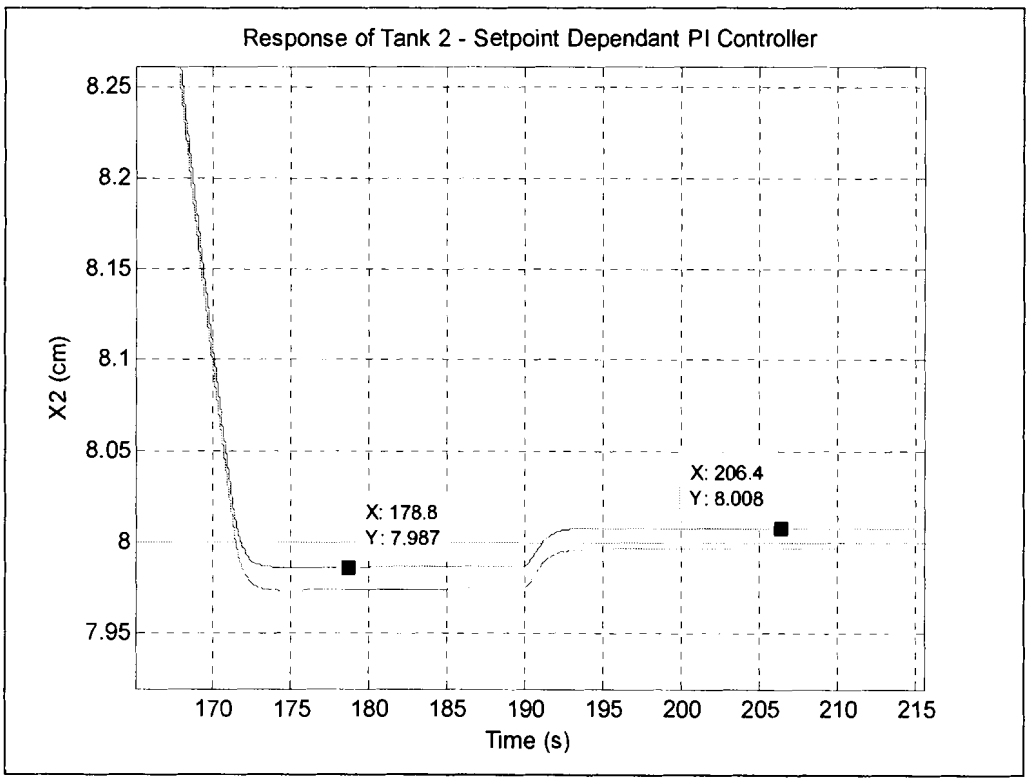

Figure 66: Experiment 1 – tank 2 same-height response, PI-controller, nonlinear

The phenomenon of reset-windup, occurred in tank 1. Figure 67 shows the response when reset-windup occurs, and figure 68 when a anti-reset windup algorithm has been successfully implemented.

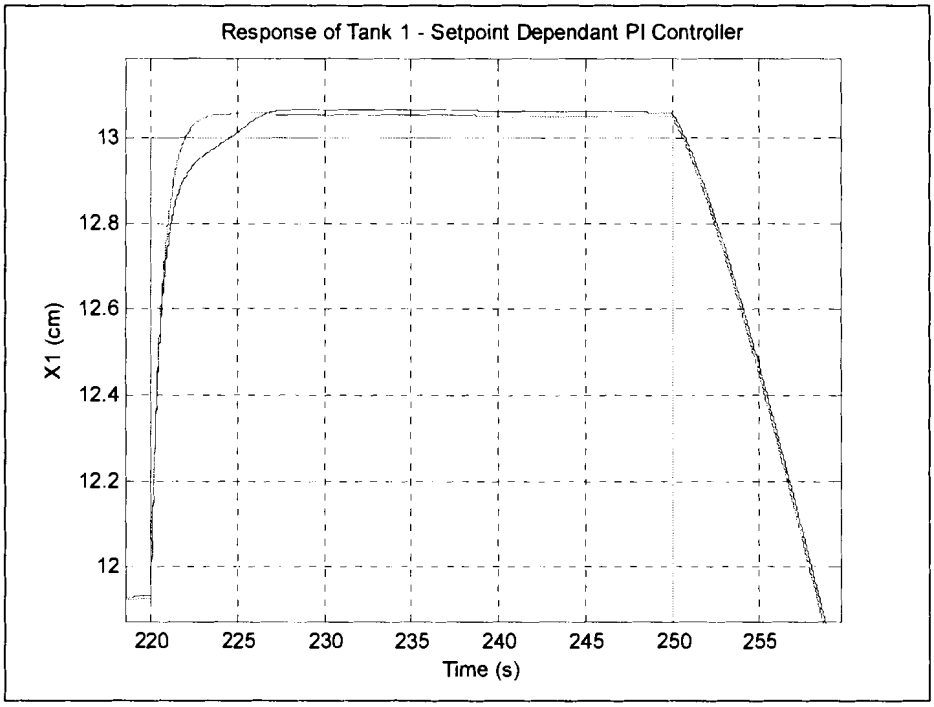

Figure 67: Anti-reset-Windup threshold=2

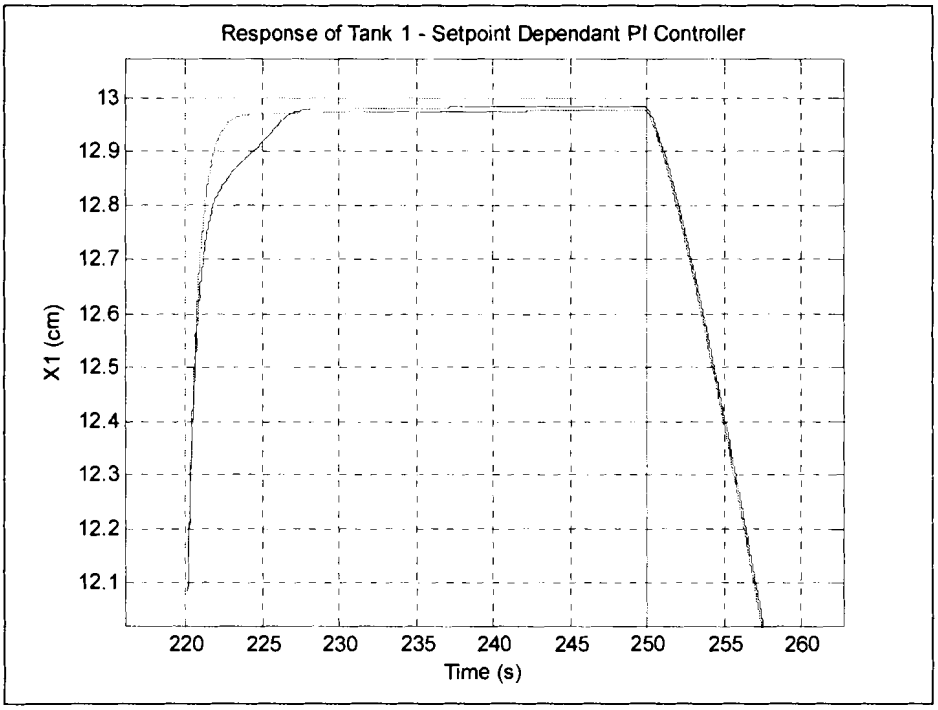

Figure 68: Anti-reset-windup threshold=0.9

PI-controller (Optimised with a GA)

As stated earlier, the GA was used on the nonlinear model of the system to optimise the controller parameters for the PI-controller,

$$\mathbf{K} = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} k_{P1}e_1(t) + k_{I1}\int_0^t e_1(t)dt \\ k_{P2}e_2(t) + k_{I2}\int_0^t e_2(t)dt \end{bmatrix}$$

The parameters within the GA itself that are of importance are the number of individuals in the population (N_{ind}), the generation gap ($GGAP$), the selection method used (rws / sus) and the objective function. Also important is the fact whether the initial population of the GA was biased or not. Different combinations of these parameters were used. These, and the results obtained are tabulated below.

The setup which yielded the best performance for the generated reference signal is used for illustration.

The optimal solution for the strategy using the GA is mainly influenced by the biasing/not of the original population, and the performance index used. The other GA parameters mostly contribute to the number of generations it takes the algorithm to find an optimal solution. Given the simulations were run long enough that all combinations would have found the optimal solution, the parameters of interest are the biasing of the initial population and the performance function. For this reason, unless stated otherwise, the GA parameters were:

N_{ind}=20;

$GGAP$=0.9

Selection method: *rws*

Crossover/recombination technique: Intermediate recombination

Mutation function[9]: mutbga

A PID-controller was also included.

[9] 'mutbga' is a function found in MATLAB®'s GA Toolbox

Type controller	Selection method	N_{ind}	Bias	Performance index	Steady-state error tank 1	Steady-state error tank 2
PI	rws	20	NO	ISE	0.03%	1.21%
PI	sus	20	NO	ISE	0.04%	1.21%
PI	sus	30	NO	ISE	0.03%	1.19%
PI	rws	20	NO	ITSE	0.04%	0.43%
PI	rws	20	YES	ISE	0.03%	1.26%
PI	rws	20	YES	ITSE	0.04%	0.1%
PID	rws	20	NO	ISE		
PID	rws	20	NO	ITSE	0.2%	0%

Table 4: GA trial runs

An optimal solution is found when the objective function value has not been decreased for a number of generations. This is illustrated below. $f(x)$ is the objective function value.

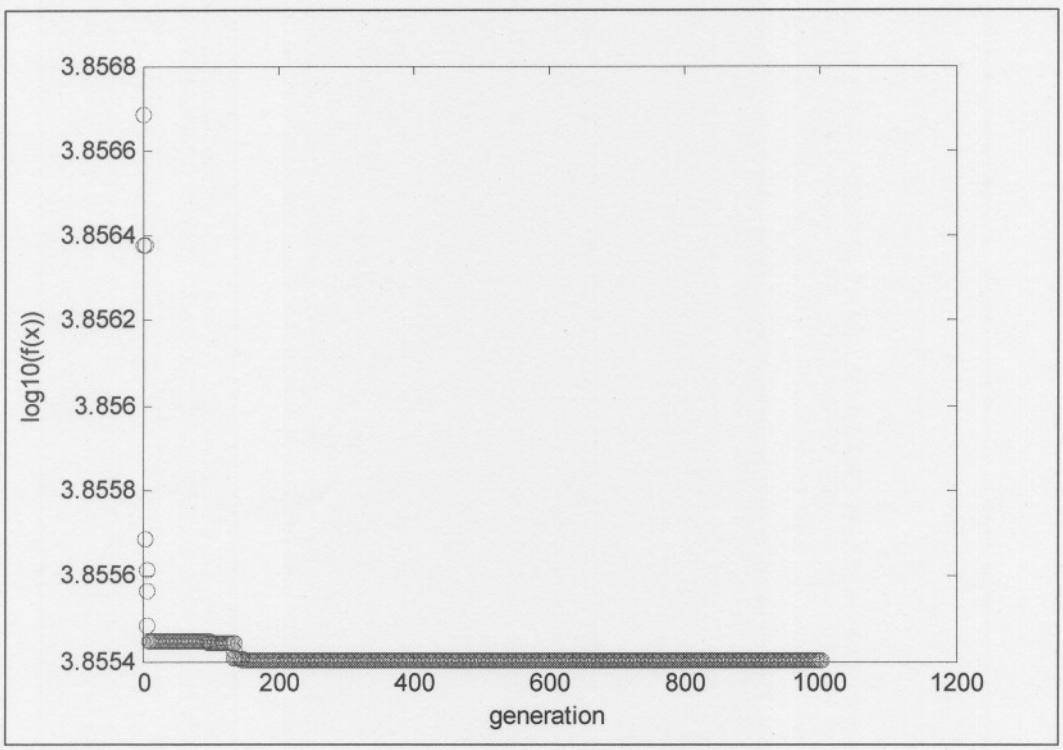

Figure 69: Optimum solution search

Because this is a probabilistic method, the possibility, how small, exists that at some point after a number of generations the solution can be improved. Thus, in this study, 1000 generations was considered enough time in which to find the optimum solution.

The illustration below shows how the GA minimises the objective function in a search for the optimum solution.

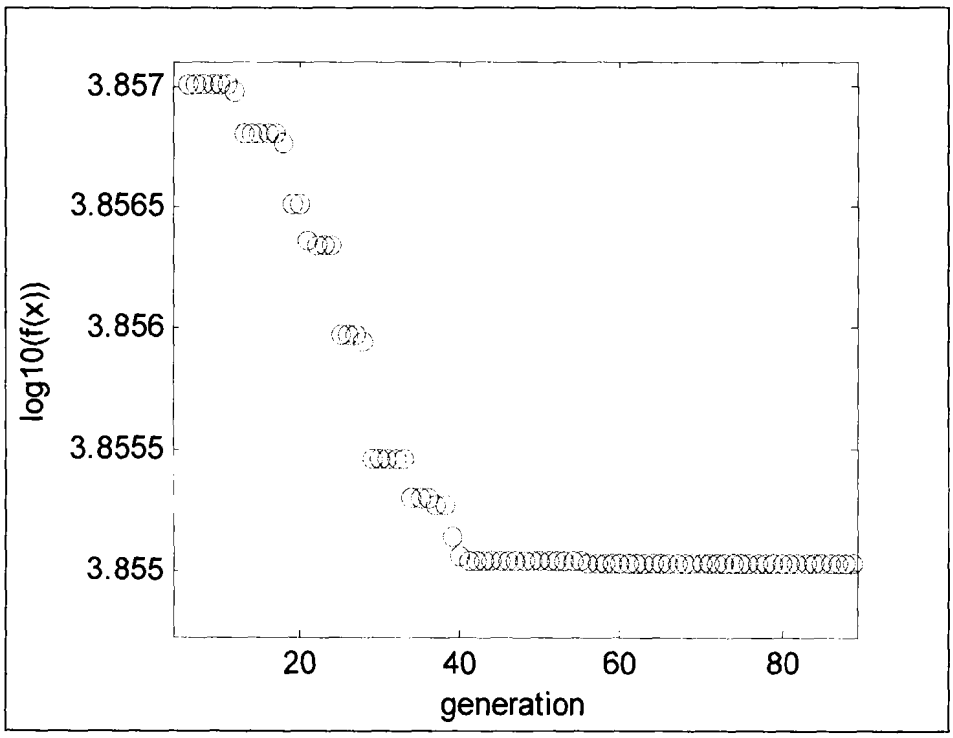

Figure 70: The learning process – minimisation of objective function

By inspection of table 4, the parameters yielding the best performance for this control strategy have been highlighted. The reason for **not choosing** the PID-controller, is that although yielding the best steady-state errors, it exhibits very poor dynamic behaviour. See figure 71.

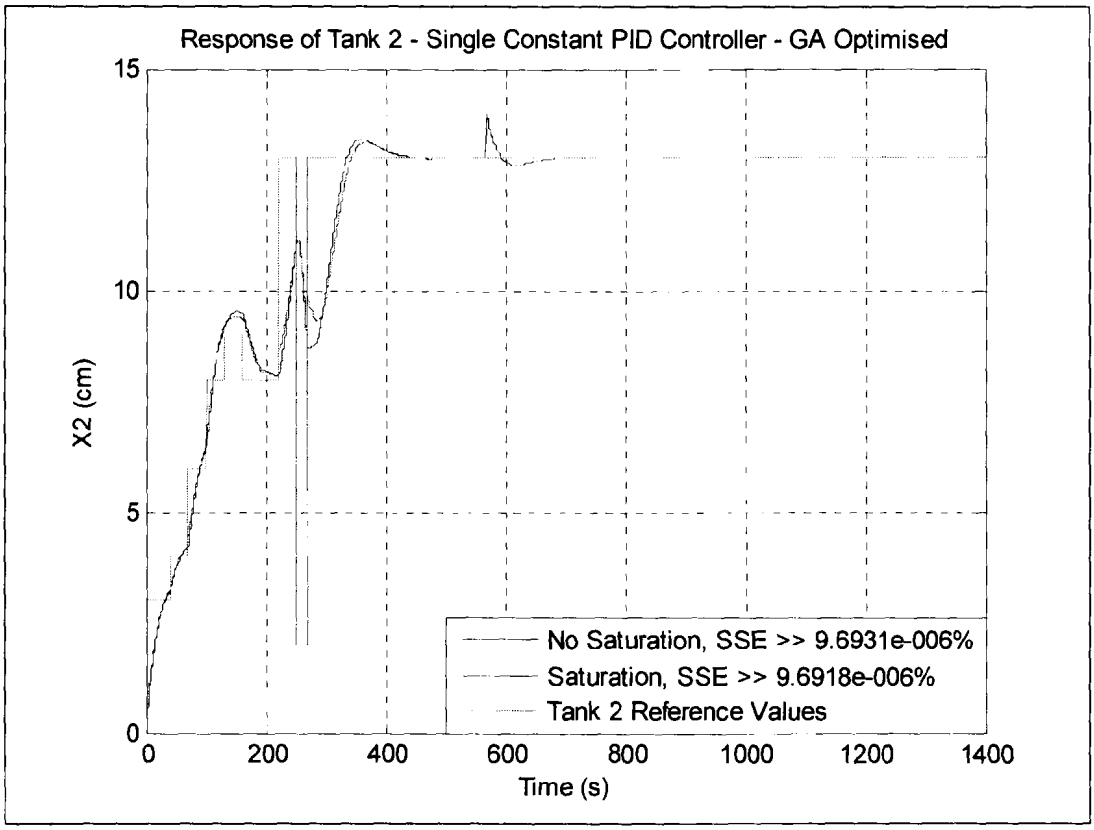

Figure 71: Experiment 1 – tank 2 response PID-controller

The highlighted parameters were used with the PI-Controller, and the anti-reset windup algorithm tweaked by adjusting the thresholds at which each tank's integrator reset itself.

This was also done using the PID-Controller setup as indicated in table 4, but it did not improve its dynamic response in any way.

The final setup is as follows:

Type of controller: PI

$$\mathbf{K} = \begin{bmatrix} k_1 = 4000e_1(t) + 0.1505\int_0^t e_1(t)dt \\ k_2 = 4000e_2(t) + 0.0959\int_0^t e_2(t)dt \end{bmatrix}$$

N_{ind} =20;

$GGAP$ =0.9

Selection method: *rws*

Crossover/Recombination technique: Intermediate recombination

Mutation function: mutbga

Performance index: ITSE

Reset windup threshold tank 1: 0.9;

Reset windup threshold tank 2: 2;

This yielded the following responses for the generated test signal:

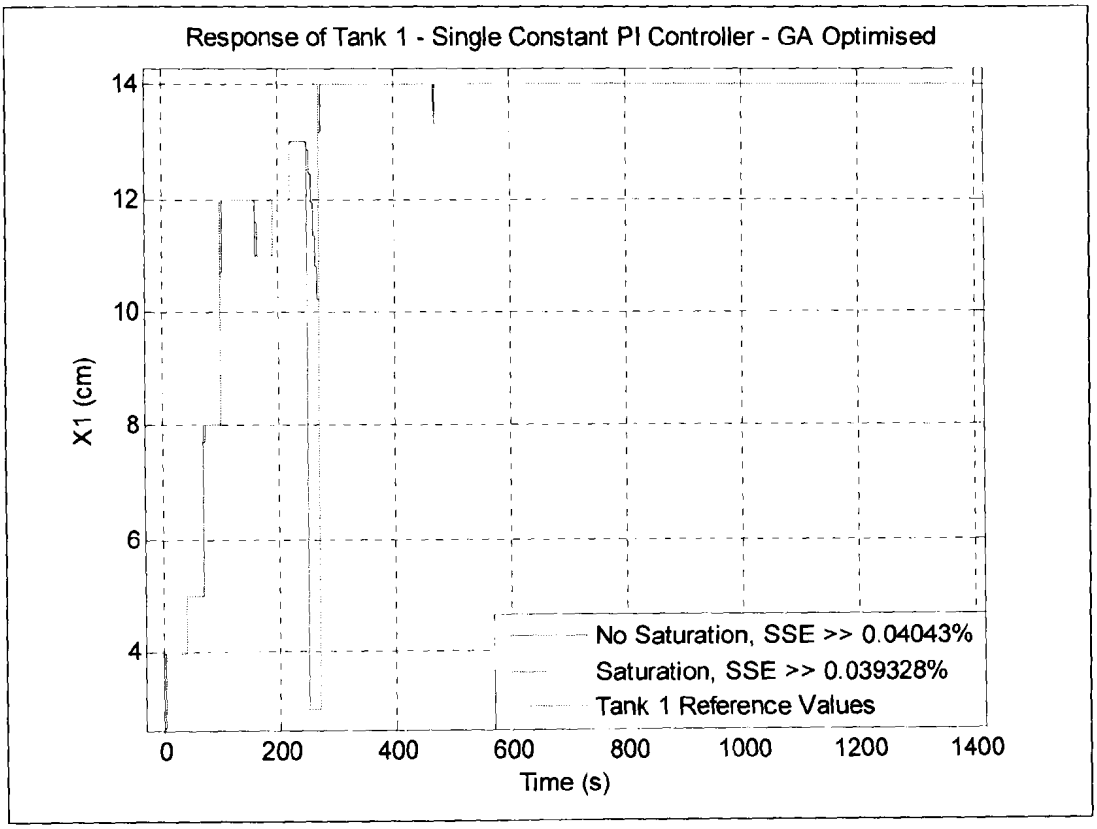

Figure 72: Experiment 1 – tank 1 response optimised PI-controller

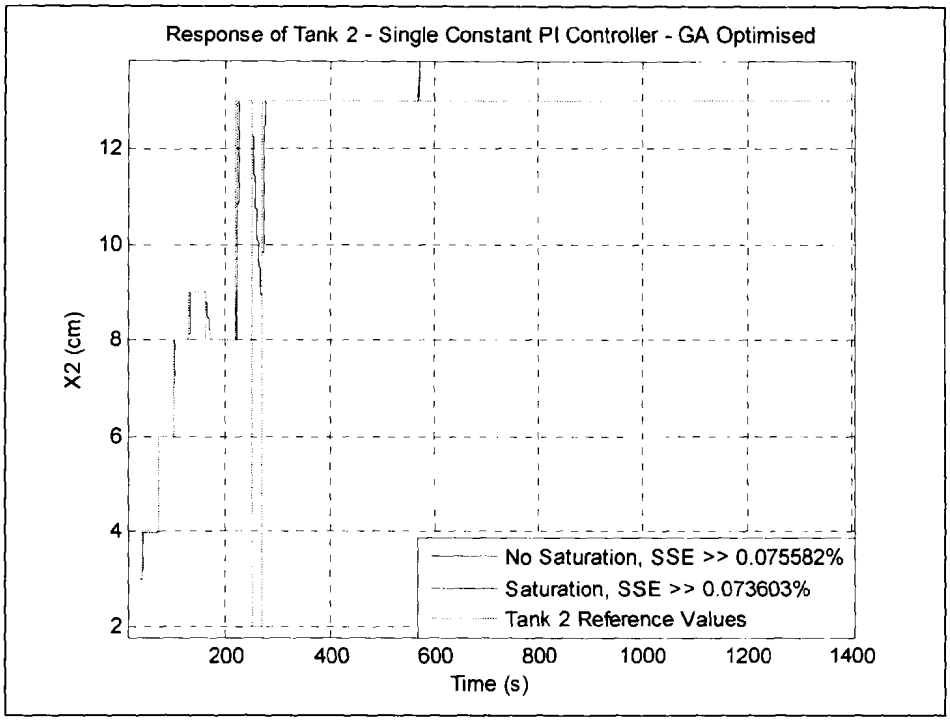

Figure 73: Experiment 1 – tank 2 response optimised PI-controller (1)

A notable improvement can be seen when comparing figures 72 and 73 with their corresponding figures using the other types of controllers. Figure 74 focuses on this improved response.

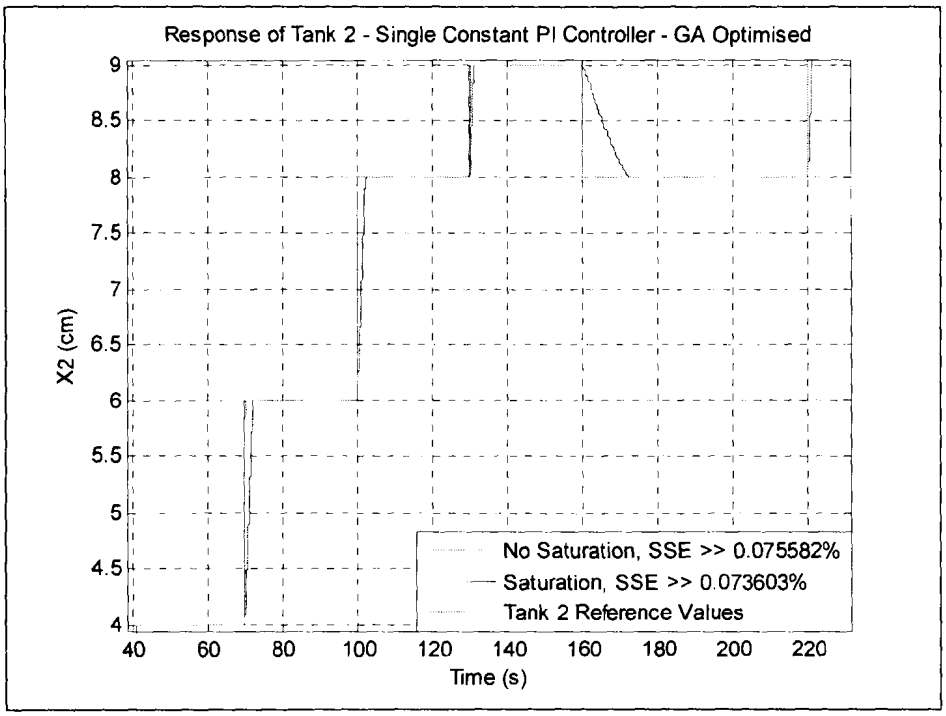

Figure 74: Experiment 1 – tank 2 response optimised PI-controller (2)

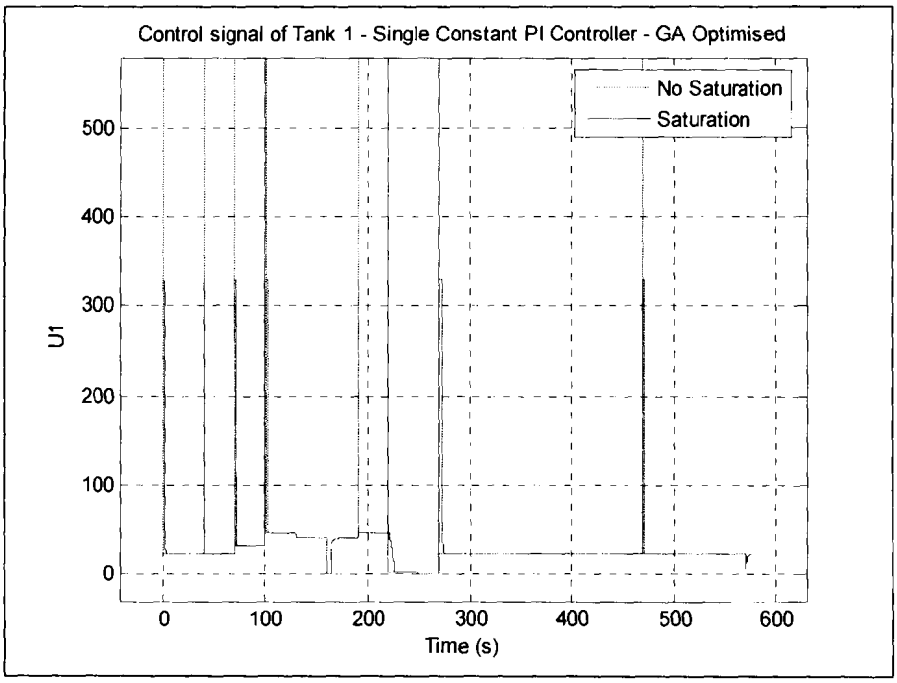

Figure 75: Experiment 1 – tank 1 control signal for optimised PI-controller

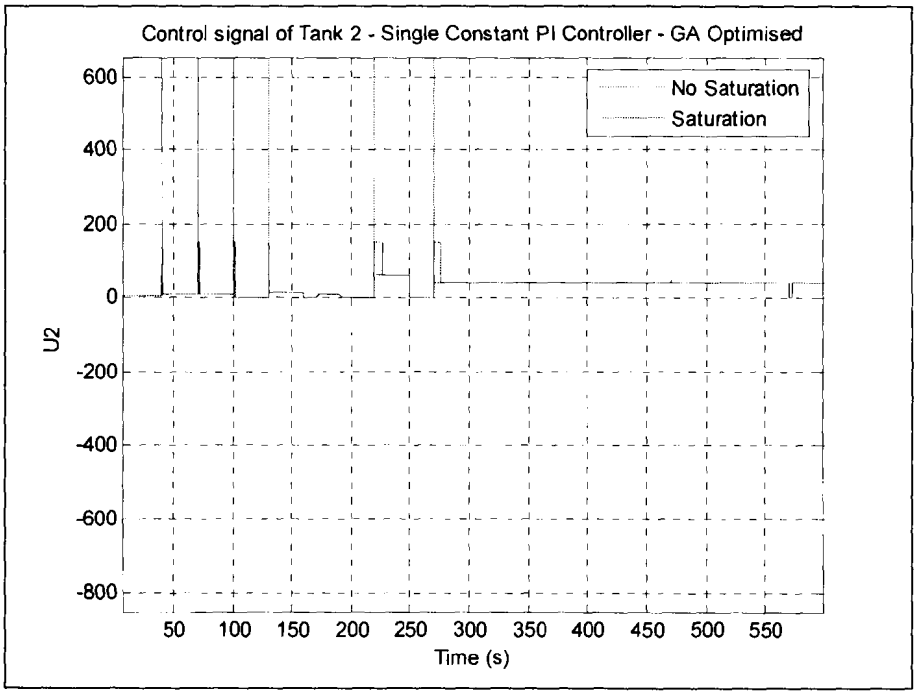

Figure 76: Experiment 1 – tank 2 control signal for optimised PI-controller

If one now looks at the optimised linear PI-controller's average control value for each tank over the simulation period (table 5), it will be seen that it is very similar to that of the linear PI-controller of table 3.

Controller type	U1 (Tank 1)	U2 (Tank 2)
PI-optimised	26.9105	36.6689

Table 5: Average control values for the linear, PI-optimised controller

These results verify our initial design for the PI-controller which did not make use of the GA.

The main difference in performance between the PI-controller which had its parameters optimised and the nonlinear PI-controller of the previous section, is in their noise rejection capabilities. Figure 77 illustrates how the noise is rejected in much less time than it took the nonlinear PI-controller of figure 64.

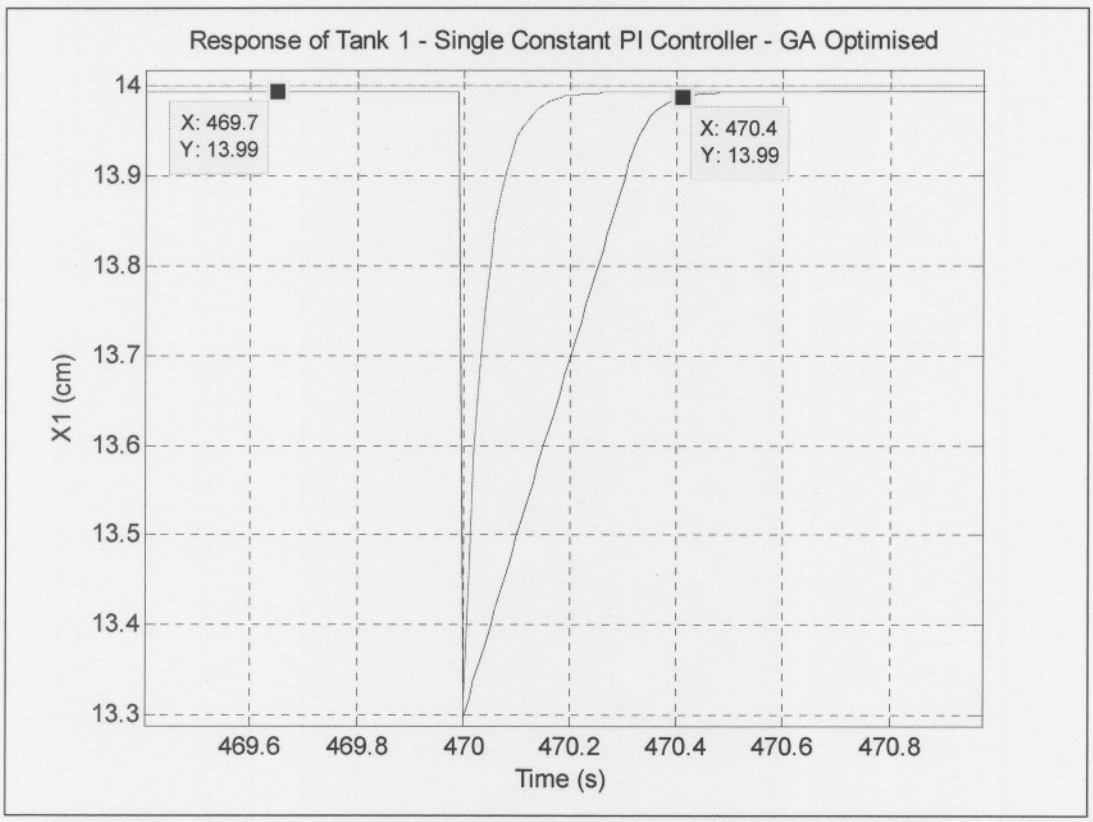

Figure 77: Experiment 1 – tank 1 noise rejection using optimised PI-controller

When comparing the above results with that of figure 64, the value of an optimal solution becomes apparent; improved dynamic response and noise rejection.

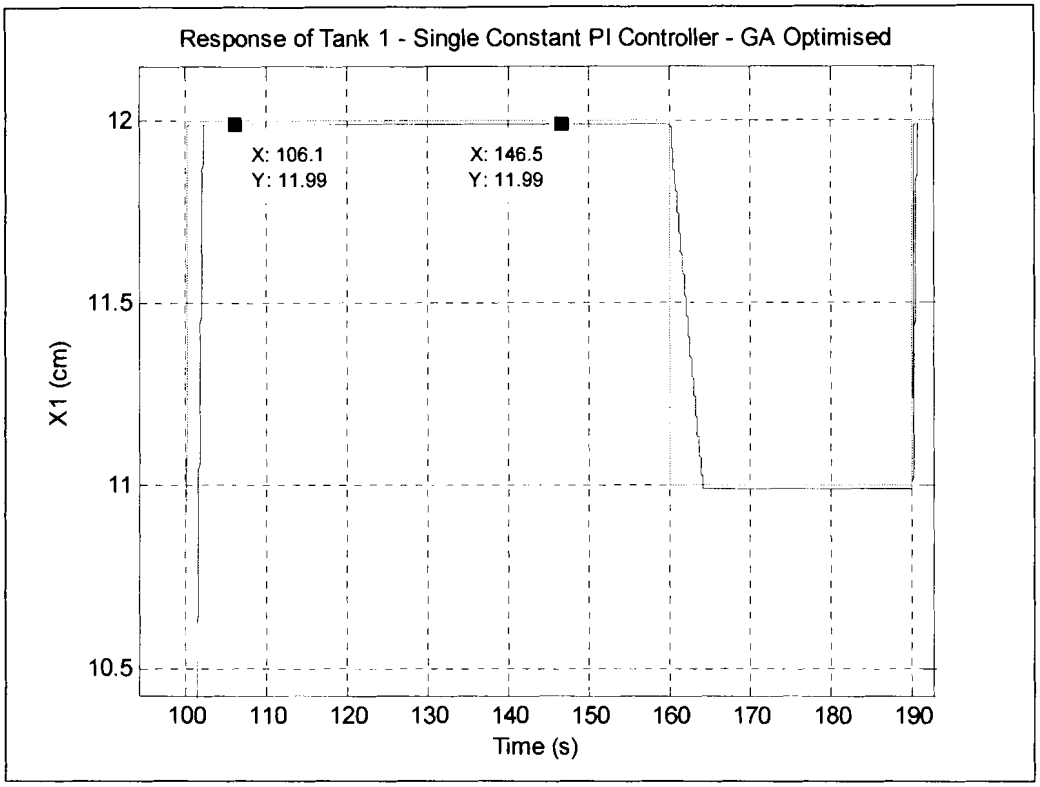

Figure 78: Experiment 1 – tank 1 same-height response, optimised PI-controller

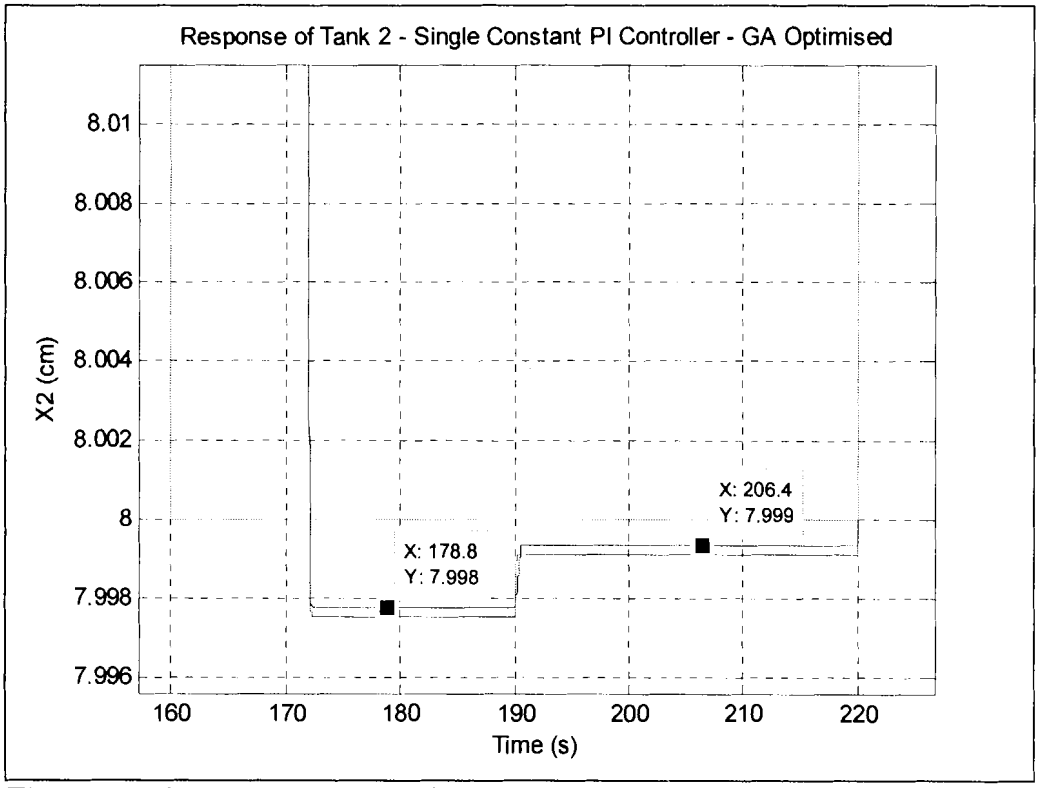

Figure 79: Experiment 1 – tank 2 same-height response, optimised PI-controller

For all practical purposes, tank 1 and tank 2's height remained the same when changing the other tank's height. This illustrates the cross-coupling effect being completely eliminated while still having an integrating action which will over time make the steady-state error zero.

In conclusion, the response of the optimised linear PI-controller, nonlinear PI-controller and nonlinear P-controller are compared to one another with regard to the test scenario where both tanks are to be controlled at the same height (scenario (b)), i.e. both tanks are to be controlled at 13 cm.

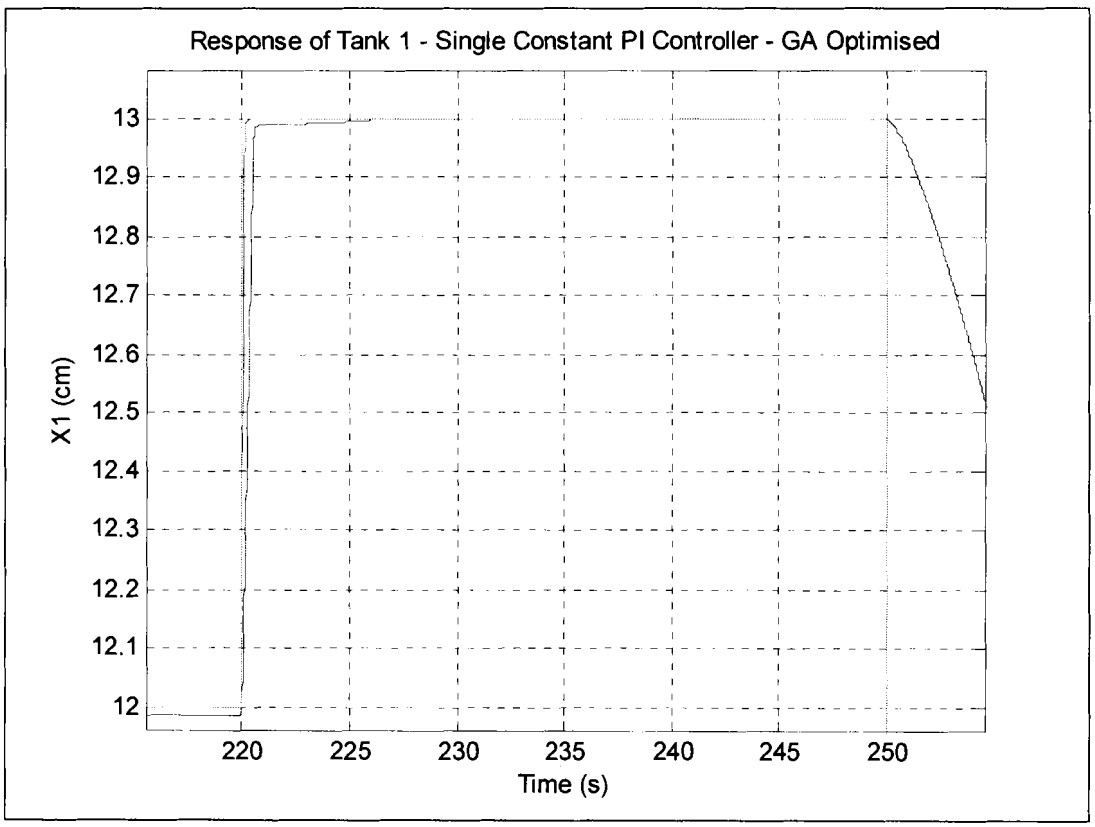

Figure 80: Experiment 1 – tank 1 scenario(b) response, optimised PI-controller

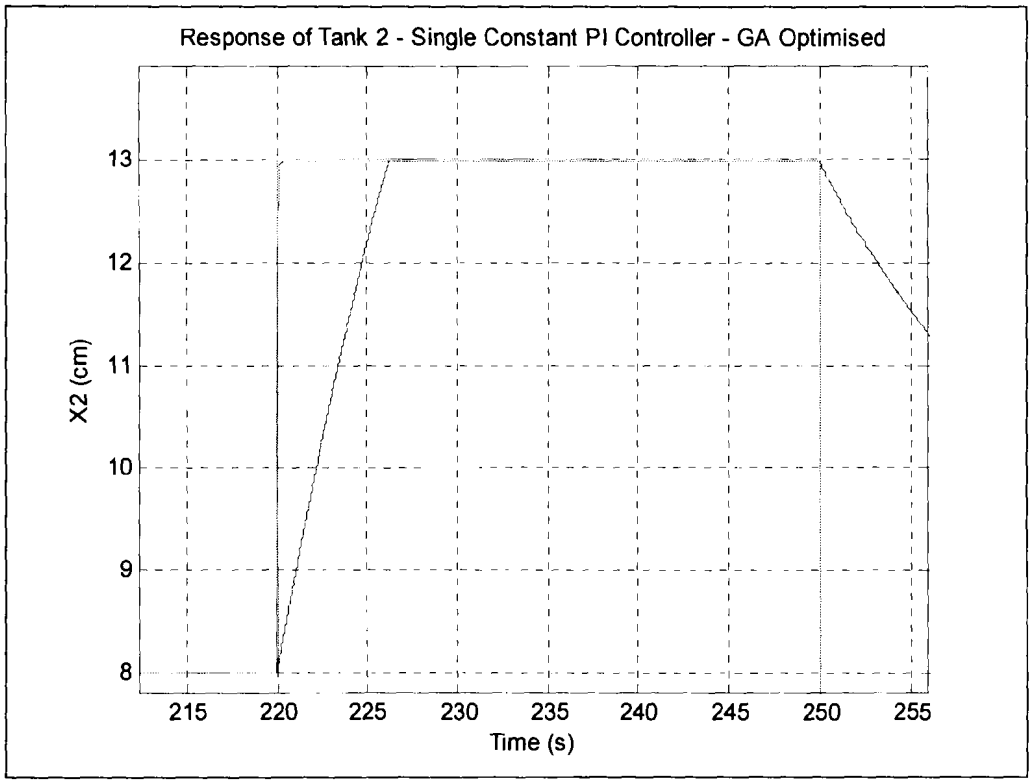

Figure 81: Experiment 1 – tank 2 scenario(b) response, optimised PI-controller

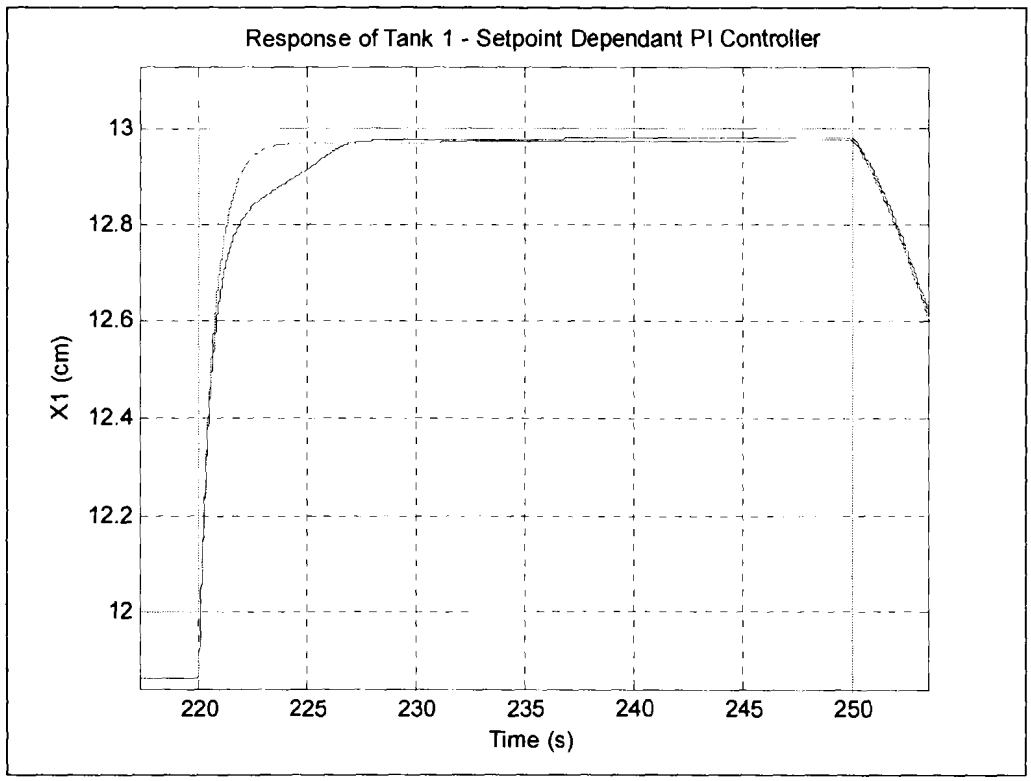

Figure 82: Experiment 1 – tank 1 scenario(b) response, PI-controller, nonlinear

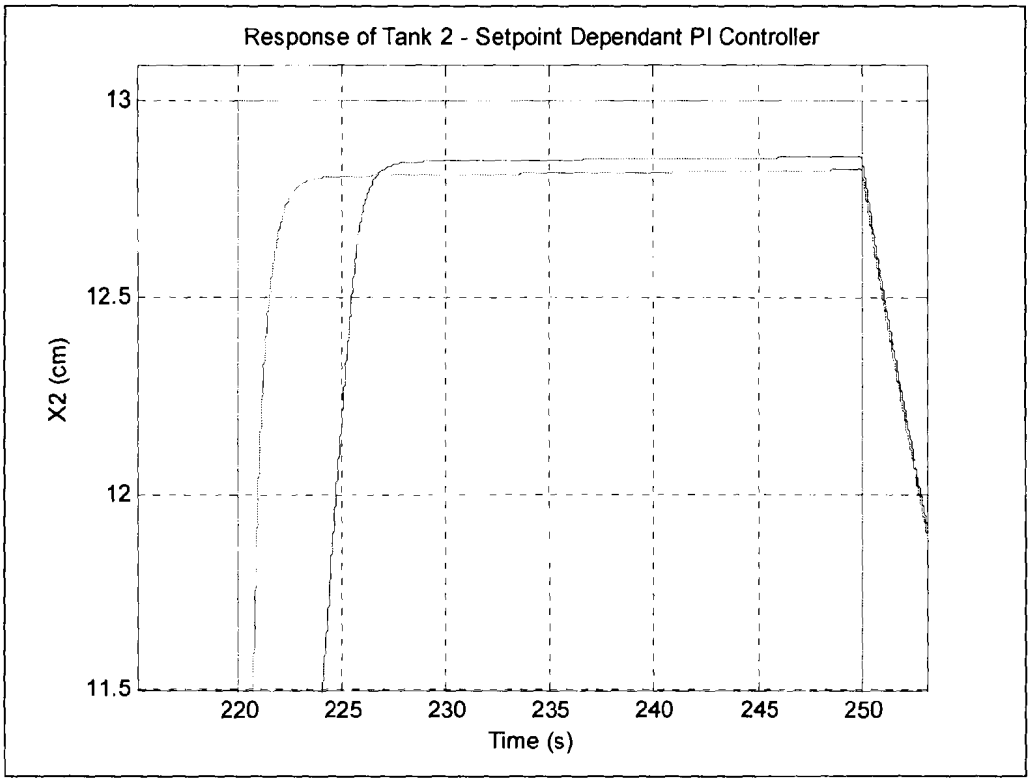

Figure 83: Experiment 1 – tank 2 scenario(b) response, PI-controller, nonlinear

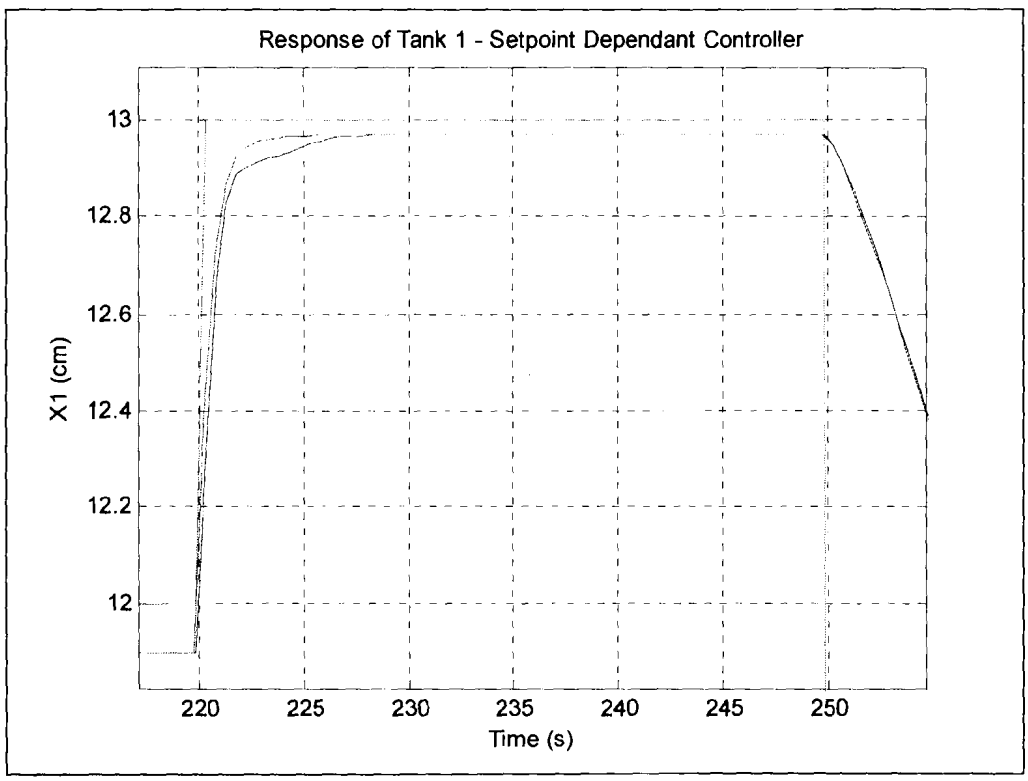

Figure 84: Experiment 1 – tank 1 scenario(b) response, P-controller, nonlinear

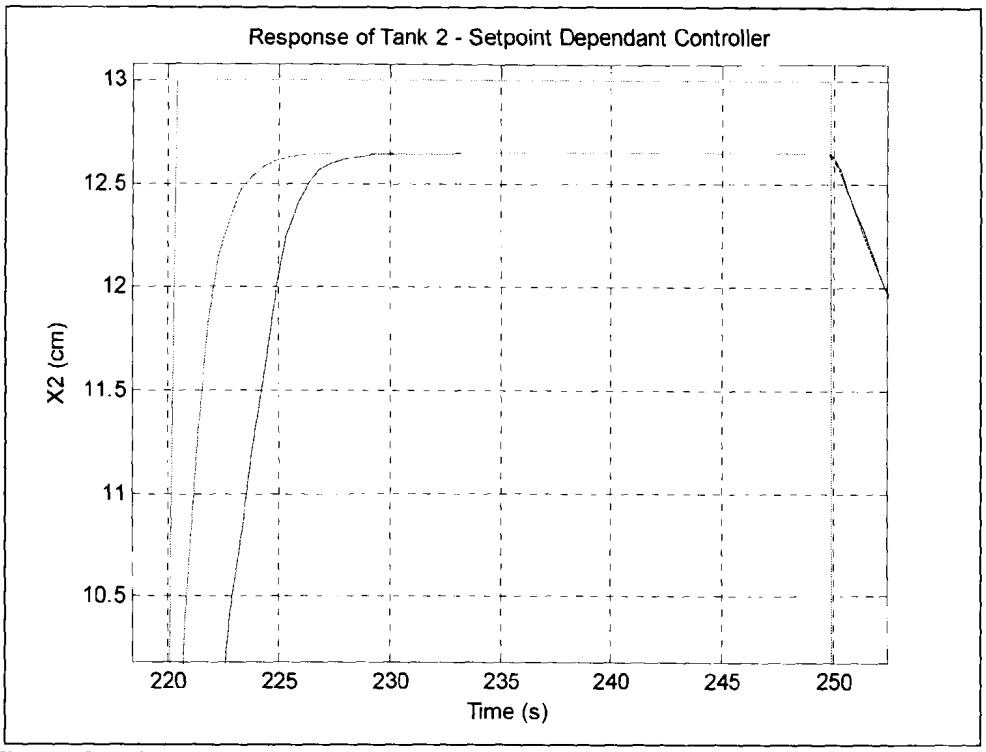

Figure 85: Experiment 1 – tank 2 scenario(b) response, P-controller, nonlinear

4.2.5 Discussion

Through trial-and-error it was found that moving the poles any significant amount left from their current locations would result in an unrealisable system.

The error, i.e. the leak or the eraser dropped into the tank, has a maximum value because it has this value only when it occurs; thereafter the controller rejects the error, and it is suppressed toward zero.

The control in this case being the inflow into each tank, as stated, can only be zero or greater than zero up to its maximum limit. The illustrations of the controls/control signals are evident of this.

The actuators for each tank can only control the inflow or stop it completely, it cannot reverse the inflow, i.e. extract liquid from the tank. This is why the system responds too slowly for it to reach its reference values from its position before 250 seconds, to its new reference values over 250

seconds to 270 seconds; the liquid has to run out naturally. The system can be made too reach this reference value if the controls can be negative, i.e. water can be sucked out of the tanks as well.

Thus, in order to improve the system response considerably, the current actuators can be placed at the bottom of each tank. Thought must then just be given regarding the actuator (pump/valve) used which will then be subjected to backpressure.

The difference in performance between the linear and nonlinear controllers in this experiment is very small; reason being, no great difference between the linear controller parameter values and that of the nonlinear controller at any given operating point. This system, thought nonlinear, has inherent smooth behaviour. The motivation for a nonlinear controller will become evident the moment one has a system which responds more variable, i.e. significantly different at different operating points.

The manner in which reset windup was dealt with, was by resetting the integrator associated with the I-constant of the PI-controller, causing its value to become zero each time en error occurred larger than the preset threshold. Tank 2 was not affected by reset-windup, and therefore its threshold value was set so high. Given the fact each tank has a theoretical range of 14 cm, an error greater than 14 cm is impossible, the tanks' size will not allow it, and neither will the controller.

The steady-state error of the PI-controller will decrease from the value on the figures to zero given the integrating action.

Even thought the nonlinear PI-controller has similar control values, the optimised linear PI-controller functions the best of all the control strategies, with better noise rejection and dynamic behaviour especially:
Faster rise time;
Faster overall transient response;
Improved steady-state error, i.e. achieving a zero percent steady-state error;

When comparing the response of the system for the test scenario where the one tank's height is changed while the other remains the same, very little to no disturbance is visible in the tank which remains at the same height.

This GA algorithm was not used as a nonlinear controller, i.e. an operating point dependant controller, because of the computational expense due to the problem being nonlinear. As stated in chapter 1, the training process is too time consuming to optimise the controller for every operating point. It is therefore not viable as an optimisation method for very nonlinear processes where the multivariable controller **must** be operating point dependant.

In an attempt to circumvent this problem, the GA was trained using the generated reference signal which included a multiple of operating points. Still, is a single controller irrespective the operating point.

The combined characteristics of the smaller average value of the controls plus the improved noise-rejection capabilities and ability to achieve a zero steady-state error, make the PI-controller the controller of choice for this system.

4.3 Experiment 2: System identification

4.3.1 Experiment description

In the previous experiment, the state-space equations were available. It does often occur that this is not the case. In this experiment the same twin-tank plant as in the previous experiment is used, but handled as a black box model in SIMULINK®.

The purpose of this experiment is to show that the developed design methodology is readily used with existing system identification methods.

The first step is to implement the system identification algorithm to obtain its state-space representation. Given that it is a nonlinear system, the proposed identification method does not

give a nonlinear state-space representation, but instead a linearised state-space representation of the unidentified plant around specified operating points.

4.3.2 Controller design

What we effectively have, is an unidentified system in SIMULINK® which we in this case know, but otherwise would have presumed to be nonlinear. We further know its operating range, and consequently, operating points of importance.

Recalling the way the controllers were designed in the previous example, if we can achieve the same state-space equation for every reference input vector, the design would follow identical to the previous experiment.

Thus, the system identification was not done at every operating point across its entire range. Instead, four indicative operating points were selected, see table 6. The resulting matrices, **A** and **B** are compared to those calculated in experiment 1 using the techniques proposed in [28], for the same operating points. If they are identical, or very nearly identical, the technique is considered to be successfully validated. Other than inspection, the matrices can be considered the same if they yield the same open-loop poles.

The chosen algorithm to perform the system identification is MATLAB®'s 'linmod2'. It obtains a linearised state-space model by taking the operating point, and then perturbing the model inputs and model states of the SIMULINK® model.

4.3.3 Test Scenarios

The operating points used to validate the system identification algorithm are tabulated below:

Operating points	Tank 1	Tank 2
Set 1	3	2
Set 2	7	5

Operating points	Tank 1	Tank 2
Set 3	13.245	11.927
Set 4	14	9

Table 6: Operating point sets

4.3.4 Results

Using 'linmod2' a linearised state-space model is retrieved of the black box system around the operating points tabulated above. The **B**-matrix for each operating point set was identical for both experiment 1 and experiment 2.

With

$$\mathbf{B} = \begin{bmatrix} 0.0064 & 0 \\ 0 & 0.0064 \end{bmatrix}.$$

The results for **A** were:

Operating points	Experiment 1	Experiment 2
Set 1	$\mathbf{A} = \begin{bmatrix} -0.0754 & 0.0755 \\ 0.0754 & -0.1155 \end{bmatrix}$	$\mathbf{A} = \begin{bmatrix} -0.0754 & 0.0755 \\ 0.0754 & -0.1155 \end{bmatrix}$
Set 2	$\mathbf{A} = \begin{bmatrix} -0.0534 & 0.0534 \\ 0.0534 & -0.0787 \end{bmatrix}$	$\mathbf{A} = \begin{bmatrix} -0.0534 & 0.0534 \\ 0.0534 & -0.0787 \end{bmatrix}$
Set 3	$\mathbf{A} = \begin{bmatrix} -0.0657 & 0.0657 \\ 0.0657 & -0.0821 \end{bmatrix}$	$\mathbf{A} = \begin{bmatrix} -0.0657 & 0.0657 \\ 0.0657 & -0.0821 \end{bmatrix}$
Set 4	$\mathbf{A} = \begin{bmatrix} -0.0337 & 0.0337 \\ 0.0337 & -0.0526 \end{bmatrix}$	$\mathbf{A} = \begin{bmatrix} -0.0337 & 0.0337 \\ 0.0337 & -0.0526 \end{bmatrix}$

Table 7: State-space matrices from system-identification vs. linearised from experiment 1

An initial value response for the uncontrolled system is illustrated below. Tank 1 has an initial value of 7 cm and tank 2 an initial value of 4 cm.

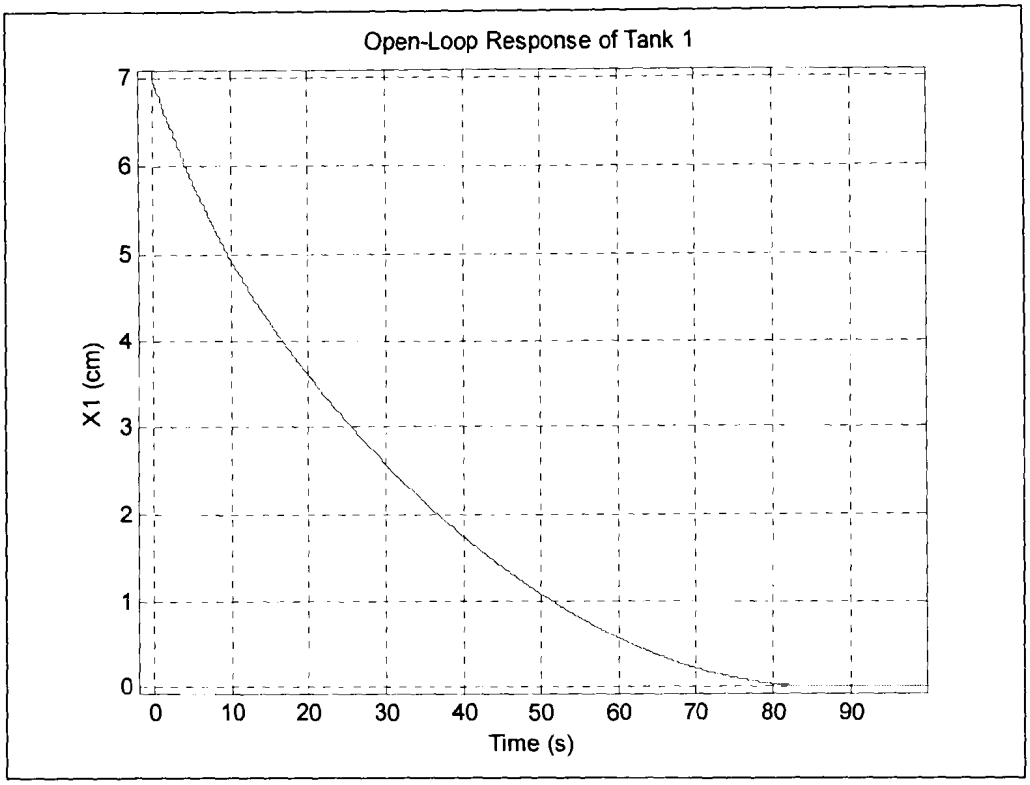

Figure 86: Experiment 2 - initial value response, tank 1 at 7 cm

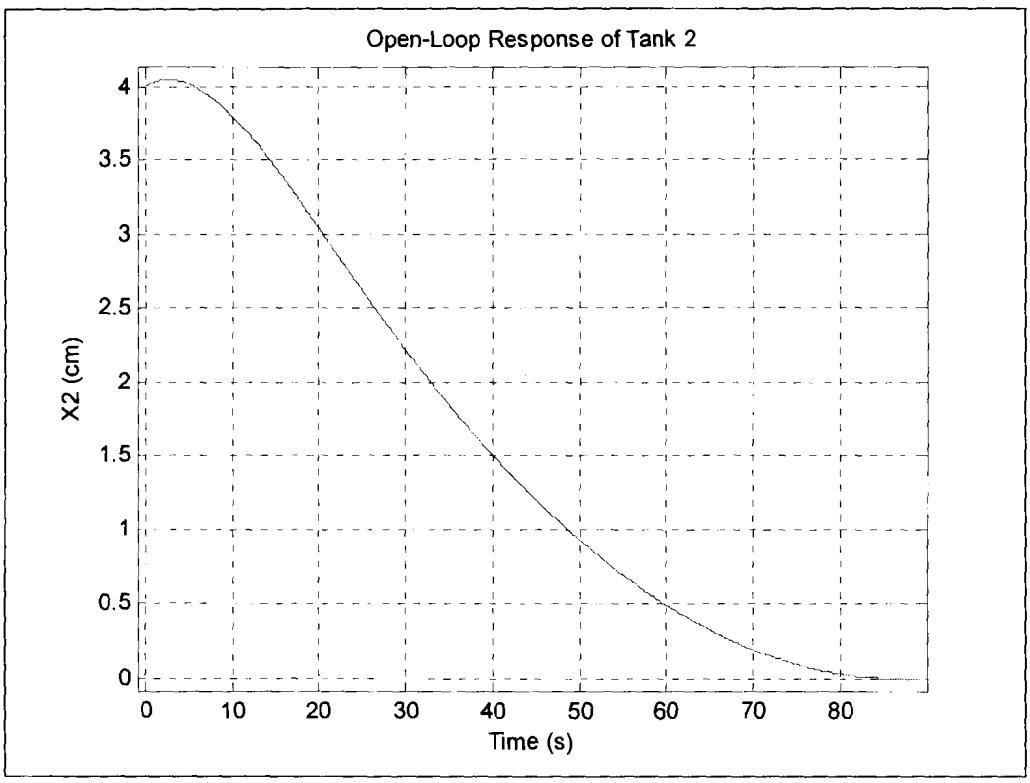

Figure 87: Experiment 2 - initial value response, tank 2 at 4 cm

When one compares this with the responses of the previous experiment for the controlled system, the motivation and necessity for controlling the system becomes apparent.

4.3.5 Discussion

The system identification algorithm yields the same state-space equations around each of the four operating point sets as the calculation method used in experiment 1. Therefore it is assumed to do likewise for the other operating points in the operating range.

This is useful, because it enables the use of the developed design methodology on black box SIMULINK® models.

4.4 Experiment 3: Pressure and level control of a plant

4.4.1 Experiment description

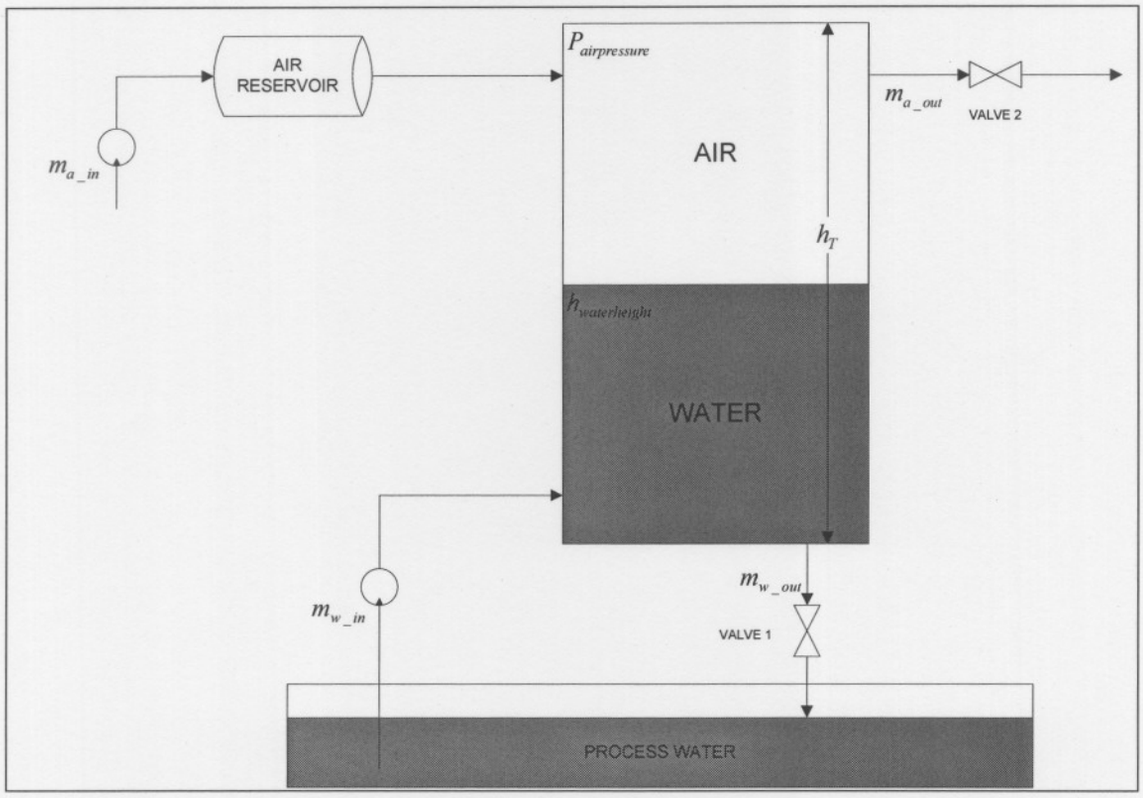

Figure 88: Experiment 3: Pressure-level control

This experiment was inspired by the pilot plant used in [37]. The above figure illustrates our model. The tank comprises a water part and an air part. Various plant processes can be attached to it via valves 1 and 2. The water is reticulated through the system, ending up in the process water tank after use, where it is available to be pumped back into the tank. The varying loads imposed on the tank by the processes can be modelled as disturbances at the operating points. In this experiment an undisturbed system was assumed, but the system's response to changing operating points was considered. The air reservoir was placed in the system as it is lengthens the time constant of the pressure loop and should smooth any pulses in the air flow [37].

The two states to be controlled by the multivariable controller is the height of the water part of the tank and the air pressure in the tank. They are respectively controlled by changing the water flow and air flow into the tank.

For the water part there was assumed that water is always available to be pumped into the tank. The rate at which the water pump pumps the water is not a function of the water in the process water tank. The compression of the water is negligible.

For the air part there was assumed that sonic flow of the gas (air) does not occur and that the process is controlled under isothermal conditions with air considered to be an ideal gas [38].

For the process to start up and be controlled, there has to be some initial values of water and air present in the tank.

The table below indicates the variables' designation for this experiment:

Name	Description
m_{a1}	Mass of water in C_1 (kg)
m_{a2}	Mass of air in reservoir (kg)
m_{a3}	Mass of air in C_2 (kg)
C_1	Water part of the tank
C_2	Air part of the tank
P	Pressure (kPa)
V	Volume (m^3)
m	Mass of gas (kg)
R	Gas constant $(\frac{J}{kg \cdot K})$
T	Temperature (K)
$P_{barometer}$	Pressure due to atmosphere (kPa)
$P(t)$	Measured (sensor) Pressure (kPa)
P_{a_abs}	Absolute air pressure (kPa)
K_1	Valve 1 constant
K_2	Valve 2 constant
$h(t)$	Measured height of water in tank (m)
h_T	Total tank height (m)
ρ	Water density $(\frac{kg}{m^3})$
g	Gravity acceleration $(\frac{m}{s^2})$
V_{res}	Volume of air reservoir (m^3)

Name	Description
A	Area of tank (m^2)
m_{a_in}	Air flow into tank ($\frac{kg}{s}$)
m_{a_out}	Air flow out of tank ($\frac{kg}{s}$)
m_{w_in}	Water flow into tank ($\frac{kg}{s}$)
m_{w_out}	Water flow out of tank ($\frac{kg}{s}$)
h_{op}	Steady-State water height (m)
$h_{equivalent}$	Height equivalent of water in the tank for air pressure (m)
β_{op}	Steady-State air Pressure in terms of $h_{equivalent}$ (kPa)
P_{op}	Steady-State Air pressure
V_{water_op}	Volume of water at operating point (m^3)
m_{a1_op}	Mass of water at operating point (kg)
$m_{w\_out\_op}$	Outflow of water by operating point ($\frac{kg}{s}$)
V_{C2}	Volume of the air part of the tank (m^3)
V_{tot_op}	Total volume of air at operating point (m^3)
ω_n	Natural frequency of the system ($\frac{rad}{s}$)
τ_{short}	Shortest time constant in the system (s)

Table 8: Experiment 3 - symbols

Where the ideal gas law was taken to be $PV = mRT$.

4.4.2 Controller design

In this experiment a state-variable model did not exist prior design, hence a model of the system using state-variable equations first had to be derived. To this end, balance equations were used.

Water part:

Dynamic behaviour of the water part is given by the equation:

$$m_{w\_in} - m_{w\_out} = \frac{d}{dt} m_{a1}$$

$$m_{w\_in} - K_1 \sqrt{P(t) + \rho g h(t)} = A\rho \frac{d}{dt} h(t)$$

Air part:

Dynamic behaviour of the air part is given by the equation:

$$m_{a\_in} - m_{a\_out} = \frac{d}{dt}(m_{a2} + m_{a3})$$

$$m_{a\_in} - K_2 \sqrt{P(t)} = \frac{V_{res}}{RT} \frac{d}{dt}(P(t) + P_{barometer}) + \frac{A}{RT}(P(t) + P_{barometer}) \cdot \frac{d}{dt}(h_T - h(t)) + \frac{A}{RT}(h_T - h(t)) \cdot \frac{d}{dt}(P(t) + P_{barometer})$$

To determine the constants K_1 and K_2, the system is assumed to be in equilibrium at an operating point, i.e.

$$m_{w\_in} = m_{w\_out}$$

$$m_{a\_in} = m_{a\_out}$$

Considering we do not have the air pressure at operating point, an equivalent water pressure was assumed for the air pressure, i.e. the air pressure was considered as if it were a certain height of water in the given area.

The following constants were assumed:

$\rho = 998.23 \frac{kg}{m^3}$; $g = 9.81 \frac{m}{s^2}$; $A = 1m^2$; $R = 287 \frac{J}{kg \cdot K}$; $T = 293K$; $P_{barometer} = 101.325kPa$;

$h_T = 2m$. R was found to be the specified amount with reference to [39]. Technically, R for moist air should have been used, i.e. R_m but for the purposes of this experiment, R for dry air is sufficient.

The following steps illustrate how K_1 and K_2 are determined using the operating points $h_{op} = 1m$ and

$$\beta_{op} = \rho g h_{equivalent}$$

where $h_{equivalent} = 2m$.

Thus,

$$V_{water\_op} = A * h_{op} = 1 * 1 = 1m^3$$

$$m_{al\_op} = V_{water\_op} * \rho = 998.23kg$$

It was decided, given the tank initially contained no water, the water needs to fill up to $h_{op} = 1m$ in 60 seconds if no water flows out of the tank. This gives us the required outflow of

$$m_{w\_out\_op} = \frac{998.23}{30} = 33.27\frac{kg}{s}.$$

Solving for $m_{w\_in} = m_{w\_out}$, yields $33.27 = K_1\sqrt{\beta_{op} + \rho g h_{op}}$ giving

$$K_1 = 0.1941$$

To determine K_2, the ideal gas law is used.

$$P_{a\_abs} = P_{barometer} * \beta_{op}$$

With

$$V_{C2} = (h_T - h_{op}) * A$$

and the volume of the air in the reservoir is chosen to be slightly smaller than that of the air part of the tank, i.e.

$$V_{res} = 0.9V_{C2}$$

This gives the total volume of air at the operating point to be

$$V_{tot\_op} = V_{C2} + V_{res}$$

Now solving the gas law gives an air mass of 0.3512kg. Following the method used for the water, if this mass of air must be moved out in 15 seconds, it would yield a required flow of $0.0234\frac{kg}{s}$.

Thus, solving for $m_{a\_in} = m_{a\_out}$, yields $0.0234 = K_2\sqrt{\beta_{op}}$ giving

$$K_2 = 1.6728 * 10^{-4}$$

This then yields **the state space equations**:

$$\frac{d}{dt}h(t) = \frac{1}{A\rho}\left(m_{w\_in} - 0.0941\sqrt{P(t) + \rho g h(t)}\right)$$

$$\frac{d}{dt}P(t) = \frac{1}{V_{res} + Ah_T - Ah(t)}\left(RT\left(m_{a\_in} - 1.6728 * 10^{-4}\sqrt{P(t)}\right) + \frac{d}{dt}h(t)\left(AP(t) + AP_{barometer}\right)\right)$$

From here, the design methodology developed in the previous chapter can be directly applied. MATLAB® was used to determine the partial derivatives necessary to yield the matrices **A** and **b** using its 'diff' function. This would yield the same matrices as if the techniques of [28] would have been used.

The problem, is that there is a derivative in the right-hand side of the equation for $\frac{d}{dt}P(t)$. The way this was dealt with, was by making that element in the equation a constant value designated by a constant, *const*. The motivation will be explained in the discussion of this experiment.

const was chosen to be $0.1\ \frac{m}{s}$.

The resulting **A** and **b** matrices:

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

and

$$\mathbf{b} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

A_{11} = -1/2/A * K_1/(P_{op} + ρ * g * h_{op})^(1/2) * g

A_{12} = – 1/2/A/ρ * K_1/(P_{op} + ρ * g * h_{op})^(1/2)

A_{21} = 1/(V_{res} * A * h_T - A * h_{op})^2 * (R * T * (m_{a_in} - K_2 * P_{op}^(1/2)) + *const* * (A * P_{op} + A * $P_{barometer}$)) * A

A_{22} = 1/(V_{res} * A * h_T - A * h_{op}) * (-1/2 * R * T * K_2/P_{op}^(1/2) + *const* * A)

Using these matrices, a PI-controller is developed for both the pressure and the height control for various operating points across the system's operating range. Four operating points were chosen, and are tabulated below.

Operating points	Reference water height (m)	Reference pressure (kPa)
Set 1	1	14700
Set 2	0.4	19320
Set 3	1.5	10111
Set 4	1.3	18001

Table 9: Experiment 3 – reference sets

The same design principle as in figure 1 was followed. Being a second order system, the controller has the same form as the controller, **K** from experiment 1. The closed-loop characteristic polynomial can then be generated as

$$CLCP = \Delta_c(s) = \det(s\mathbf{I} - \mathbf{A} + \mathbf{bK})$$

using the linearised matrices, **A** and **b** around operating points, and

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The coefficients can then be compared to that of

$$CLCP_{desired} = s^4 + 2.1\omega_n s^3 + 3.4\omega_n{}^2 s^2 + 2.7\omega_n{}^3 s + \omega_n{}^4$$

where

$$\omega_n = 2\Pi\left(\frac{1}{0.1\tau_{short}}\right) = 4.189 rad/s$$

Thus,

$$CLCP_{desired} = s^4 + 8.796s^3 + 59.6563s^2 + 198.44s + 307.861$$

The PI-controller was again chosen to be $\mathbf{K} = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} k_{P1}e_1(t) + k_{I1}\int_0^t e_1(t)dt \\ k_{P2}e_2(t) + k_{I2}\int_0^t e_2(t)dt \end{bmatrix}$.

$e_1(t)$ being the error between the reference level and measured water head and $e_2(t)$ the error with regard to the pressure, k_{P1} and k_{I1} the gain and integrating parameters of the level controller, and k_{P2} and k_{I2} the pressure controller parameter values.

The integrating constants for the pressure and level were respectively chosen

$$k_{I1} = 0.2\tau_{short}$$
$$k_{I2} = 2\tau_{short}$$

The controller is designed for the various operating points, see table 9, and implemented in the nonlinear model of the system. Since it adjusts its value depending on the operating point, it is nonlinear in same fashion as the controllers of the first experiment.

4.4.3 Test Scenarios

The model will be validated to show that the equilibrium state used for the derivation is in fact achieved. This will also illustrate the open-loop uncontrolled response of the system revealing the system's transient behaviour.

The system will then be given an initial state, and controlled at the reference points, table 9, which change during the simulation according to the criteria set below. These changes will be exacted in the following way:

- The water height reference will remain constant while changing the pressure reference.
- The pressure reference will remain the same while changing the water height reference

These criteria for the changes will be:

The reference points are set from the start of simulation. The pressure is increased by a value of 2000 kPa at 300 seconds and the water head is decreased by 0.3 m at 500 seconds.

This will illustrate the cross-coupling of the two controllers due to the height of the water and the pressure in the tank directly influencing each other.

4.4.4 Results

The following parameter values were used:

$\rho = 998.23\frac{kg}{m^3}$; $g = 9.81\frac{m}{s^2}$; $A = 1m^2$; $R = 287\frac{J}{kg \cdot K}$; $T = 293K$; $P_{barometer} = 101.325kPa$;

$h_T = 2m$; $K_1 = 0.1941$; $K_2 = 1.6728 * 10^{-4}$;

Initial air pressure: $5000kPa$

Initial water head: 0.3 m

Simulation start time: 0 seconds

Simulation stop time: 800 seconds

The results for figure 89 were obtained using reference set 1 from table 9. For the steady-state design to be verified and validated, the system should settle at these reference values if the correct air flow and water flow is given to the system and it operated at under the conditions specified above. In this case, the reference points are *not* changed during the simulation. The resulting response is illustrated.

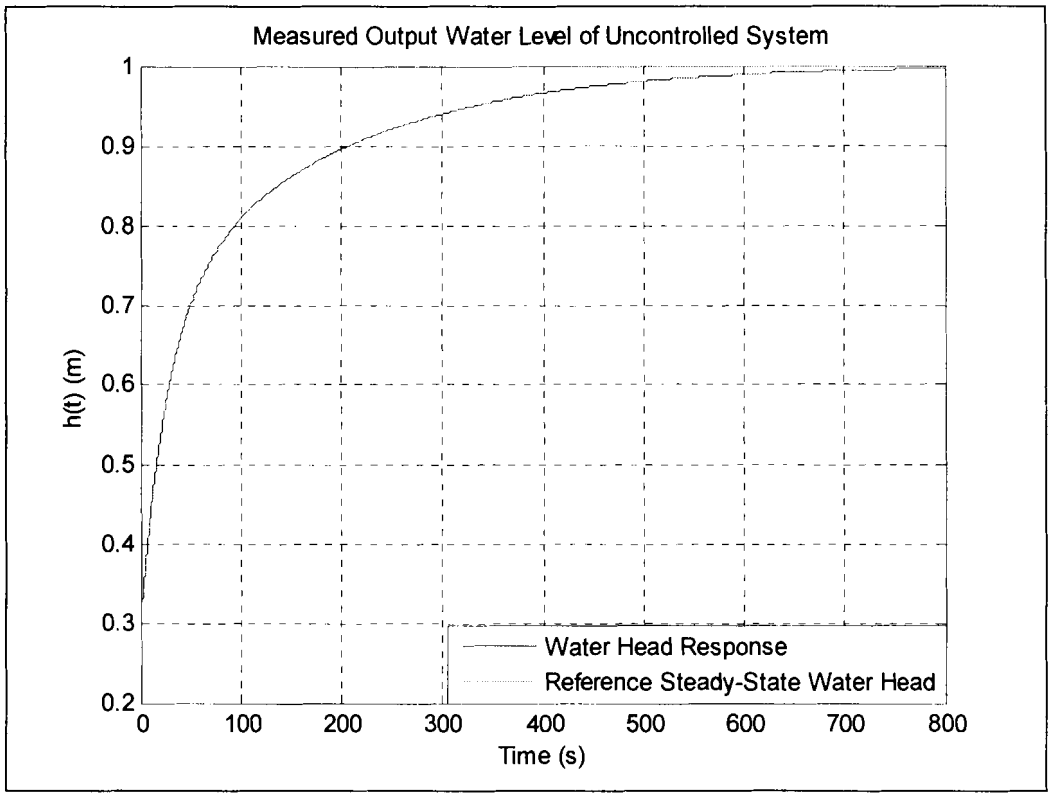

Figure 89: Experiment 3 – open-loop, uncontrolled transient response, water level

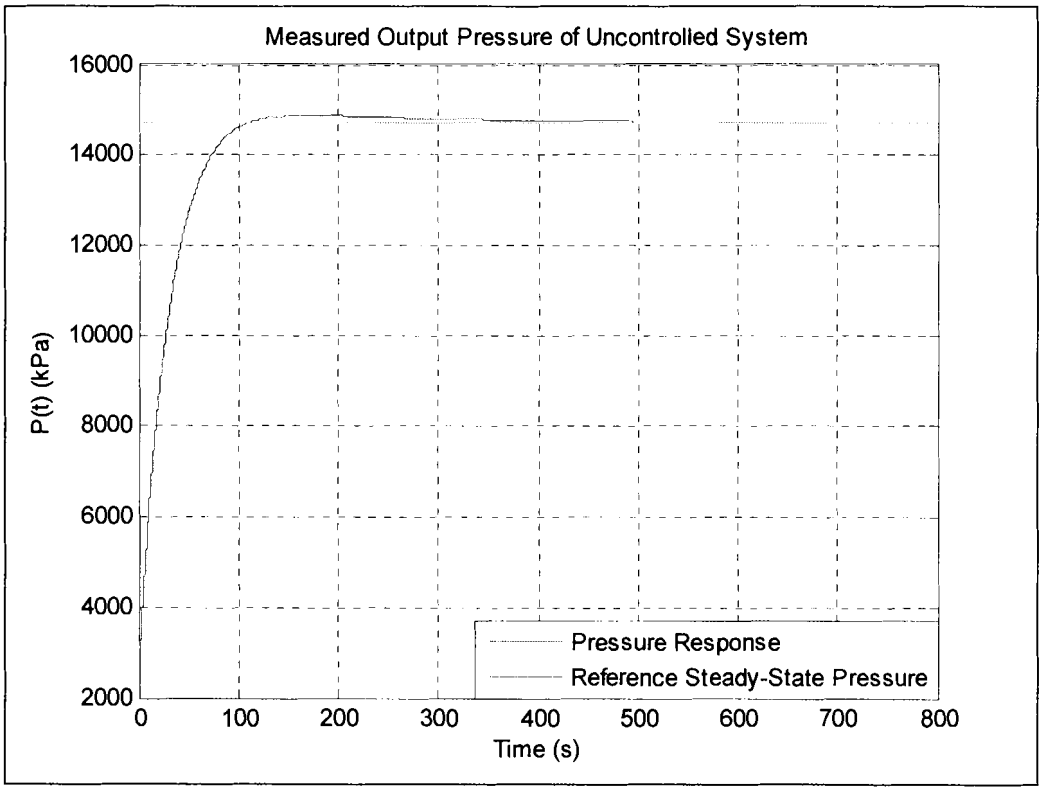

Figure 90: Experiment 3 – open-loop, uncontrolled transient response, air pressure

The closed loop response for the same reference set will now be supplied.

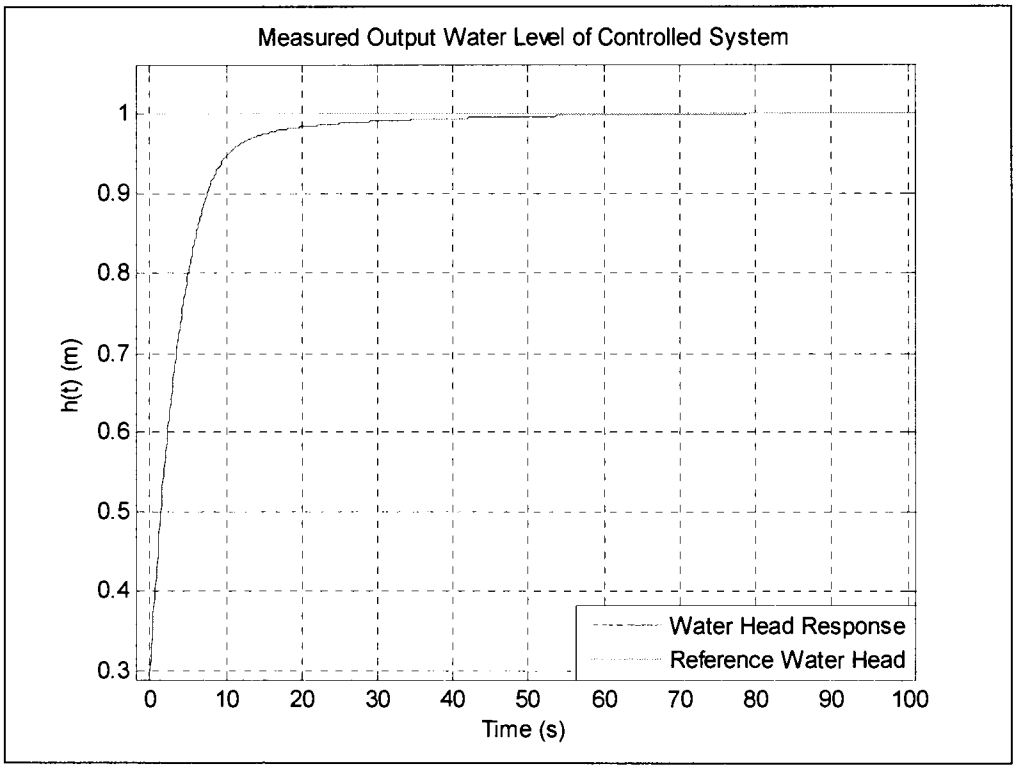

Figure 91: Experiment 3 – closed-loop, transient response, water level

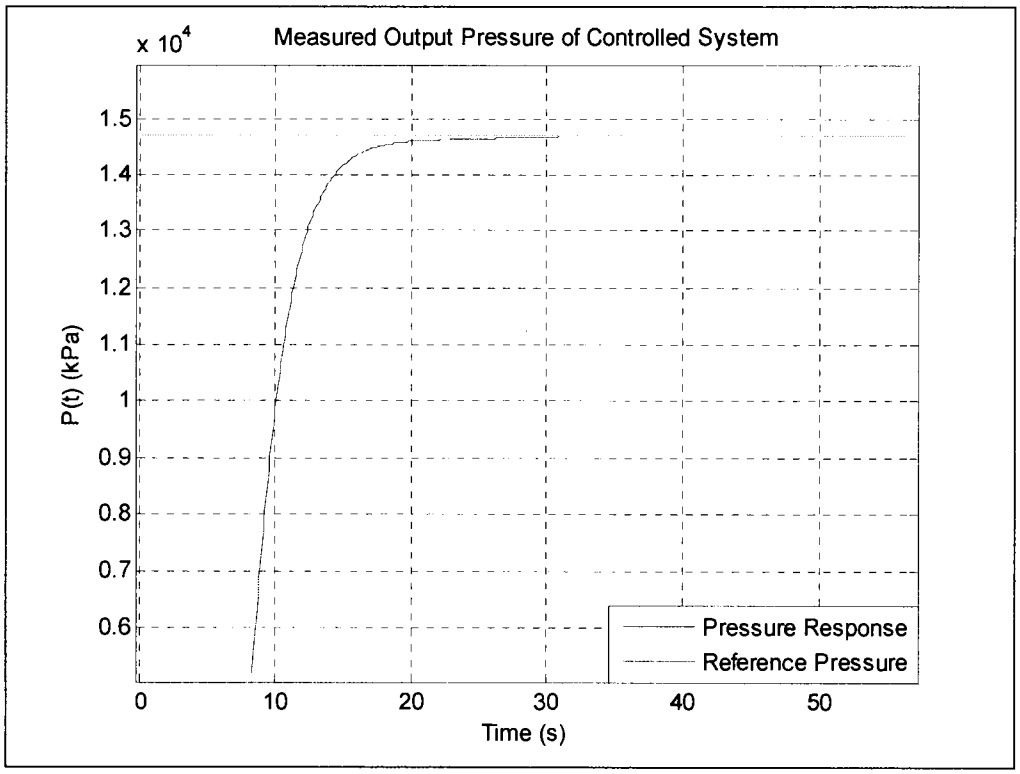

Figure 92: Experiment 3 – closed-loop, transient response, air pressure

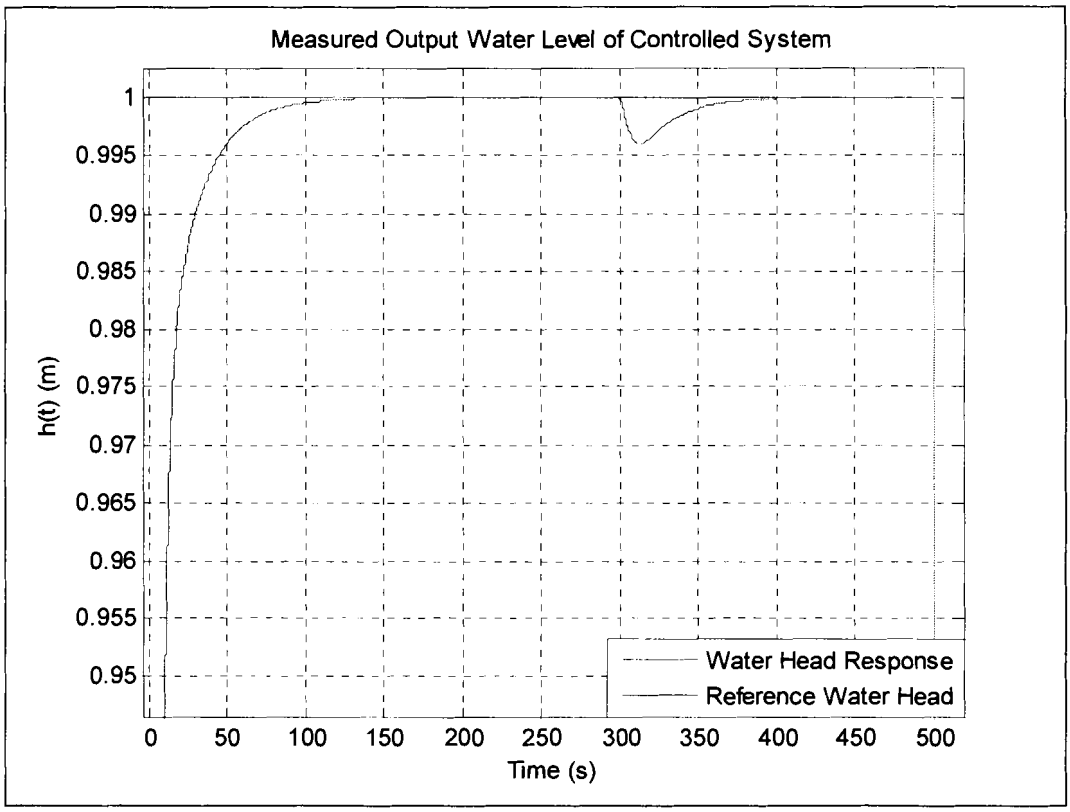

Figure 93: Experiment 3 – closed-loop, cross-coupling rejection, water Level

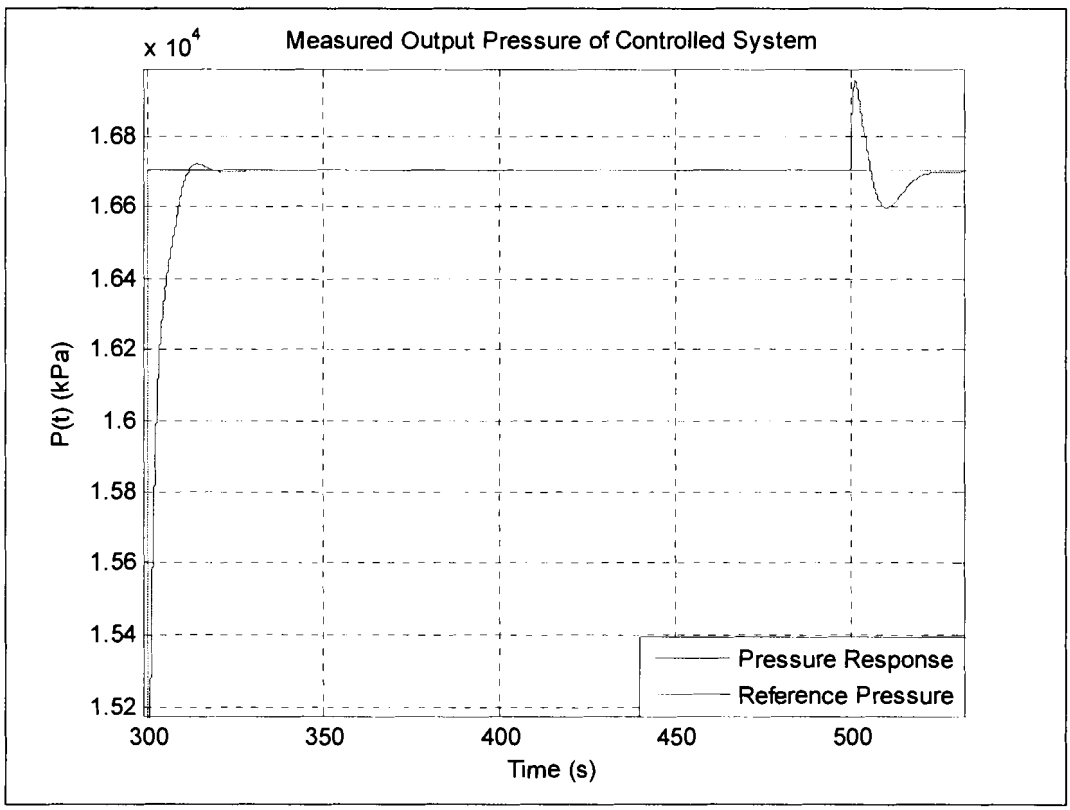

Figure 94: Experiment 3 – closed-loop, cross-coupling rejection, air pressure

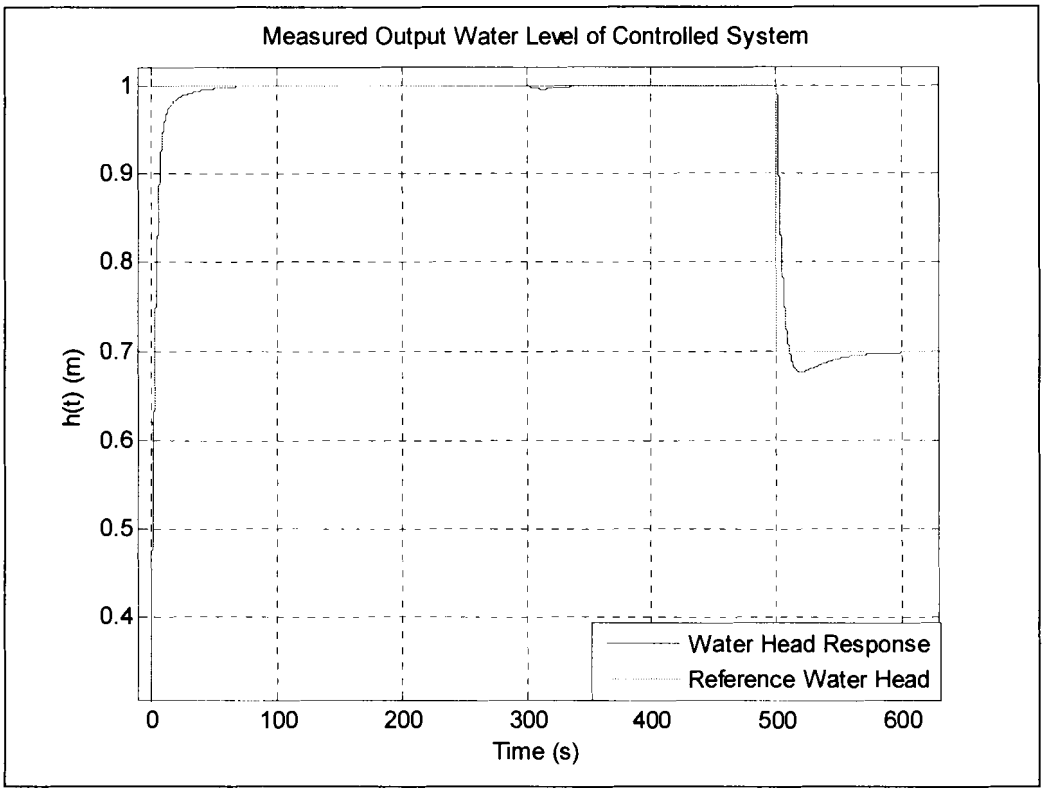

Figure 95: Experiment 3 – closed-loop, changing reference, water level

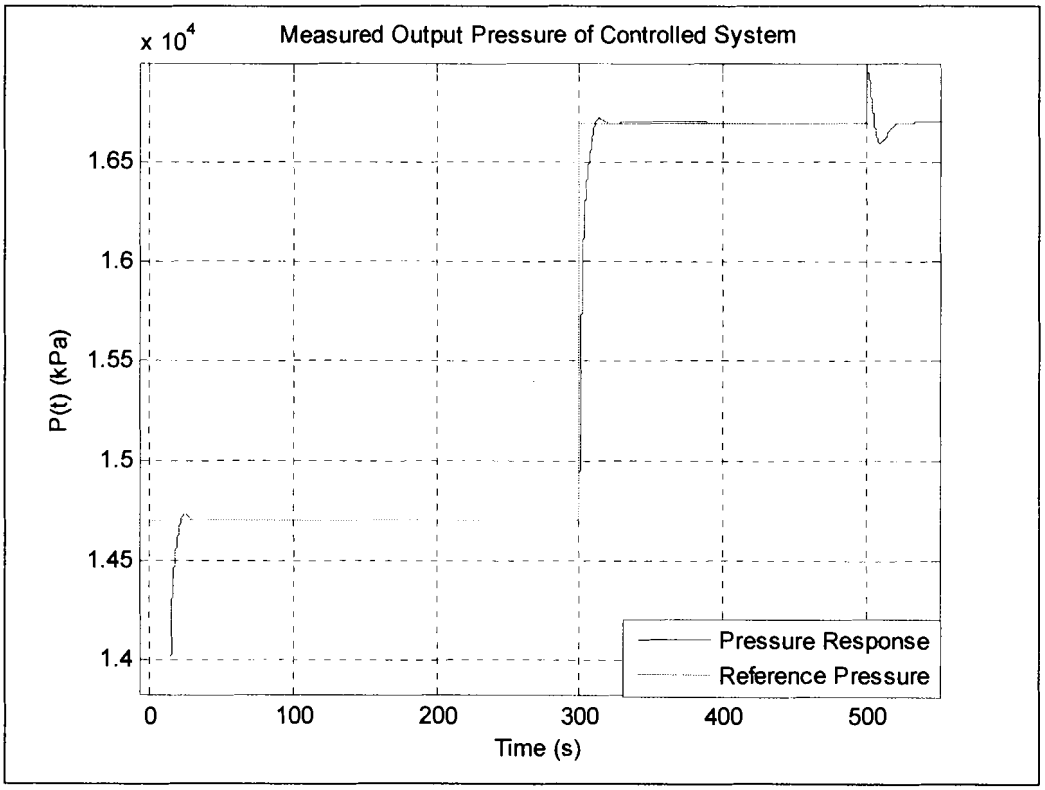

Figure 96: Experiment 3 – closed-loop, changing reference, air pressure

Note the improved settling time, rejection of the cross-coupling effect of the other states and the accurate tracking of the reference signal.

PI-controller optimised with a genetic algorithm

The genetic algorithm was used to optimise the linear PI-controller parameters that make up the multivariable controller **K** using the same initial values and the following settings.

N_{ind} =20;

$GGAP$ =0.9

Selection method: *rws*

Crossover/Recombination technique: Intermediate recombination

Mutation function: mutbga

Performance index: ITSE

> Note:
> The performance index works on the error between the reference value and the actual system output. Thus, one has two ITSE indices, one for the water level and one for the pressure. To use the GA, a single criterion is required. The index for the GA was the sum of the two indices, since both pressure and level is of equal importance. When one variable is of greater importance, the respective index can be scaled relative to the variable of greatest interest, in a sense functioning in the same way as the weighting matrix **H** mentioned by optimal control theory in chapter 2.

For the learning process, a reference signal for the water head and the air pressure was generated respectively which varied, taking each variable through its operating range. For the air pressure this was approximately 13000 kPa to 19000 kPa, and for the water level approximately 0.1 m to 1.2 m.

Since this is an academic investigation, the boundaries for the proportional constants for both pressure and level were chosen to be between 0.001 and 300 and between 0.001 and 30 for the integral constants.

Using the same setup and reference signal as for figures 95 and 96, figure 97 illustrates the distinct improvement of the settling time, at the expense of some overshoot. The cross-coupling effect visible with figure 95 at 300 seconds, is also now sufficiently suppressed with it not visible in the figure below.

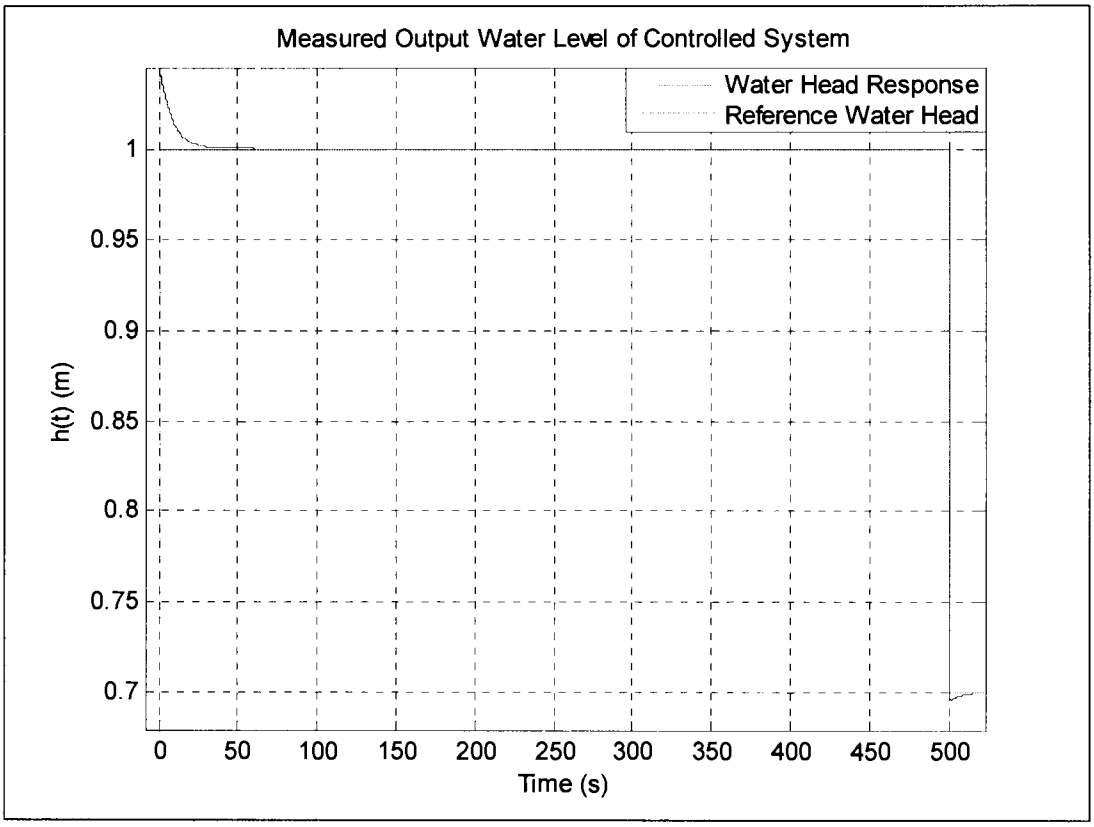

Figure 97: Experiment 3 – closed-loop, changing reference, water level, optimised PI-controller

For the air pressure, the there is still the improved settling time after the reference has been changed, i.e. at 300 seconds.

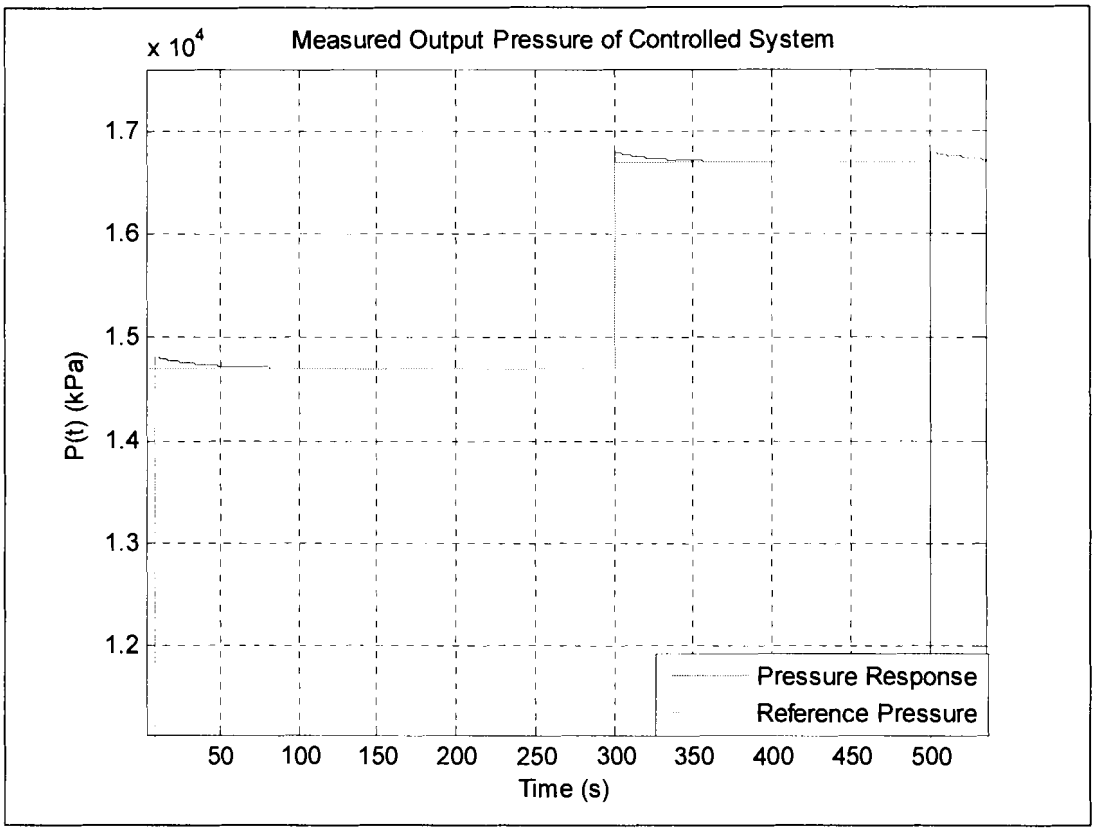

Figure 98: Experiment 3 – closed-loop, changing reference, air pressure, optimised PI-controller

Figure 99 focuses on the occurrence at 500 seconds when the air pressure controller attempts to suppress the cross-coupling effect from the water reference being changed. Notice a transient.

This transient behaviour is referred to as ping-pong vs. onions [40]. It is the result of the order in which MATLAB® and SIMULINK® attempt to solve the differential equations. Recall the derivative of the height of the tank found in the right hand of equation for the pressure, $\frac{d}{dt}P(t)$ in the derived state-variable equations for this experiment; the solution for $\frac{d}{dt}h(t)$ is thus necessary to solve $\frac{d}{dt}P(t)$. We do not have access to the order in which MATLAB® operates on this kind of problem. The time span over which the occurrence takes place also supports this hypothesis.

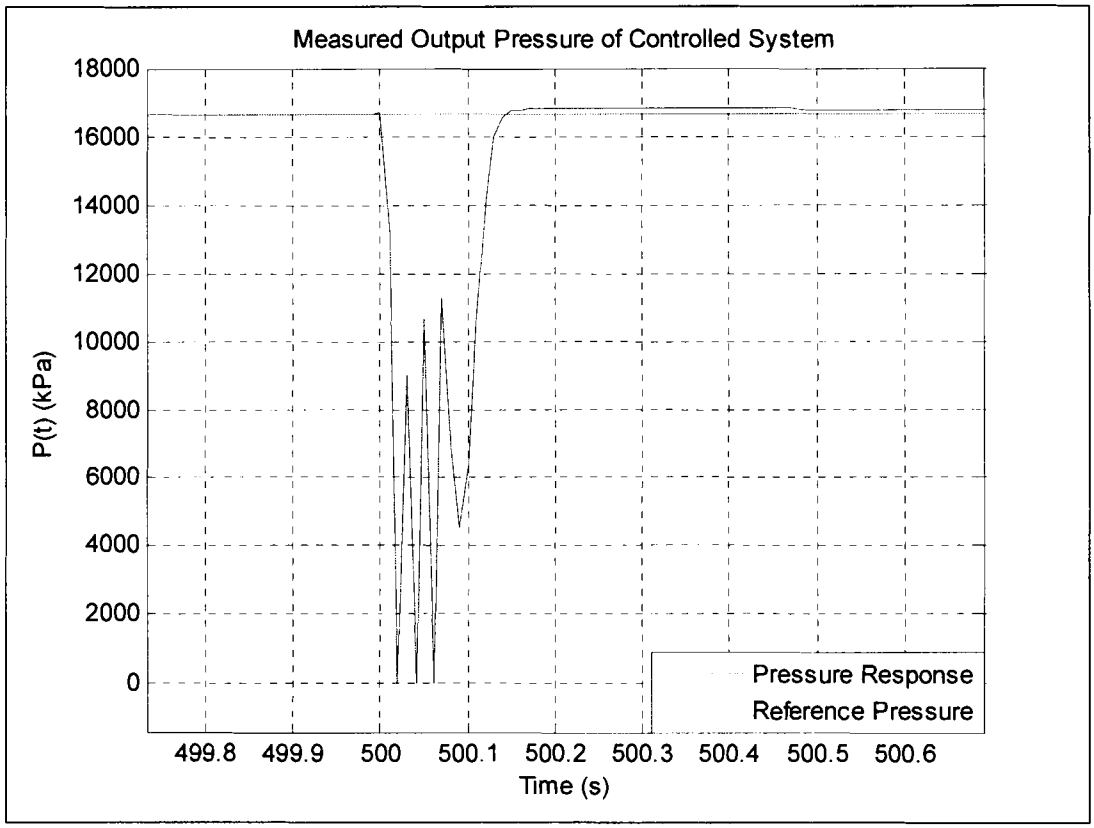

Figure 99: Experiment 3 – Ping-pong vs. onions

The occurrence being due to solving technique, the pressure response to the cross-coupling effect is still improved, settling faster with less overshoot, suppressing the cross-coupling better than the conventionally designed PI-controller used in figure 96, see figure 100.

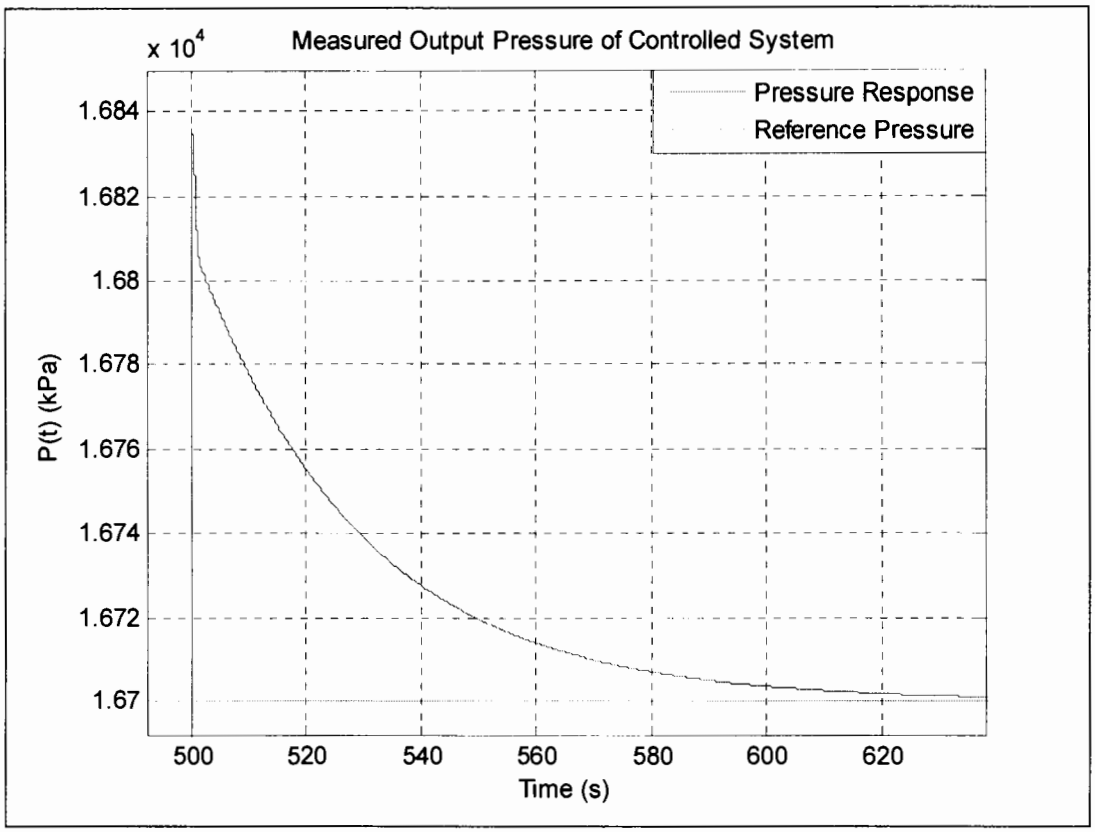

Figure 100: Experiment 3 – Cross-coupling rejection, air pressure, optimised PI-controller

If one considers practical implementation, a correctly sized[10] controller should be designed. For this, the boundaries for the proportional gain constants are between 0.001 and 10, and 0.001 and 3 for the integral constants.

The figures below illustrate the system's response after training.

The transient behaviour and cross-coupling suppression is not as impressive as with the previous case, however, the system will be able to handle greater variation.

[10] See the discussion of this experiment to understand what is meant by the term 'sized'

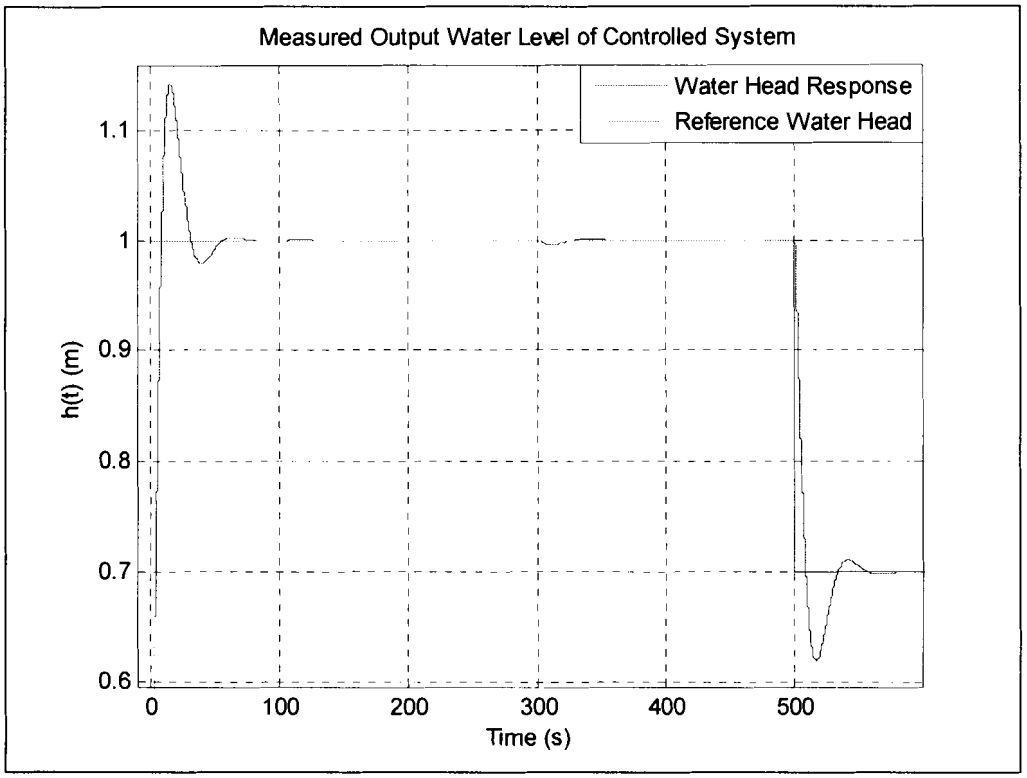

Figure 101: Experiment 3 – changing reference, water level, sized optimised PI-controller

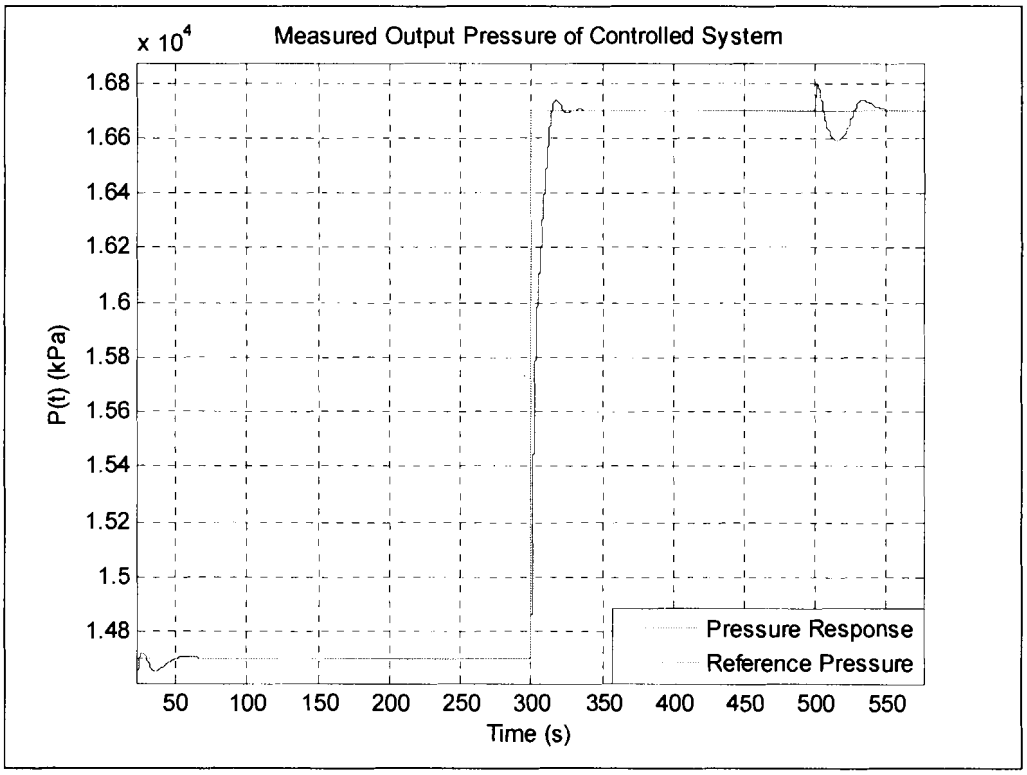

Figure 102: Experiment 3 – changing reference, air pressure, sized optimised, PI-controller

4.4.5 Discussion

The derivative in the right-hand side $\left(\frac{d}{dt}h(t)\right)$, of the equation for $\frac{d}{dt}P(t)$, represents a water head differential. This rate is obviously dependant on the pressure and water head at the time of calculation. However, to simplify the state-space equation, and to have a workable model, an average value for all state values was assumed. It was entered as an independent constant in the equation for MATLAB® to differentiate. This resulted in the **A** and **B** matrices as stated above. Note the improved settling time, and rejection of the cross-coupling effect of the other states

The reason only four operating points were chosen rather than many predetermined as was the case with experiment one, is because per inspection of the controller values, little variation occurred for the operating point sets of table 9, which cover the operating range of interest. Thus, four was deemed sufficient.

When a controller is designed for a plant, the idea is that it is correctly sized. The gain constants cannot just be increased until the desired response is achieved. This means that a value of one for the proportional gain controller parameter allows for variations across the system's entire operating range, i.e. the design allows for large disturbances. A gain constant of 300 would allow variations $\frac{1}{300}^{th}$ its operating range. Thus, the higher the gain constants are made, the better the response, but then only small variations around operating points can occur.

5 Conclusions and recommendations

Conclusions are drawn from the research performed and some recommendations made.

Conclusions

The nature of the system is very important. To avoid non-compulsive control design (NCCD), one must design the *best* controller for the system. If a steady-state error of less than 2% is acceptable, why design anything more complex than a proportional controller. However, if noise rejection becomes an issue, introduce an integrator. Certain systems using integrators in their controller may suffer from reset windup, which can be addressed with an anti-reset windup algorithm.

As illustrated and discussed by experiment I, the owner of that system would achieve the necessary performance with the minimum equipment; this increases his profit margin.

Controllers can be in almost any form, i.e. P, PI, PID, or user defined. One has to be mindful though, considering the pole placement technique used to determine the controller parameters. The more controller parameters is required to be calculated, the more equations, or designer input is needed.

Even though the GA achieved better results for the PI-Controller, it is computationally too expensive when it comes to a multivariable nonlinear system. This promotes the idea of decreasing the initial solution domain by, for example, biasing the population with the results obtained from the conventionally designed PI-controller. Alternatively, conventional design methods for a nonlinear PI-Controller which make use of optimal control theory should be applied. This is also the reason why a nonlinear version of the GA-optimised PI-Controller was not implemented in this study, it would take too long to optimise constants for every (enough) operating points across a system's operating range.

The nonlinear controllers are best suited for systems which are not smooth, i.e. systems that have significantly different behaviour depending on the operating point. If the behaviour for different operating points are similar, the nonlinear controllers achieve comparable performance with that of the linear controllers.

A very good understanding of the system is essential, not just to avoid NCCD, but also to enable a better and thorough design.

A PI-Controller has the following main advantages:
Very good noise rejection;
Zero steady-state error;
Very easy to implement;

For systems that are to be impervious to noise, a PI-controller is optimal if it can achieve the required response time. If not, a PID-controller can be considered, having improved response time, but being susceptible to noise.

The control of nonlinear systems is paramount due to their importance and widespread occurrence in the aerospace-, power generation-, chemical-, and petrochemical industries. This design methodology can readily be applied in all these industries.

The design methodology can be applied to systems of which we do not have the state-space equations.

This can be done by performing system identification on the black box model and deriving a linear state-space model around each operating point of importance. From here, the methodology follows suit. This is also true when one has a system that is not necessarily a black box SIMULINK® model, but a plant of which the input-output data is available or can be generated. In this case there exist Kalman estimators that can derive a state-space model.

This knowledge makes the design readily applicable to my sponsors where the simulated plants, in most cases, do not have a state-space model.

The derived design methodology is general enough to be used on any multivariable plant which may be nonlinear. Controllers can take almost any form, with PID-type controllers being very easily implemented. The performance of the designed controller can easily be adapted by using different performance indices, making it applicable in most scenarios; this makes the design very pliable.

The methodology is encapsulated in the algorithm that has been developed, making it very easy to implement the abovementioned control strategies. The algorithm also contains methods of handling most commonly occurring phenomenon which hamper control, like reset windup.

Good results can still be generated even if the phenomenon of "ping-pong vs. onions" [40] occurs.

This algorithm was implemented using PI-type controllers, which are widespread in the industry.

With little effort the algorithm can be developed into a marketable product.

Other than its widespread application, from a safety point-of-view, the controllers which are developed have deterministic behaviour in a classical sense, eliminating any suspicions associated with non-deterministic methods like neural networks, fuzzy logic or a combination thereof.

Why NCCD is an important philosophy to have, is because it ensures the simplest solution; the more complex the controller and/or control technique, the greater the financial burden to the client.

The client is thus ensured of the **best**, **necessary** design for his/her system, given the constraints and performance criteria. This generally leads to an increased profit margin while maintaining a safe work environment.

Recommendations

In order to improve the assumptions for the I-constants of the nonlinear PI-Controller, one might use a GA for critical operating points, and for a generated reference signal. The generated reference signal would just be used, as to enable interpolation between the constants calculated for critical operating points, and other user defined operation.

Further research should be performed to enable the design of a nonlinear, multivariable controller which yields the optimum solution, which is also stable in a classical control sense. A further requirement is that it is not computationally expensive.

6 References

The references used for this study are listed.

References

[1] Y. Takahashi, M. J. Rabins and D.M. Auslander, *Control and Dynamic Systems.* Menlo Park, California: Addison-Wesley, 1972.

[2] K.E. Willcox, 'Dynamical Engineering Systems', Last modified: 2002-08-20, Available: http://web.mit.edu/aa-math/www/modules/node2.html.

[3] C. Schmid, 'Control Objectives', Last modified: 2005-05-09, Available: http://virtual.cvut.cz/dynlabmodules/syscontrol/node3.html.

[4] D.E. Kirk, *Optimal Control Theory An Introduction.* Englewood Cliffs, New Jersey: Prentice Hall Inc., 1970.

[5] C.L. Phillips and H.T. Nagle, *Digital Control Systems Analysis and Design Third Edition.* Upper Saddle River, New Jersey: Prentice-Hall Inc. Pearson Educational International, 1998.

[6] J.W. Nilsson and S.A. Riedel, *Electric Circuits Sixth Edition.* Upper Saddle River, New Jersey: Prentice-Hall Inc., 1972.

[7] R.C. Dorf and R.H. Bishop, *Modern Control Systems Ninth Edition.* Upper Saddle River, New Jersey: Prentice-Hall, 2000.

[8] H.H. Rosenbrock, 'Distinctive problems of process control'. *Chem. Eng. Progress, 58, No. 9, 1962-09, pp.43-50.*

[9] M.A. Murray-Lasso, 'The modal analysis and synthesis of linear distributed control systems'. Ph.D.Thesis, M.I.T., Cambridge, 1966.

[10] L.A. Gould and M.A. Murray-Lasso, 'On the modal control of distributed systems with distributed feedback'. *IEEE Trans. On Automatic Control,* AC-11, No. 4, October 1966, pp. 729-737.

[11] F.M. Schlaefer, 'Modal control of an ammonia reactor'. *Report ESL-R-293,* M.I.T. Project DSR 74994, December 1966.

[12] S. Sastry and M. Bodson, *Adaptive Control Stability, Convergence and Robustness.* Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1989.

[13] C. Edwards and S.K. Spurgeon, *Sliding Mode Control Theory and Applications.* T.J. International Ld, Padstow, UK: Taylor and Francis, 1998.

[14] M.K. Khan and S.K. Spurgeon, 'Robust MIMO water level control in interconnected twin-tanks using second order sliding mode control', *Control Engineering Practice,* Volume 14, pp. 375-386, Year 2006.

[15] R.W. Bass and I. Gura, 'High-order systems design via state-space considerations'. *1965 JACC Preprint*, pp. 311-318.

[16] A.M. Letov, 'Analytical controller design I', *Avtomatika I Telemekhanika,* 21, No. 4, April 1960, pp. 436-441.

[17] J.S. Tyler, 'Use of optimal control theory in multivariable control systems design'. *1964 JACC Preprint,* pp. 40-51.

[18] R.H. Willis and R. W. Brockett, 'Frequency domain solution of regulator problems', *1965 JACC Preprint,* pp. 228-234.

[19] A. Chipperfield, et al, 'User's Guide version 1.2'. *Genetic Algorithm Toolbox for use with* MATLAB®, Department of Automatic Control and Systems Engineering, University of Sheffield, 1994.

[20] M.D. Mesarovic, *The Control of Multivariable Systems.* New York: Wiley, 1960.

[21] C. Schmid, 'Dynamical Behaviour of a closed loop system', Last modified: 2005-05-09, Available: http://virtual.cvut.cz/dynlabmodules/syscontrol/node51.html.

[22] C. Schmid, 'Open Loop vs Closed Loop', Last modified: 2005-05-09, Available: http://virtual.cvut.cz/dynlabmodules/syscontrol/node4.html

[23] K.E. Willcox, 'More Comments About State', Last modified: 2002-08-20, Available: http://web.mit.edu/aa-math/www/modules/node3.html

[24] S. Lawrence, et al, Available: http://citeseer.ist.psu.edu/context/98078/0

[25] J.M. Goncalves and M.A Dahleh, 'Necessary and Sufficient Conditions for Robust Stability of a Class of Nonlinear Systems', in *Proceedings of the 34th Conference on Decision and Control.* New Orleans, 1995-12.

[26] K.E. Willcox, 'Linearization', Last modified: 2002-08-20, Available: http://web.mit.edu/aa-math/www/modules/node5.html

[27] J. van de Vegte, *Feedback Control Systems.* Englewood Cliffs, New Jersey: Prentice Hall, 1994.

[28] L. Ljung and T. Glad, *Modeling of Dynamic Systems.* Englewood Cliffs, New Jersey: PTR Prentice Hall, 1994, Appendix B.

[29] K.E. Willcox, 'The Real World Linearization', Last modified: 2002-08-20, Available: http://web.mit.edu/aa-math/www/modules/node8.html

[30] K.E. Willcox, 'Nonlinear Effects. Conclusions', Last modified: 2002-08-20, Available: http://web.mit.edu/aa-math/www/modules/node10.html

[31] B. de Schutter, 'Predictive Control of Nonlinear Systems in the Process Industry', Last modified: 2002-06-10, Available: http://lcewww.et.tudelft.nl/~crweb/research/node40.html

[32] K.E. Willcox, 'Why is Linear Good? Applicaions', Last modified: 2002-08-20, Available: http://web.mit.edu/aa-math/www/modules/node9.html

[33] E.W. Weisstein, et al, 'Lyapunov Function', Available: http://mathworld.wolfram.com/LyapunovFunction.html

[34] L.W. Johnson, R.D. Riess andJ.T. Arnold, *Introduction to Linear Algebra.* Menlo Park, California: Addison-Wesley, 1999.

[35] Mathworks, et al, 'Singular Coefficient Matrix', MATLAB® *R14 Help.* The Mathworks, Inc., 2004.

[36] M.T. Tham, 'Discretised PID Controllers - Implementation and Performance of Discrete PID Controllers', Available: http://lorien.ncl.ac.uk/ming/digicont/digimath/dpid1.htm

[37] M. Atanasijević-Kunc, R. Karba and B. Zupančič, 'Multipurpose modelling in the evaluation of laboratory pilot plant', *Simulation Practice and Theory,* Volume 5, pp. 751-776, Year 1997.

[38] D. Halliday, R. Resnick and J. Walker, *Fundamentals of Physics Extended Sixth Edition.* New York: Johan Wiley & Sons, Inc., 2001.

[39] R.H.B. Exell, 'Data for Dry Air', King Mongkut's University of Technology. Last modified: 2001. Available: http://www.jgsee.kmutt.ac.th/exell/JEE661/JEE661Lecture2.html

[40] S. Lawrence, et al, 'SYNOPSIS By means of a case study involving a severe case of coupled heat and air flow in buildings'. Available: http://citeseer.ist.psu.edu/cis?q=ping-pong+vs+onions&submit=Search+Documents&cs=1

7 Appendix

The material unable to be included in this document, is supplied.

Appendix

Please see the accompanying compact disc.

The folders in the root directory of the compact disc each contain its namesake's relevant data. This data could comprise of the necessary MATLAB® m-files or web pages.

Within the folder named: "Experiments", a folder corresponding to each experiment exists.