GENETIC ALGORITHMS FOR CONTROL SYSTEMS DESIGN

By

Naim Ajlouni

A THESIS SUBMITTED FOR THE DEGREE OF

Doctor of Philosophy

OF THE

University of Salford

DEPT. AERONAUTICAL, MECHANICAL & MANUFACTURING ENG.

DECEMBER 1995

TO MY FRIENDS & BROTHERS

KAYED, FIRAS

&

TO MY BELOVED WIFE & CHILDREN

ACKNOWLEDGMENTS

I wish to express my respect and thanks to my supervisor, Dr Tony Jones, for his guidance, friendship, encouragement, and support during the entire course of this research.

I also wish to express my thanks to my friend Stefan Kenway for all the help and support during the research.

Finally, my special thanks goes to my wife Hazel, my Children for their understanding, encouragement, and support

SYNOPSIS

The automated design of non-linear control systems has long eluded the control engineer, because of the lack of sufficient mathematical tools to solve the design problem. Indeed it is considered that if a design route is available it is unlikely to via the mathematical approach. This has inspired the concept of using Genetic Algorithms (GAs) to obtain a solution to the non-linear control system design problem. The GAs are search procedures based on the mechanics of natural genetics to guide themselves along paths to solutions in the search space. GAs provide a mean to search non-linear spaces.

This Thesis uses GAs to develops automated design techniques for the design of non-linear control systems. In this context the non-linear control system design problem has been sub-divided into the sub-problem of designing controllers for:

(i) non-linear controllers for linear plants. In this case the aim was to design high performance controllers, using non-linear PID gains. In this case the design methods employs three different techniques to map the non-linear gain functions. The research thus provides a comparison between Lagrangian Polynomial, Neural Networks, and Fuzzy Logic. The results shows that GAs are highly effective for the design of non-linear controllers for linear plants;

(ii) non-linear controllers for non-linear plants. In this case the aim was to design high performance and robust controllers using PID controllers. The problem can be considered as designing gain-scheduled PID controllers. In this case the gain scheduled functions were mapped methods using Lagrangian Polynomial. The results shows that GAs are highly effective for the design of gain scheduled PID controllers;

(iii) linear controllers for non-linear plant. In this case the aim was to design a fixed gain PID controller that would give similar performance through out the operating envelope of the non-linear plant. In this design method a max-min strategy was adopted to produce similar performance through out the operating envelope of the non-linear plant.

This research provides a profound insight into the different methodologies involved in implementing non-linear control systems. Furthermore, it clearly shows that the genetic design methods developed are powerful and flexible enough to be applied to design non-linear control systems for a broad spectrum of problems.

CONTENTS

Page

PART I : INTRODUCTION TO CONTROL SYSTEMS AND GENETIC ALGORITHMS

Chapter 1 Introduction

Chapter 2 Genetic Algorithms

CONTENTS

Page

PART II DESIGN OF NON-LINEAR CONTROLLERS FOR LINEAR PLANTS

Chapter 3 Design of Zoned Non-Linear Interpolated Controllers for Linear Plants

Page

CONTENTS

Page

CONTENTS

Page

PART IV DESIGN OF LINEAR CONTROLLERS FOR NON-LINEAR PLANTS

Chapter 8 Design of linear Controllers For Non-linear Plants

CONTENTS

Page

PART I

Introduction To Control Systems & GA

CHAPTER 1

Introduction

Chapter 1
INTRODUCTION

INTRODUCTION

Automatic control has played a vital role in the advance of engineering and science. In addition to its extreme importance in missile guidance systems, robotic systems, and many others, automatic control has become an important and integral part of modern manufacturing and industrial processes. Thus, for example, automatic control is essential in the numerical control of machine tools, in the manufacturing industries. It is also essential in such industrial operations as controlling pressure, temperature, humidity, viscosity and flow in the process industries. The centrifugal speed governor developed by James Watt for the speed control of a steam engine can be considered as the first widely used feedback control system not involving a human being. Over the years, control systems have become more sophisticated in order to improve the quality of manufactured goods and to speed up the production process. Therefore, a systematic approach to control system design was needed for the development of modern engineering.

Control systems are in general classified into two categories: open-loop and closed-loop systems. This distinction is determined by the control action, which is that quantity responsible for activating the system to produce the output. In open-loop systems, unlike closed loop systems the control action is independent of the output. Open-loop systems ability to perform accurately depends very much on their calibration, i.e. the procedure used to establish the input-output relation to obtain a desired system accuracy, also they are not generally troubled with the problem of instability. However, such open-loop control systems are frequently unsatisfactory because any unexpected disturbance to the system or changes in the system can cause deviation of the output from the desired value.

In closed-loop control, feedback is used to reduce the effects of both disturbances and changes in plant parameters. Feedback involves comparing the actual plant output with a desired value and using this difference to reduce the error between the two values.

The significance of feedback control was not understood clearly until the introduction of Laplace transform [1] and associated frequency-response techniques. Nyquist [2] who is the creator of the Nyquist frequency domain stability criteria, showed analytically the trade-off between stability and large loop gains in feedback control systems. The approach determines whether a closed-loop system is stable or unstable, and it also points the way to improving both the transient and steady-state response of the system. The Nyquist plot also carries information about the values of roots of the closed loop characteristic equation, and thus about the transient response characteristics. The stability characteristics are usually specified by two values called gain margin and the phase margin.

The exact relationship between the gain and phase margin and the transient response are different for different plants. Therefore, there is no guarantee that designing to a specified gain and phase margin will result in a good transient response.

The work done by Nyquist was extended by Black [3] and Bode [4]. The extension involved plotting the magnitude of the transfer function versus frequency on a logarithmic scale. Both Black and Bode realised that both the magnitude and the phase of the open-loop frequency response are needed to analyze the performance of the system, and that a major disadvantage in working with polar coordinates (i.e. the Nyquist plot) is that the curve changes its original shape when a simple modification such as a change in integral gain is made.

Another advance in controller design occurred in 1948 when Evans [5] presented his root-locus theory. This is an analytical graphical method for the choice of controller gains or the plant parameters for satisfactory plant behaviour. An important advantage of the root-locus method is that the roots of the characteristic equation of the system can be obtained directly, which results in a complete and accurate solution of the transient and steady-state response of the controlled variable. However, although the root-locus method can give a good indication of the type of transient response provided, it is not exact and always requires fine tuning on real applications. In the late fifties, control system scientists and engineers accordingly sought a different approach to control theory.

In this, differential equations replaced transfer functions for describing the dynamics of processes; stability was approached via the theory of Liapunov instead of the frequency-domain methods of Bode and Nyquist; and optimization of the system performance was studied by Pontrygin et al [6].

The classical ideas of Bode and Nyquist, and the design techniques based on them which were originally developed for the study of SISO systems, were generalised to deal with the multi-variable feedback control problem. The design of a succession of feedback loops one at a time using well established SISO feedback theory could not be used because the interactive effects between different loops can lead to instability in certain situations and may also cause a reduction of stability margins. Therefore, it was essential to design a multi-variable control systems in such a way as to deal with the interactive effects which would otherwise be prejudicial to system stability.

In the 1970's, Rosenbrock [7][8][9] and McMorran [10] accordingly extended the frequency-domain ideas to multi-variable systems and developed the Inverse Nyquist Array, Whilst Macfarlane and Belletrutti [11] developed the multi-variable theory of characteristic Loci. There have been various modifications, extensions and possible applications of these methods (Cook [12], Murno [13], Rosenbrock [14], and Yeung etal [15]). However, one disadvantage of these techniques is they rely heavily on the on the interpolation of graphs in the frequency domain, which can soon become very complex-even for experts in the use of these methods. The robustness of the resulting controllers is another problem with these techniques as they are generally un-robust because plant variation was not explicitly taken into account.

1.2 LINEAR CONTROL

Feedback control is a method to force some physical process, or plant to conform to a desired behaviour. Through feedback, one can obtain the desired behaviour with only partial and imprecise knowledge of the plant. However, the complexity of most plants forces one to construct oversimplified and approximate models for the purpose of analysis and design of a feedback control system. One class of models for which both analysis and design are well understood is the class of linear time-invariant plant. Unfortunately, in some cases a linear time invariant description of a plants dynamics is inadequate. Hence there was a need to seek alternate methods for the design of controllers for systems with widely varying non-linear and/or time-varying parameter dependent dynamics.

1.3 NON-LINEAR CONTROL

Non-linear systems with either inherent nonlinear characteristics or non-linearities deliberately introduced into the system to improve their dynamic characteristics have found wide application in the most diverse fields of engineering. The principal task of nonlinear system analysis is to obtain a comprehensive picture of what happens in the system if the variables are allowed, or forced, to move far away from the operating points. This is called the Global behaviour. Local behaviour of the system can in most cases be analyzed on a linearized model of the system. Therefore, the local behaviour can be investigated by linear methods that are based upon the powerful superposition and homogeneity principles. If linear methods are extended to the investigation of the global behaviour of a nonlinear system, the results can be erroneous both quantitatively and qualitatively since the nonlinear characteristics may be not be revealed by the linear methods. Therefore, there is a strong emphasis on the development of methods and techniques for the analysis and design of nonlinear systems.

The development of nonlinear methods faces real difficulties for various reasons. There is no universal mathematical method for the solution of nonlinear differential equations which are the mathematical models for nonlinear systems. The methods deal with specific classes of nonlinear equations and therefore have only limited applicability to system analysis. The classification of a given system and the choice of an appropriate method of analysis are not at all an easy task. Furthermore, even in simple nonlinear problems, there are numerous new phenomena qualitatively different from those expected in linear system behaviour, and it is impossible to encompass all these phenomena in a single and unique method of analysis.

Although there is no universal approach-to the analysis of nonlinear systems, one can conclude that all the methods available fall under the category of stability analysis methods. Such as, the phase-space topological techniques,(i.e. stability analysis method, and the approximate methods of nonlinear analysis).

Unlike the situation for linear systems most analytical techniques for nonlinear systems are directed at trying to obtain solutions to specific questions, involving system stability, and may have limited or no use for answering other questions which might be of concern, such as the response to a particular form of input. In addition, in several methods assumptions are required to start the analysis and these are usually based on the anticipated behaviour of the system.

1.3.1 EARLY WORK IN NON-LINEAR CONTROL

Although many scientists and engineers who studied some forms of control systems in the nineteenth century, for example Maxwell and Airy, were aware that the systems were not linear, it was in most instances a common procedure to linearize the differential equations involved. The only serious attempts at trying to analyze nonlinear feedback systems prior to the 1940's appear to be some varied and infrequent efforts to examine the behaviour of relay systems beginning with the work of Leaute in 1885 and ending with that of Hazen in 1934. Significant developments only started to take place in the early 1940's using methods developed for second order nonlinear differential equations and slightly later in the decade using methods based on the pioneering frequency response work of Bode and Nyquist.

The Phase-Space, or more specifically the Phase-Plane, approach has been used for solving problems in mathematics and physics at least since Poincare. The approach gives both the local and the global behaviour of the nonlinear system and provides an exact topological account of all possible system motion under various operating conditions. It is convenient, however, only in the case of second-order equations, and for high-order cases the phase-space approach is cumbersome to use. It can be extended to the study of high-order differential equations in those cases where a reasonable approximation can be made to find an equivalent second-order equation. However, this may lead to either erroneous conclusions about the essential system behaviour, such as stability and instability, or various practical difficulties as time scaling.

The Stability Analysis of nonlinear systems, is heavily based on the work of Liapunov, this is a powerful approach to the qualitative study of the system global behaviour. By this approach, the global behaviour of the system is investigated utilizing the given form of the nonlinear differential equations but without explicit knowledge of their solutions. Stability is an inherent feature of wide classes of systems, thus system theory is largely devoted to the stability concept and related methods of analysis. Among the methods used for stability analysis and investigation of sustained nonlinear oscillations, sometimes called a limit cycle, is the describing function. The theoretical basis of the describing function analysis lies in the Van Derpol [16] method of slowly varying coefficients as well as in the methods of harmonic balance and equivalent linearization proposed by Krylov and Bogoliubov [17] for solving certain problems of nonlinear mechanics.

Numerous ideas connected with stability analysis were founded by Liapunov. Liapunov proposed two methods for stability analysis. The first method did not find wide application to stability problems, the second method of Liapunov offered much promise for further advance in stability theory of nonlinear systems. A significant problem in Liapunov stability theory for a wide class of nonlinear systems was proposed by and partially solved by Lur'e. Popov expressed a different form of solution to the Lur'e Problem known as absolute stability. The relationship between the Popov absolute stability criterion and Liapunov function was first established by Yakubovich, and later refined by Kalman and Meyer.

Stability analysis, however, does not constitute the complete picture. A satisfactory method for the design of nonlinear control systems is also required. It is evident therefore that the area of non-linear control systems design is a fruitful area for research. In this thesis genetic algorithms are proposed as a mean of providing a solution to designing non-linear control systems. It will be shown and illustrated by examples that the use of genetic algorithms is highly effective in designing a nonlinear control system such that a time-domain cost function involving performance and /or robustness can be optimised. It is believed that the use of evolutionary processes in the design of non-linear control systems as described in this thesis will constitutes a significant contribution to the design of non-linear control systems, with wider application than any other previous design method.

1.4 AN EVOLUTIONARY APPROACH TO CONTROL SYSTEM DESIGN

Over the years, many methods have been developed for optimizing the design and operation of engineering systems. In general, optimization consists of searching the space of design parameters as a function of some performance index to determine where the performance index is maximised or minimised. Optimization has been studied for a great many years, both as an abstract mathematical problem and also from the view point of engineering design. A great many methods have evolved which are detailed in a sizeable literature with each method having its champions and detractors. Some methods, such as calculus-based gradient search or zero-finding procedures, converge nicely for their intended problem class, but one of the difficulties of these methods is that the results are best in the neighbourhood of the current point. In addition, calculus-based methods depend upon the existence of derivatives of the objective function. Even if numerical approximations of derivatives are considered, this is a severe shortcoming since many practical parameter spaces have little respect for the notion of derivative and smoothness this implies (Goldberg [18]). Enumerative schemes start looking at objective function values at every point in space one at a time. Such search techniques are not efficient and fail in many practical problems in which the search space is too large to search one at a time even if an enumerative scheme such as dynamic programming is employed. Random search algorithms are another search technique, but random walks and random schemes that search and save the best must also be discounted because of the efficiency requirement.

A common feature of these methods is that they date from before the time of powerful computers. They have thus tended to be developed with low order problems in mind since the computational power to test them on high-dimensional problems was not available at their inception, clearly, there is a need for methods which are both global and efficient, and which are also more robust over the broad spectrum of the problems. In recent years, a new search technique called Genetic Algorithms (GAs for short) has accordingly been developed by Holland [19]. GAs are search algorithms based on the mechanics of natural selection and natural genetics, and starts with a population of structures that are coded into binary strings. These structures are evaluated within some environment and a measure of the fitness of a structure is defined. The fitness of each structure is calculated and a new population of structures is then obtained by a process of selection. Each structure is selected with a probability determined by its fitness, so that those best-fitted for the environment will survive and those not fitted will become extinct. The selected population of structures then undergoes the genetic operations of crossover and mutation, which provide a structured yet randomised information exchange among the structures. GAs are different from more normal optimization and search procedures in the following respects;

1. GAs work with a coding of the parameter set, not the parameters themselves.
2. GAs search from a population of points, not a single point.

3. GAs use objective function information, not derivatives or other auxiliary knowledge.
4. GAs use probabilistic transition, not deterministic rules.

In the period of 1980-1987, Goldberg examined and demonstrated the power of GAs to solve a variety of problems such as computer aided pipeline operation [20]. The application of genetic algorithms to the self-learning of diagnostic rules for a pilot-scale mixing process was studied by Zhang and Roberts [21]. Indeed, despite the simplicity of their operators, genetic algorithms have quickly found near optimal solutions in a variety of problem domains including mathematical optimization (Dejong [22]), engineering optimization (Goldberg [19]). The demand for GAs with fast response time has led to the investigation of parallel implementation, and two level GAs were investigated by Grefenstette [23]. Furthermore, Roa [24], and Onoda and Hanawa [25], used GAs to find optimal locations of the actuators in large space structure.

In addition most control engineering problems can be considered as constrained optimization problems. In order to convert constrained problems to unconstrained problems different algorithms have been developed, such as external penalty function methods, moving parameter penalty function method, and multiplier methods. (Ariel [26], Luenberger [27], Polak [28], and Davison [29]). It is of considerable interest to find a reliable algorithm to minimize on objective function algorithms, the difficulty arises when the designer does not know how large to choose the penalty function weighting parameters so as to achieve a given degree of accuracy in the solution of the problem. In general, standard penalty function algorithms require considerable manipulation of their penalty parameters in order to obtain a solution. However, Porter and Borairi [30] showed that constraints can be handled directly by GAs. Porter and Jones [31] used GAs to obtain the optimal PID controller gains. Also Mingwu and Zalzala presented a genetic-based approach to mobile robot motion planing with a distance-safety criterion [91]. Weller, Summers, and Thompson, investigated the use of genetic algorithms to evolve the optimum set of inputs for neural networks [36]. Porter, Mohamed, and Jones [33] demonstrated that GAs provide a much simpler approach to the tuning of multi-variable controllers. All of these techniques represents an evolutionary approach to control system design using genetic algorithms to effect the design .

1.5 OBJECTIVE OF THE THESIS

This thesis develops the concept of using GAs as a design technique for the design of non-linear control systems. In this context the non-linear control system design problem will be sub-divided into the sub-problems of designing controllers for:

(i) *non-linear controllers for linear plants*;

(ii) *non-linear controllers for non-linear plants*;

(iii) *linear controllers for non-linear plants*.

The efficient design of non-linear control systems has long eluded the control engineer, because of the lack of tools to solve the design problem. Indeed, it is considered that if a design route is available it is unlikely to be via a mathematical approach. This has inspired the concept of investigating whether Genetic Algorithms (GAs) can be used to obtain a solution to the non-linear control system design problem. This new approach is made possible by the advent of powerful computing facilities and may revolutionise the way control systems are designed in the future.

1.6 OUTLINE OF THE THESIS

This thesis consists of five major sections, which have been structured so as that three non-linear control system cases are considered and design techniques for each case are developed, the sections are:

Section i) an introduction to control system theory and genetic algorithms;

Section ii) genetic design of non-linear PID controllers for linear plants;

Section iii) genetic design of non-linear controllers for non-linear plants;

Section iv) genetic design of a linear robust PID controllers for non-linear plants;

Section v) conclusions and recommendations.

SECTION I INTRODUCTION

Chapter 1: Introduces the control problem, provides some historical background to control systems and gives a brief survey of engineering optimization, and provides an outline of the objectives of the thesis.

Chapter 2 describes the genetic algorithms, what they are ,and where they come from, how they work and describes the way the GAs can be applied in the field of control systems design.

SECTION II DESIGN OF NON-LINEAR CONTROLLERS FOR LINEAR PLANTS

Chapter 3 introduces the concept of a dual-zone PID controllers, it also shows how GAs can be applied to the design of high performance dual-zone PID controllers. Furthermore it demonstrates the ease of application of the GA, by genetically designing linear and non-linear PID controllers for linear plants using straight line interpolation methods.

Chapter 4 shows how GAs can be applied to the design of high performance non-linear PID controllers using a polynomial interpolation function.

Chapter 5 shows how GAs can be used to design high performance fuzzy controllers. In this case the fuzzy logic is used to map the non- linear gain functions of an incremental dual-zoned PID controller.

Chapter 6 shows GAs can be used to design high performance neural controllers. In this case three neural networks are used to map the non-linear gain functions of an incremental dual-zoned PID controller.

SECTION III DESIGN OF NON-LINEAR CONTROLLERS FOR NON-LINEAR PLANTS

Chapter 7 the concept of non-linear controllers for non-linear plants is introduced and explained. In this case a design method for high performance controllers using gain-scheduled PID control is introduced, and the Lagrangian polynomial interpolation is used to map the gains of the gain-scheduled controllers.

SECTION IV DESIGN OF LINEAR CONTROLLERS FOR NON-LINEAR PLANTS

Chapter 8 in this chapter the concepts of using genetic algorithms to design linear PID controllers for non-linear plants is introduced. In this case the aim is to design a linear controller that would guarantee good performance through out the operating envelope of the non-linear plant.

SECTION V CONCLUSIONS & RECOMMENDATIONS

Chapter 9 in this section the use of genetic algorithms for non-linear control systems design are reviewed and discussed. The important results of the thesis are summarised, and recommendations for future work in this field are suggested.

CHAPTER 2

Genetic Algorithms

Chapter 2
GENETIC ALGORITHMS

2.1 INTRODUCTION

The genetic algorithm (GA), was first suggested by John Holland [19] in his book Adaptation in Natural and Artificial Systems. The goals of his research have been twofold:(1) to abstract and rigorously explain the adaptive processes of natural systems, and (2) to design artificial systems software that retains the important mechanisms of natural systems. This approach has led to important discoveries in both natural and artificial systems science.

Over the last 20 years, the GA's have been used to solve a wide range of search, optimisation, and machine learning problems [18]. P J Angeline, Sanders, and Pollac [35], used GAs to constructs recurrent neural networks. Weller [36], used GAs to evolve an optimum input set for a predictive neural network. Kelemen [37] produced a design for a robot controller based on machine learning control system using GAs. Jones [38] produced a genetic tuning algorithm for PID controllers. Jones and Ajlouni [39][40] produced a genetic design method for gain-scheduled controllers for non-linear plants. Kim [41], designed fuzzy Neural controllers using GAs. Jones and Olivera [38], produced an auto-tuner for PID controllers. Gensing [43], produced a penalty algorithm for solving general constrained parameter optimisation problems. Jones [45], produced a genetic algorithm for tuning neural non-linear PID controllers. and numerous others [46 - 59], all used GAs in control systems design.

As the name implies genetic algorithms attempt to solve problems in a fashion similar to the way in which biological evolutionary process seem to operate. Random search algorithms have achieved increasing popularity as researchers have recognised the shortcoming of calculus-based and enumerative schemes. Yet, random walks and random schemes that search and save the best have been discounted because of the efficiency requirement. Random searches in the long run, can be expected to do no better than numerative schemes. Even though random search methods have been discounted, they must be carefully separated from randomised techniques. The GA is an example of a search procedure that uses random choice as a tool to guide a highly exploitative search through a coding of parameter space. At this stage it is useful to ask *"How are GAs diferent from traditional methods?"*, in order for GAs to surpass their more traditional cousins, GAs must differ in some very fundamental ways. GAs are different from more normal optimization and search procedures in four ways:

i) GAs work with coding of the parameter sets, not the parameters themselves;

ii) GAs search from a population of points, not a single point;

iii) GAs use pay-off (objective function) information, not derivatives or other auxiliary knowledge;

iv) GAs use probabilistic transition rules, not deterministic rules.

The GAs basically maintain a population of knowledge structures that represents candidate solutions for the problem. The population evolves over time through competition (survival of the fittest) and controlled variation (recombination and mutation). In this way the best elements of the current population are used to form the new population. If this is done correctly then the new population will, on average be better than the old population.

2.2 BASIC IDEA OF GAs

GAs are a search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structure with a structured yet randomized information exchange to form a search algorithm. The algorithms starts with a population of structures that has been coded into binary strings. These strings are evaluated within some environment and a measure of the fitness of a string is defined, then a new population of strings is obtained by a process of selection and replication.

Genetic algorithms involve three operations by which to abstract and rigorously represent the adaptive processes of natural systems:

(I) Reproduction operation,

(II) Crossover operation,

(III) Mutation operation.

Reproduction is a process where an old string is carried through into a new population depending on the performance index values. In this process, the fitness values are calculated for each candidate string using a fitness function, which depends on a goal for optimization problems. According to the fitness values, string with larger fitness values give rise to a larger number of copies in the next generation. Following reproduction, the strings are randomly mated using the crossover operation. Each pair of candidate strings will undergo crossover with the probability pcross(Pc). This operation provides randomised information exchange among the strings. Mutation is simply an occasional random alteration of the value of a string position (based on the probability of mutation). In a binary code, this involves changing a 1 to 0 and vice versa. The mutation process helps to escape local minima in the search space. The sequence of successive stages of genetic algorithms is shown in figure(2.1). In this chapter a full description of how such algorithms can be used to design digital PID controllers is presented.

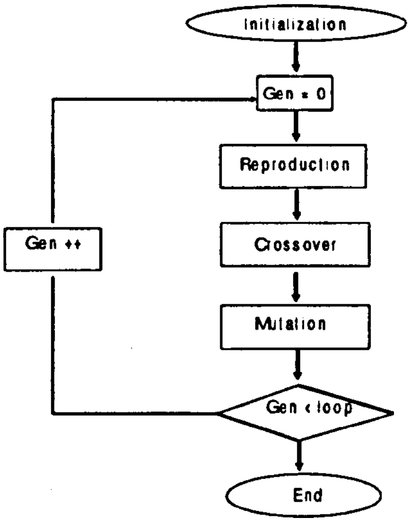


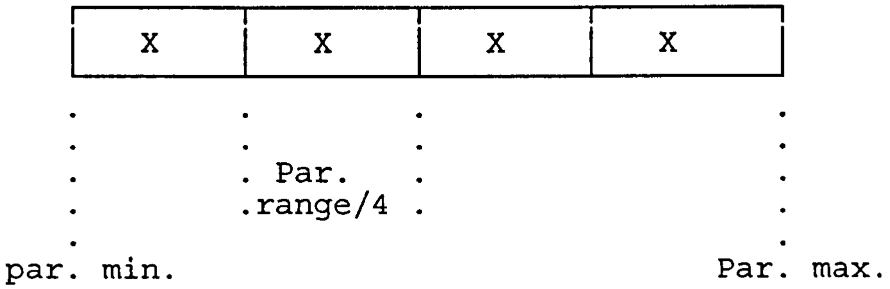
Figure (2.1) sequence of genetic algorithms.

2.3 INITIALIZATION

Genetic algorithms require the natural parameter set of the optimization problem to be coded as a string of binary digits. Thus, in the genetic design of digital PID controllers, each element of the set (K_p, K_i, K_d) of the controller gains are represented by a string of binary digits. Therefore, if the sub-string length is equal to an integer L, then each of the tuning parameters is in the set (K_p, K_i, K_d) $\in$ $[0,2^L]$. However, this apparent limitation can be circumvented by mapping the decoded unsigned integer linearly from $[0,2^L]$ to a specified interval [PAR_{min}, PAR_{max}] where PAR is any element of the tuning parameters in the set (K_p, K_i, K_d). In this way, the range and precision of the decision variables can be controlled. It should be realised that each encoded parameter may have its own-string length, and its own maximum and minimum values. Furthermore, the scaling factor can be calculated as

$$scale\ (i) = \frac{PAR_{\max(i)} - PAR_{\min(i)}}{2^L - 1} \qquad (i=1,2,....j), \qquad 2.1$$

where j is the number of variables to be optimized.



figure(2.2) Typical string representation.

Random initialisation is the most commonly used approach to form the initial population(Goldberg [19]). This approach requires the least knowledge acquisition effort and provides a lot of diversity for GAs to work with. The process of interview [19] is introduced to make sure the randomly generated variables do not violate any constraints on the function to be optimized. By using the process of interview, there will be more "fit" strings in the initial population in comparison with the case of generating initial population without the process of interview.

It is important to note that relaxing maximum and minimum values of the variables without the process of interview may result in an initial population in which some, or even all, candidate strings will violate the constraints on the objective function. On the other hand, considering too small a variable space interval may confine the solution to an undesirable local minimum.

2.4 REPRODUCTION

Reproduction is a process in which individual strings are copied according to their objective function values, f (fitness function). Intuitively, one can think of the function f as some measure of profit, utility, or goodness that we want to maximize. Copying strings according to their fitness values means that strings with higher value have higher probability of contributing one or more offspring in the next generation. This operator, of course, is an artificial version of natural selection, a Darwinian survival of the fittest among string creatures. In natural populations fitness is determined by a creatures ability to survive predators, pestilence, and the other obstacles to adulthood and subsequent reproduction. The reproduction operator may be implemented in algorithmic form in a number of ways. One method is to create a biased roulette wheel where each current string in the population has a roulette wheel slot sized in proportion to its fitness. Suppose the sample of four strings in table(2.1) has objective fitness function values f(the values in the table are chosen randomly). Summing the fitness over all four strings, we obtain a total of 1170. The percentage of population total fitness is also shown in the table. To reproduce one simply spin the weighted roulette wheel thus defined four times.

NO	String	Fitness	% of Total
1	01101	400	20
2	11000	100	5
3	01000	500	25
4	10011	1000	50
Total		2000	100

Table (2.1) Sample Problem String and Fitness Values

For the example problem, string number 1 has a fitness value of 400, which represents 20 percent of the total fitness. As a result, string 1 is given 20 percent of the biased roulette wheel, and each spin turns up string 1 with probability 0.20. Each time one requires another offspring, a simple spin of the weighted roulette wheel yields the reproduction candidate. In this way, more highly fit strings have a higher number of offspring in the succeeding generation. Once the string has been selected for reproduction it is entered into a mating pool, a tentative new population, for further genetic operator action. From the above it can be seen that the reproduction process comprises of a number of steps. The first being the testing of each parameter against the fitness function, so as to allocate a fitness against the parameter. The second is the mapping of the objective function to the fitness function.

2.4.1 FITNESS FUNCTION

After the initialisation of the elements of the tuning parameters, the objective function is introduced and the value of the objective function is calculated using decoded values of the parameters in each string. Thus, for example if minimum ISE (integral square of the error) is regarded as the ultimate design requirement, GAs can be readily used to select the set { K_p, K_i, K_d } for tuning parameters such that the generalised ISE is minimised. This performance index is computed by subjecting the plant concerned to a set-point change. In each case, the function

$$ISE = \sum_{j=1}^{j=N} e^2{}_j \qquad 2.10$$

is evaluated, where

$$e_j = v - y_j$$

and $e_j \in \Re$, is the error signal, $y_j \in \Re$,is the output signal, T is sampling time,

and $N = \frac{T}{\tau}$

where τ is an appropriately chosen settling time.

This criteria requires that the ISE of the dynamic response be minimum, i.e. the area between the integral of the response curve and the set-point be a minimum. The ISE is only one criterion IAE (Integral Absolute Error) or ITAE (Integral Time Absolute Error) or any other could equally be used. Indeed, actuator limits, or rate limits can also be easily included in the cost function.

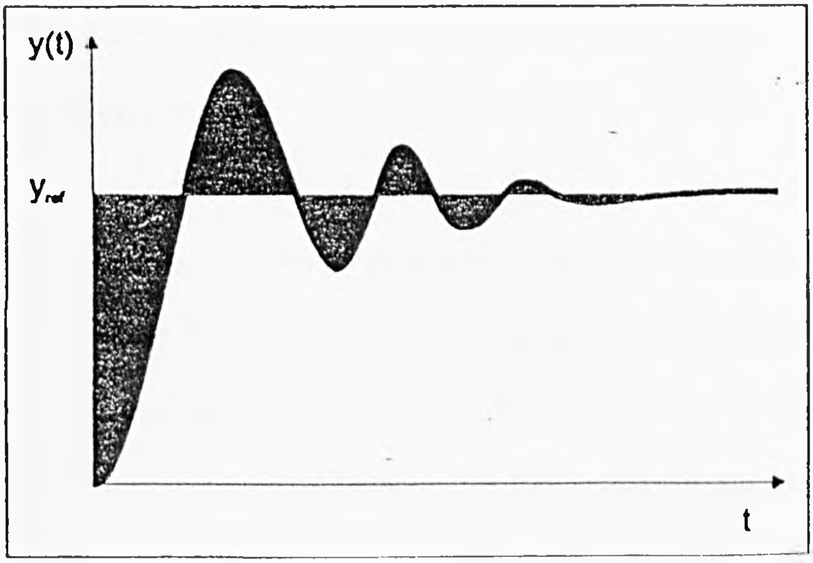


Figure (2.3) Integral of Square of error criteria.

2.4.2 MAPPING OBJECTIVE FUNCTION TO FITNESS FUNCTION

In general a genetic algorithm maximises its cost function. However in designing digital PID controllers (as in many other optimization problems) the objective is more naturally stated as the minimization of some cost function, g(x), rather than it's maximization. It is therefore often necessary to map the underlying natural objective function to a fitness function through one or more mapping. In normal operations, to transform a maximization problem to a minimization problem the objective function can simply be multiplied by a minus one. However, this operation alone is insufficient because the objective function is not guaranteed to be non-negative in all instances. In the case of GAs, the most common objective function-to-fitness transformation is therefore of the form

$$\text{fitness}(x) = W_{max} - g(x) \qquad \text{when } g(x) < W_{max} \qquad 2.12$$
$$\text{fitness}(x) = 0 \qquad \text{when } g(x) \geq W_{max}$$

where W_{max} is design parameter. There are a variety of ways to choose the coefficient W_{max} . W_{max} may be taken as an input coefficient, as the largest g(x) value observed so far, as the largest g(x) value in the current population, or the largest of the last k generation. For the purpose of this thesis there are two fitness functions available, one far more selective than the other, and is only available after a set number of generations within the GAs (i.e. when the change fitness variable is set to one). Otherwise the less selective function is used to allocate chromosome fitness. The functions are designed for cost function minimisation and allocation of fitness to scaled cost as yielded by the formula given by

$$\text{scaled chrome cost} = 1 + \frac{\text{chrome cost - population minimum cost}}{\text{population max cost - population min cost}} \qquad 2.13$$

The above scaled cost is used by the least selective fitness function. The mapping of the least selective fitness function is as shown by figure (2.4), it can be seen that each member has a finite chance of fitness.

The most selective fitness function uses the following scaled cost:

most selective chrome cost = 3.6125 - 2.375 X scaled chrome cost 2.14

The mapping of the most selective fitness function is shown in figure(2.4), it can be seen that only the members close to the optimal solution has a chance of being selected. Hence, this fitness function can only be used after the GA has been running for a few generations, which gives the GA a chance to reject bad members before

the most selective function is applied.

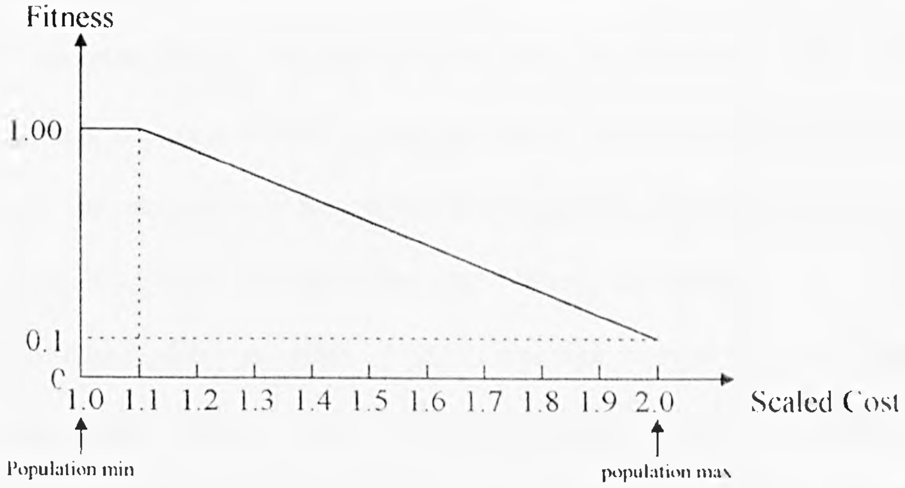


Figure 2.4 Least selective Fitness Function

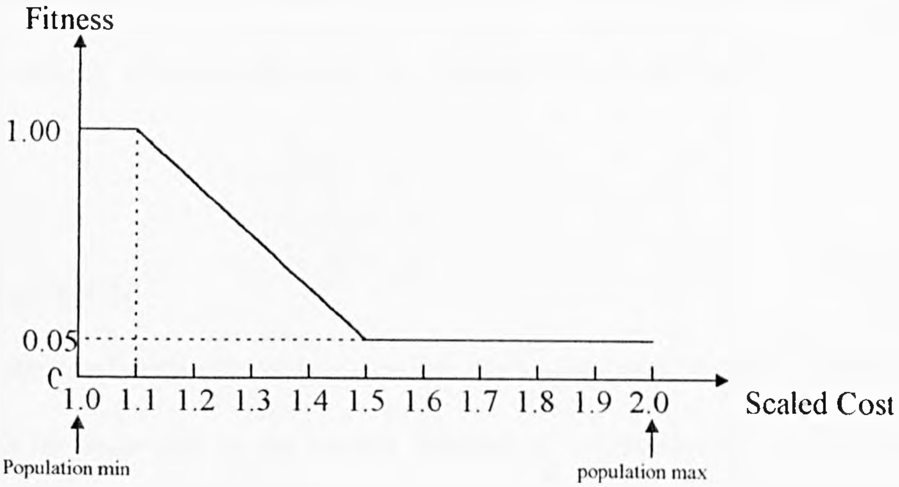


Figure 2.5 Most Selective Fitness Function

2.4.3 CONSTRAINTS

Many practical problems contain one or more constraints that must also be satisfied. Constraints are usually classified as equality or inequality relations. Inequality constraints may be subsumed into a system model. A GA generates a sequence of parameters to be tested using the system model, and the constraints. It runs the model, evaluates the fitness function, and checks to see if any constraints are violated. If not, the parameter set is assigned the fitness val·e corresponding to its evaluation during simulation. If constraints are violated, the solution is infeasible and thus has no fitness. This procedure is fine except that many practical problems are highly constrained; finding a feasible point is almost as difficult as finding the best. As a result, it is usual to get some information out about some infeasible solutions, per results by degrading their fitness ranking in relation to the degree of constraint violation. This is what is done in a *penalty method.* For example if the minimum ISE for a controller being designed is higher than the maximum ISE value set for this design, this controller will be assigned a low cost function.

2.4.4 SELECTION

After the fitness of each candidate string has been calculated, a new population of strings with the same size as the current population is produced by the process of selection. In this process, each string is selected with a probability determined by its fitness, so that those best-fitted for the environment will survive and those not fitted will become extinct. The probability of a string being selected is

$$P(i) = \frac{F(i)}{\sum_{j=1}^{q} F(j)} \qquad 2.15$$

where P(i) is the probability that the ith candidate string will be selected, F(i), is the fitness of the ith string, and q is the total number of strings in the current population. There may exist situations in which the best set of parameters in the current generation is inferior to the best set in the previous generation. This could be caused by crossover disruptions. Therefore, to preserve the best discovered set of tuning parameters, it is proposed to replace the worst fitted candidate string in the current generation by the best found so far.

2.5 CROSSOVER

Each pair of the selected strings is subjected to the probability of the crossover operation. A simple crossover may proceed in two steps. First, members of the newly reproduced strings in the mating pool are mated at random. Second each pair of strings undergoes crossing over as follows: an integer position K along the string is selected uniformly at random between 1 and the string length less than one [1, L-1]. Two new strings are created by swapping all characters between position (k+1) and 1 inclusively. For example, consider strings A_1 and A_2 from table (2.1) shown earlier:

A_1 = 0110 | 101

A_2 = 1100 | 001

Suppose in choosing a random number between 1 and 4, a (k = 4) can be obtained (as indicated by the separator symbol |). The resulting crossover yields two new strings where the prime(') means the strings are part of the new generation:

A'_1 = 0110001

A'_2 = 1100101

2.6 MUTATION

In a simple GA, mutation is the occasional (with small probability) random alteration of the value of a string position. In the binary coding of the example given above, this simply means changing a 1 to a 0 and vice versa. By itself, mutation is a random walk through the string space. When used sparingly with reproduction and crossover, it is an insurance policy against premature loss of important notations. Mutation operators plays a secondary role in the simple GA, to obtain good results in a genetic algorithm the frequency of mutation is in the order of one mutation per thousand bit.

2.7 EVOLUTIONARY ALGORITHMS

The use of evolutionary algorithms (EA) has increased considerably in recent years largely because computer science has encouraged researchers to study the potential and ability of EA to solve real problems, and to assess their ability to produce effective results.

Recent results in simulated evolutions [92][93][94] have demonstrated that search processes based on natural evolution are robust and can address difficult optimization problems across a variety of domains. Simulated evolution is based on the collective learning processes within a population of individuals, each of which represents a search point in the space of possible solutions to a given problem. There are currently three main lines of research in simulated evolution: genetic algorithms (GA), evolution strategies (ES), and evolutionary Programming (EP)[95].

In each of these methods, a population of individuals is initialized, and evolves toward successively better regions of the search space by means of a stochastic process of selection, mutation and if appropriate crossover. The methods differ with respect to the specific representation, mutation operations and selection procedures.

2.8 WHY USE GAs FOR EVOLUTIONARY DESIGN

One of the primary reasons to use GAs is that they are broadly competent algorithms [19] [95]. Empirical work has long suggested this [95], it appears that GAs can solve problems that have many difficult-to-find optima. Because GAs use very little problem-specific information, they are remarkably easy to connect to existent application code. Moreover, because of a GAs noise tolerance, discrete-event simulations and other noisy evaluators can be used directly as long as population sizing is performed to account for the stochastic variations in the evaluation process [95]. With practical successes growing in number and the application results paving the way for practical, will increasingly be used by industrialist to solve their problems. With increasing number of practical applications in existence, the future of GAs seems fairly bright. By having a look at the applications it reveals a surprising breadth of application area as well as the use of different coding, operators, and objective functions. On the other hand, the applications are surprisingly similar in their underlying motivation and approach. Many of the applications demonstrated a fairly rigid separation between model and searcher, and this is likely to be the case in any other application as well. Also many of the applications had useful heuristics and local search techniques and found it was useful to bring those on board to improve convergence times. Perhaps most importantly, each of the applications come to GAs for performance. It is becoming clear that GAs and EAs are changing the vision of what is possible to design and operate.

Finally it is evident that the evolutionary algorithms are becoming even more powerful design tools. It is believed that in the future all Computer Aided Control System Design systems will incorporate such evolutionary algorithms in their design tools. With this tool the emphasis of the design changes to selecting the most appropriate cost function to solve the control problem at hand.

PART II

CHAPTER 3

Design of Non-Linear Controllers for Linear Plants

Chapter 3
GENETIC DESIGN OF ZONED NON-LINEAR PID CONTROLLERS FOR LINEAR PLANTS

3.1 INTRODUCTION

The potential of computers to address the problem of intelligent control was realised as early as 1958, when Kalman [44] examined the problem of building a machine which adjusts itself automatically to control an arbitrary dynamical system. Whilst it must be acknowledged that in the intervening years computer technology has made the implementation easier and more efficient, it must also be acknowledged that this technology has not made the practice of control any easier. Recently, considerable effort has been deployed in the field of intelligent control systems to develop techniques for designing better and more robust controllers, and to achieve that, different algorithms have been used, (GA's, fuzzy, and neural techniques [41 - 50]). The genetic tuners for digital PID controllers developed by Porter and Jones [42] offer an alternative solution to the tuning problem, which can overcome many of the limitations of other techniques. Controller tuning is a parameter optimisation problem subject to plant-dependent constraints, and hence, amenable to the genetic algorithm approach, since the genetic algorithm approach can be seen as a natural evolutionary tuning mechanism which has all the features of a good adaptive controller. The genetic approach to fixed gain PID controller tuning has been previously presented [31][33] for both single-input/single-output (SISO) plants, and linear multi-input/multi-output (MIMO) plants. However, it is well-known that if non-linear gains are used in the PID controller then superior performance can be achieved. The major problem with such a technique is how to define and tune the non-linear gain functions. The problem of defining the function can be solved in a number of ways by using peicewise and homogeneous interpolation, fuzzy logic, or neural networks. In this chapter interpolation is proposed as the means of defining the non-linear gain functions.

Optimal tuning of the functions can also be achieved using GA's. Indeed it is shown that the genetic algorithms greatly facilitates the design of such control systems. The enhanced performance obtained from interpolated non-linear PID controllers is illustrated by contrasting the performance of GA optimised controller with that of GA optimised linear PID controller when controlling the same plant through the same task.

3.2 SYNTHESIS

The linear SISO plants under consideration are governed on the continuous time set $T = [0,\infty)$ by state and output equation of respective forms

$$\dot{x}(t) = Ax(t) + bu(t) \tag{3.1}$$

and

$$y(t) = cx(t) \tag{3.2}$$

where

$x(t) \in \Re^n$ is the state vector,

$y(t) \in \Re$ is the scalar output from the plant,

$u(t) \in \Re$ is the scalar input to the plant,

$A \in \Re^{nxn}$ is the plant matrix,

$b \in \Re^{nx1}$ is the input matrix,

$c \in \Re^{1xn}$ is the output matrix.

It is assumed that the plant is functionally controllable, so that none of the transmission zeros of the plant lies at the origin in the complex plane and therefore that any and all solutions for s in

$$\begin{vmatrix} sI_n-A, & -b \\ c, & 0 \end{vmatrix} = 0 \qquad 3.3$$

are non-zero [Rosenbrock (1974)]. This assumption ensures that rank M = n + 1 [Porter and Power (1970)], where M is the system matrix given by

$$M = \begin{bmatrix} A, & b \\ c, & 0 \end{bmatrix} \qquad 3.4$$

In order to design non-linear PID controllers for SISO linear plants governed by equations (3.1) and (3.2), it is convenient to consider the behaviour of such plants on the discrete-time set $T_T = \{0, T, 2T, ...\}$.

The behaviour is governed by state and output equations of the respective forms [Kwakernaak and Sivan (1972)]

$$x_{k+1}=\Phi x_k+\Psi u_k \qquad 3.5$$

and

$$y_k=\Gamma x_k \tag{3.6}$$

where

$$\Phi=e^{AT} \tag{3.7}$$

$$\Psi=\int_0^T e^{A\tau}b\,d\tau \tag{3.8}$$

and

$$\Gamma = \mathrm{c} \tag{3.9}$$

In these equations, $x_{kT} \epsilon \Re^n$, $u_{kT} \epsilon \Re$, $y_{kT} \epsilon \Re$, $\Phi \epsilon \Re^{nxn}$, $\Psi \epsilon \Re^{nx1}$, $\Gamma \epsilon \Re^{1x}$, and T $\epsilon \Re^+$ is the sampling period.

The system to be controlled under the action of error-actuated PID controllers is governed on the discrete-time set $T_T = \{0,T,2T,\ldots,kT,..\}$ by control law equations of the form

$$u_k=T(k_pe_k+Tk_iz_k+k_d(e_k-e_{k-1})) \tag{3.10}$$

where e_k is the tracking error and is defined by

$$e_k = v_k - y_k \qquad 3.11$$

and z_k is the integral error, and given by

$$z_{k+1} = z_k + Te_k \qquad 3.12$$

where

v_k is the set-point command input,

K_p is the value of the proportional gain,

K_i is the value of the integral gain,

K_d is the value of the derivative gain,

z_k is the integral state,

T is sampling time.

This structure for the PID controller is different from the conventional PID controller in two respect:

i) the sampling time "T" is included in the controller equations so that the controller can be "de-tuned" to ensure close loop stability [42];

ii) the sampling time "T" has been removed from the derivative action to reflect the fact that the inclusion of the sampling time in the derivative gain as K_d/T could result in the derivative action dominating the controller response in the limits as $T \rightarrow 0$; $\{K_d/T\} \rightarrow \infty$.

The resulting controller has not been proposed before, but it is believed that this controller structure is more appropriate for analysing PID controllers.

It is evident that if equations (3.10), is incremented back in time by one sample, the equation become

$$u_{k-1}=T(k_p e_{k-1}+Tk_i z_{k-1}+k_d(e_{k_1}-e_{k-2})) \qquad 3.13$$

It therefore follows from equation (3.10), and (3.13) that the incremental controller can be described by equation of the form

$$u_k-u_{k-1}=T(k_p(e_k-e_{k-1}) +Tk_i e_{k-1}+k_d(e_k-2e_{k_1}+e_{k-2})) \qquad 3.14$$

However it is more convenient to describe equation (3.14) as

$$\Delta u_k = T(k_p \Delta e_k + Tk_i e_k + k_d \Delta^2 e_k) \qquad 3.15$$

where

Δu_k is the incremental change in input,

Δe_k is the first order backward difference in error,

$\Delta^2 e_k$ is the second order backward difference in error.

It can be seen from the comparison of equations (3.14), and (3.15) that the incremental change in input can be expressed as

$$\Delta\ u_k = u_k - u_{k-1} \qquad 3.16$$

and the first order backward difference in error can be written as

$$\Delta\ e_k = (e_k - e_{k-1}) \qquad 3.17$$

It follows that the second order backward difference of error can be expressed as

$$\Delta^2 e_k = (e_k - 2e_{k-1} + e_{k-2}) \qquad 3.18$$

It is evident by looking at equations (3.15),(3.17), and (3.18) that they can be used at each sampling instant to compute the next value of the controller output.

It is interesting to consider the behaviour of the controller to a set point change from an initially quiescent conditions (i.e. $y_0 = x_0 = e_{-1} = e_{-2} = 0$, and the set-point change is v).

However, if the behaviour of the controller is considered at the initial sampling instant following a set-point change, then from equation (3.11) the error equation of an initially plant condition is given by

$$e_0 = v - y_0$$

and by substituting in the above equation for y_0 the above equation becomes

$$e_0 = v \tag{3.19}$$

the first order backward difference in error given by equation (3.17) at the initial sampling instant can be represented by

$$\Delta e_0 = e_0 - e_{-1}$$

By back substituting for e_0, and e_{-1}, in the above equation it follows that

$$\Delta e_0 = v - 0$$

which can be written as

$$\Delta e_0 = v \qquad 3.20$$

More over the second order backward difference of error given by equation (3.18) at the initial sampling instant can be represented as

$$\Delta^2 e_0 = e_0 - 2e_{-1} + e_{-2}$$

By back substituting for e_0, e_{-1}, and e_{-2} from equations (3.19, and 3.20)in the above equation, it follows that

$$\Delta^2 e_0 = v \qquad 3.21$$

However the input to the system given by equation (3.14) assuming initial quiescent conditions can be described by

$$u_0 = u_{-1} + T(k_p \Delta e_0 + Tk_i e_0 + k_d \Delta^2 e_0)$$

and

$$u_{-1} = 0.$$

then

$$u_0 = 0 + T(k_p \Delta e_0 + Tk_i e_0 + k_d \Delta^2 e_0)$$

By back substituting for the parameters (Δe_0, e_0, and $\Delta^2 e_0$), as given by equations (3.19, 3.20, and 3.21), the above equation can be represented as

$$u_0 = Tk_p v + T^2 k_i v + Tk_d v$$

If the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$u_0 = Tv(k_p + k_d) + 0(T^2) \qquad 3.22$$

Furthermore the change in input to the system given by equation (3.15) assuming initial quiescent conditions can be described by

$$\Delta u_0 = T(k_p \Delta e_0 + Tk_i e_0 + k_d \Delta^2 e_0)$$

By back substituting for the parameters (Δe_0, e_0, and $\Delta^2 e_0$), as given by equations

(3.19, 3.20, and 3.21), the above equation can be represented as

$$\Delta u_0 = Tk_p v + T^2 k_i v + Tk_d v$$

and if the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$\Delta u_0 = Tv(k_p + k_d) + 0(T^2) \qquad 3.23$$

This equation clearly demonstrate that the initial change in controller output following a set-point change is a function of the proportional ,and derivative gain functions, and the set point v.

If the system behaviour is now considered at the first sampling instant then the control equation given by equation (3.5) can be described as

$$x_1 = \Phi x_0 + \Psi u_0$$

and since the values of $x_0 = 0$, then the above equation can be written as

$$x_1 = 0 + \Psi u_0$$

By substituting in the above equation the value of u_0 from equation (3.22), the above equation can be written as

$$x_1 = T\ \Psi v(k_p + k_d) \qquad 3.24$$

Moreover the output equation given by equation (3.6), at the first sampling instant can be written as

$$y_1 = cx_1$$

Hence, by substituting in the above equation the value of x_1 from equation (3.23), then the above equation can be written as

$$y_1 = T\ c\,\Psi v(k_p + k_d) \qquad 3.25$$

Moreover the error given by equation (3.11) at the first sampling instant is given by

$$e_1 = v - y_1$$

By back substituting in the above equation the of y_1 from equation (3.24), the above equation can be written as

$$e_1 = \dot{v} - T\, c\, \Psi\, v\, (k_p + k_d) \qquad 3.26$$

Moreover the first order backward difference in error at the first sampling instant is given by

$$\Delta e_1 = e_1 - e_0$$

By back substituting in the above equation the values of e_1, and e_0 from equations (3.26, 3.19), the above equation can be written as

$$\Delta e_1 = v - T\, c\, \Psi\, v\, (k_p + k_d) - v$$

Moreover it is evident that the above equation now becomes

$$\Delta e_1 = -T\, c\, \Psi\, v\, (k_p + k_d) \qquad 3.27$$

Moreover it is evident that the second order backward difference in error given by equation (3.18), at the first sampling instant would be

$$\Delta^2 e_1 = e_1 - 2e_0 + e_{-1}$$

Hence, by back substituting in the above equation the values of e_1, and e_0, from equations (3.26, 3.19), the above equation can be written as

$$\Delta^2 e_1 = v - T\, c\, \Psi\, v\,(k_p + k_d) - 2v + 0$$

given that $e_{-1} = 0$, the above equation now becomes

$$\Delta^2 e_1 = -v - T\, c\, \Psi\, v\,(k_p + k_d) \qquad 3.28$$

However, it is evident that the control input to the system at the first sampling instant can be found by

$$u_1 = u_0 + T(k_p \Delta e_1 + T k_i e_1 + k_d \Delta^2 e_1)$$

By back substituting in the above equation the values for (u_0, Δe_1, e_1, and $\Delta^2 e_1$), from equations (3.22, 3.27, 3.26, and 3.28), the above equation can be written as

$$u_1 = T v\,(k_p + k_d) + T^2\, k_p(-c\, \Psi\, v\,(k_p + k_d)) + T^2 k_i(v - T\, c\, \Psi\, v\,(k_p + k_d))$$

$$+ T k_d(-v - T\, c\, \Psi\, v\,(k_p + k_d))$$

Moreover it can be seen that the above equation can be written as

$$u_1 = Tv(k_p + k_d) + Tvk_d + T^2\ k_p(-c\Psi v(k_p + k_d)) + T^2 k_i(v - T\ c\Psi v(k_p + k_d))$$

$$+T^2\ k_d(-c\Psi v(k_p + k_d))$$

and if the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$u_1 = Tv(k_p + k_d) - Tvk_d + 0(T^2) + 0(T^3)$$

Moreover it can be seen that the above equation can be written as

$$u_1 = Tvk_p + 0(T^2) \qquad 3.29$$

It can also be seen that the change in the control input at the first sampling instant, as given by equation 3.15 can be expressed as

$$\Delta u_1 = T(k_p \Delta e_1 + Tk_i e_1 + k_d \Delta^2 e_1)$$

by back substituting in the above equation the value of Δe_1, e_1, and $\Delta^2 e_1$, from equations (3.27, 3.26, and 3.28), the above equation can be written as

$$\Delta u_1 = T^2\, k_p(-\Psi c v\,(k_p+k_d)) + T^2 k_i(v - T\,\Psi c v\,(k_p+k_d)) + T k_d(-v - T\,\Psi c v\,(k_p+k_d))$$

and if the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$\Delta u_1 = -Tvk_d + 0(T^2) \qquad 3.30$$

This equation clearly shows that the first change in controller output following a set-point change is a function of the derivative gain function.

If the behaviour of the system is now considered at the second sampling instant then the control equations can be as

$$x_2 = \Phi x_1 + \Psi u_1$$

By substituting for x_1, and u_1, as given by equations (3.24, and 3.29), the above equation becomes

$$x_2 = T\ \Phi\Psi(k_p + k_d) + T\ \Psi k_p \qquad 3.31$$

The out-put equation of the controller at the second sampling instant is given by

$$y_2 = cx_2$$

By substituting the value of x_2, from equation (3.31), the above equation becomes

$$y_2 = T\ c\Phi\Psi(k_p + k_d) + T\ c\Psi k_p \qquad 3.32$$

Hence, it is evident that the tracking error given by equation (3.11), at the second sampling instant is given by

$$e_2 = v - y_2$$

By substituting in the above equation for y_2, from equation (3.32), the above equation becomes

$$e_2=v-Tc(\Phi\Psi v(k_p+k_d)+\Psi v k_p) \qquad 3.33$$

Also, the first order backward difference in error at the second sampling instant can be given by

$$\Delta e_2=e_2-e_1$$

By substituting in the above equation for e_2, and e_1, from equations (3.33, and 3.26), the above equation becomes

$$\Delta e_2=(v-Tc(\Phi\Psi v(k_p+k_d)+\Psi T v k_p))-(v-Tc\Psi v(k_p+k_d))$$

Therefore it is evident that the above equation can be written as

$$\Delta e_2=Tc(-\Phi\Psi v(k_p+k_d)-\Psi v k_p+\Psi v(k_p+k_d))) \qquad 3.34$$

Further more it can be seen that the second order backward difference of error at the second sampling instant would be

$$\Delta^2 e_2=e_2-2e_1+e_0$$

By back substituting in the above equation for (e_2, e_1, and e_0), form equations (3.33, 3.26, and 3.19), the above equation can be written as

$$\Delta^2 e_2 = v - T\,c(\Phi\Psi(k_p+k_d)+\Psi k_p)-2(V-T\,c\,\Psi(k_p+k_d))+v$$

Therefore it is evident that the above equation can be written as

$$\Delta^2 e_2 = T\,c(-\Phi\Psi(k_p+k_d)-\Psi k_p+2\,\Psi(k_p+k_d)) \qquad 3.35$$

However, it is evident that the control input to the system from equation (3.14), at the second sampling time can be written as

$$u_2 = u_1 + T(k_p\Delta e_2 + Tk_i e_2 + k_d\Delta^2 e_2)$$

By back substituting in the above equation for u_1 from equation (3.29), the above equation can be written as

$$u_2 = Tvk_p + T(k_p\Delta e_2 + Tk_i e_2 + k_d\Delta^2 e_2)$$

By back substituting in the above equation for (Δe_2, e_2, and $\Delta^2 e_2$), from equations (3.34, 3.33, and 3.35), the above equation can be written as

$$u_2 = Tvk_p + T^2k_p\ c(\Phi\Psi v(k_p+k_d) - \Psi vk_p + \Psi v(k_p+k_d)) + T^2k_i\ c(v$$

$$-T\ c(\Phi\Psi v(k_p+k_d) + \Psi vk_p)) + T^2k_d\ c((-\Phi\Psi v + 2\Psi v)(k_p+k_d) - \Psi vk_p)$$

and if the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$u_2 = Tvk_p + 0(T^2 + T^3) \qquad 3.36$$

Also it is evident that the change in the input to the controller given by equation (3.15), at the second sampling instant is given by

$$\Delta u_2 = T(k_p\Delta e_2 + Tk_ie_2 + k_d\Delta^2 e_2)$$

By back substituting for the parameters (Δe_2, e_2, and $\Delta^2 e_2$), as given by equations (3.34, 3.33, and 3.35), the above equation can be represented as

$$\Delta u_2 = T^2k_p\ c((\Phi\Psi v + \Psi Tv)(k_p+k_d) + \Psi vk_p) + T^2k_i(v - (T\ c\,\Phi\Psi v(k_p+k_d)$$

$$+T\ \Psi vk_p) + T^2k_d\ c((-\Phi\Psi v + 2\Psi v)(k_p+k_d) - \Psi vk_p)$$

and if the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$\Delta u_2 = T^2\ c((\Phi\Psi v + \Psi v)(k_p{}^2 + k_p k_d) + \Psi v k_p{}^2) + T^2\ v k_i$$

$$T^2\ c((-\Phi\Psi v k_d + 2\Psi v k_d)(k_p + k_d) - \Psi v k_p k_d) + 0(T^3) \qquad 3.37$$

This equation clearly demonstrate that the change in controller output at the second sampling time is of the order T^2, as apposed to that at the initial and first which were of order T. It can also be seen that it is a function of proportional, integral, and derivative gain functions.

However if the behaviour of the system is now considered at the third sampling instant then the control equations can be as

$$x_3 = \Phi x_2 + \Psi u_2$$

By substituting in the above equation for x$_2$, from equation (3.31) ,the above equation can be written as

$$x_3=T\Phi(\Psi v\,(k_p+k_d)+\Psi vk_p)+\Psi u_2$$

By back substituting in the above equation for u$_2$, from equation (3.36), the above equation can be written as

$$x_3=T(\Phi(\Psi v\,(k_p+k_d)+\Psi vk_p)+\Psi vk_p) \qquad 3.38$$

Also the out-put equation of the controller at the third sampling instant is as

$$y_3=cx_3$$

By substituting in the above equation for x$_3$, from equation (3.38), the above equation then becomes

$$y_3=T(c\Phi\,(\Psi v(k_p+k_d)+\Psi vk_p)+\Psi vk_p) \qquad 3.39$$

Hence, it is evident that the tracking error equation (3.11), at the third sampling instant is given by

$$e_3 = v - y_3$$

and by substituting in the above equation for y_3, from equation (3.39), the above equation can be written as

$$e_3 = v - T(c\Phi(\Psi v(k_p + k_d) + \Psi v k_p) + \Psi v k_p)$$

Moreover it is evident that the above equation can be written as

$$e_3 = v - T\, c\,(\Phi\Psi v k_d - \Psi v k_p) \qquad 3.40$$

Also the first order backward deference in error at the third sampling instant can be written as

$$\Delta e_3 = e_3 - e_2$$

By substituting in the above equation for e_3 , and e_2 from equations (3.40, and 3.33), the above equation can be written as

$$\Delta e_3 = (v - T\, c\,(\Phi\Psi v(k_p + k_d) - \Phi\Psi v k_p - \Psi v k_p)) -$$

$$(v - T\, c\,(\Phi\Psi v(k_p + k_d) + \Psi v k_p))$$

Therefore it is evident that the above equation can be written as

$$\Delta e_3 = T\,c\,(-\Phi\Psi v k_d - 2\Psi v k_p) \tag{3.41}$$

And the second order backward deference in error at the third sampling instant can be written as

$$\Delta^2 e_3 = e_3 - 2e_2 + e_1$$

By back substituting in the above equation for the (e_3, e_2, and e_1), from equations

(3.40, 3.33, and 3.26), the above equation can be represented as

$$\Delta^2 e_3 = (v - T\,c\,(\Phi\Psi v\,(k_p + k_d) - \Phi\Psi v k_p - \Psi v k_p)) - 2(v - (T\,c\,(\Phi\Psi v\,(k_p + k_d)$$

$$+ \Psi v k_p)) + v - T\,c\,\Psi v\,(k_p + k_d)$$

Therefore it is evident that the above equation becomes

$$\Delta^2 e_3 = T\,c(\Phi\Psi v k_d - \Psi v k_d) \qquad 3.42$$

However it is evident that the change in the control input from equation (3.14), at the third sampling instant is written as

$$u_3 = u_2 + T(k_p \Delta e_3 + T k_i e_3 + k_d \Delta^2 e_3)$$

By substituting in the above equation for u_2 , from equation (3.36), the above equation becomes

$$u_3 = (T v k_p) + T(k_p \Delta e_3 + T k_i e_3 + k_d \Delta^2 e_3)$$

By back substituting in the above equation for the (Δe_3, e_3, and $\Delta^2 e_3$), from equations (3.41, 3.40, and 3.42), the above equation can be represented as

$$u_3 = (T v k_p) + T(T k_p\,(-\,c(-\Phi\Psi v k_d - 2\Psi v k_p)) - T k_i (v - T\,c(\Phi\Psi v k_d - \Psi v k_p))$$

$$+ T^2 k_d\, c(\Phi\Psi k_d - \Psi v k_d))$$

and if the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$u_3 = Tvk_p + 0(T^2 + T^3) \qquad 3.42$$

However, it can also be seen that the change in the control input from equation (3.15), at the third sampling instant is given by

$$\Delta u_3 = T(k_p \Delta e_3 + Tk_i e_3 + k_d \Delta^2 e_3)$$

By back substituting in the above equation for the (Δe_3, e_3, and $\Delta^2 e_3$), from equations (3.41, 3.40, and 3.42), the above equation can be represented as

$$u_3 = T(Tk_p\ (-\ c(-\Phi\Psi vk_d - 2\Psi vk_p)) - Tk_i(v - T\,c(\Phi\Psi vk_d - \Psi vk_p))$$

$$+Tk_d\ c(\Phi\Psi k_d - \Psi vk_d))$$

and if the controller tuning parameter T is chosen to be very small (which corresponds to fast sampling) then the above equation can be written as

$$u_3 = T^2\ c(\Phi\Psi vk_p k_d + 2\Psi vk^2{}_p) - Tk_i v + T^3\ c(\Phi\Psi vk_i k_d - \Psi vk_i k_p)$$

$$+T^2 k^2{}_d\ c(\Phi\Psi v - \Psi v)) \qquad 3.43$$

It is evident from the analysis done above during the first four sampling instances following a set-point change, the controller operates in two distinct regions within its operating envelope. The first region is at a single point at v during the initial and subsequent sampling instances, when the change in controller input is of the order "T" as was shown by equations (3.23, and 3.30. The second region is a small region of the order "T^2" close to and including zero. The system remains in the second region during all subsequent sampling instances, due to the control equations being of order of T^2 as shown by equations (3.37, and 3.43).

Hence it can be concluded that the controller operates in two distinct regions within its operating envelope. The first region is at the set-point change, and the second is for the rest of the transient response. It is therefore, evident that since there are distinct regions, any non-linear design should make use of this fact, to define the regions over which the non-linear controller operates.

3.2.1 NON-LINEAR INCREMENTAL CONTROLLERS

The incremental PID controller given by equation (3.15), can take one of two forms, one is linear, and the other is non-linear. In the linear form the gains k_p, k_i, and k_d are scalars, representing the values of proportional, integral, and derivative gains respectively.

The other form however, is a non-linear PID controller. In both the linear and non-linear controllers, the objective is to design a PID controller such that a good tracking behaviour is achieved.

One way of designing non-linear controllers, where the non-linearities are a function of the plant error e_k, is to choose the controller gains k_p, k_i, and k_d as functions of the first order backward difference in error, error, and second order backward difference in error respectively. It follows from equation (3.15), that the incremental non-linear PID controller can be described by an equation of the form

$$\Delta u_k = f_p(\Delta e_k)\Delta e_k + f_i(e_k)e_k + f_d(\Delta^2 e_k)\Delta^2 e_k \qquad 3.44$$

where

$f_p(\Delta e_k)$ is a function representing proportional gain,

$f_i(e_k)$ is a function representing integral gain,

$f_d(\Delta^2 e_k)$ is a function representing derivative gain,

Moreover, it follows from equation (3.44), that the gain functions can be described by equations of the form

$$\mathscr{K}_p = f_d(\Delta e_k) \qquad 3.45$$

$$\mathscr{K}_i = f_i(e_k) \qquad 3.46$$

$$\mathscr{K}_d = f_d(\Delta^2 e_k) \qquad 3.47$$

Hence it follows from equation (3.44), that the incremental non-linear PID controller can be conveniently described by an equation of the form

$$\Delta u_k = \mathscr{K}_p \Delta e_k + \mathscr{K}_i e_k + \mathscr{K}_d \Delta^2 e_k \qquad 3.48$$

It is important to note that equation (3.48) is implemented in the incremental form. By deploying the PID controller in incremental form avoids any bumpless transfer techniques associated with the integral state is avoided. This is particularly important in the case of non-linear controllers incorporating integral control, because the integral state will requires bumpless transfer every time the integral gain is changed. The non-linear controller involves defining non-linear gain functions for the controller, instead of linear gains, for the linear controller .

In general it can be seen that the complexity of the above problem has made it difficult to design such controllers. Hence, in order to overcome the difficulty, the techniques of genetic algorithms is therefore proposed as a mean of designing and tuning such non-linear controllers. Moreover the design criteria of dual-zone PID controllers previously developed in this chapter can be deployed to define the effective work-space of the non-linear controller. Figure (3.1), shows a block diagram of the non-linear control system.

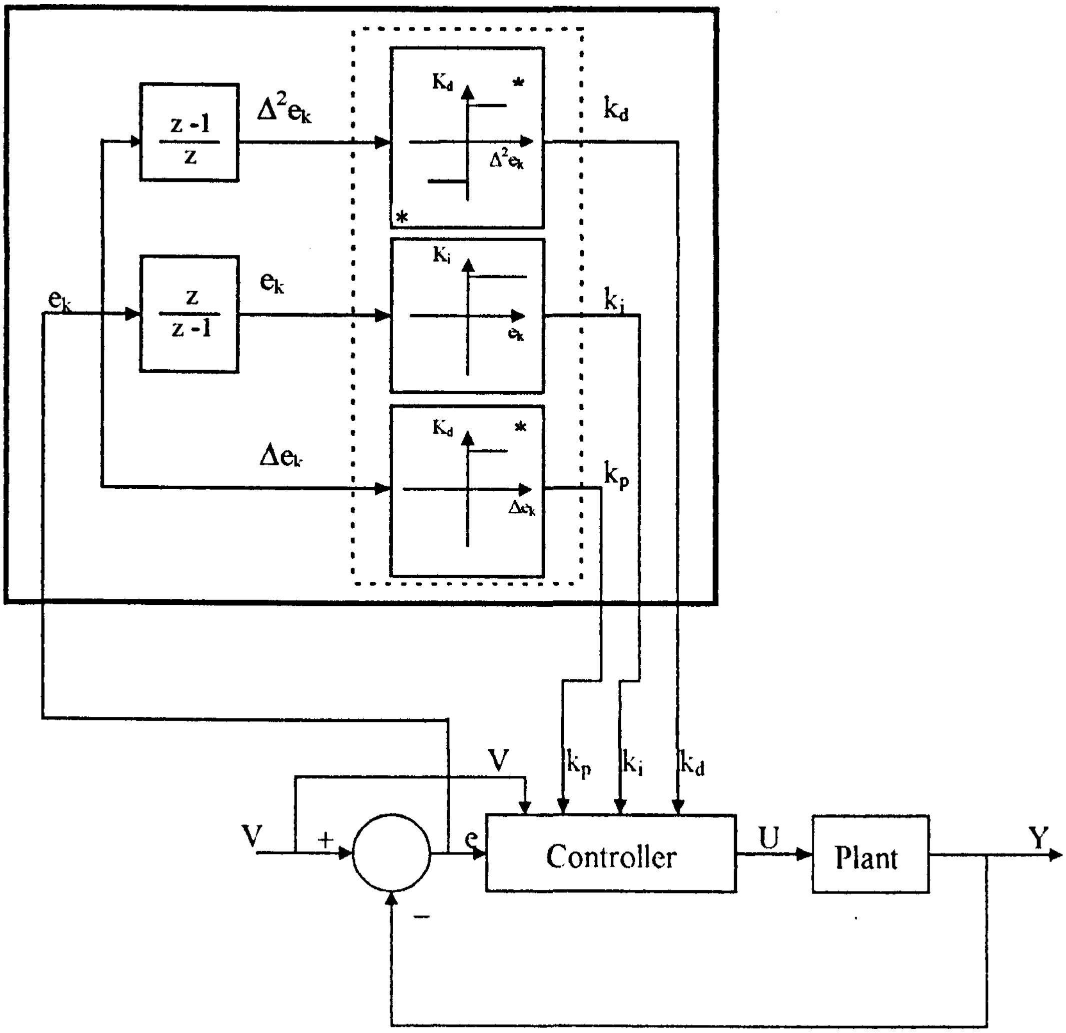
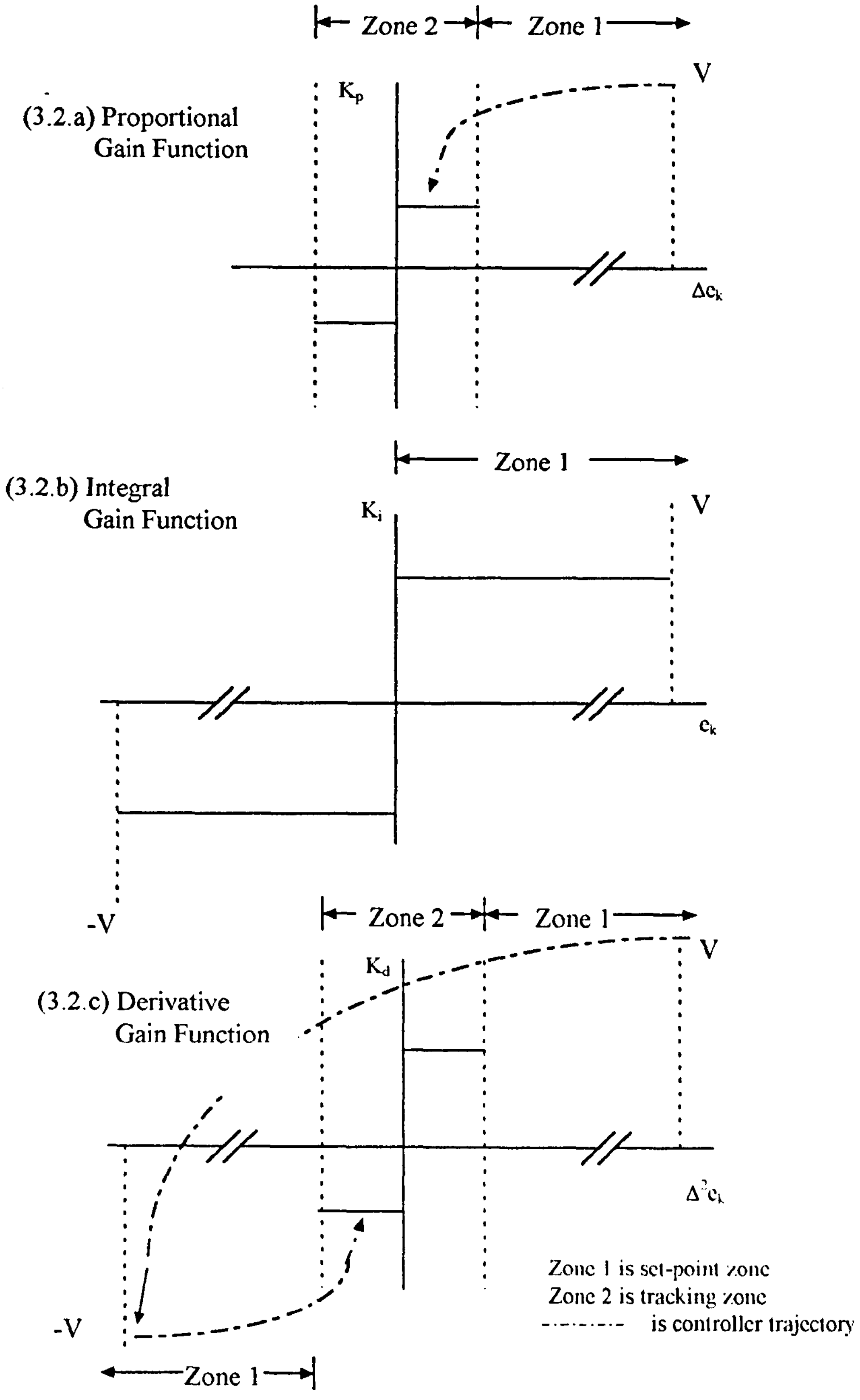


Figure 3.1 System Block Diagram

3.2.2 DUAL ZONE CONTROLLER

The dual-zone controller is split into two major zones. The first zone is used by the controller for the first two sampling periods following any set point change, and is thus called the set-point change zone. The gains needed for this zone can be considered as a form of feed forward, since, the gains are in the form of a single point at the start of the set-point change. The second zone is a small region close to the origin entered into after the second sampling period after a set-point change. The controller operates within this region for the reminder of the transient time. Hence, this zone is called the tracking zone.

It is evident from the dual-zone control theory introduced previously that it can only be implemented in the incremental form. From the analysis done involving dual zone control it can be seen that the gains of the dual zone controller have to be chosen to operate in the two zones. Therefore the new incremental proportional gain K_p operates in two distinct regions. Initially at a the set-point v, then for the rest of the transient time it operate in a small region close to and include zero. It was also noticed that the derivative gain function behaves in the same manner with one exception, it operates in two distinct points during the set-point change, as shown by figure (3.2c). Hence, the derivative gain operates in three distinct areas as shown by figure (3.2c). The derivative gain operates initially at two points, the first at the set-point v, the second at the set-point -v, and for the rest of the transient time it operate in a small region close to and include zero, as $\Delta^2 e_k << v$ in the tracking zone.



(3.2.a) Proportional Gain Function

(3.2.b) Integral Gain Function

(3.2.c) Derivative Gain Function

Figure 3.2 Typical Dual Zone Gain Function Mapping

3.2.3 GENETIC DESIGN OF LINEAR DUAL ZONED PID CONTROLLERS

In order to use Genetic Algorithms to design dual-zoned PID controllers, the parameters associated with the dual-zoned controller have to be precisely defined. It is evident from the previous discussion of dual-zoned controllers that the controller behaviour can be split into two distinct zones. The zones are defined as the

1) set-point zone;

2) tracking zone.

The parameters for the controller in the two zones are thus considered separately. The performance of the dual-zoned controller will be contrasted by considering a number of plants controlled by both the genetically designed dual-zoned controller, and the genetically designed linear PID controller.

3.2.3.1 PARAMETERS FOR THE SET POINT ZONE

From the analysis done for the dual-zone control , the controller output parameters for the initial two sampling instances are given by equations (3.23, and 3.30), the parameters are

$$\Delta u_0 = T(K_p v + K_d v) + 0(T^2),$$

It is evident from the equation describing Δu_0 that the GA will be searching for a sum of proportional, and derivative gain to produce the best results, and since it is a sum of gains, the exact proportion of each of the two gains is not important. Therefore the sum of the two gains can be represented by a single value F_0, this would results in equation (3.23) being rewritten as

$$\Delta u_0 = F_0 T v,$$

where $F_0 = (K_p + K_d)$.

Therefore it is evident that only F_0 needs to be coded into the GA to represent the sum of the two gain above.

Also from equation (3.30) it can be seen that controller input parameters describing Δu_1 is given by

$$\Delta u_1 = -TK_d v,$$

where $F_1 = -K_d$.

Hence F_1 need to be coded in the GA to represent the derivative gain during the second sampling instance.

From the above it can be seen that during the set-point zone only two parameters are searched for by the GA.

3.2.3.2 PARAMETERS FOR THE TRACKING-ZONE

In the tracking zone it was found that the proportional and derivative gains that the controller operate in a very small region close to zero, and from the equations of the input parameters for the tracking zone it can be seen that the controller must have individual gain values for the proportional, integral, and derivative gains. The gains used in this zone are mapped as shown in figure (3.3). By utilizing the frame work illustrated in figure (3.3), the equations representing the desired gains can be defined. From figure (3.3a), if

δE_p represent the zone length,

δU_p represent the incremental change in the input value Δu_k over the zone length.

K_p is the resulting gain given by

$$K_p = \Delta U_p/\delta E_p \qquad 3.49$$

This gain is used to evaluate $\Delta u^p{}_k$ within that tracking zone for any value of Δe_k. Also from figure(3.3b), if

δE_i represent the zone length of e_k,

δU_i represent the incremental change in the input value Δu_k over the zone length.

K_i is the resulting gain given by

$$K_i = \Delta U_i/\delta E_i \qquad 3.50$$

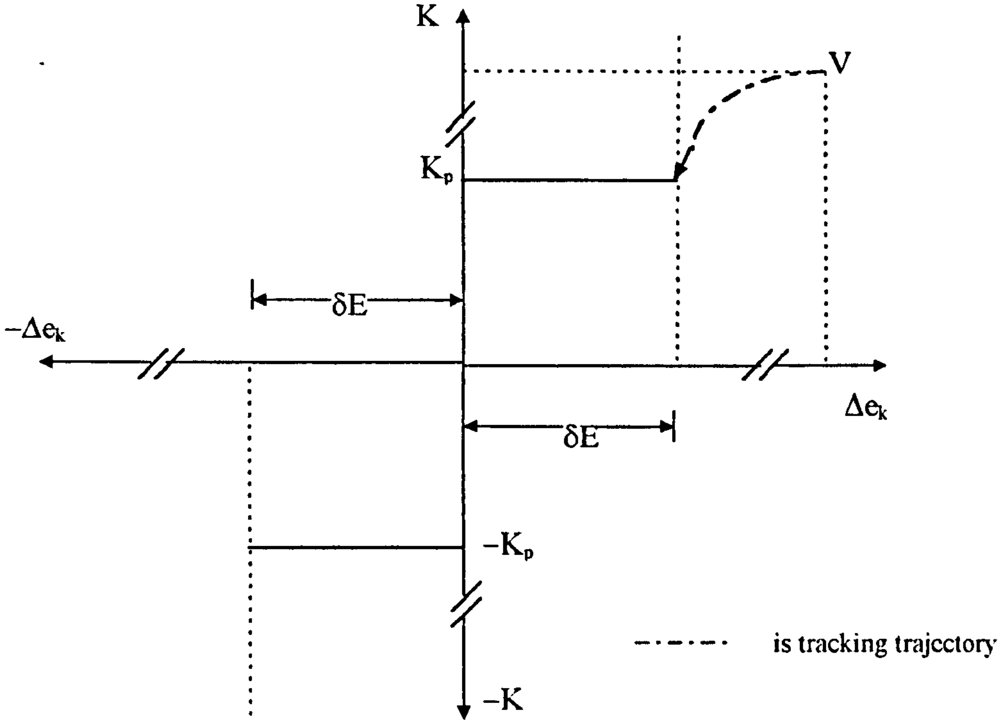


Figure 3.3.a Single Zoned Proportional Gain Function Profile Using Dual Zone Method.

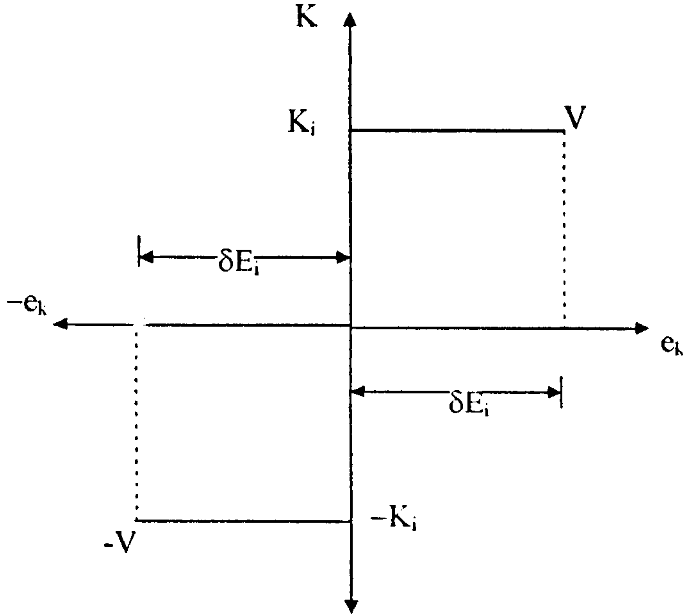


Figure 3.3.b Single Zoned Integral Gain Function Profile Using Dual Zone Method.

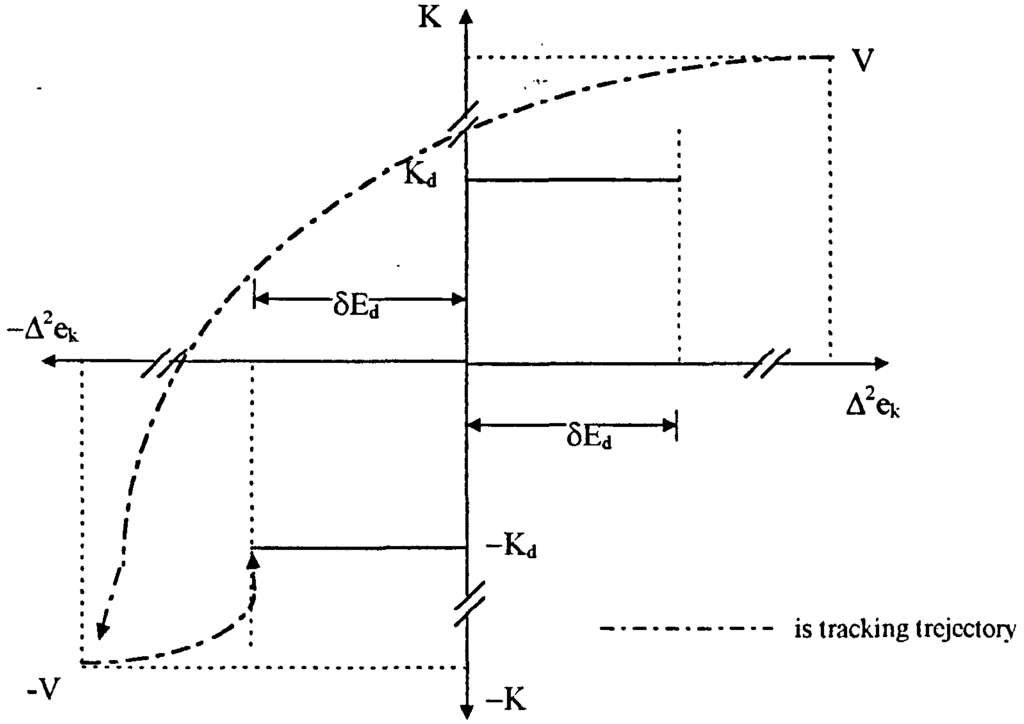


Figure 3.3.c Single Zoned Derivative Gain Function Profile Using Dual Zone Method.

This gain is used to evaluate Δu^i_k within that tracking zone for any value of Δe_k.

And from figure(3.3c), if

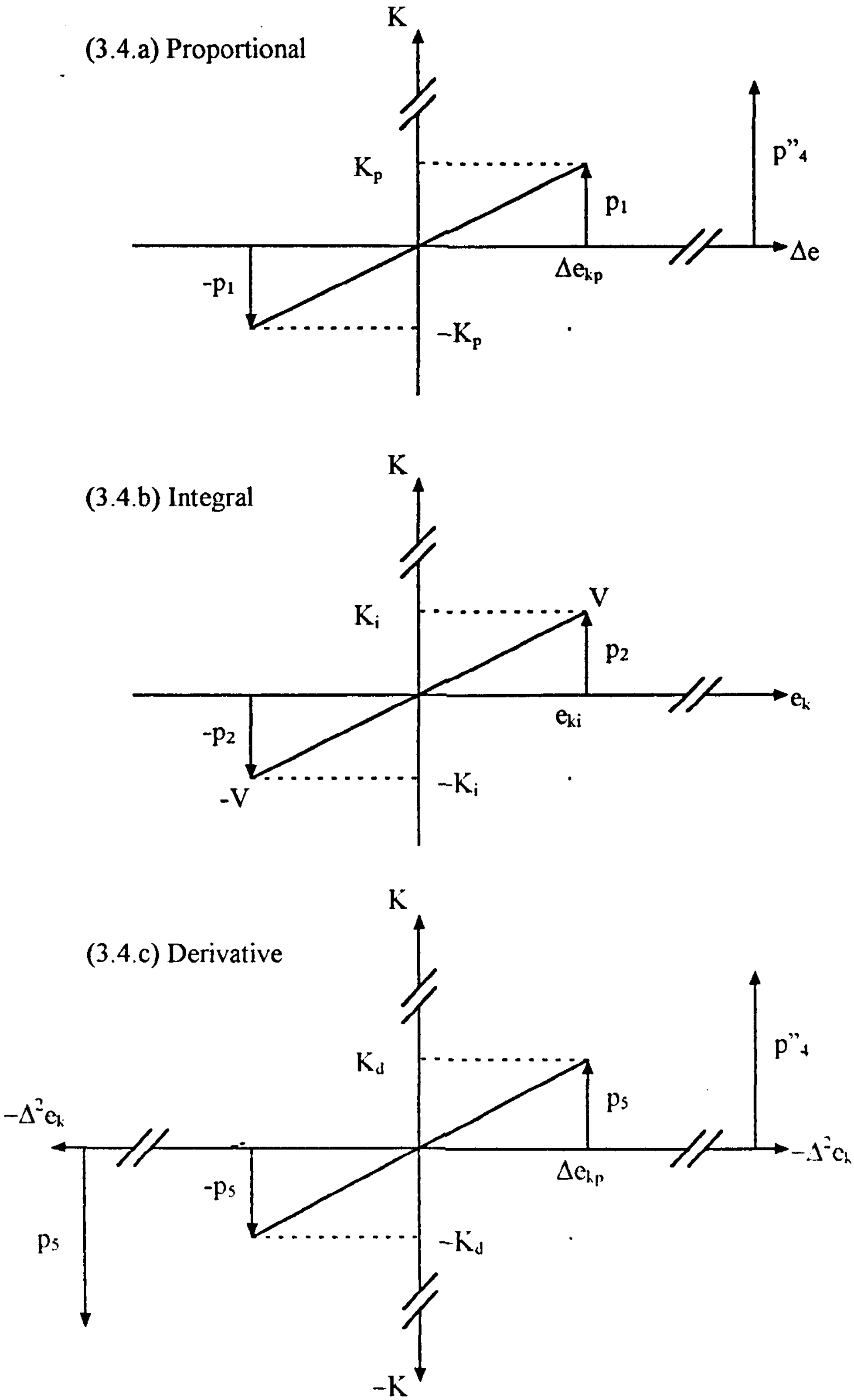
δE_d represent the zone length of $\Delta^2 e_k$ access,

ΔU_d represent the incremental change in the input value Δu_k over the zone length

K_d is the gain given by

$$K_d = \Delta U_d/\delta E_d \qquad 3.51$$

This gain is used to evaluate Δu^d_k within that tracking zone for any value of Δe_k.



(3.4.a) Proportional

(3.4.b) Integral

(3.4.c) Derivative

Figure 3.4 Typical GA String Mapping for Single Zone

population. This type of initialization is important in this type of design, since otherwise there would be a danger of creating an initial population in which many of its members violate the constraints on the controllers being designed, Figure 3.5 shows the sequence of genetic algorithms. Following the initialization, the objective function is introduced and its value is calculated using decoded values of the parameters in each string. Thus for example, if minimum ISE is regarded as the ultimate design requirement, genetic algorithms can be readily used to select the best set of tracking gains, and the feed forward gains such that the ISE is minimised.

In the genetic design of non-linear PID controllers the plant under consideration is subjected to a command input (i.e. unit step), then the performance index is computed for the plant. therefore, for each member in the population the function

$$ISE = \sum_{j=1}^{j=N} e^{2}_{j} \qquad 3.53$$

where $e_j = v - y_j \qquad N = \frac{\tau}{T}$

and $e_j \in \Re$, is the error signal, $y \in \Re$, is the output signal, T is sampling, and τ is an appropriate chosen settling time.

Minimization of this performance index over the entire population can be rapidly obtained by using the genetic operations of reproduction, crossover, and mutation.

It is interesting to note that the conditions for the existence of a non-linear PID controllers for the plant under consideration will be automatically satisfied by the genetic algorithms. This is obvious, because in the case of violation of the constraints the corresponding set of tracking gains, and the feed forward gains produces large ISE and the result will be zero or low fitness, and by the action of the selection operator it would not be chosen for the next generation.

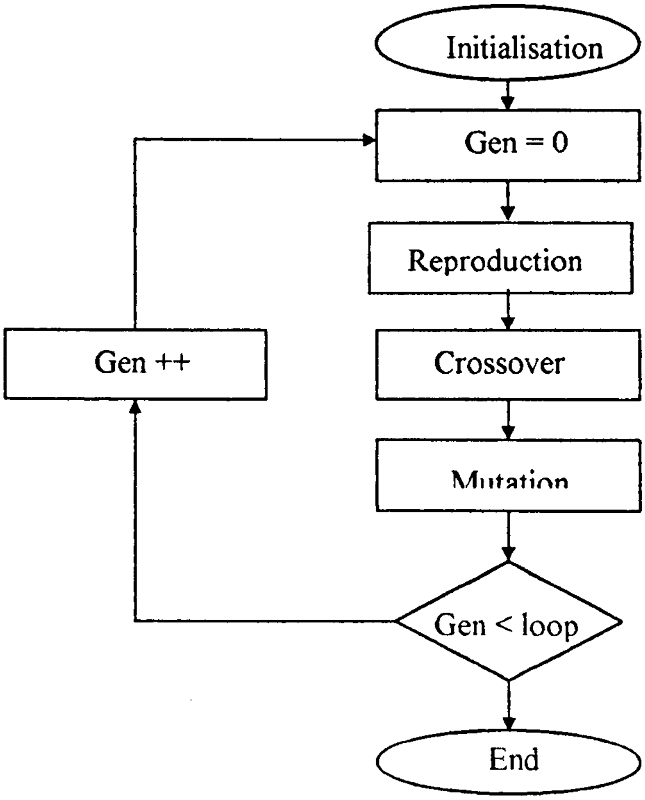


Figure 3.5 Sequence of Genetic Algorithms

The parameters for the controller in the two zones are thus considered separately. The performance of the multiple zoned controller will be contrasted by considering a number of plants controlled by both the genetically designed multiple zoned controller, and the genetically designed linear PID controller.

3.2.1.2.1 PARAMETERS FOR THE SET POINT ZONE

The parameters in this zone are exactly the same as was introduced in section 3.2.3.1 i.e. the GA will search the space for two feed forward gains for the set-point zone.

3.2.1.2.2 PARAMETERS FOR THE TRACKING ZONE

In the tracking zone it was found that the proportional and derivative gains that the controller operates in a very small region close to zero, and from the equations of the input parameters for the tracking zone it can be seen that the controller must have individual gain values for the proportional, integral, and derivative gains. The gains used in this zone are mapped as shown in figure (3.6). By utilizing the framework illustrated in figure (3.6), the equations representing the desired gain functions can be defined. Assuming equal sub-zone lengths it is evident from figure (3.6a) that the equation representing the sub-zone length for the proportional gain can be represented as

$$\delta E_p = \Delta E_p / n \qquad 3.54$$

where ΔE_p is the over all zone length, and n is the number of sub-zones.

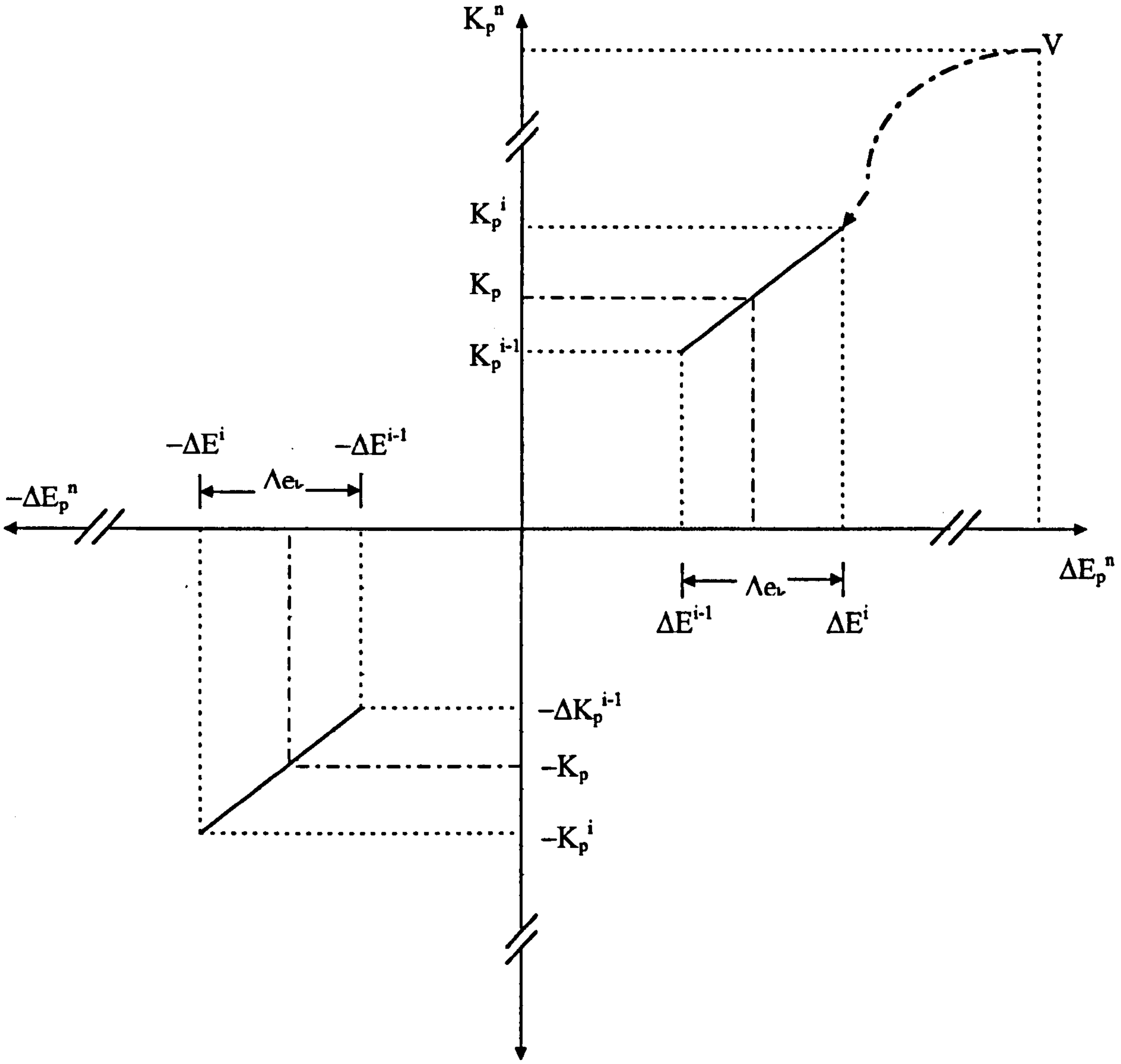


Figure (3.6.a) Multipule-Zoned Interpolated Gain Functions For Proportional Gain (Implemented using dual-zone method)

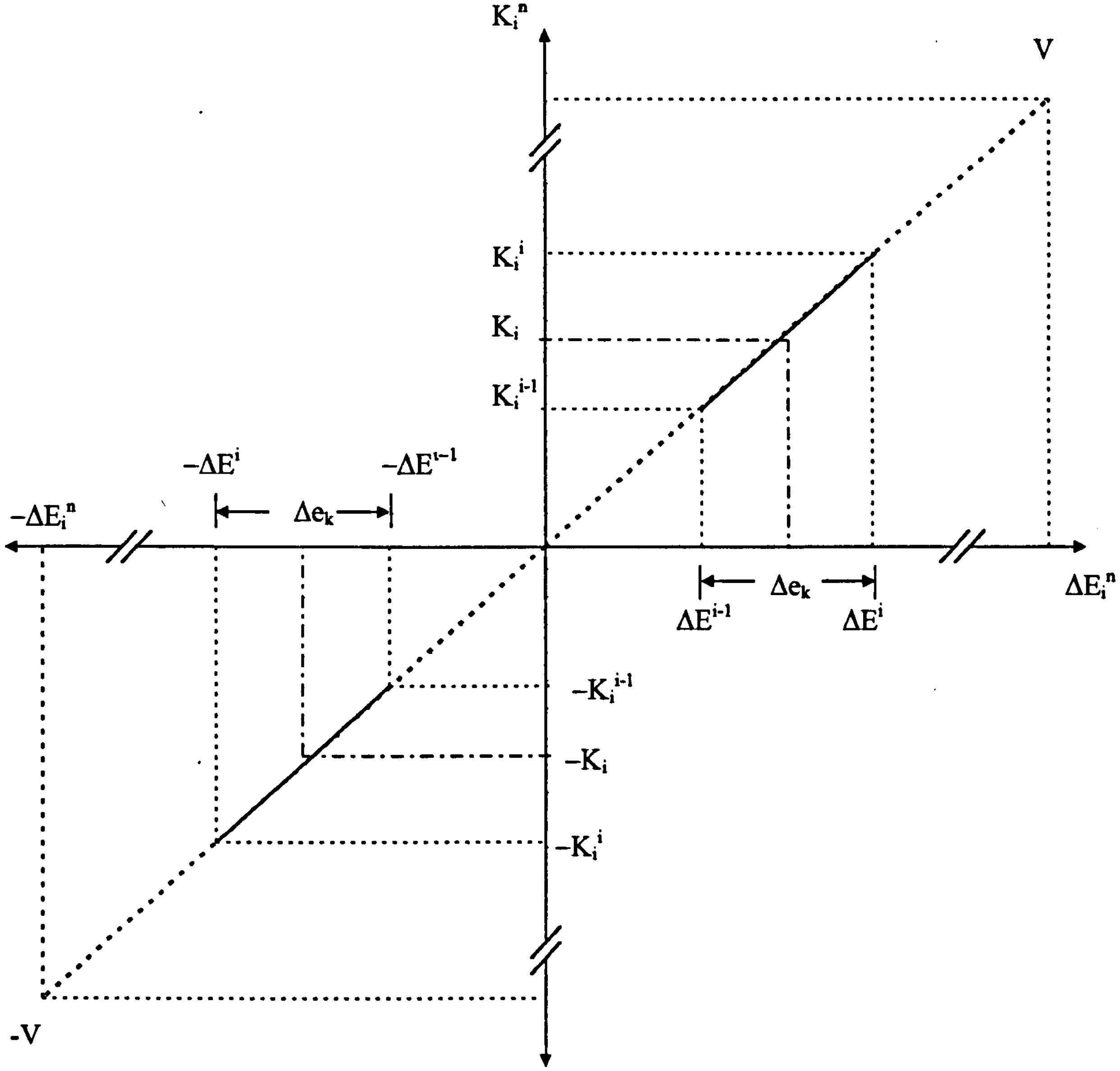


Figure (3.6.b) Multipule-Zoned Interpolated Gain Functions For Integral Gain (Implemented using dual-zone method)

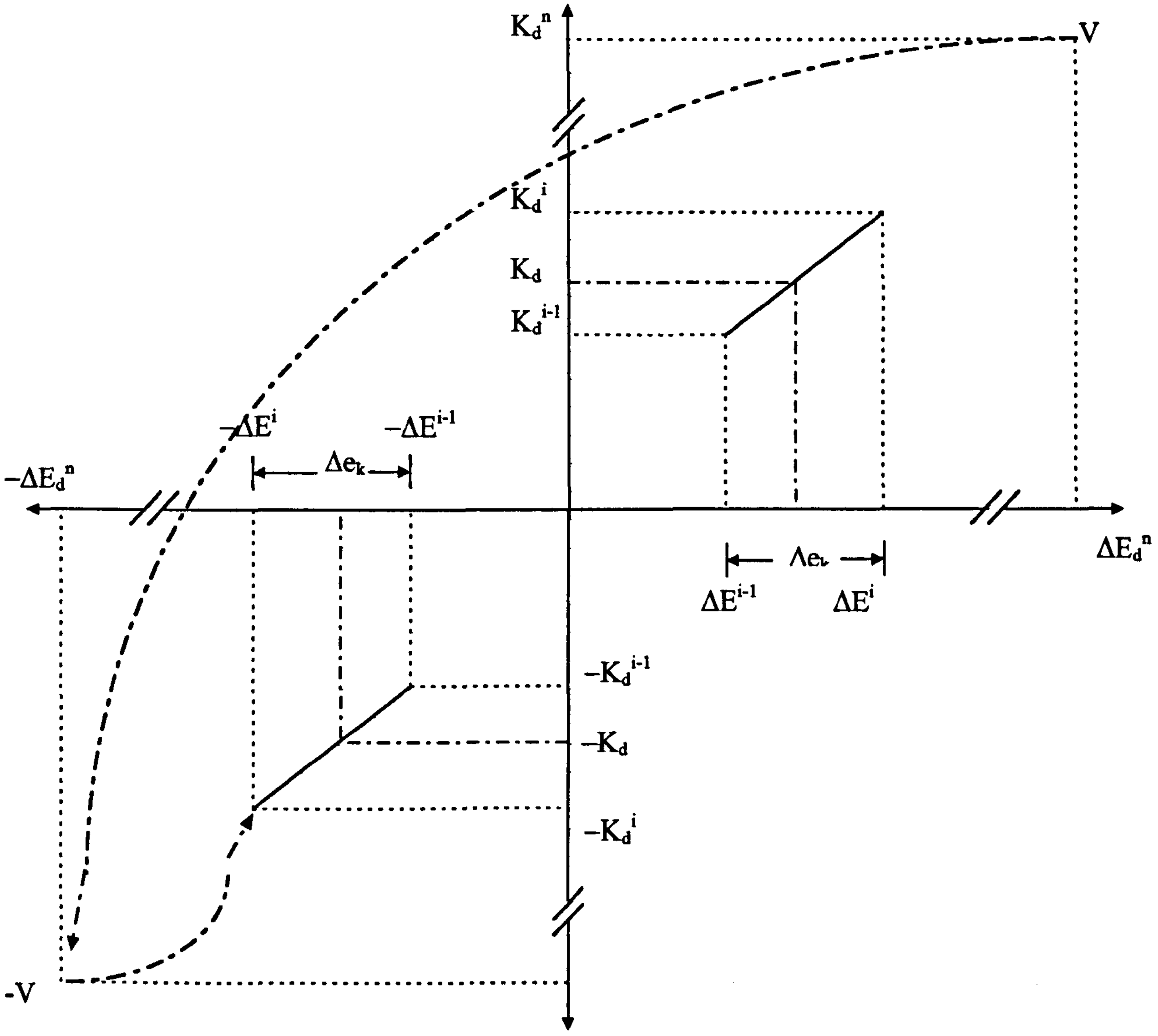


Figure (3.6.c) Multipule-Zoned Interpolated Gain Functions For Derivative Gain
(Implemented using dual-zone method)

The current sub-zone for any proportional gain can be found by using an equation of the form

$$i = \Delta e_k * \delta E_p \qquad 3.55$$

where

* is integer division, $e_k \epsilon R$, $\Delta u_k \epsilon R$, $n \epsilon U$, i is current sub-zone number,

The equation representing the proportional gain within any sub-zone is of the form

$$K_p = K^{i-1}_p + (K^i_p - K^{i-1}_p)\ (\Delta e_k - \Delta E^{i-1})\ /\ \delta E_p \qquad 3.56$$

where K^i_p is the value of the proportional gain in the ith sub-zone.

Figure (3.6b) represents the profile of the integral gain function. By considering figure(3.3b), and assuming equal sub-zone lengths, the equation representing the sub-zone length for the integral gain can be represented as

$$\delta E_i = \Delta E_i\ /\ n \qquad 3.57$$

where ΔE_i is the over all zone length, and n is the number of sub-zones.

The current sub-zone for any integral gain can be found by using equation of the form

$$i = e_k * \delta E_i \qquad 3.58$$

where

$*$ is integer division, $e_k \in R$, $\Delta u_k \in R$, $n \in U$, i is current sub-zone number,

The equation representing the integral gain within any sub-zone is of the form

$$\mathcal{K}_i = \mathcal{K}^{i-1}_i + (\mathcal{K}^i_i - \mathcal{K}^{i-1}_i)(e_k - \Delta E^{i-1}) / \delta E_i \qquad 3.59$$

where $\mathcal{K}^i_i$ is the value of the integral gain in the ith sub-zone.

Figure (3.6c) represents the profile of the derivative gain function. By considering figure(3.6c), and assuming equal sub-zone lengths, the equation representing the sub-zone length for the derivative gain can be represented as

$$\delta E_d = \Delta E_d / n \qquad 3.60$$

where ΔE_d is the over all zone length, and n is the number of sub-zones.

The current sub-zone for any derivative gain can be found by using equation of the form

$$i = \Delta^2 e_k * \delta E_p \qquad 3.60$$

where

* is integer division, $\Delta^2 e_k \epsilon R$, $\Delta u_k \epsilon R$, $n \epsilon U$, i is current sub-zone number,

The equation representing the derivative gain within any sub-zone is of the form

$$\mathcal{K}_d = \mathcal{K}^{i-1}_d + (\mathcal{K}^i_d - \mathcal{K}^{i-1}_d)\,(\Delta^2 e_k - \Delta E^{i-1}) \,/\, \delta E_d \qquad 3.61$$

where $\mathcal{K}^i_d$ is the value of the derivative gain in the ith sub-zone

The zoning method is a method used currently by most designers. But the above implementation of the zone method has not been used before by any designers. This is because of the complexity of the implementation, as the design method is impossible to implement without the use of GAs. The GA is used to search the operating envelope for the best values of the non-linear controller gains for the controllers being designed. The GA is then used for the tuning of the controllers, as illustrated in figure(3.5).

In order to use GA's to select the tuning parameters in such a way as to produce satisfactory response in the case of a step input response, it is only necessary to encode the elements of the tracking zone gain functions , plus the feed forward gains needed for the set point change zone, as binary strings.

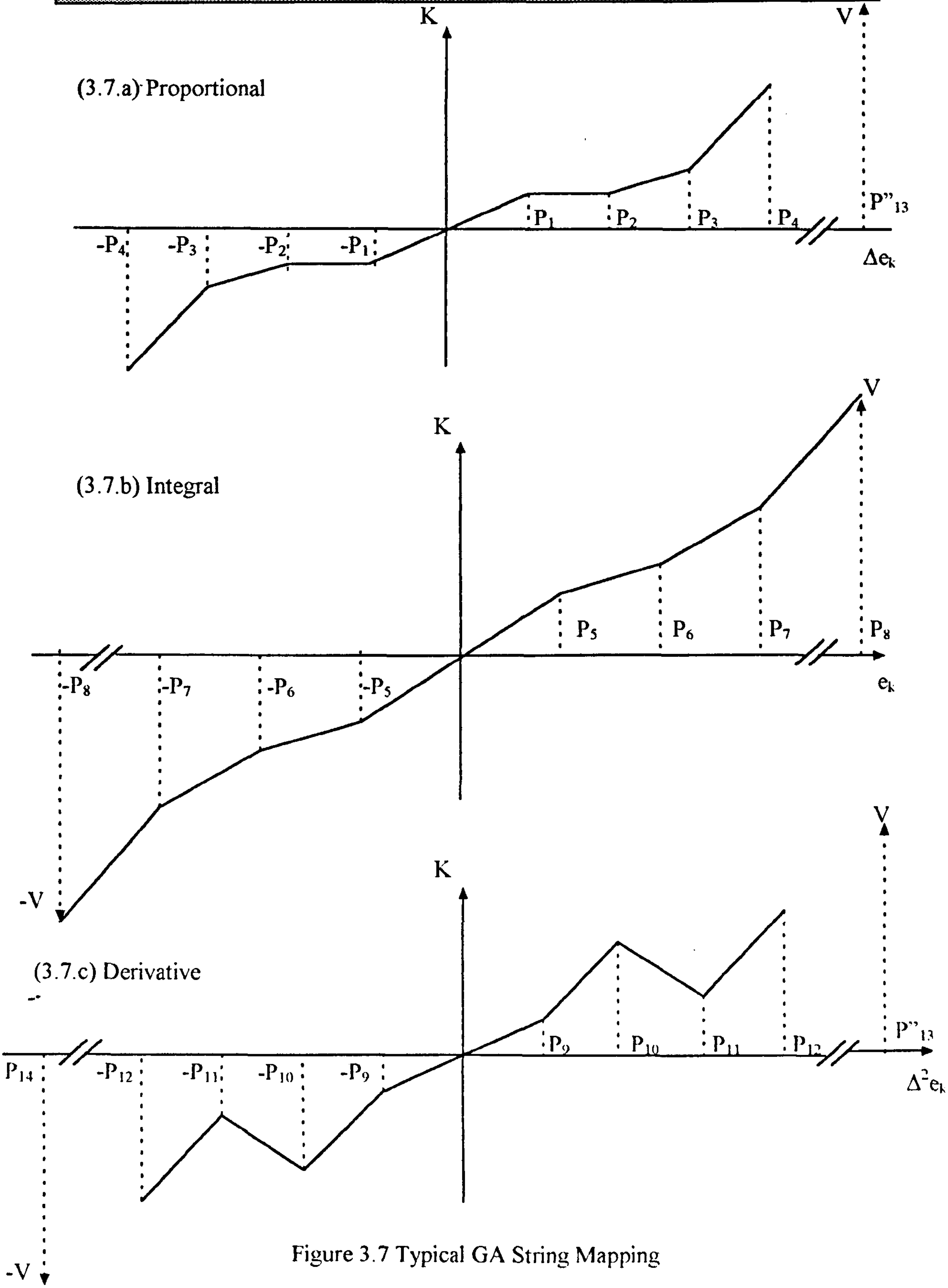
The binary string would be represented as

$$\{ \, (\mathcal{K}^{1}{}_{p} \, \ldots\ldots\ldots \, \mathcal{K}^{n}{}_{p}) \;\; (\mathcal{K}^{1}{}_{i} \, \ldots\ldots\ldots \, \mathcal{K}^{n}{}_{i}) \;\; (\mathcal{K}^{1}{}_{d} \, \ldots\ldots\ldots \, \mathcal{K}^{n}{}_{d}) \;\; (\quad F_0 \quad F_1 \quad) \, \}$$

proportional integral derivative set point change points

Where the whole string contains elements of the tracking zone for the three gain functions plus the feed forward gains.

Random initialization is the approach used to initialize the initial population, in this case as well, Since this approach requires the least knowledge-acquisition effort and provides a lot of diversity for the GA's to work with. The process of interviewing is also used to insure that the randomly generated variables do not initially violate any constraints on the function to be tuned. The system incorporates both the linear plant and the non-linear digital PID controllers, the controllers are designed by randomly generated sets of tracking and feed forward gains by the GA's. A stability test is then carried on all the controllers. Incase of violation of stability in any of the cases, the randomly generated set of Δu_{kn}'s, and the feed forward parameters will not be included in the initial population. This type of initialization is essential in this type of design, since otherwise their would be a danger of creating an initial population which so many of its members violate the constraints on the controllers



(3.7.a) Proportional

(3.7.b) Integral

(3.7.c) Derivative

Figure 3.7 Typical GA String Mapping

being designed, Figure 3.9 shows the sequence of genetic algorithms.

Following the initialization, the objective function is introduced and its value is calculated using decoded values of the parameters in each string. Thus for example, if minimum ISE is regarded as the ultimate design requirement, genetic algorithms can be readily used to select the best set of tracking and the feed forward gains such that the ISE is minimised.

In the genetic design of non-linear PID controllers the plant under consideration is subjected to a command input (i.e. unit step), then the performance index is computed for the plant. Therefore, for each member in the population the function for ISE given by equation (3.58) is valid.

Minimization of this performance index over the entire population can be rapidly obtained by using the genetic operations of reproduction, crossover, and mutation. It is interesting to note that the conditions for the existence of a non-linear PID controllers for the plant under consideration will be automatically satisfied by the genetic algorithms. This is obvious, because in the case of a violation of the constrained the corresponding set of tracking and feed forward gains produces large ISE and the result will be zero or low fitness, and by the action of the selection operator it would not be chosen for the next generation.

3.4 ILLUSTRATIVE EXAMPLES

3.4.1 PLANT 1

The procedure for the tuning of genetic control systems can be conveniently illustrated by designing a genetic non-linear PID control system for the open loop SISO plant with transfer function of the form

$$g(z) = \frac{0.8}{z^4(z-0.9)} \qquad 3.77$$

the sampling period is 0.1 sec.

The Arma model for the plant is of the form

$$y_k = a_0 y_{k-1} + b_0 u_{k-5} \qquad 3.78$$

The plant variables are given by

$a_0 = 0.9$

$b_0 = 0.8$

where the incremental PID controller is given by equation (3.48). The controller is implemented in the incremental form so as, to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. The controller is designed using the dual zone method introduced earlier.

3.3.1.1 DUAL-ZONED CONTROLLER

Initially the gains K_p, K_i and K_d are chosen as a function of the plant error such as that given by equations (3.51, 3.54, 3.57), i.e. dual zoned interpolated PID controller. Then the controller was designed by means of genetic algorithms, so as that the Integral square error is minimised for the plant. In this case, a population of 100, a crossover probability, $P_c=0.65$, and a mutation probability, $P_m = 0.01$, was used, also for this case also the maximum value for both the first and second order backward difference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design. Figure 3.10 shows the transient response of the genetically designed controller.

3.3.1.2 MULTI-ZONED CONTROLLER

The gains K_p, K_i and K_d are chosen as a function of the plant error such as that given by equations (3.62, 3.68, 3.74), i.e. multi-zoned interpolated PID controller, the tracking zone is split into 4 sub-zones. Then the controller was designed by means of genetic algorithms, so as that the integral square error is minimised for the plant. In this case, a population of 100, a crossover probability, $P_c=0.65$, and a mutation probability, $P_m = 0.01$, was used, also for this case also the maximum value for both the first and second order backward deference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design. Figure 3.11 shows the transient response of the genetically designed controller, figure 3.11a, 3.11b, and 3.11c shows the resulting profiles for the proportional, integral, and derivative gains respectively.

3.3.1.3 LINEAR CONTROLLER

To contrast the difference between linear and non-linear PID controllers, a genetically designed linear PID controllers was also considered, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, and K_d chosen by the GAs to minimise the ISE for the fixed controller. In this case, a population of 100, a crossover probability, p_c = 0.65, and a mutation probability, P_m = 0.01 was used. Figure 3.12 shows the transient response of the genetically designed controller. Table (3.1) shows the ISEs for the three controllers.

CONTROLLER	DUAL PID	MULTI ZONED	LINEAR
ISE	7.03	5.72	9.16

Table (3.1).

Table (3.2) shows the gains for the two linear controllers.

CONTROLLER	K_P	K_i	K_d
LINEAR	3.53	2.71	0.45
DUAL-ZONED	0.65	1.363	0.6

Table (3.2).

Table (3.3) shows the feed forward gains for the linear dual-zoned PID controller.

Feed Forward Gains	2.818	7.848

Table (3.3).

Table (3.4) shows the gains for the multi-zoned non-linear PID controller.

Gains	Zone1	Zone2	Zone3	Zone4
$\mathcal{K}_p$	0.696	0.678	0.315	0.434
$\mathcal{K}_i$	3.383	6.340	9.614	6.044
$\mathcal{K}_d$	0.958	0.877	0.669	0.009

Table (3.4).

Table (3.5) shows the feed forward gains for the multi-zoned non-linear PID controller.

Gains	2.458	6.753

3.3.2 PLANT 2

A second plant was considered, to investigate the effectiveness of the genetic algorithms in designing linear, and non-linear controller. The plant considered has a transfer function of the form:

$$g(z) = \frac{0.03573 + 0.044625\,z}{z^4(0.0513423 - 1.4331\,z^2)} \tag{3.79}$$

the sampling period is 0.1 sec.

The Arma model for the plant is of the form as

$$y_k = a_0 y_{k-1} + a_1 y_{k-2} + b_0 u_{k-5} + b_1 u_{k-6} \tag{3.80}$$

The plant variables are given by,

a_0 = 1.4331,

a_1 =-0.51342,

b_0 = 0.044625,

b_1 = 0.03573,

and the incremental PID controller is governed by equation (3.48). The plant was tested under different operating conditions as shown in the examples bellow:

3.3.2.1 DUAL-ZONED CONTROLLER

Initially the gains K_p, K_i, and K_d, where chosen as in equations (3.51, 3.54, 3.57), i.e. dual-zoned PID controller with linear plant. The controller was designed by means of genetic algorithms, such that the Integral Square Error ISE is minimised for the plant. In this case, a population of 100, a crossover probability, $P_c = 0.65$, and mutation probability, $P_m = 0.01$ was used, also for this case also the maximum value for both the first and second order backward deference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design. Figure 3.13, shows the transient response of the genetically designed non-linear controller for the above plant.

3.3.2.2 MULTI-ZONED CONTROLLER

The gains $\mathcal{K}_p$, $\mathcal{K}_i$ and $\mathcal{K}_d$ are chosen as a function of the plant error such as that given by equations (3.62, 3.68, 3.74), i.e. multi-zoned interpolated PID controller, the tracking zone is split into 4 sub-zones. Then the controller was designed by means of genetic algorithms, so as that the Integral square error is minimised for the plant. In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m = 0.01, was used, also for this case also the maximum value for both the first and second order backward deference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design. Figure 3.14 shows the transient response of the genetically designed controller, figure 3.14a, 3.14b, and 3.14c shows the resulting profiles for the proportional, integral, and derivative gains respectively.

3.3.2.3 LINEAR CONTROLLER

Finally to contrast the deference between linear and non-linear PID controllers, a genetically designed linear PID controller was also considered, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, K_d, are chosen by the GA to minimise the ISE for the fixed controller. In this case, a population of 100, a crossover probability, p_c = 0.65, and a mutation probability, P_m = 0.01 was used. Figure 3.15 shows the transient response of the genetically designed controller. Table (3.6) shows the ISEs for the three controllers.

CONTROLLER	DUAL-ZONED	MULTI ZONED	LINEAR
ISE	7.15	6.153	10.57

Table(3.6)

Table (3.7) shows the gains for the two linear controllers.

CONTROLLER	K_p	K_i	K_d
LINEAR	4.353	7.711	0.647
DUAL-ZONED	0.167	18.926	0.014

Table (3.7).

Table (3.8) shows the feed forward gains for the linear dual-zoned PID controller.

Feed Forward Gains	3.704	5.161

Table (3.8).

Table (3.9) shows the gains for the multi-zoned non-linear PID controller.

Gains	Zone1	Zone2	Zone3	Zone4
$\mathcal{K}_p$	0.534	0.295	0.141	0.019
$\mathcal{K}_i$	1.632	1.188	1.533	1.266
$\mathcal{K}_d$	0.165	0.054	0.086	0.004

Table (3.9).

Table (3.10) shows the feed forward gains for the multi-zoned non-linear PID controller.

Gains	3.294	3.323

Table(3.10).

3.3.3 ROBUSTNESS TEST

This test is aimed at finding out how robust are the genetically designed dual zone PID controllers are for changes in the plant. In this case the plants used in the illustrative example were modified to produce different plants . To do this test consider plant 1 with transfer function of the form

$$g(z) = \frac{0.8}{z^4(z-0.9)} = \frac{\beta}{z^{\gamma}(z-\alpha)}$$

To produce the new plants the value of α, β, and γ are changed. Tables (3.11) shows the range of plants considered in this robustness test, and the respective ISEs obtained,

Plant	Lin ISE	D-Z ISE	M-Z ISE	β	α	γ
1	9.16	7.15	6.153	0.8	0.90	5
2	11.764	8.442	6.637	0.8	0.90	6
3	13.07	16.677	8.109	1.2	0.90	6
4	14.867	83.01	52.05	1.2	0.94	6
5	9.976	27.887	35.52	1.2	0.94	4
6	13.967	-- 14.855	X	1.2	0.60	4
7	22.652	42.486	X	0.5	0.60	4

Table (3.11)

Figure (3.16 to 3.21) show the response of the plants for the linear dual-zoned controller designed using the original plant1. Figure (3.22 to 3.25) show the response of the plants for the non-linear dual-zoned controller designed using plant.

Figure (3.26 to 3.31) show the response of the plants for the linear controller designed using the plant.

3.3.4 CONTROLLER ERROR CLIPPING

From the results obtained during the robustness test carried out in the previous section, it is evident that in some cases the change in plant has forced the values of Δe_k and $\Delta^2 e_k$ out of the working region of the controller. Once this has occurred the gains are extrapolated rather than interpolated. In general extrapolation can be unsatisfactory. Therefore it has been decided to introduce the concept of controller clipping. This would mean that if during the transient the first or the second order backward difference in error is greater that the maximum design value it would make the values equal to zone length (i.e. If $\Delta e_k > \Delta e_{kmax}$ $\Delta e_k = \Delta e_{kmax}$). The diagram shown in figure (3.16) shows how the clipping concept would be used in practice.

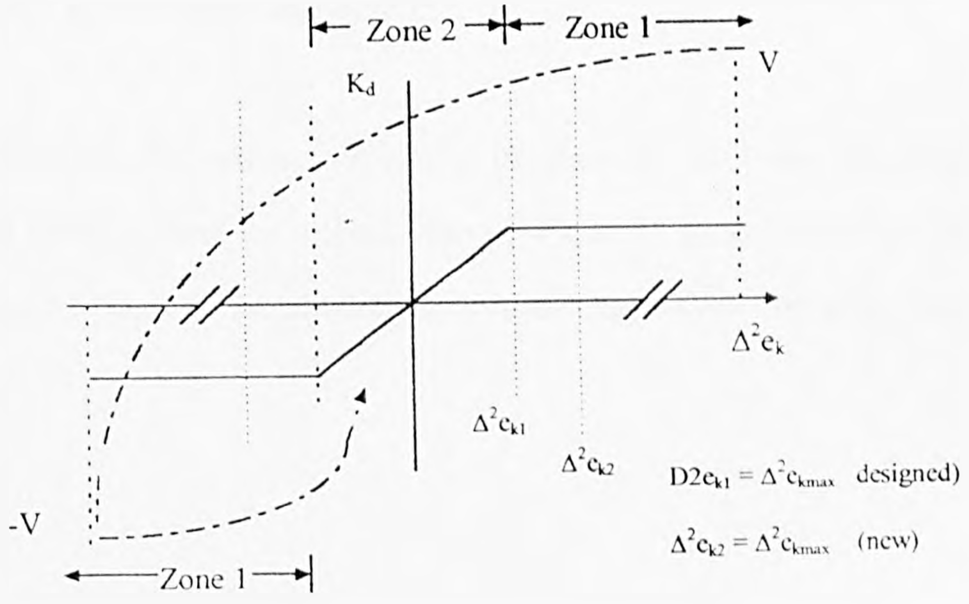

Zone 2
Zone 1
V
K_d
$\Delta^2 e_k$
$\Delta^2 e_{k1}$
$\Delta^2 e_{k2}$
$D2e_{k1} = \Delta^2 e_{kmax}$ designed)
$\Delta^2 e_{k2} = \Delta^2 e_{kmax}$ (new)
-V
Zone 1

Figure 3.16 Controller Error clipping

This technique is often used in fuzzy logic controllers and frequently improves robustness. The results obtained using the clipped controller for the robustness test is given in table (3.12). The clipping process will reduce the amount of gain used at higher values of Δe_k and $\Delta^2 e_k$ and makes the controller more robust.

Plant	Lin ISE	C-DZ ISE	C-MZ ISE	β	α	γ
1	9.16	7.15	6.153	0.8	0.90	5
2	11.76	8.442	6.637	0.8	0.90	6
3	13.07	8.879	8.109	1.2	0.90	6
4	14.867	12.987.	23.693	1.2	0.94	6
5	9.976	5.6324	21.641	1.2	0.94	4
6	13.967	10.142	23.908	1.2	0.60	4
7	22.652	20.794	34.782	0.5	0.60	4

Table (3.12)

where C-DZ ISE is clipped dual zone ISE,

C-MZ ISE is clipped muli-zone ISE.

Figure (3.32 to 3.37) show the response of the plants for the linear dual-zoned controller designed using the original plant1. Figure (3.38 to 3.43) show the response of the plants for the non-linear dual-zoned clipped controller using plant in table (3.12).

3.4 CONCLUSIONS

The technique of genetic algorithms has been proposed as a mean of designing non-linear PID controllers for linear plants. It has been shown that the use of genetic algorithms for this purpose greatly facilitates the design of such controllers so that the ISE is minimised. It has also been shown that the use of dual zone control theory to design the non-linear controller improves the effectiveness of the controller performance. The straight line interpolation technique was used in this chapter to map the non-linear controller gains. The results have been illustrated by genetically designing digital PID controllers for linear plants. It has thus been shown in this chapter that genetic algorithms are a powerful, reliable, and simple means of handling constrained optimization problems. Indeed, comparing the results for the linear and dual-zoned linear controller indicates that the dual-zoned design is a more effective design method to employ in this thesis. Also from the results obtained it is evident that the GA is the only method available which can be used to design such controllers. The robustness test indicates that the non-linear PID controllers are not as robust compared to the linear controllers. Furthermore the clipped non-linear PID controllers exhibits far better robustness properties than the un-clipped non-linear controllers. This is because the clipping forces the controller to be rate limited.

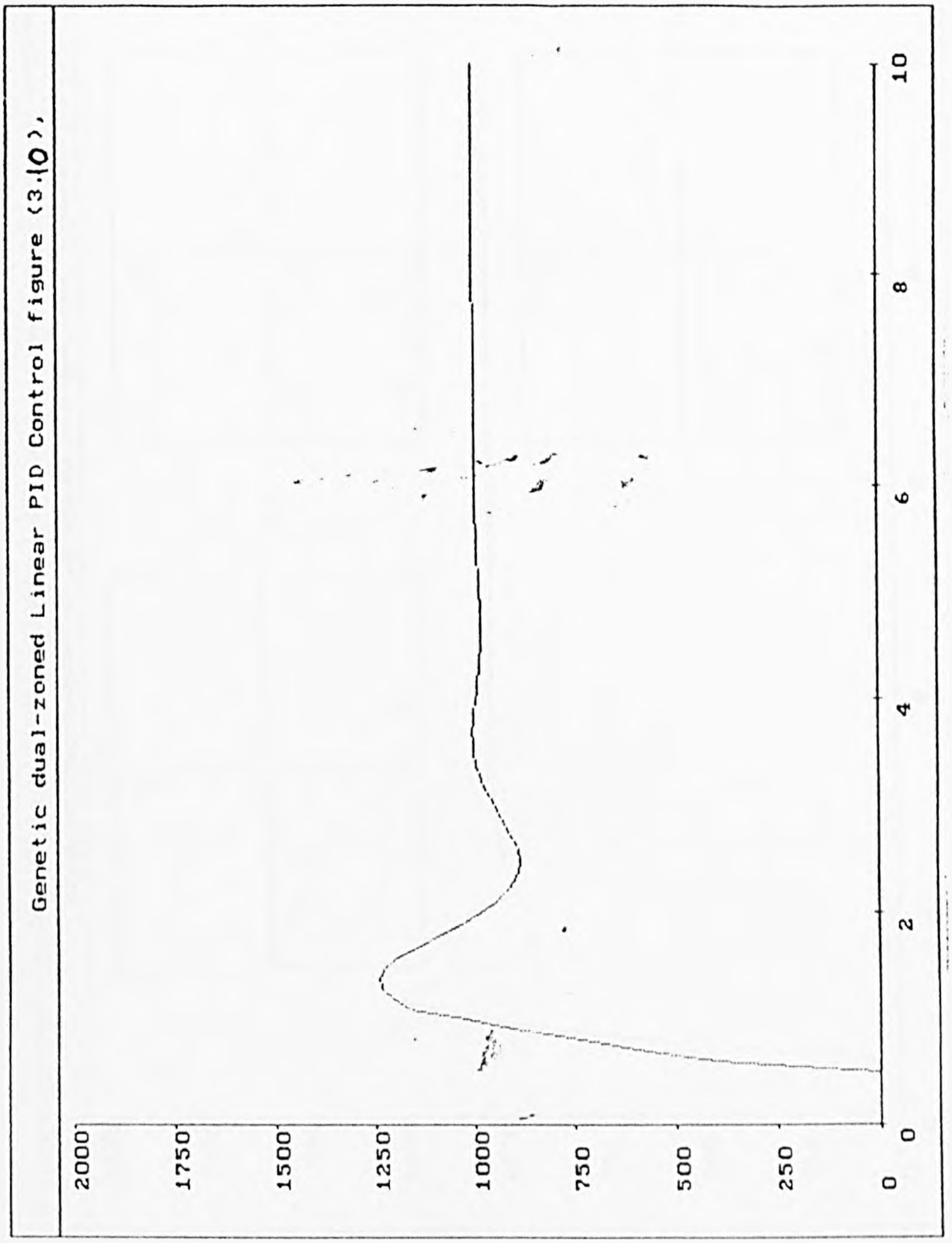


Figure (3.10) Transient response of the genetically designed dual-zoned PID controller for plant (1)

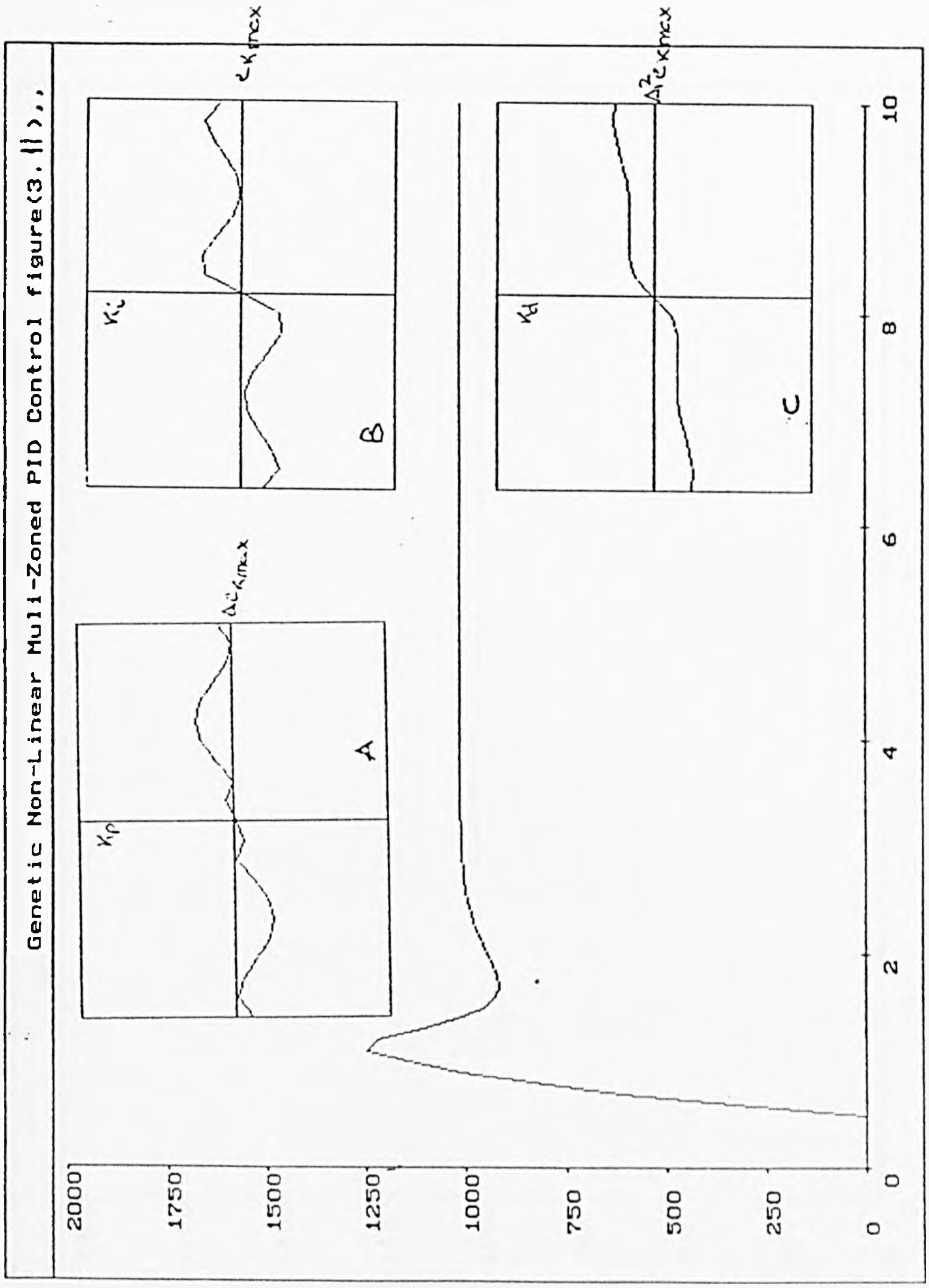


Figure (3.11) Transient response of the genetically designed multi-zoned PID controller for plant (1)

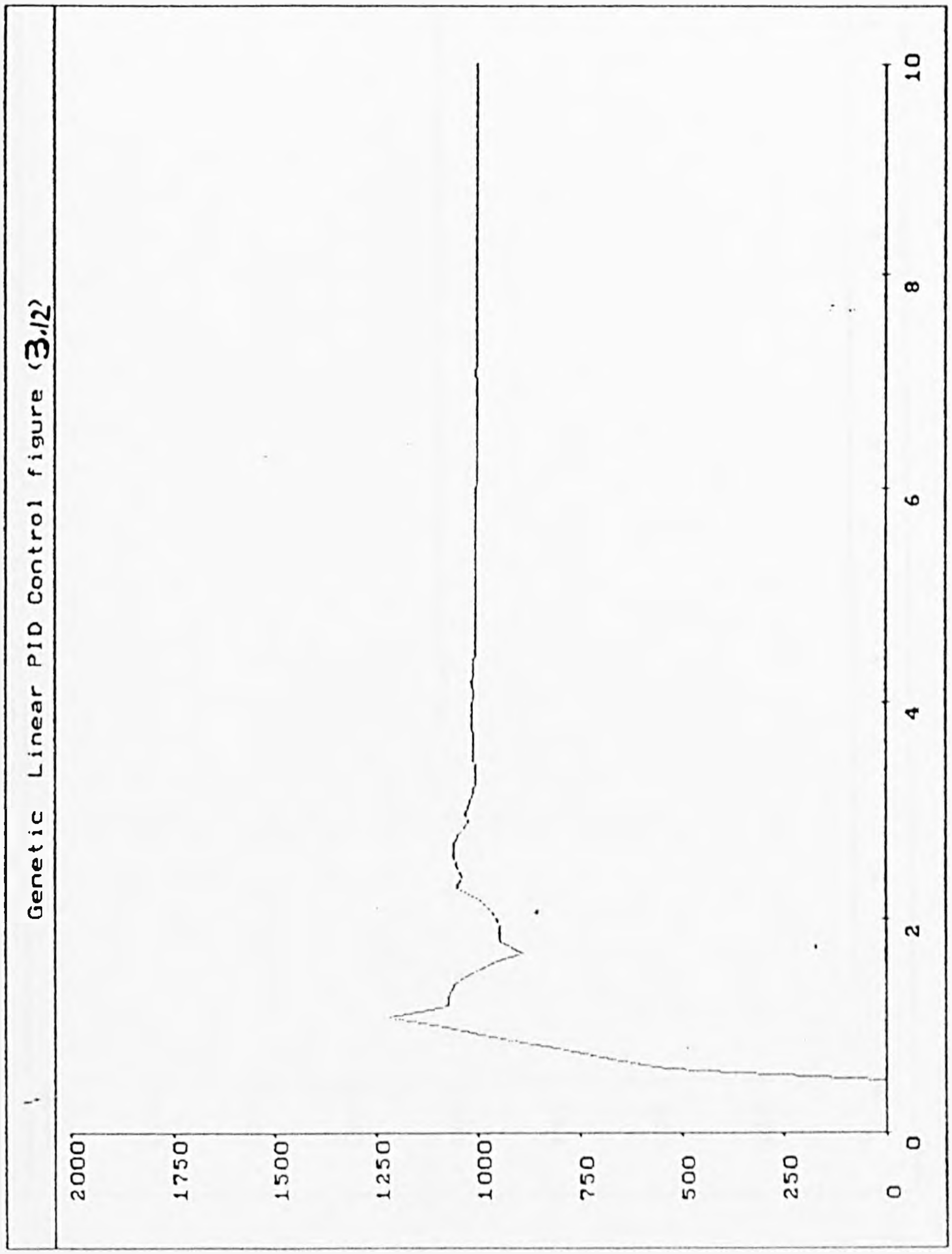


Figure (3.12) Transient response of the genetically designed linear PID controller for plant (1)

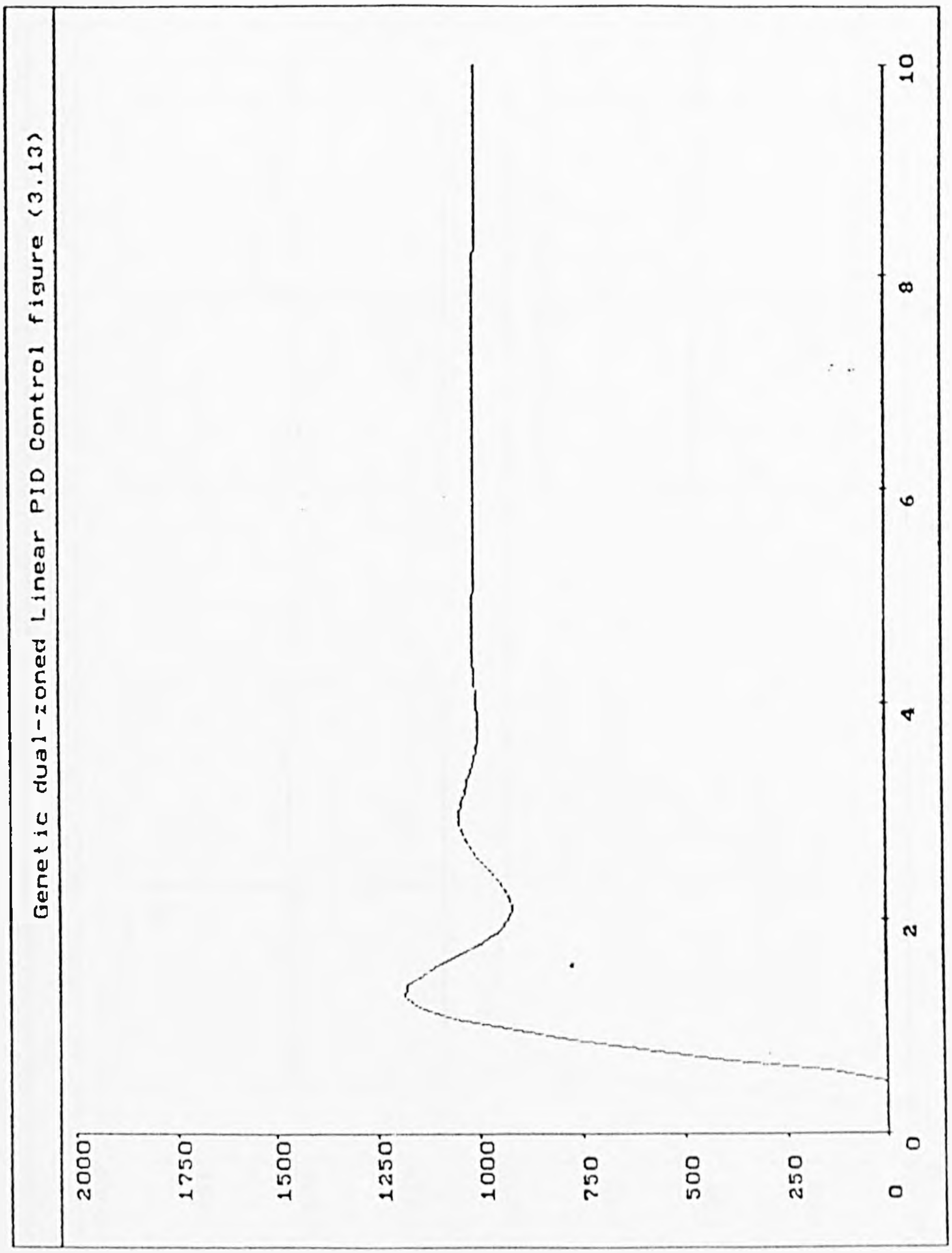


Figure (3.13) Transient response of the genetically designed dual-zoned PID controller for plant (2)

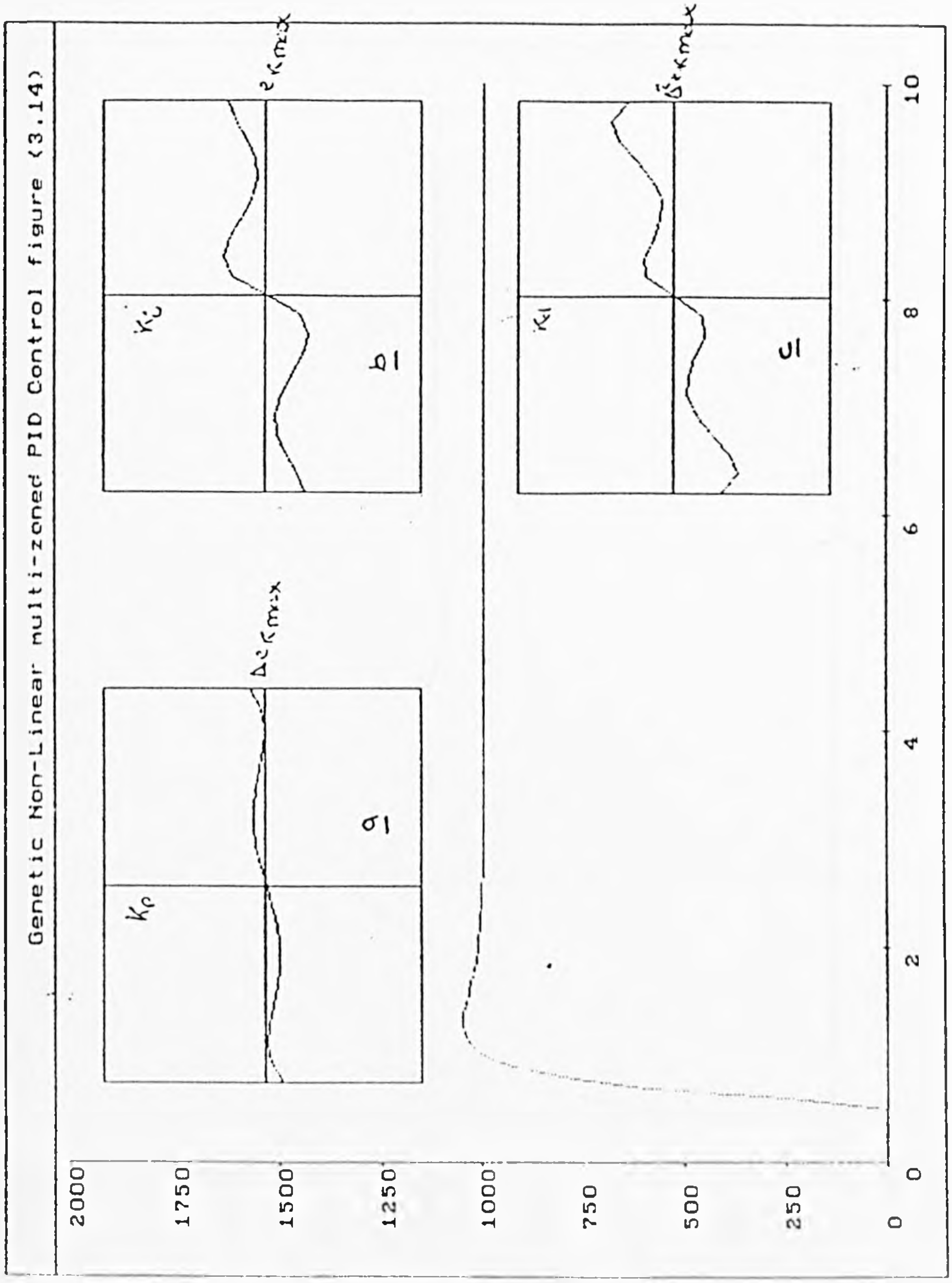


Figure (3.14) Transient response of the genetically designed multi-zoned PID controller for plant (1)

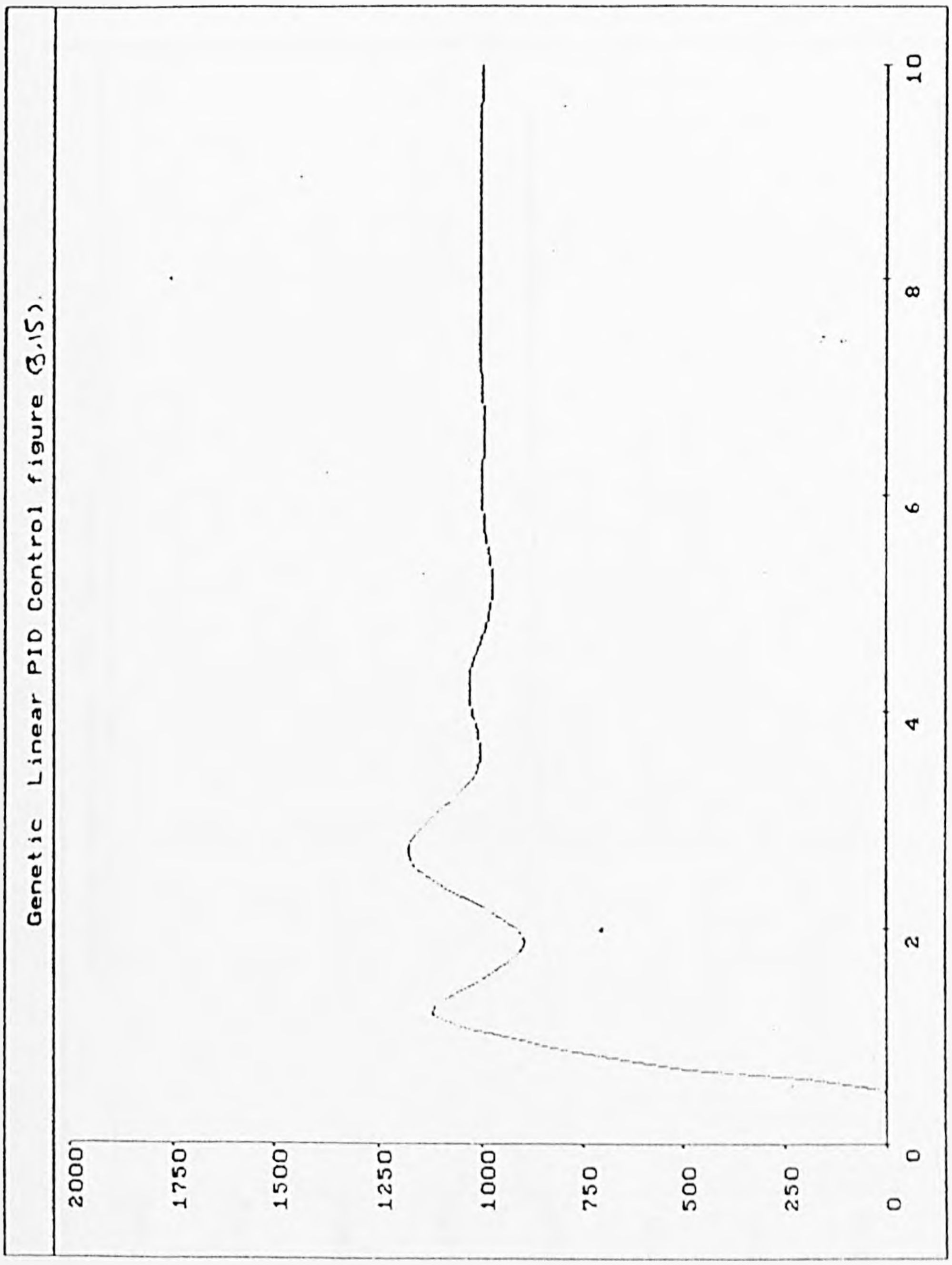


Figure (3.15) Transient response of the genetically designed linear PID controller for plant (2)

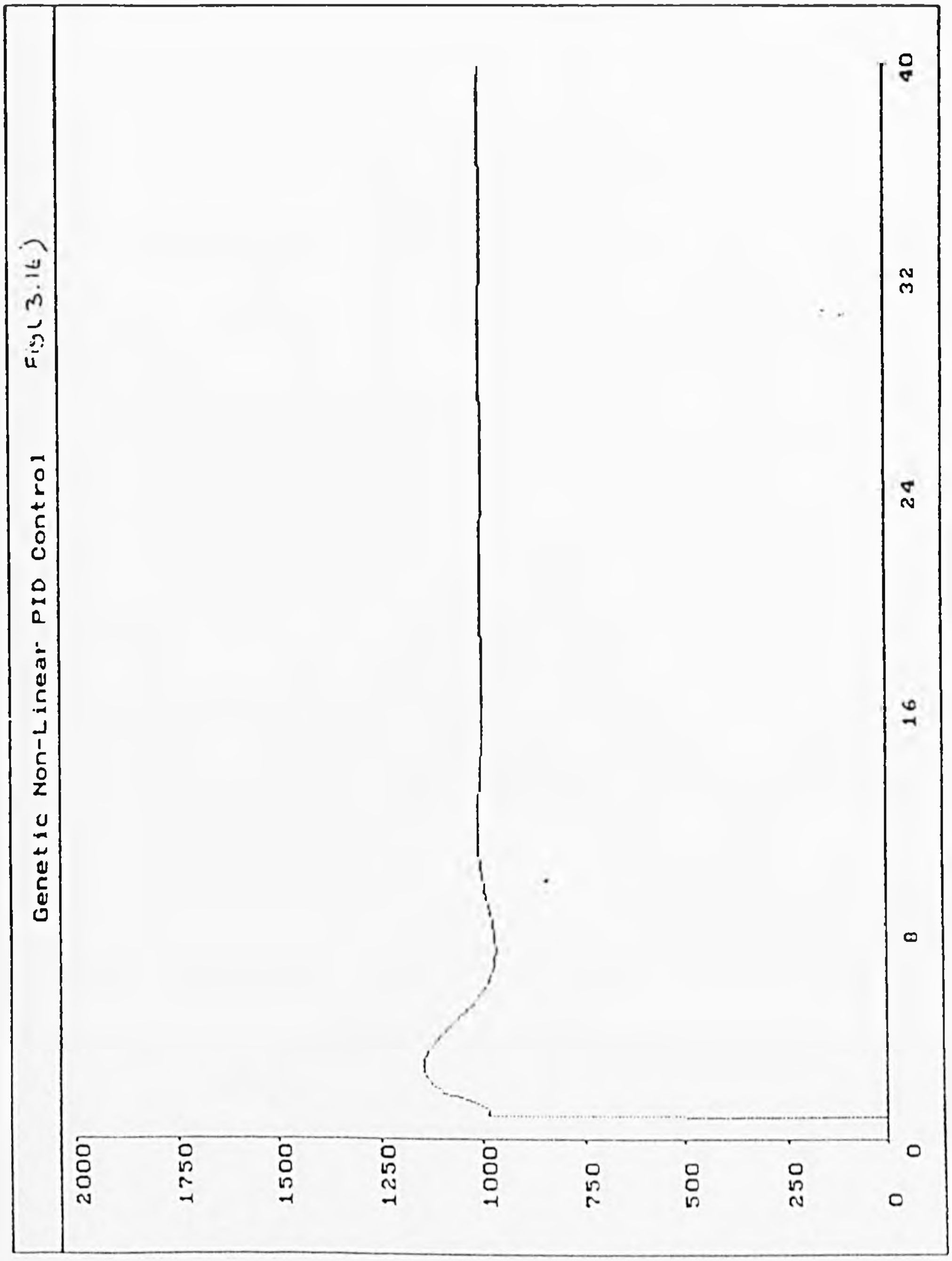


Figure (3.16) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (2), table (3.11).

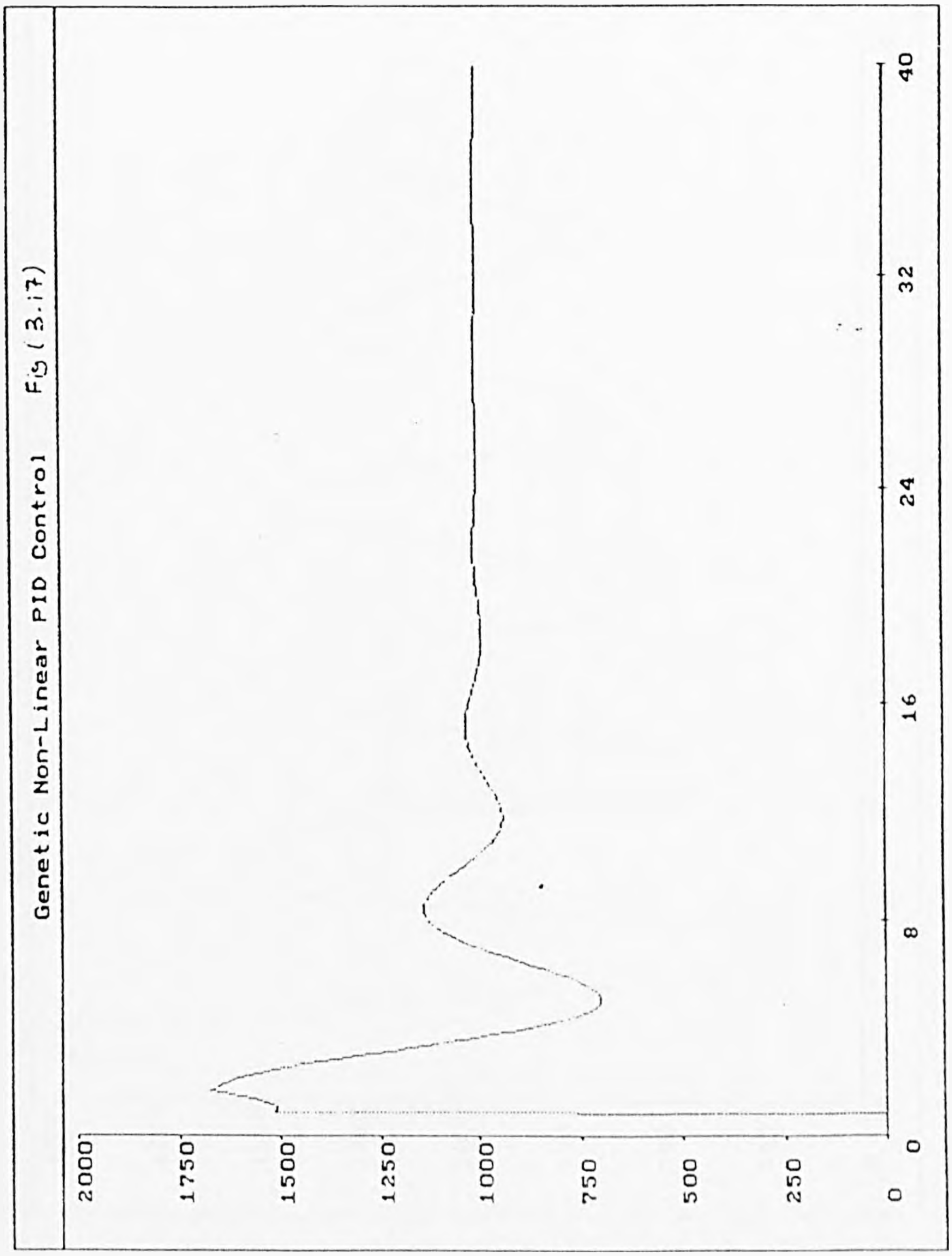


Figure (3.17) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (3), table (3.11). Figure (3.10)

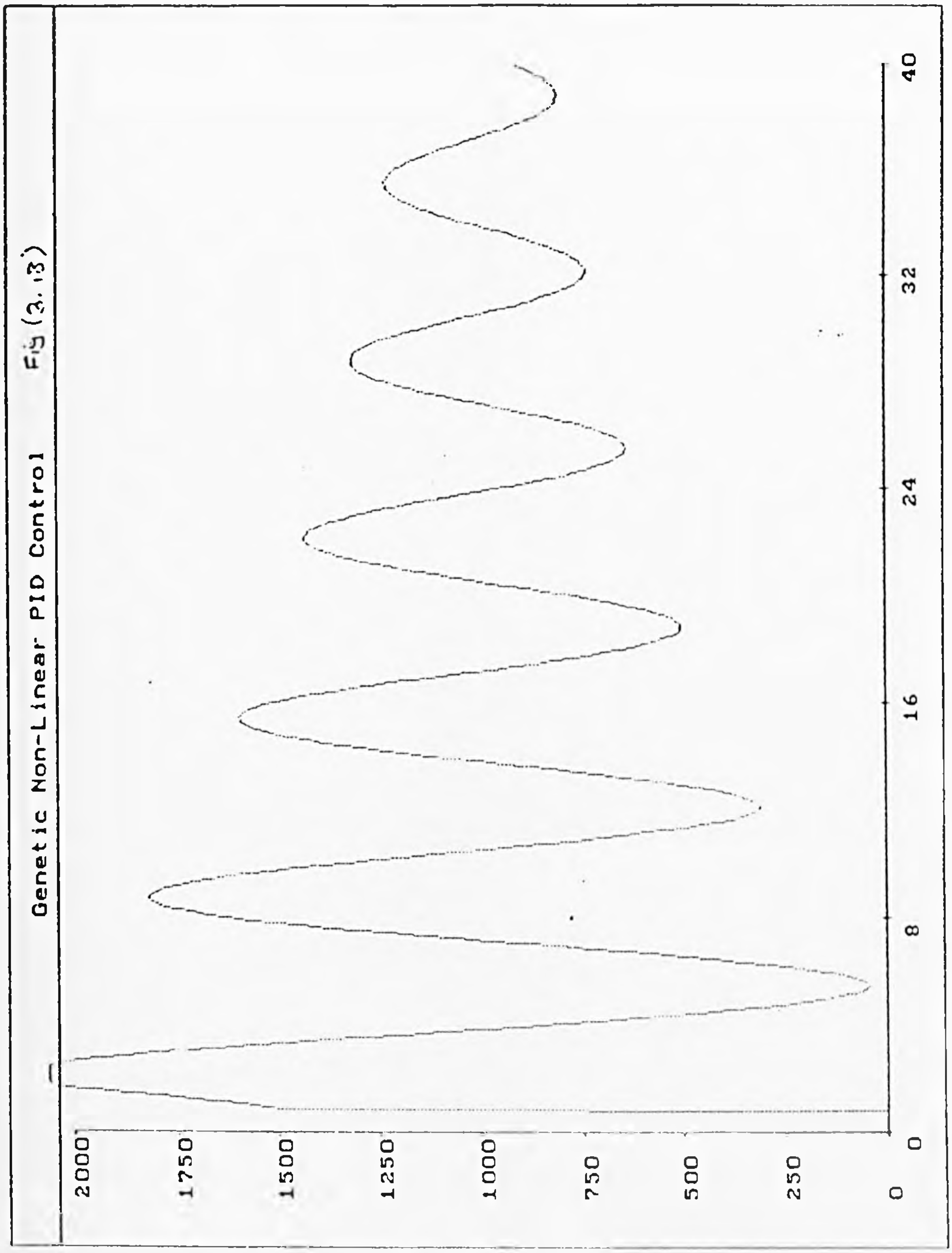


Figure (3.18) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (4), table (3.11).

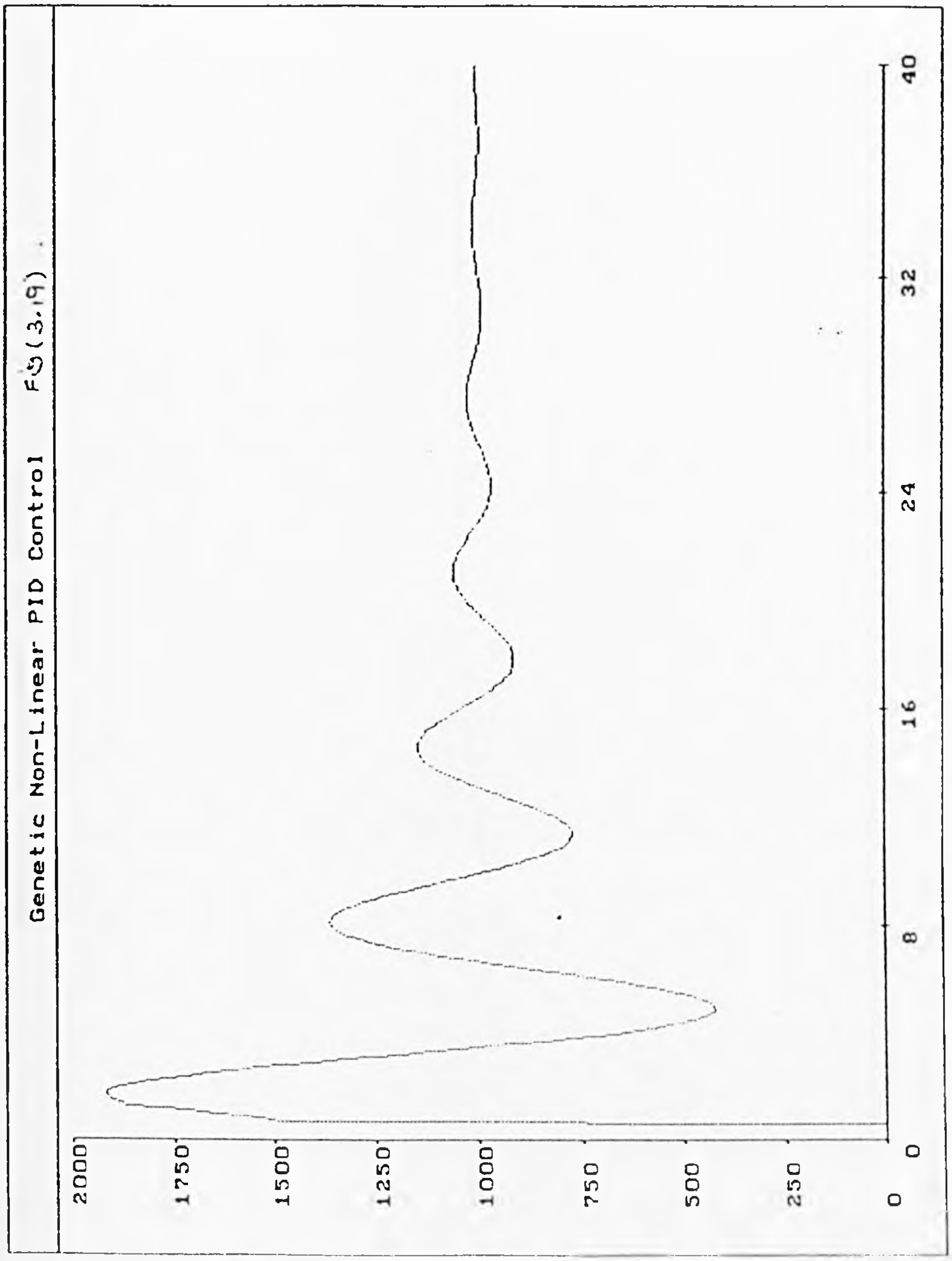


Figure (3.19) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (5), table (3.11).

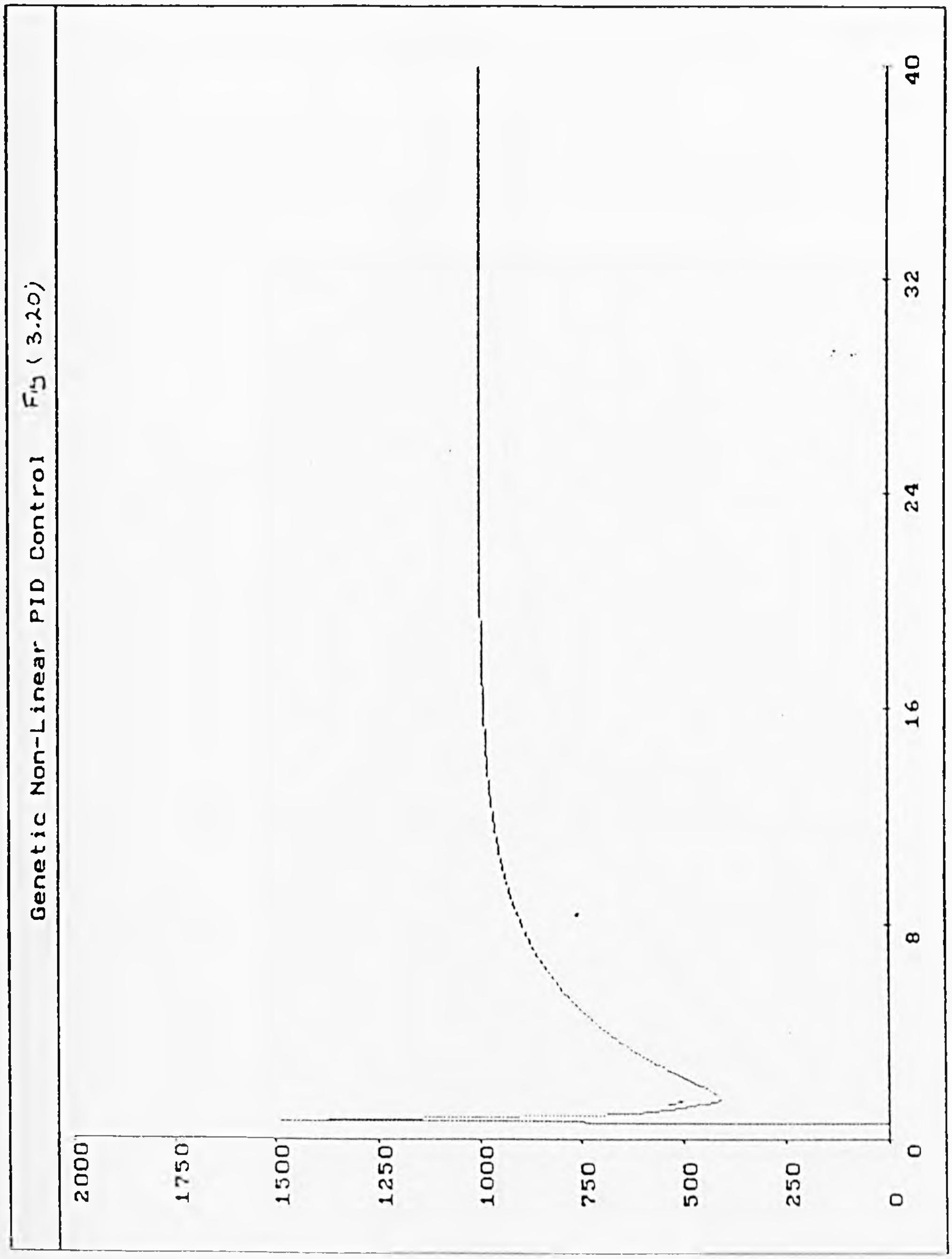


Figure (3.20) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (6), table (3.11).

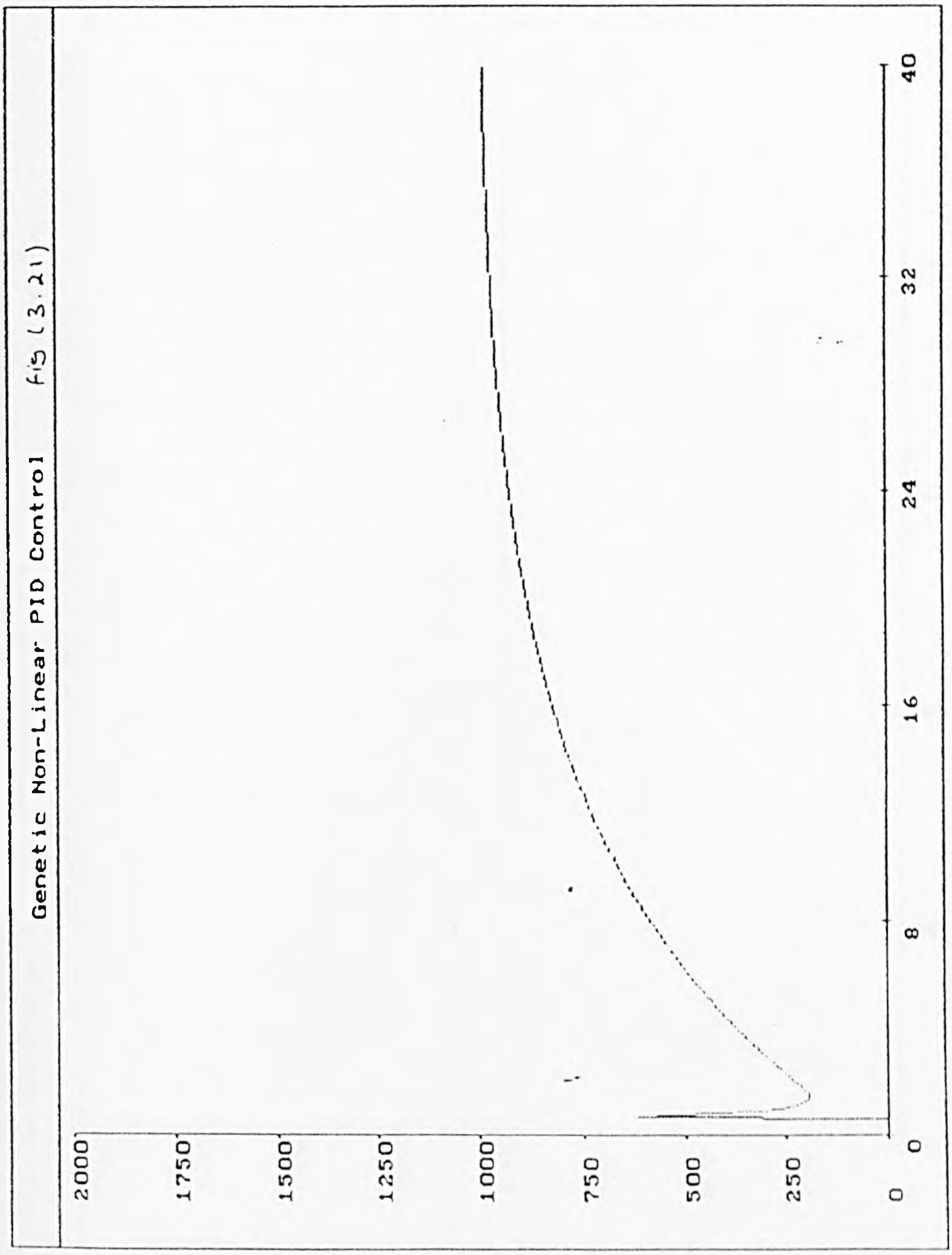


Figure (3.21) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (7), table (3.11).

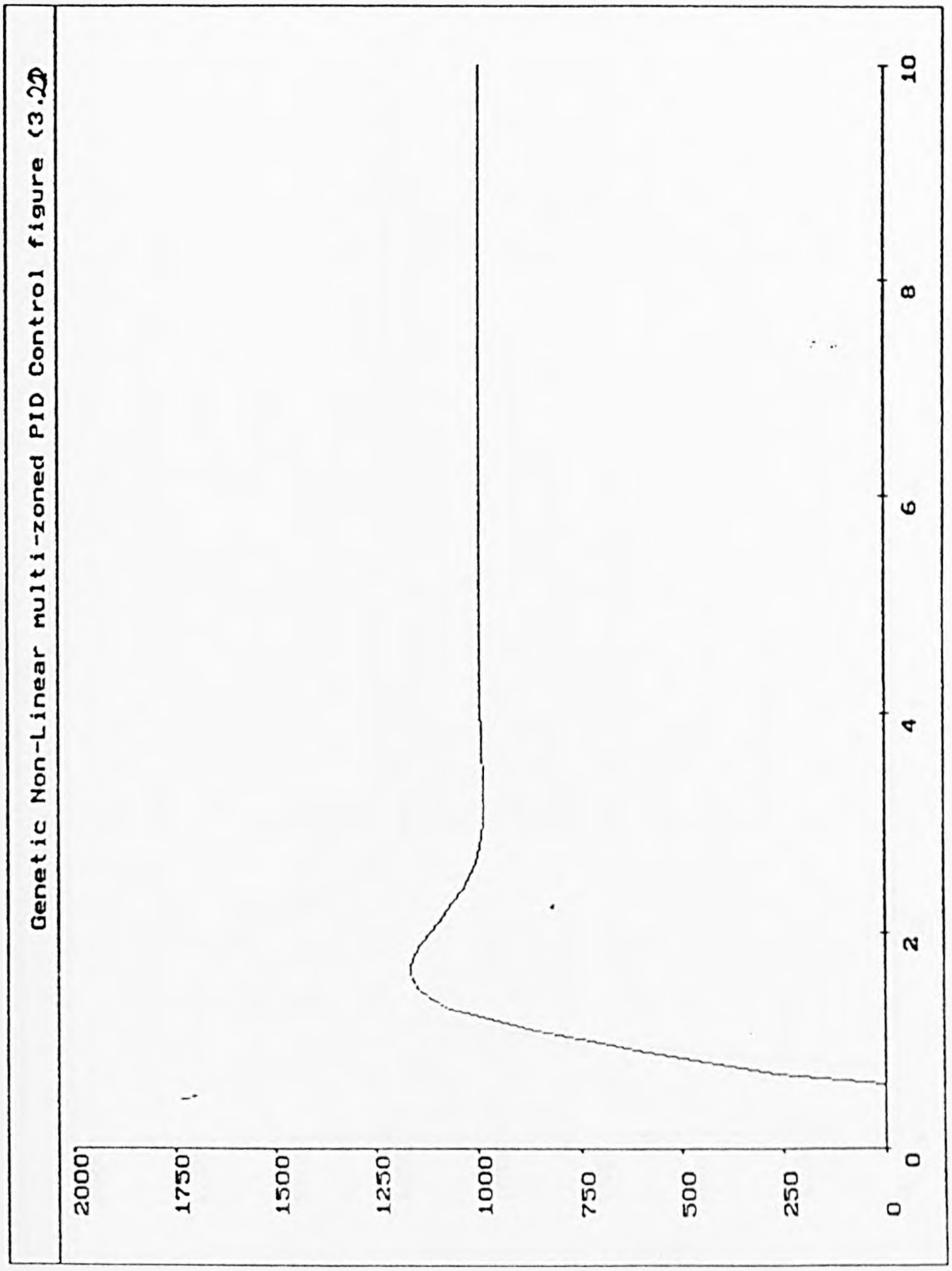


Figure (3.22) Transient response of robustness test using genetically designed non-linear dual-zoned PID controller for plant (2), table (3.11).

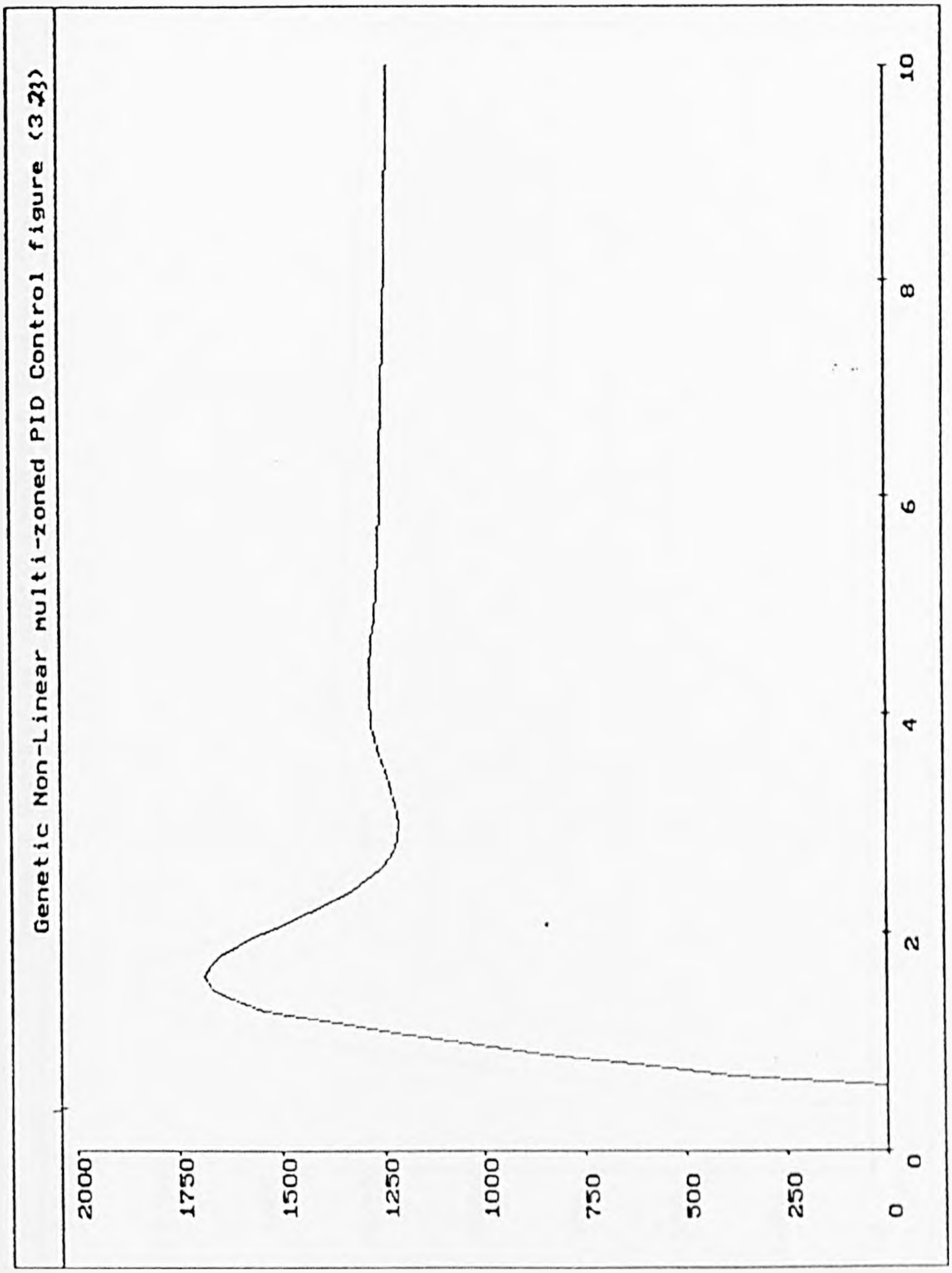


Figure (3.23) Transient response of robustness test using genetically designed non-linear dual-zoned PID controller for plant (3), table (3.11).

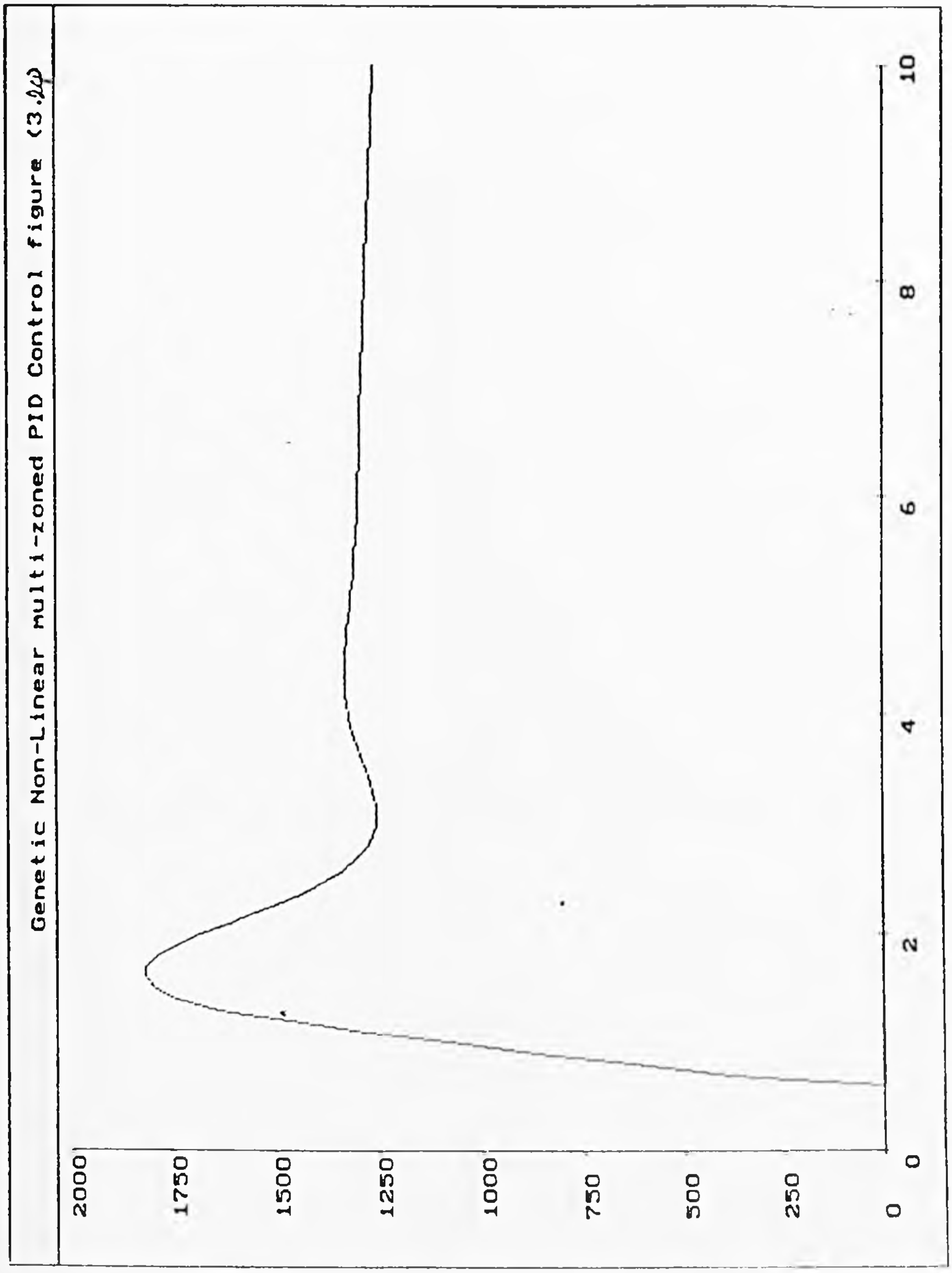


Figure (3.24) Transient response of robustness test using genetically designed non-linear dual-zoned PID controller for plant (4), table (3.11).

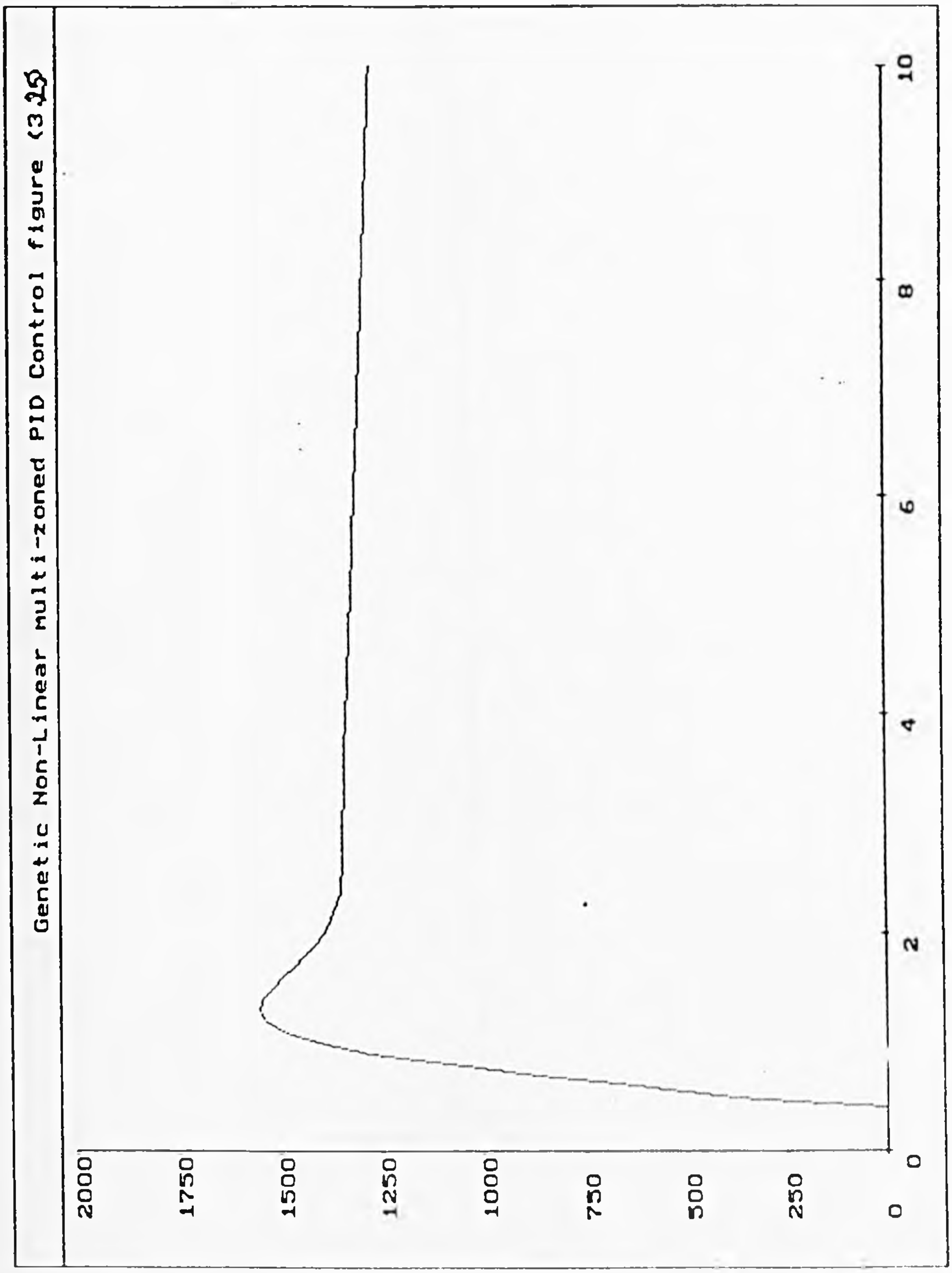


Figure (3.25) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (5), table (3.11).

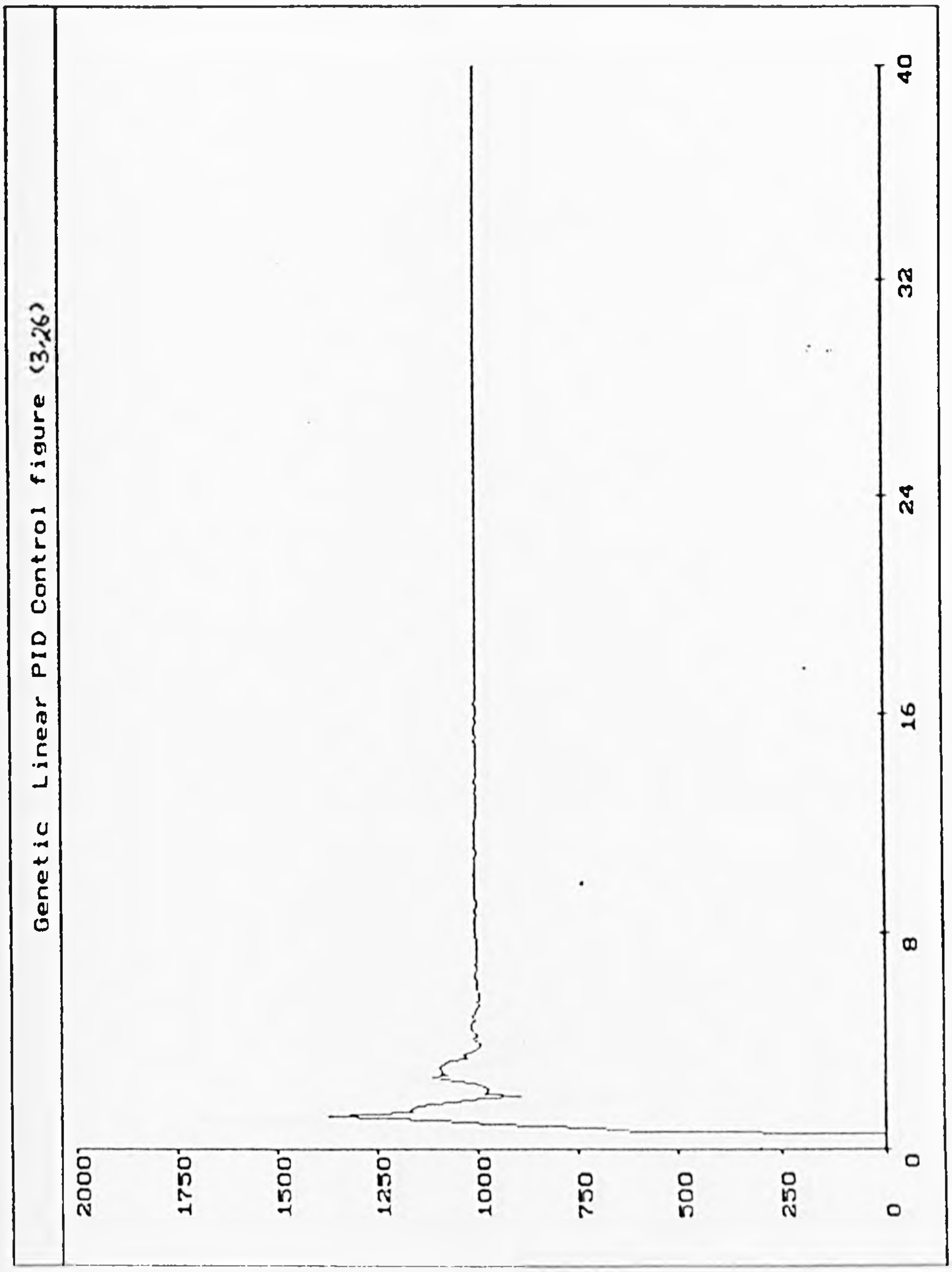


Figure (3.26) Transient response of robustness test using genetically designed linear PID controller for plant (2), table (3.11).

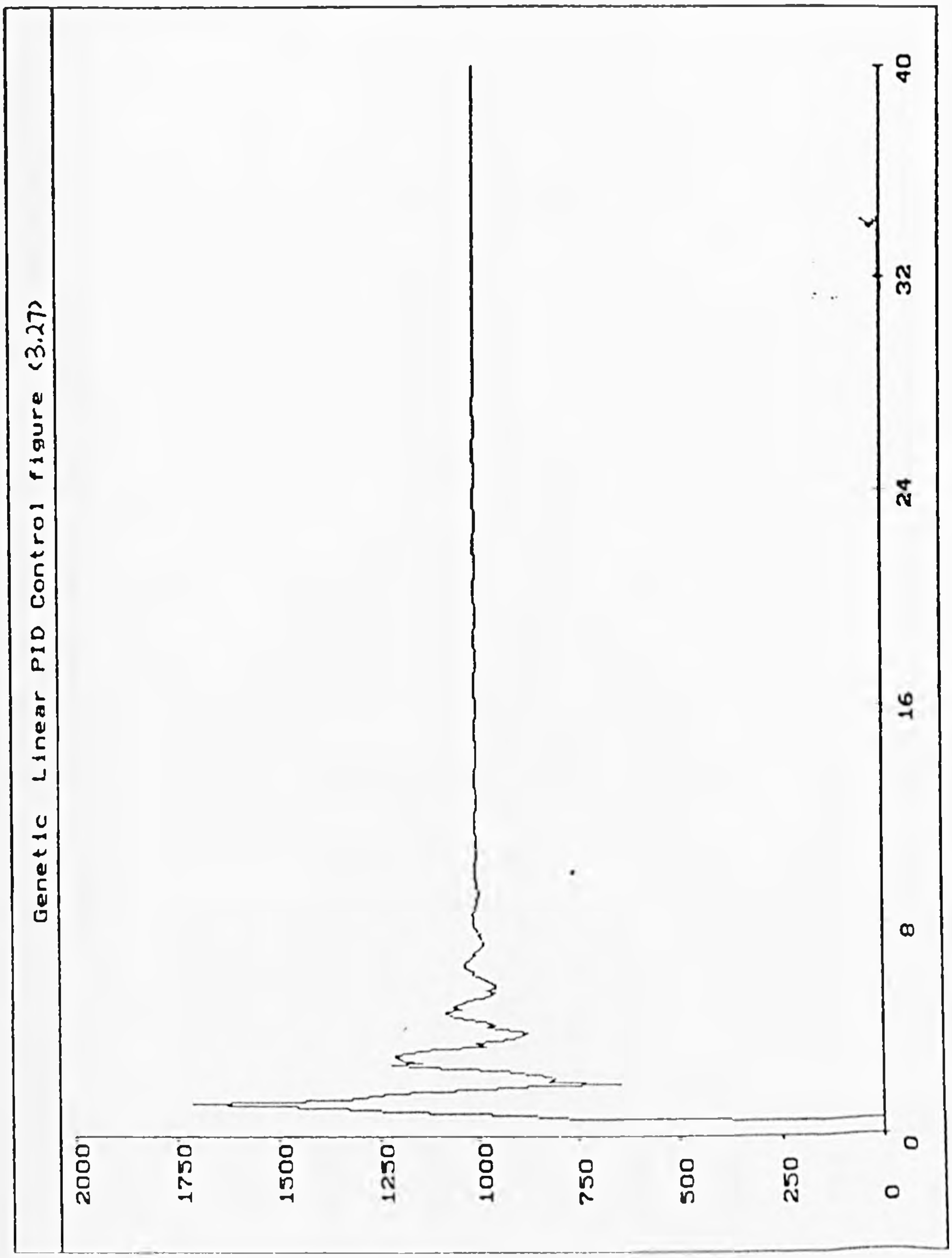


Figure (3.27) Transient response of robustness test using genetically designed linear PID controller for plant (3), table (3.11).

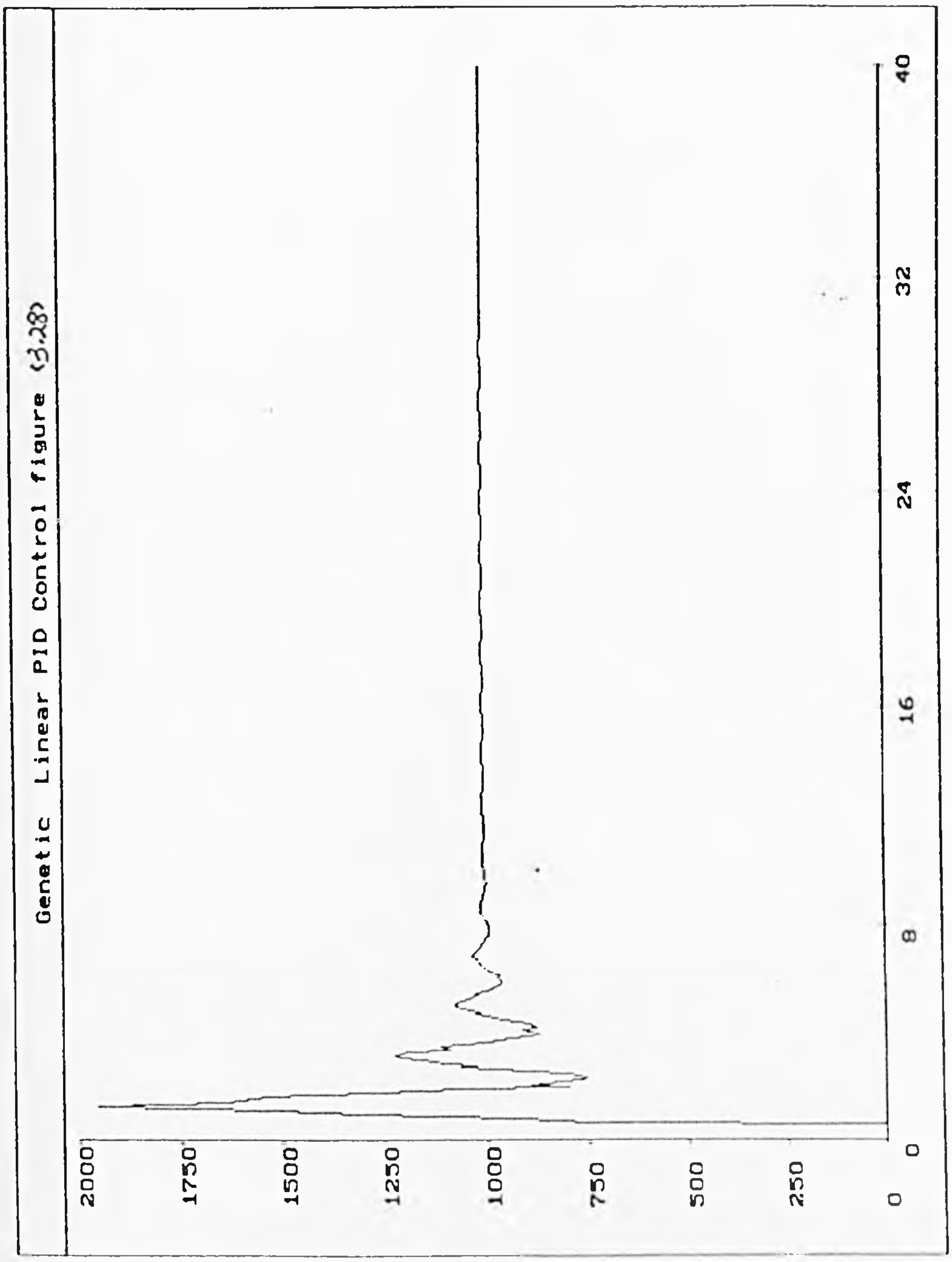


Figure (3.28) Transient response of robustness test using genetically designed linear PID controller for plant (4), table (3.11).

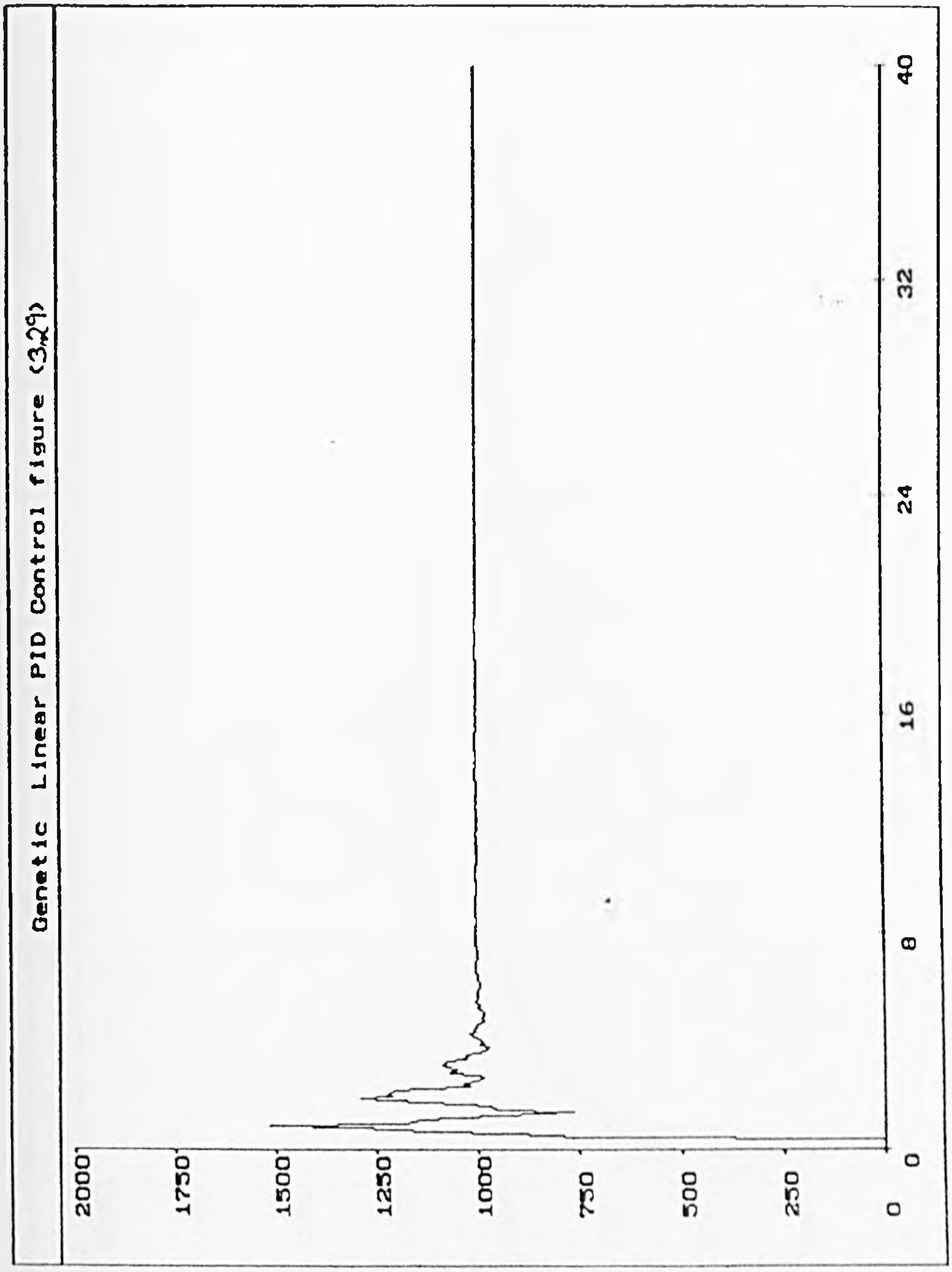


Figure (3.29) Transient response of robustness test using genetically designed linear dual-zoned PID controller for plant (5), table (3.11).

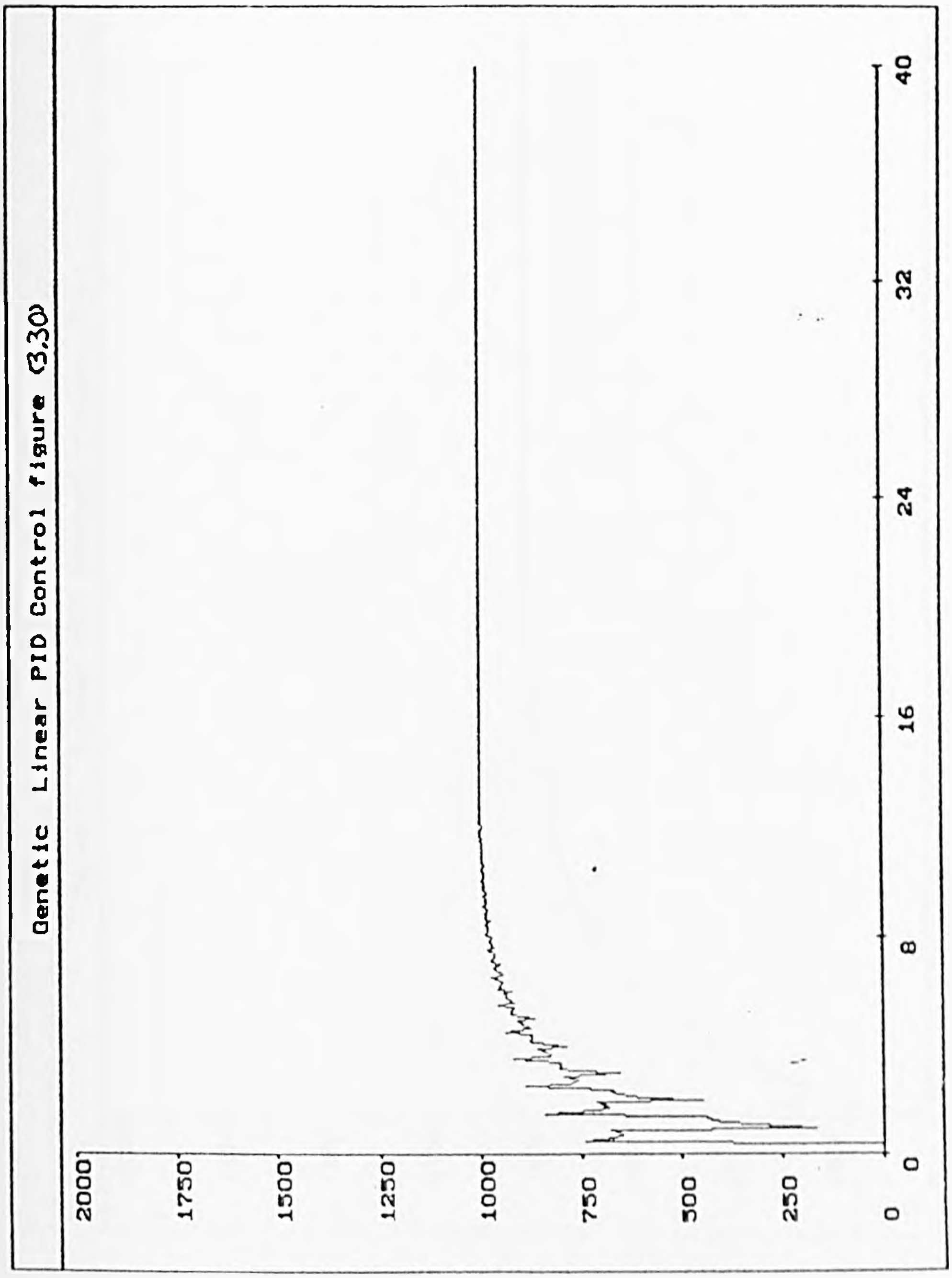


Figure (3.30) Transient response of robustness test using genetically designed linear PID controller for plant (6), table (3.11).

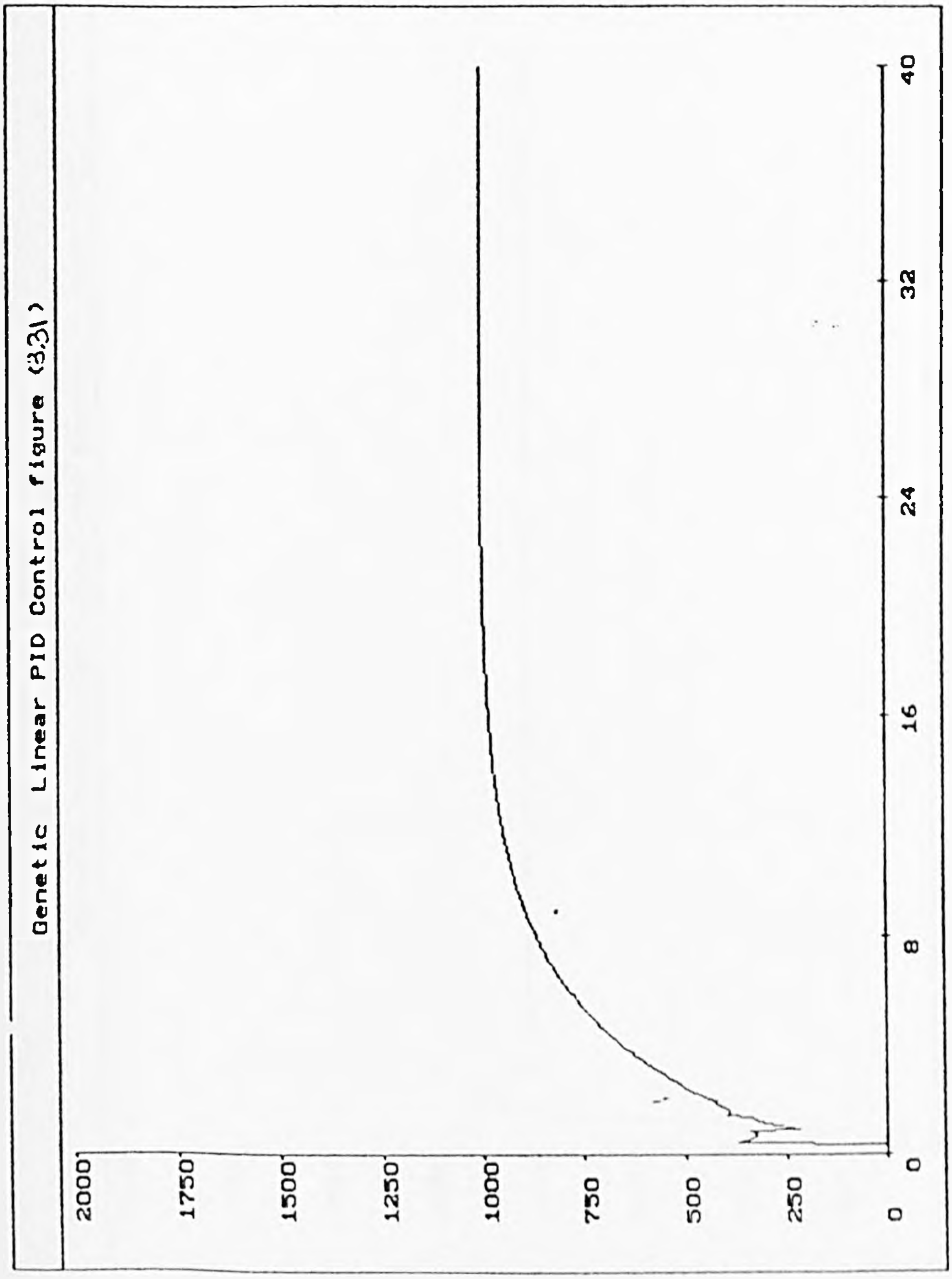


Figure (3.31) Transient response of robustness test using genetically designed linear PID controller for plant (7), table (3.11).

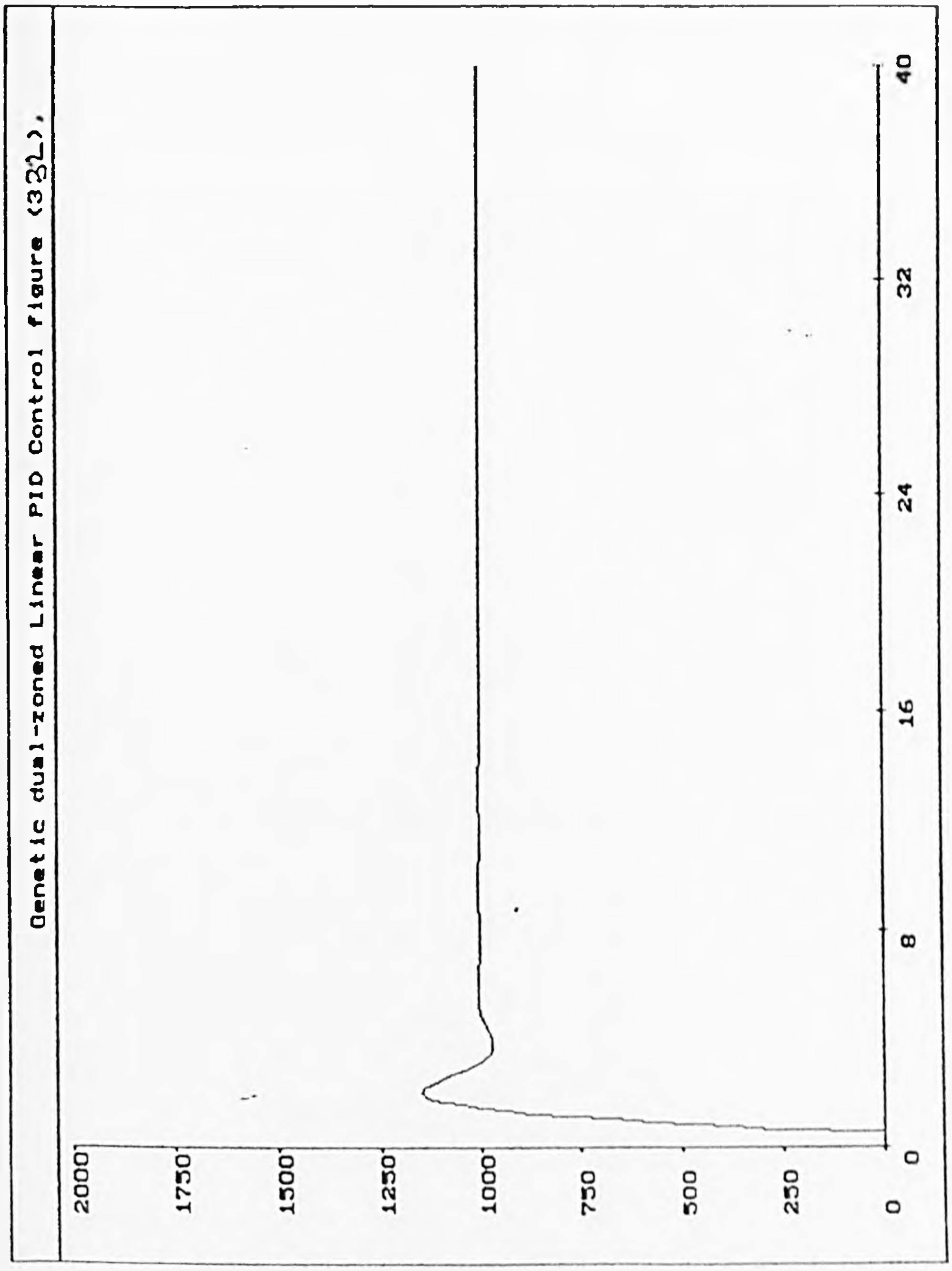


Figure (3.32) Transient response of robustness test using clipped genetically designed linear dual-zoned PID controller for plant (2), table (3.11).

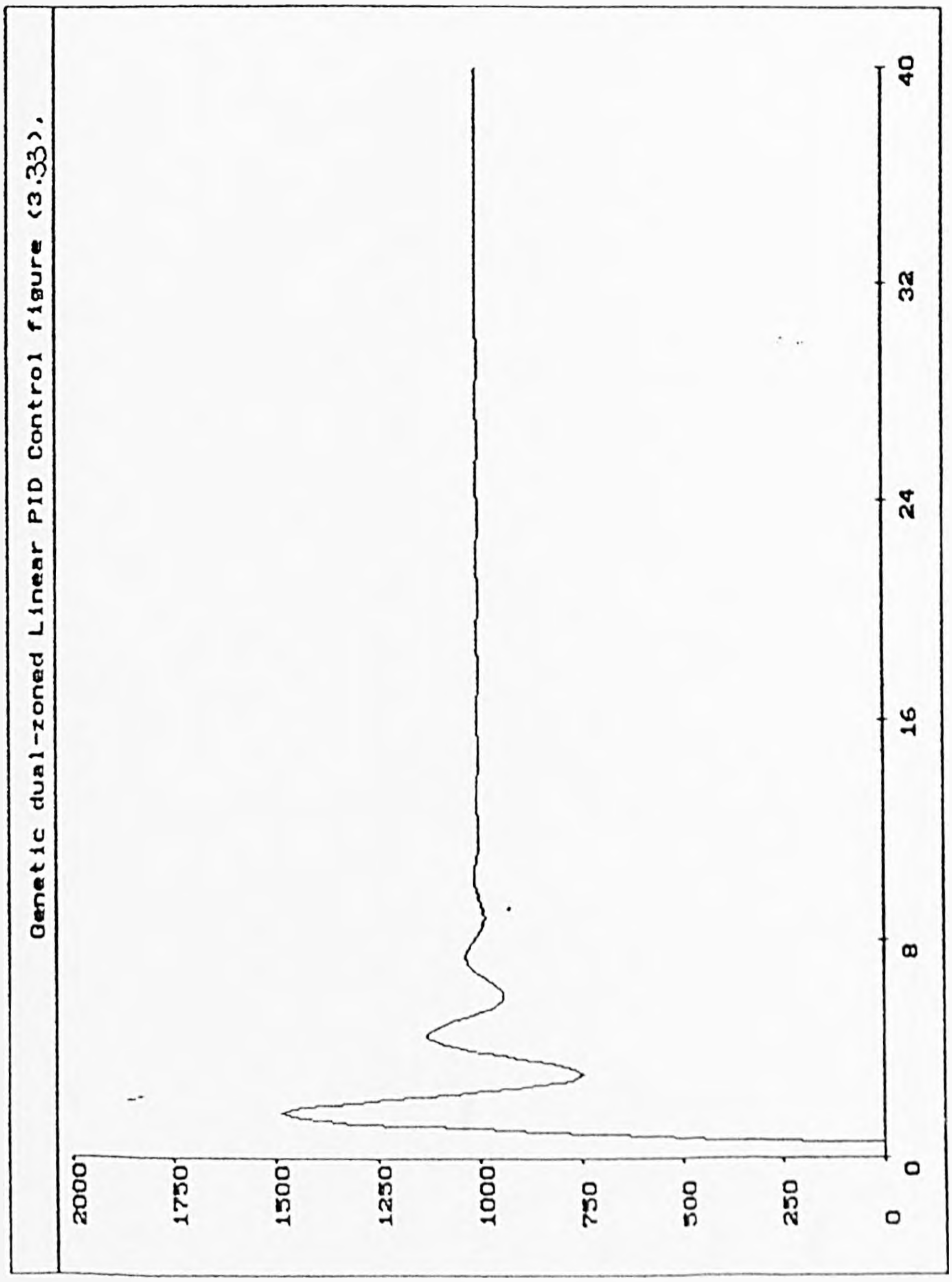


Figure (3.33) Transient response of robustness test using clipped genetically designed linear dual-zoned PID controller for plant (3), table (3.11).

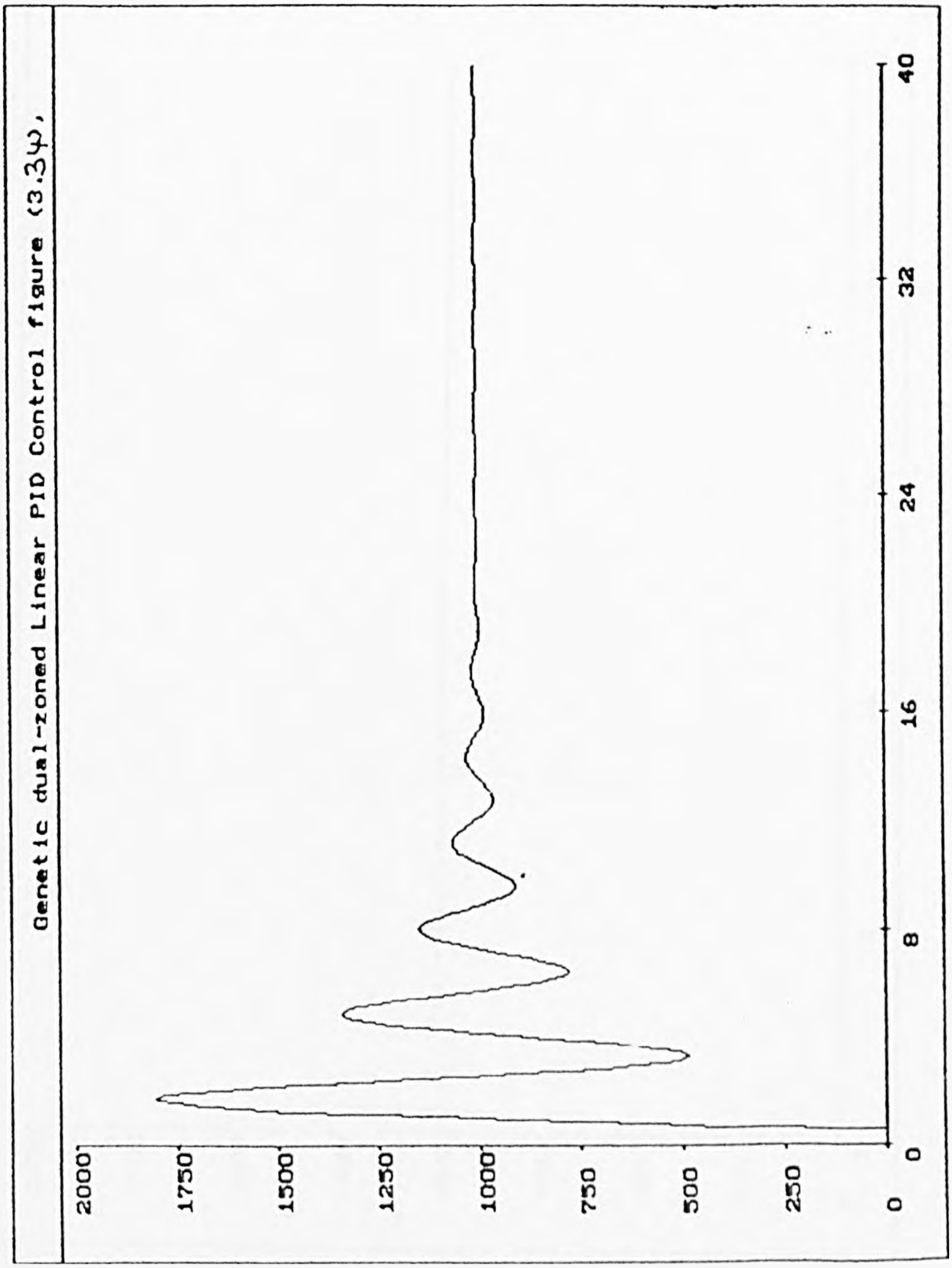


Figure (3.34) Transient response of robustness test using clipped genetically designed linear dual-zoned PID controller for plant (4), table (3.11).

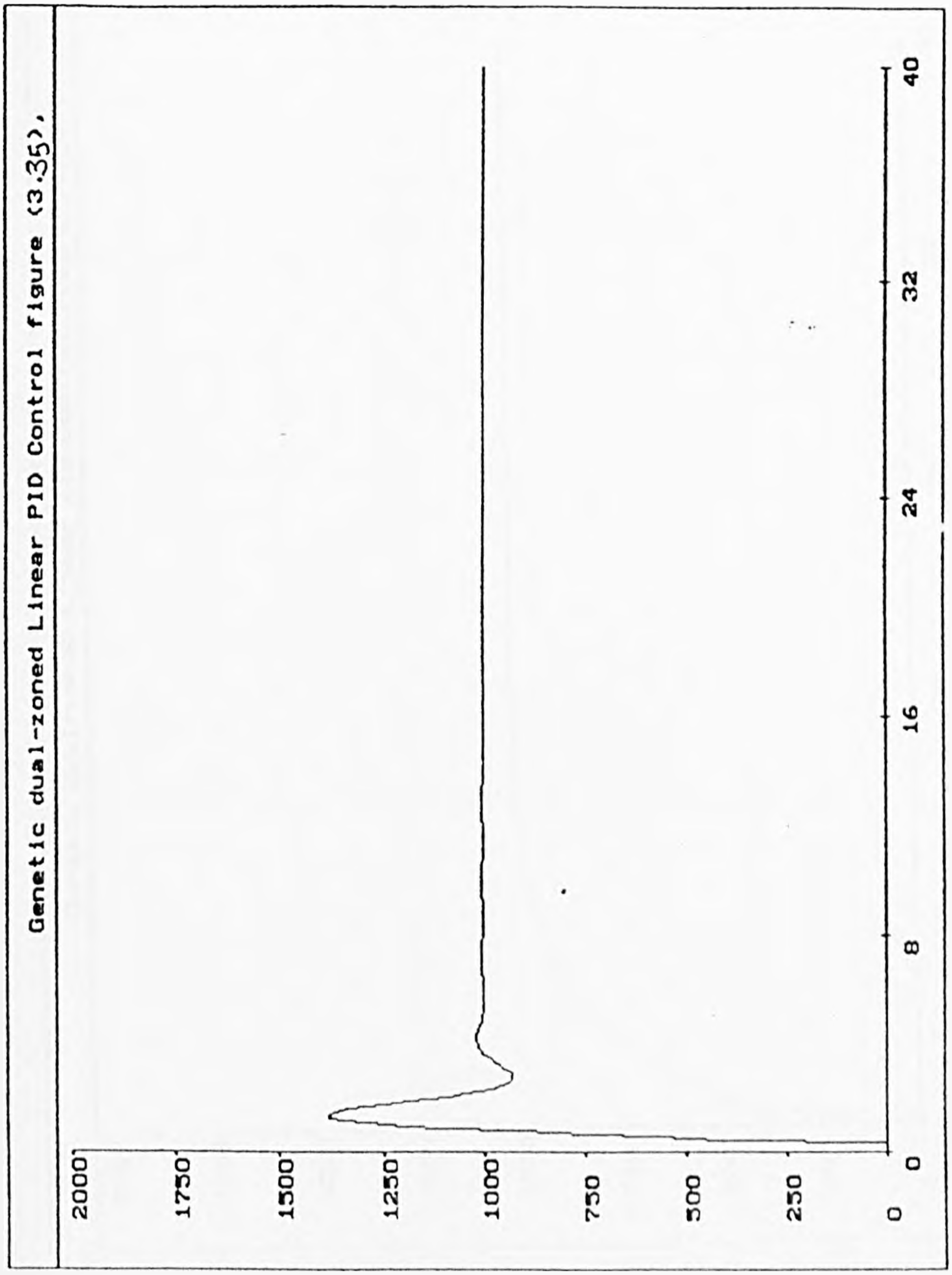


Figure (3.35) Transient response of robustness test using clipped genetically designed linear dual-zoned PID controller for plant (5), table (3.11).

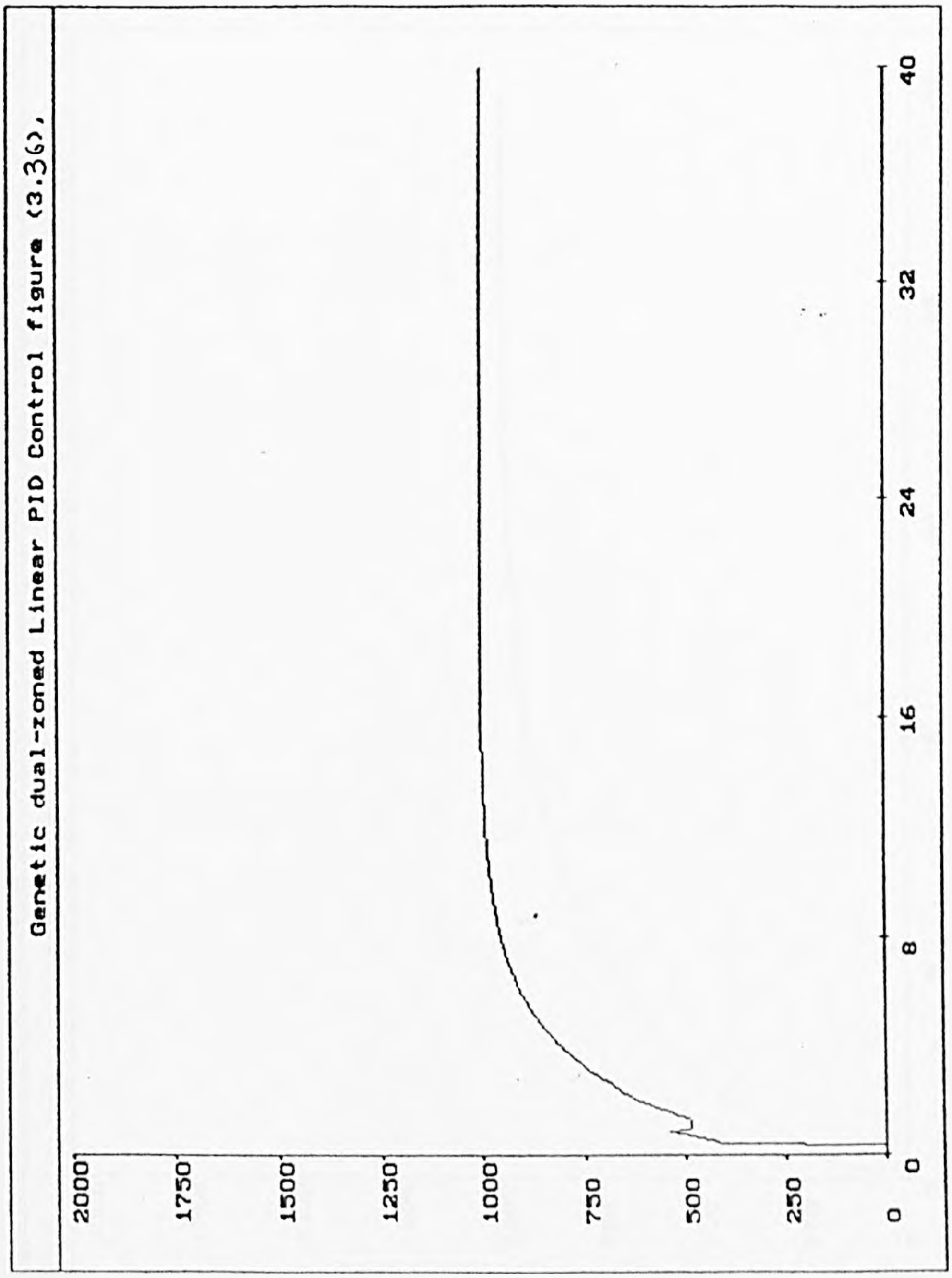


Figure (3.36) Transient response of robustness test using clipped genetically designed linear dual-zoned PID controller for plant (6), table (3.11).

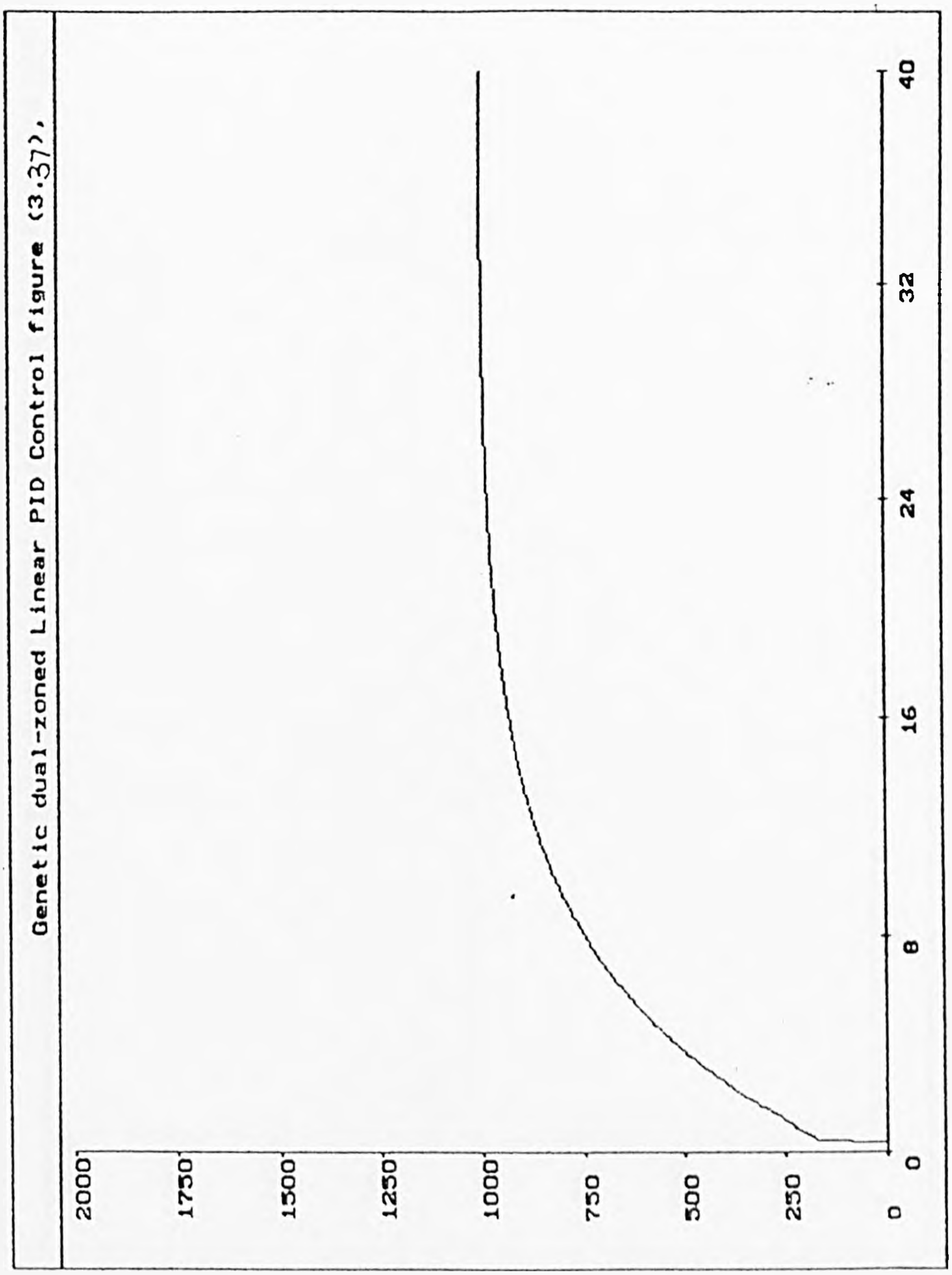


Figure (3.37) Transient response of robustness test using clipped genetically designed linear dual-zoned PID controller for plant (7), table (3.11).

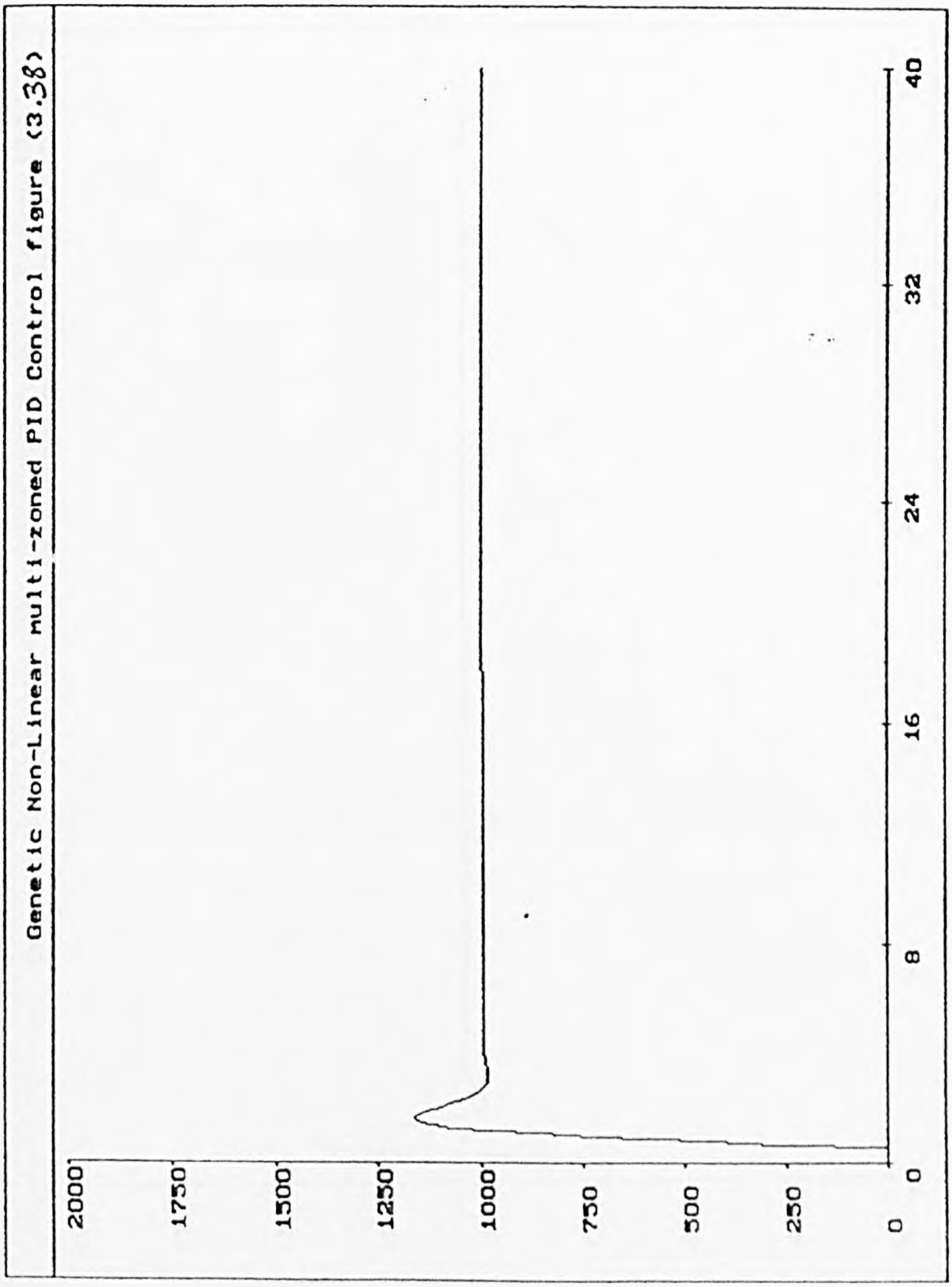


Figure (3.38) Transient response of robustness test using clipped genetically designed non-linear dual-zoned PID controller for plant (2), table (3.11).

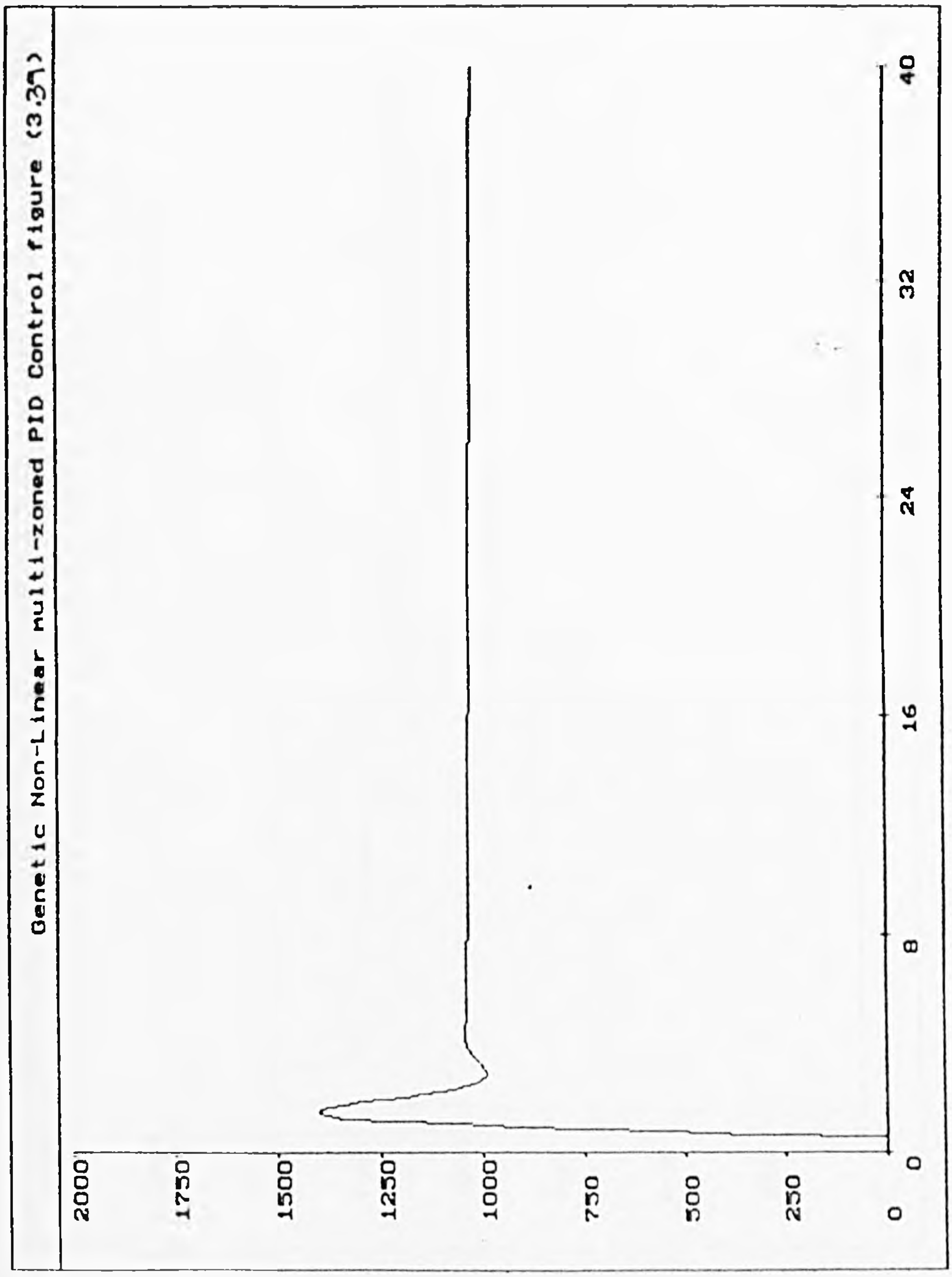


Figure (3.39) Transient response of robustness test using clipped genetically designed non-linear dual-zoned PID controller for plant (3), table (3.11).

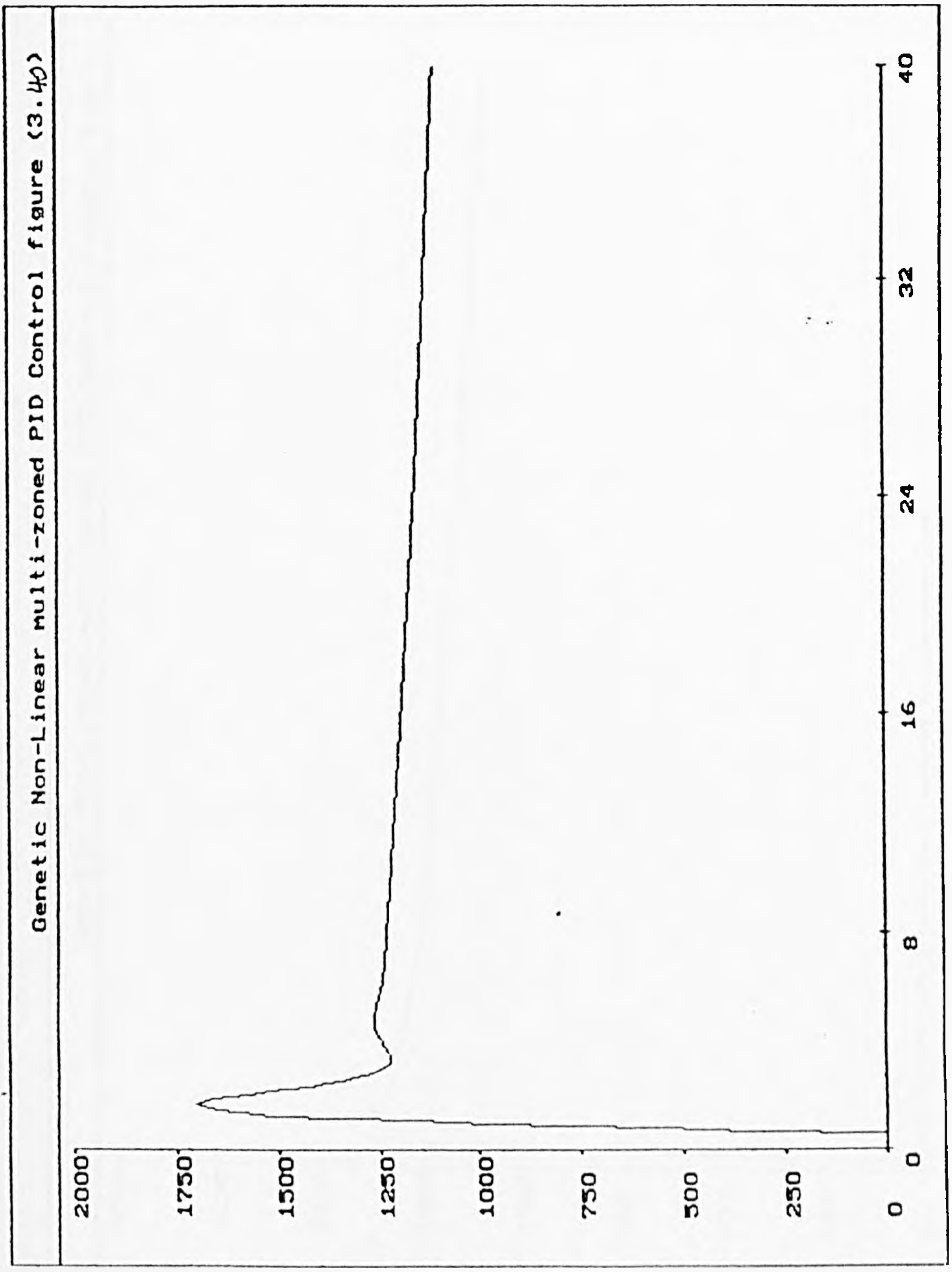


Figure (3.40) Transient response of robustness test using clipped genetically designed non-linear dual-zoned PID controller for plant (4), table (3.11).

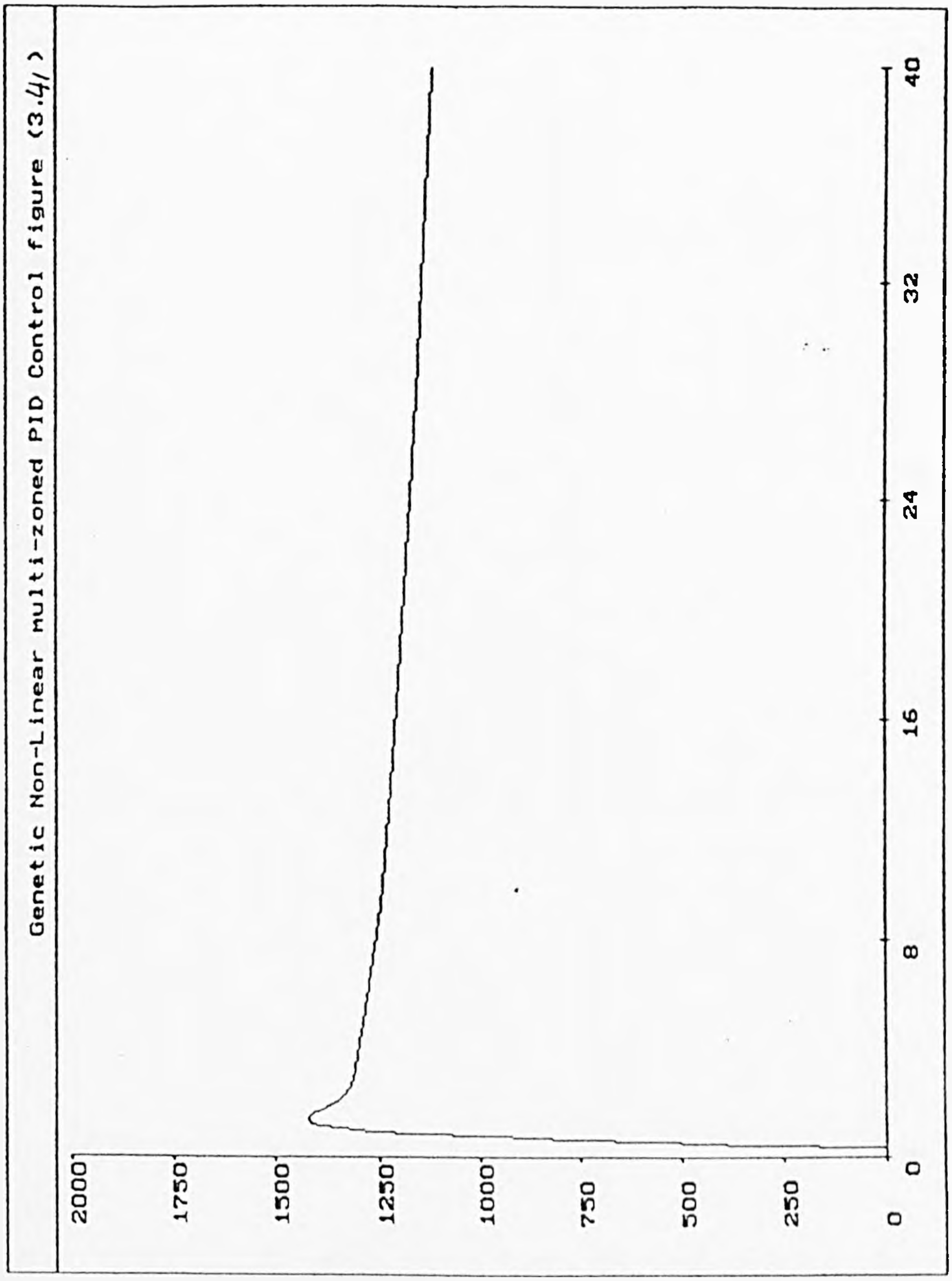


Figure (3.41) Transient response of robustness test using clipped genetically designed non-linear dual-zoned PID controller for plant (5), table (3.11).

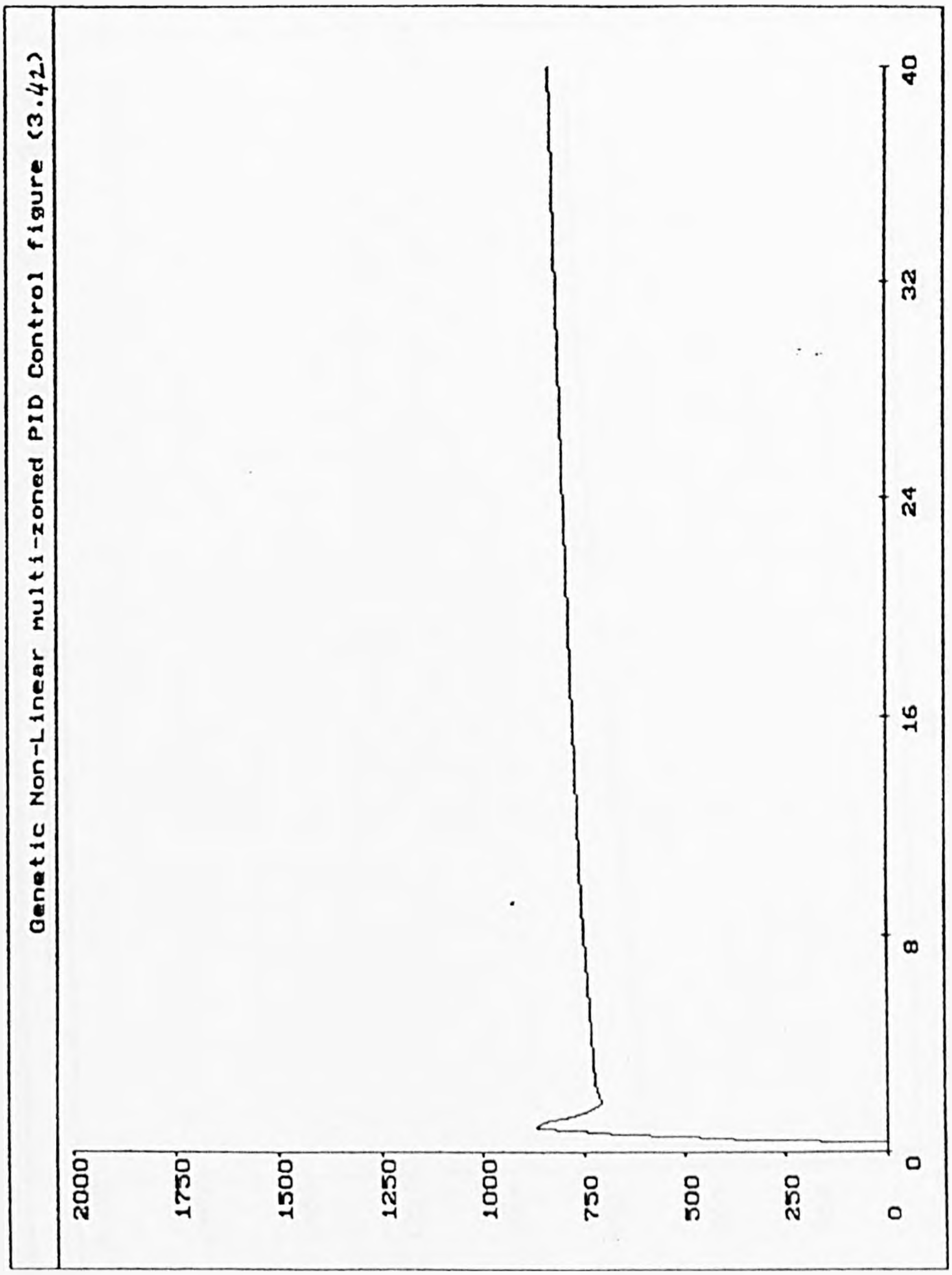


Figure (3.42) Transient response of robustness test using clipped genetically designed non-linear PID controller for plant (2), table (3.11).

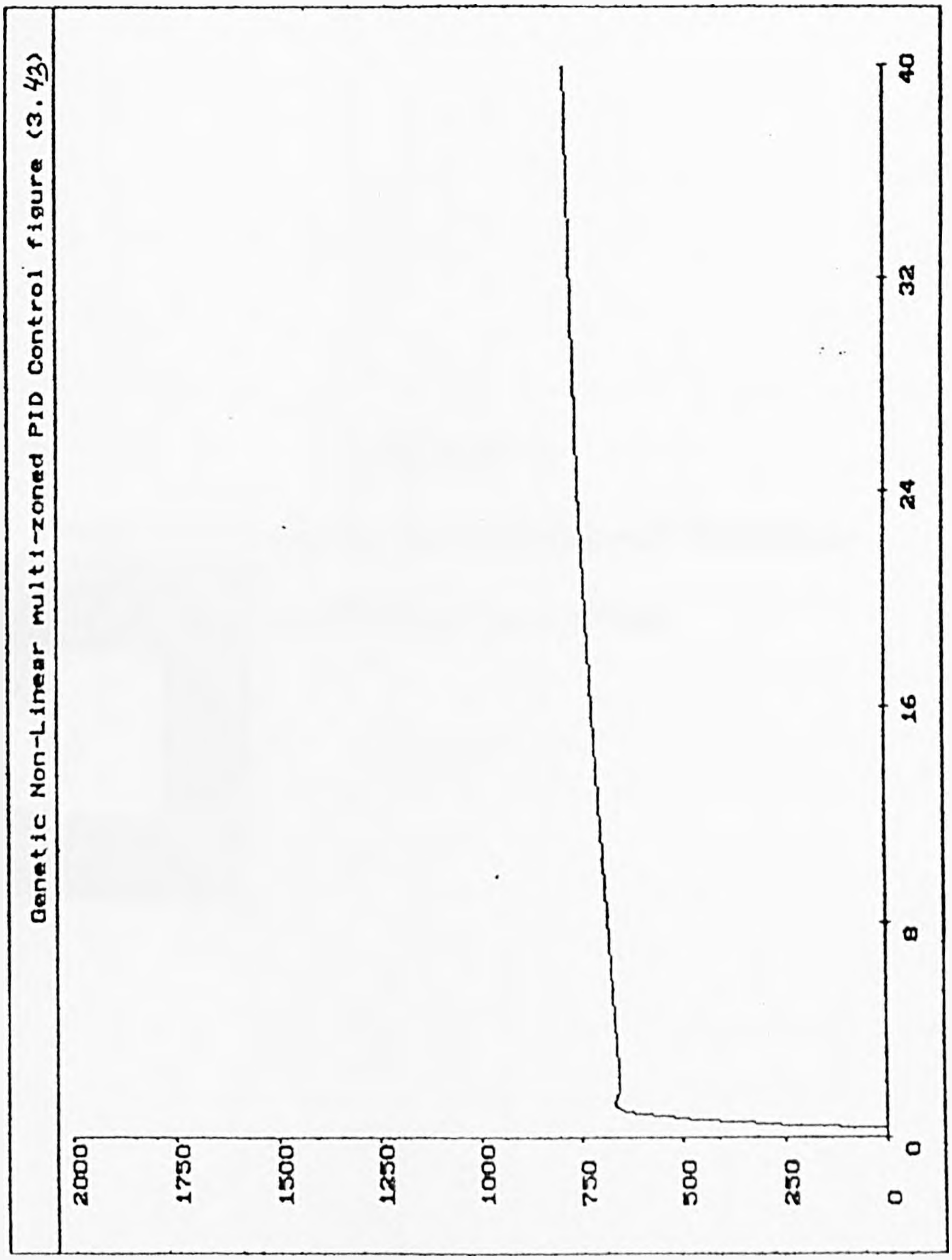


Figure (3.43) Transient response of robustness test using clipped genetically designed non- linear PID controller for plant (3), table (3.11).

CHAPTER 4

Design of Zoned Polynomial Interpolated Non-Linear Controllers for Linear Plants

Chapter 4
GENETIC DESIGN OF POLYNOMIAL NON-LINEAR PID CONTROLLERS FOR LINEAR PLANTS

4.1 INTRODUCTION

In the previous chapter linear interpolation routines were used to represent the non-linear PID controller gains of a PID controller. In this chapter it is proposed to replace the linear interpolation by a non-linear interpolation technique based on polynomial interpolation to represent the PID controller gains of a non-linear PID controllers.

The linear interpolation routine used in the previous chapter possess sharp edges at the point of change between sub-zones. This non-continuous derivative could cause some degradation in the performance of the controller. Hence, in this chapter a non-linear interpolation technique is to be used to provide a continuous derivative through all zones by removing the sharp edges. It is also hoped that the smoothness of the non-linear interpolation technique will further improve the performance of the non-linear controller.

Thus the task is to find an interpolation function that can provide the smoothness, and the computational efficiency required. Moreover, the technique should also be sufficiently general so as to be able to approximate large classes of functions which might arise in practice. Polynomials are the bases for most interpolation functions are the most commonly used functions, and they would provide most if not all the requirements for this task. They are also well documented, and all the different forms have been already used and tested for their ability to give a good and accurate results.

4.2 SYNTHESIS

In order to design an efficient non-linear controller using the Polynomial Interpolated Non-linear (PIN) method, it is proposed to use the dual zoned method introduced in chapter 3. The SISO plants under consideration, are governed on the continuous time set T = $(0,\infty]$ by state and output equation of respective forms

$$\dot{x}(t) = Ax(t) + bu(t) \tag{4.1}$$

and

$$y(t) = cx(t) \tag{4.2}$$

where

$x(t) \in \Re^n$ is the state vector,

$y(t) \in \Re$ is the scalar output from the plant,

$u(t) \in \Re$ is the scalar input to the plant,

$A \in \Re^{nxn}$ is the plant matrix,

$b \in \Re^{nx1}$ is the input matrix,

$c \in \Re^{1xn}$ is the output matrix.

It is assumed that the plant is functionally controllable, so that none of the transmission zero of the plant lies at the origin in the complex plane and therefore that any and all solutions of s in

$$\begin{vmatrix} sI_n-A, & -b \\ c, & 0 \end{vmatrix} = 0 \qquad 4.3$$

are non-zero [Rosenbrock (1974)]. This assumption ensures that rank M = n + 1 [porter and power (1970)] where the system matrix is given by

$$M = \begin{bmatrix} A, & b \\ c, & 0 \end{bmatrix} \qquad 4.4$$

In order to design non-linear PID controllers for SISO linear plants governed by equations (4.1) and (4.2), it is convenient to consider the behaviour of such plants on the discrete-time set T_T = {0, T, 2T,....}.

This behaviour is governed by state and output equations of the respective forms [Kwakernaak and Sivan (1972)]

$$x_{k+1}=\Phi x_k+\Psi u_k \qquad 4.5$$

and

$$y_k = \Gamma x_k \qquad 4.6$$

where

$$\Phi = e^{AT} \qquad 4.7$$

$$\Psi = \int_0^T e^{A\tau} b \, d\tau \qquad 4.8$$

and

$$\Gamma = c \qquad 4.9$$

In these equations, $x_k T \in \Re^n$, $u_k T \in \Re$, $y_k T \in \Re$, $\Phi \in \Re^{nxn}$, $\Psi \in \Re^{nx1}$, $\Gamma \in \Re^{1x}$, and $T \in \Re^+$ is the sampling period.

It is evident from chapter 3 that the incremental controller can be described as

$$\Delta u_k = T(k_p \Delta e_k + T k_i e_k + k_d \Delta^2 e_k) \qquad 4.10$$

where

Δu_k is the incremental change in input,

Δe_k is the first order backward difference in error,

$\Delta^2 e_k$ is the second order backward difference in error.

K_p is the current effective value of the proportional gain,

K_i is the current effective value of the integral gain,

K_d is the current effective value of the derivative gain,

T is sampling time.

It is evident from chapter 3 that the error can be written as

$$e_k = v_k - y_k \qquad 4.11$$

Also from chapter 3 that the first order backward difference in error can be written as

$$\Delta e_k = e_k - e_{k-1} \qquad 4.12$$

Further more from chapter 3 the second order backward difference in error can be expressed as

$$\Delta^2 e_k = e_k - 2e_{k-1} + e_{k-2} \qquad 4.13$$

4.2.1 NON-LINEAR INCREMENTAL PID CONTROLLER

The incremental PID controller given by equation (4.10), can take one of two forms, one is linear, and the other is non-linear. In this chapter the non-linear form is to be investigated. The non-linearities are a function of the plant error.

It follows from equation (4.10), that the incremental non-linear PID controller can be described by an equation of the form

$$\Delta u_k = \beta_p(\Delta e_k)\Delta e_k + T\beta_i(e_k)e_k + \beta_d(\Delta^2 e_k)\Delta^2 e_k \qquad 4.14$$

where

$\beta_p(\Delta e_k)$ is a function representing the proportional gain,

$\beta_i(e_k)$ is a function representing the integral gain,

$\beta_d(\Delta^2 e_k)$ is a function representing the derivative gain.

It is evident from equation (4.14), that the gain functions can be represented by equation of the form

$$\mathcal{K}_p = \beta_p(\Delta e_k) \qquad 4.15$$

$$\mathcal{K}_i = \beta_i(e_k) \qquad 4.16$$

$$\mathcal{K}_d = \beta_d(\Delta^2 e_k) \qquad 4.16$$

The above gain functions are to be based on the polynomial interpolation function, as will be illustrated later in this chapter.

It follows from equation (4.14) that the incremental non-linear PID controller can be conveniently described by equation of the form

$$\Delta u_k = \mathcal{K}_p \Delta e_k + T\mathcal{K}_i e_k + \mathcal{K}_d \Delta^2 e_k \qquad 4.17$$

It is important to note that the PID controller has been implemented in the incremental form to make use of the dual zone tuning technique, and to avoid any bumpless transfer techniques associated with the integral state.

Figure 4.1 shows a black diagram representing the control system using polynomial interpolation routine to represent the gain functions.

2.2.2 INTERPOLATION ROUTINES

There are a number of interpolation routines available that can be implemented which could give a solution to the non-linear control system design problem, such routines are:

i) polynomial (Lagrange);

ii) cubic spline;

iii) polynomial coefficients.

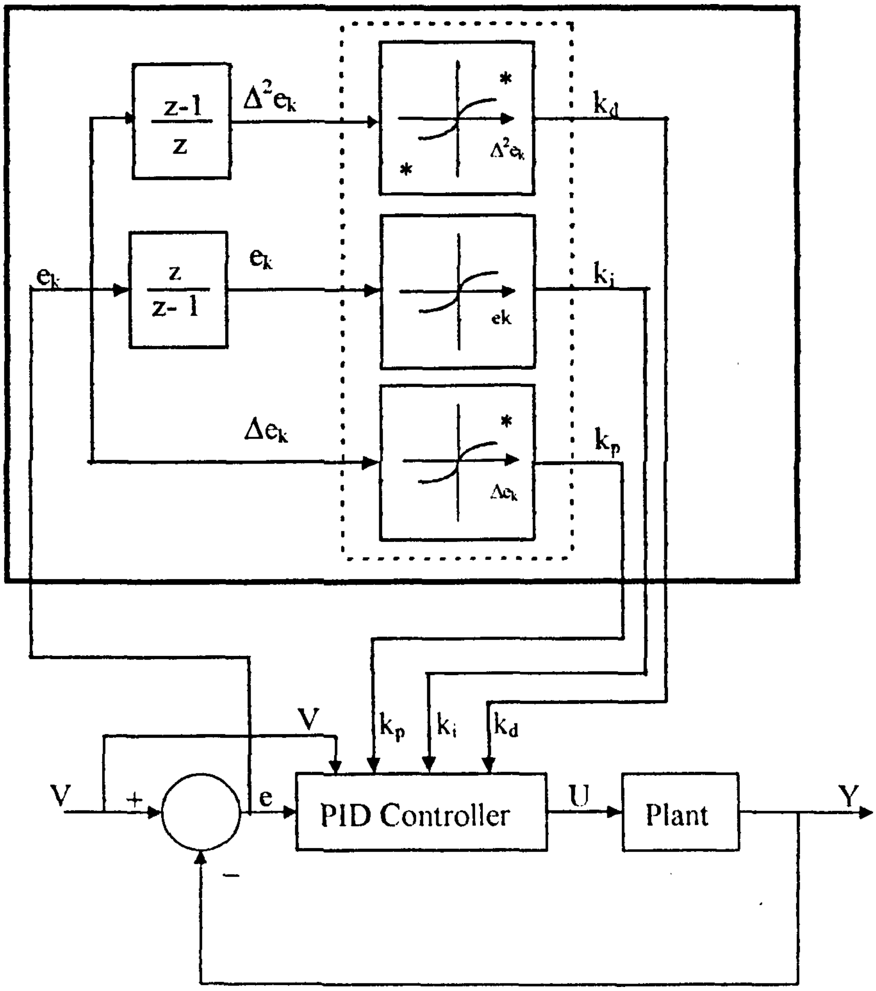


Figure 4.1 System Block Diagram

All three routines are polynomial interpolation routines. A choice has to be made, on which of the above would best suit the design requirement. Such a choice can be made on the bases that the gains in the problem have to be coded as a two dimensional non-linear function. The function also needs to be easy to implement, and fast to compute. From consideration of the above and on the basis of the constraints imposed it was decided to use the Lagranges polynomial interpolation routine. The principle reasons behind this design are:

i) initialising the coefficients of the Lagrangian routine is similar to choosing gains,and reasonabley easy to achieve;

ii) initialising the coefficients of the polynomial interpolation routine can be difficult to achieve;

iii) the cubic spline interpolation routine is a further extension of the Lagrange interpolation routine, and hence, it requires longer computational time to achieve what a basic Lagrange routine can achieve;

iv) also the cubic splines requires careful matching of the first and second derivatives at the start and end which can be difficult to achieve.

4.2.2.1 POLYNOMIAL INTERPOLATION

There is a unique line through any two points. Through any three points, a unique quadratic. The interpolating polynomial of degree N-1 through the N points y_1 =

$f(x_1)$,....,y_N = $f(x_N)$ is given explicitly by Lagrange's Formula,

$$P(x) = \sum_{r=1}^{N} f_r \prod_{k=1}^{N} \frac{(x - x_k)}{(x_r - x_k)} \qquad 4.18$$

where $\prod$ is the product of terms, r=(1,2,.....,N), k=(1,2,.....N), and k $\neq$ r.

The best way to explain how the above formula can be used to map a required function is by considering an example for a two point function.

For two points the above equation can be represented as

$$P(x) = \frac{(x - x_2)}{(x_1 - x_2)} f_1 + \frac{(x - x_1)}{(x_2 - x_1)} f_2 \qquad 4.19$$

By substituting the values into the above function a value for P(x) can be found.

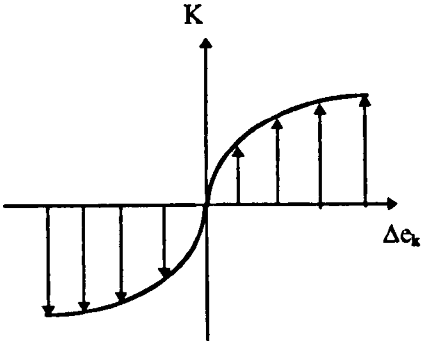


Figure (4.2) General Gain Function Mapping Using Polynomial Interpolation to Represents Proportional Gain

4.2.3 NON-LINEAR GAIN FUNCTION MAPPING

By utilizing equation 4.18 a non-linear gain function representing the gains needed for the non-linear PID controller can be obtain as

$$K(e_k)=\sum_{i=1}^{N} f_i \prod_{j=0,j\neq i}^{N} \frac{(e_k-E_k^j)}{(E_k^i-E_k^j)} \qquad 4.20$$

where

e_k is the tracking error;

E^N_k is the over all zone length.

As it can be seen from the above equation, the polynomial interpolation equation can be used as a non-linear gain function. This gain function can then be used in the PID controller to produce a non-linear incremental PID controller. Figure (4.2) shows how the above gain function can be mapped for a gain using the equation (4.20) for a dual-zone , or a multiple-zone.

Moreover from equation (4.20), it is evident that a gain function can be produced for each of the non-linear gains needed for the non-linear controller. The three gain functions are thus given by

$$K_p=\sum_{i=1}^{N} f_i \prod_{j=0,j\neq i}^{N} \frac{(\Delta e_k-E_{k_p}^{\ j})}{(E_{k_p}^{\ i}-E_{k_p}^{\ j})} \qquad 4.21$$

where

Δe_k is the first order backward difference in error;

$E^N{}_{k_p}$ is the over all proportional gain zone length.

This equation represents the proportional gain function for an N number of parameters, using the Lagranges smooth interpolation routine.

Thus the integral non-linear gain function is given by

$$K_i = \sum_{i=1}^{N} f_{i_i} \prod_{j=0, j\neq i}^{N} \frac{(e_k - E_{k_i}{}^j)}{(E_{k_i}{}^i - E_{k_i}{}^j)} \qquad 4.22$$

where

e_k is the tracking error;

$E^N{}_{k_i}$ is the over all proportional gain zone length.

This is an integral gain function for an N number of parameters, using the Lagranges smooth interpolation routine.

It is therefore evident that the derivative gain function can be represented as

$$K_d = \sum_{i=1}^{N} f_i \prod_{j=0, j \neq i}^{N} \frac{(\Delta^2 e_k - E_{k_d}^{\,j})}{(E_{k_d}^{\,i} - E_{k_d}^{\,j})} \qquad 4.23$$

where

$\Delta^2 e_k$ is the second order backward difference in error;

$E^{N}_{k_d}$ is the over all proportional gain zone length.

This is a derivative gain function for N parameter, using the Lagranges smooth interpolation routine.

4.3 GENETIC DESIGN OF ZONED NON-LINEAR PID CONTROLLERS

In order to use genetic algorithm to design dual zoned non-linear PID controllers, the parameters associated with the dual-zoned controller have to be precisely defined. It is evident from the previous chapter that the behaviour of dual controller can be split into two distinct zones, the set-point zone, and the tracking zone.

The parameters for the controllers in the two zones are designed together. The performance of the dual-zoned controller will be contrasted by considering a number of plants controlled by both the genetically designed dual-zoned non-linear controller, and the genetically designed linear PID controller.

In representing the gain functions using this method, any future changes in the design can easily be achieved. For example the gain function K_d can be interpolated using Δe_k instead of $\Delta^2 e_k$, and this is easily achieved by changing interpolating variable.

Figure (4.4) shows the function mapping for the three gains as they would be presented by the GAs.

4.3.1 PARAMETERS FOR THE SET POINT ZONE

The parameters in this zone are exactly the same as was introduced in section 3.2.3.1 of the previous chapter i.e. the GA will search the space for two feed forward gains for the set-point zone.

4.3.2 PARAMETERS FOR THE TRACKING ZONE

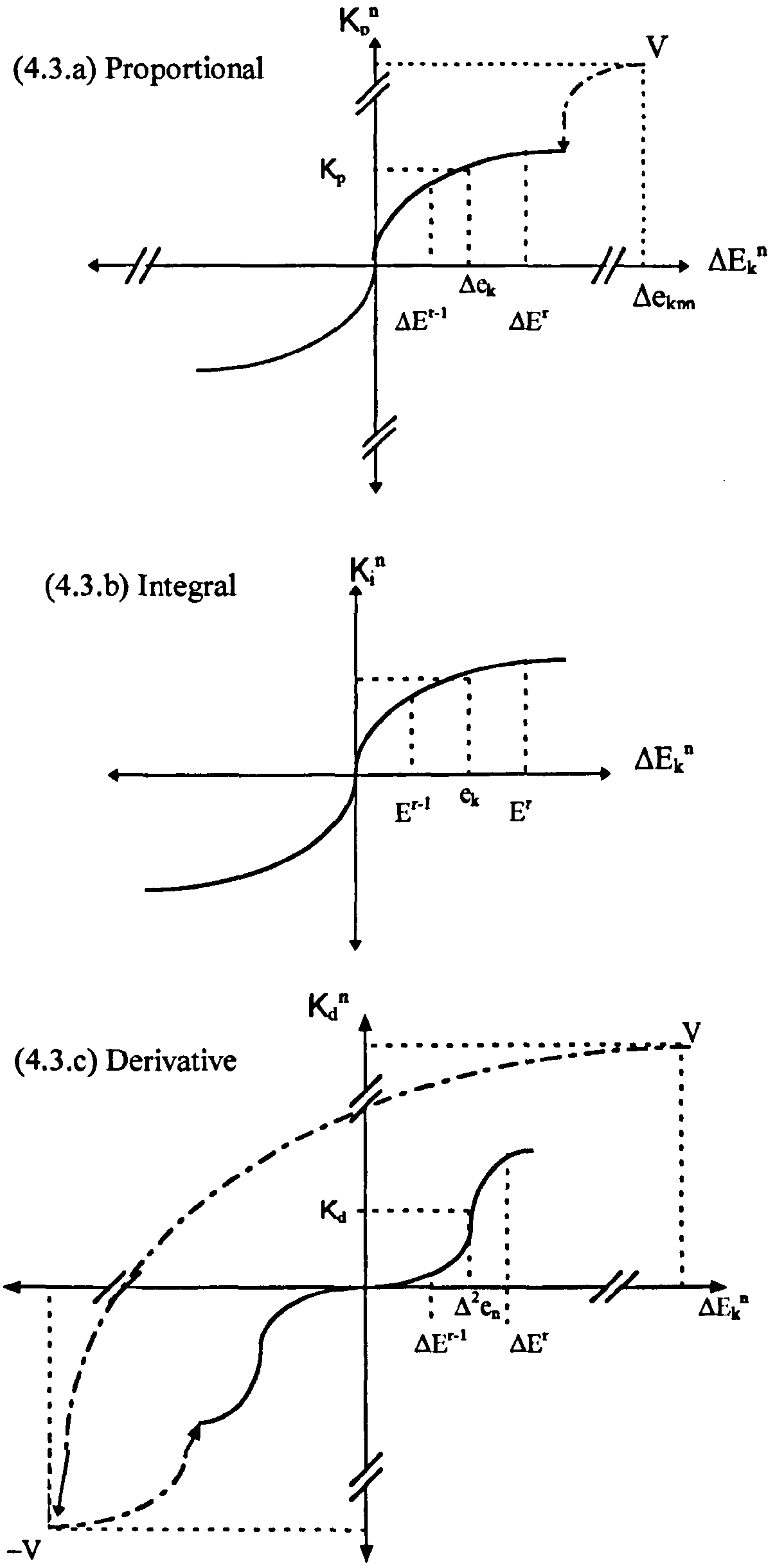
In the tracking zone it was found that the proportional and derivative gains that the controller operate in a very small region close to zero, and from the equations of the input parameters for the tracking zone it can be seen that the controller must have individual gain values for the proportional, integral, and derivative gains. The gains used in this zone are mapped as shown in figure (4.3).

In order to use GAs to select the tuning parameters in such a way that as to produce satisfactory response in the case of a step input, it is only necessary to encode the elements of the tracking zone gain functions, plus the feed forward gains needed for the set point change zone, as binary strings. The binary string would be represented as

$$\{ \; \underset{\text{proportional}}{(f^1_p \ldots\ldots f^N_p)} \;\; \underset{\text{integral}}{(f^1_i \ldots\ldots f^N_i)} \;\; \underset{\text{derivative}}{(f^1_d \;\; \ldots\ldots f^N_d)} \;\; \underset{\text{set point change}}{(F_0 \;\; F_1)} \; \}$$

where the whole string contains the three gain functions elements, plus the feed forward gains. The mapping of the string is shown in figure (4.5).

Random initialization is the approach used to initialize the initial population, Since this approach requires the least knowledge-acquisition effort and provides a lot of diversity for the GAs to work with. The process of interviewing introduced in chapter 2 is used to insure that the randomly generated variables do not initially violate any constraints on the function to be tuned. The system incorporates both the linear plant, and the PIN controllers, the controllers are designed by randomly generated sets of: tracking zone gains elements; and feed forward gains by the GA's. A stability test is then carried out on all the controllers. In the case of a violation of stability in any of the cases, the randomly generated set of tracking zone gains elements, and feed forward gains will not be included in the initial population. This type of initialization is essential in this type of design, since otherwise their would be a danger of creating an initial population in which many of its members violate the constraints on the controllers being designed, figure (4.4) shows the sequence of genetic algorithms. Following the initialization, the


(4.3.a) Proportional

(4.3.b) Integral

(4.3.c) Derivative

Figure 4.3 G ain Functions Mapping for PIN PID Controller

objective function is introduced and its value is calculated using decoded values of the parameters in each string. Thus for example, if minimum ISE is regarded as the ultimate design requirement, genetic algorithms can be readily used to select the best set of tracking gains elements, and the feed forward gains such that the ISE is minimised. In the genetic design of non-linear PID controllers the plant under consideration is subjected to a command input (i.e. unit step), then the performance index is computed for the plant. therefore, for each member in the population the function

$$ISE = \sum_{j=1}^{j=N} e^{2}{}_{j} \qquad 4.24$$

is evaluated, where

$$e_j = v - y \qquad N = \frac{T}{\tau}$$

and $e_j \in \Re$, is the error signal, $y_j \in \Re$,is the output signal, T is sampling time, and τ is an appropriately chosen settling time.

Minimization of this performance index over the entire population can be rapidly obtained by using the genetic operations of reproduction, crossover, and mutation. It is interesting to note that the conditions for the existence of a non-linear PID controllers for the plant under consideration will be automatically satisfied by the genetic algorithms. This is obvious, because in the case of violation of the constrained the corresponding set of tracking zones gains elements and the feed forward gains produces large ISE and the result will be zero or low fitness, and by the action of the selection operator it would not be chosen for the next generation.

4.3 ILLUSTRATIVE EXAMPLES

To design the dual zone non-linear PID controller the analysis done earlier in this chapter to produce gain functions using the method as in equation (4.20), will now be extended to produce a 5 points gain function for the gains required to produce a non-linear incremental PID controller. The five points gain function produced will map the whole tracking zone for each of the gains (i.e. proportional, integral, and derivative). The function given in equation (4.21), which represents a proportional gain function for an N parameters can hence, be used to produce a 5 points function as required for the proportional gain, then equation (4.21) can be written as

$$\begin{aligned}
\mathcal{K}_p = &+ \frac{(\Delta e_k - H)(\Delta e_k - 2H)(\Delta e_k - 3H)(\Delta e_k - 4H)K_{p0}}{(24H^4)} \\
&- \frac{(\Delta e_k)(\Delta e_k - 2H)(\Delta e_k - 3H)(\Delta e_k - 4H)K_{p1}}{(6H^4)} \\
&+ \frac{(\Delta e_k)(\Delta e_k - H)(\Delta e_k - 3H)(\Delta e_k - 4H)K_{p2}}{(4H^4)} \\
&- \frac{(\Delta e_k)(\Delta e_k - H)(\Delta e_k - 2H)(\Delta e_k - 4H)K_{p3}}{(6H^4)} \\
&+ \frac{(\Delta e_k)(\Delta e_k - H)(\Delta e_k - 2H)(\Delta e_k - 3H)K_{p4}}{(24H^4)}
\end{aligned} \tag*{4.24}$$

This equation, represents proportional gain function for all five points in one function.

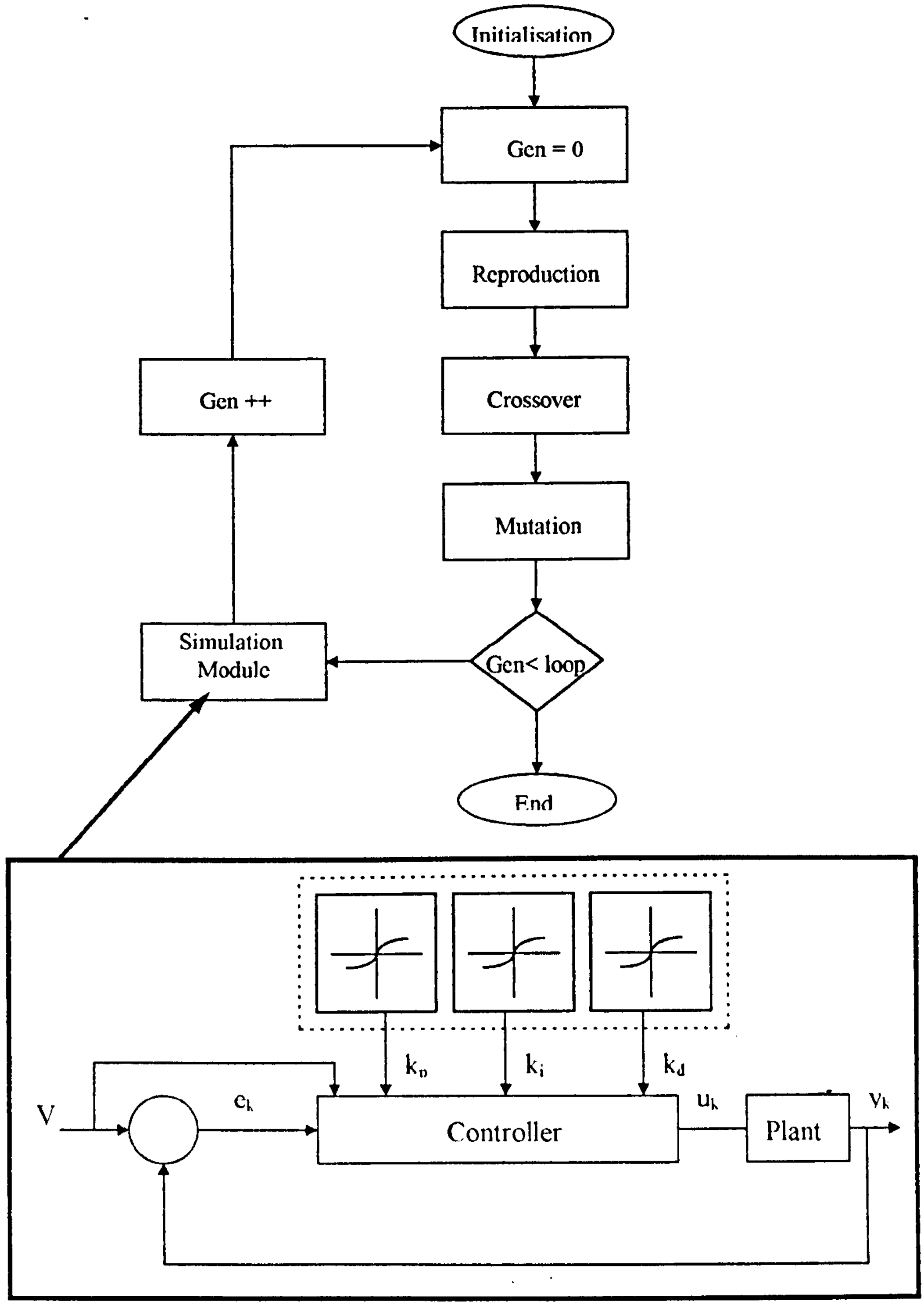


Figure 4.4 Sequence of Genetic Tunig Process

The analyses done for the proportional gain function can be used to produce the remaining two gains. By adopting the analysis of equation (4.22), a 5 points integral gain function can be produced as

$$\begin{aligned}\mathcal{K}_i = &+\frac{(e_k-H)(e_k-2H)(e_k-3H)(e_k-4H)K_{i0}}{(24H^4)}\\ &-\frac{(e_k)(e_k-2H)(e_k-3H)(e_k-4H)K_{i1}}{(6H^4)}\\ &+\frac{(e_k)(e_k-H)(e_k-3H)(e_k-4H)K_{i2}}{(4H^4)}\\ &-\frac{(e_k)(e_k-H)(e_k-2H)(e_k-4H)K_{i3}}{(6H^4)}\\ &+\frac{(e_k)(e_k-H)(e_k-2H)(e_k-3H)K_{i4}}{(24H^4)}\end{aligned} \tag{4.25}$$

This equation, represents integral gain function for all five points in one smooth function. Figure (4.3b) shows the mapping of the integral gain function.

Finally a 5 points function based on the analysis done in equation (4.21, and 4.22), can be used to produce a derivative gain function as

$$
\begin{aligned}
\mathscr{K}_{\mathbf{d}} =& + \frac{(\Delta^2 e_k - H)(\Delta^2 e_k - 2H)(\Delta^2 e_k - 3H)(\Delta^{e_k} - 4H) K_{d0}}{(24 H^4)} \\
& - \frac{(\Delta^2 e_k)(\Delta^2 e_k - 2H)(\Delta^2 e_k - 3H)(\Delta^2 e_k - 4H) K_{d1}}{(6 H^4)} \\
& + \frac{(\Delta^2 e_k)(\Delta^2 e_k - H)(\Delta^2 e_k - 3H)(\Delta'' e_k - 4H) K_{d2}}{(4 H^4)} \\
& - \frac{(\Delta^2 e_k)(\Delta^2 e_k - H)(\Delta^2 e_k - 2H)(\Delta^2 e_k - 4H) K_{d3}}{(6 H^4)} \\
& + \frac{(\Delta^2 e_k)(\Delta^2 e_k - H)(\Delta^2 e_k - 2H)(\Delta^2 e_k - 3H) K_{d4}}{(24 H^4)} \qquad 4.26
\end{aligned}
$$

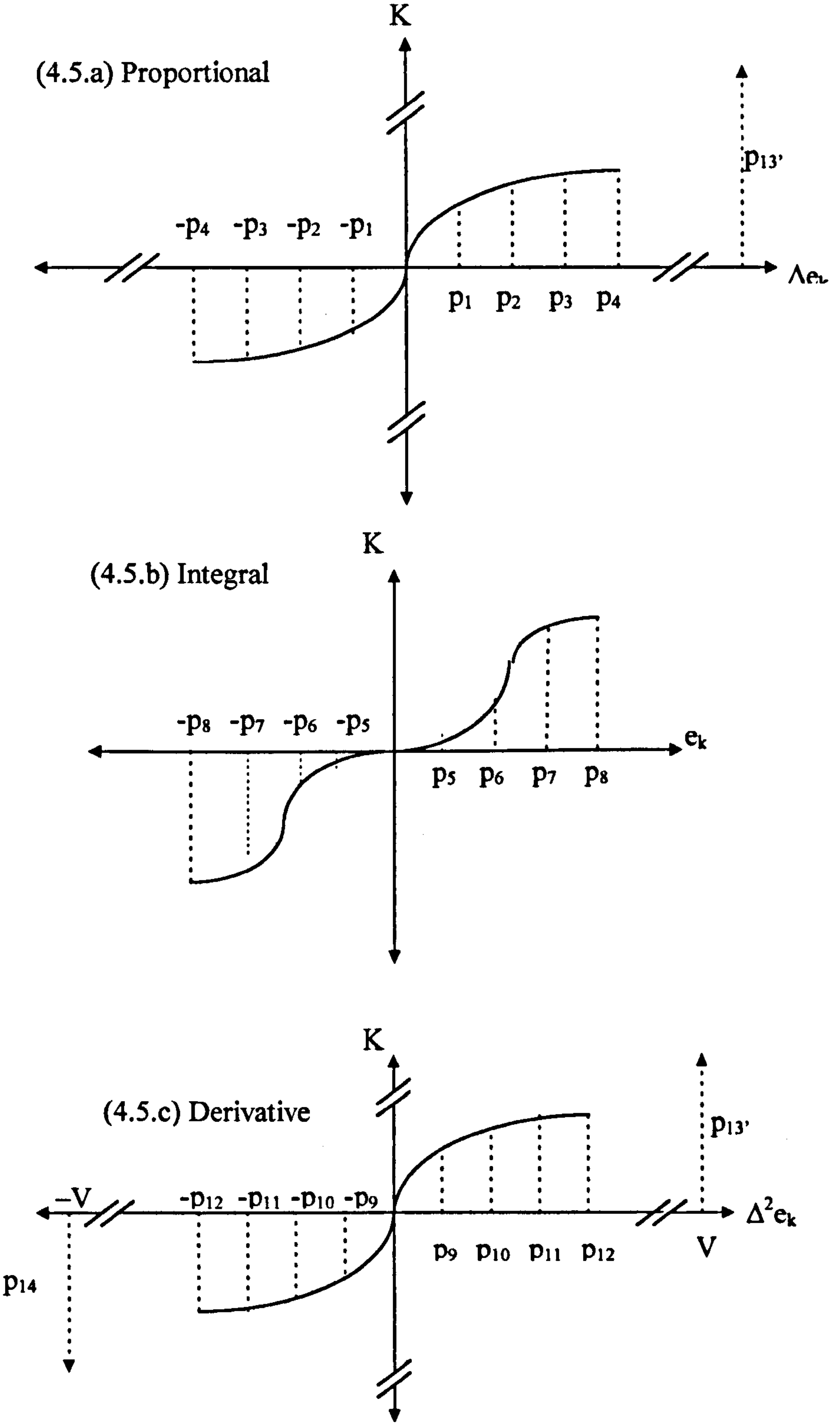
This equation, represents the derivative gain function for all five points in one smooth function. Figure (4.3c), shows the mapping of the derivative gain function.

It should also be pointed out that (K_{p0}, K_{i0}, and K_{d0}) all are equal to zero in the above equations, which forces the functions to go through the origin.

Also if the tracking zone length is equal to E^N , then

$$H = \frac{E^4}{4} \qquad 4.27$$

where E^4 is used since the fifth point is a zero to allow the function to go through the origin.



(4.5.a) Proportional

(4.5.b) Integral

(4.5.c) Derivative

Figure 4.5 Genetic polynomial interpolation mapping of gains

The mapping of the gain functions given by equations(4.24, 4.25, and 4.26), are show in figure (4.3), where

e_k is the error,

Δe_k is the first order backward difference in error,

$\Delta^2 e_k$ is the second order backward difference in error.

In order to use GAs to select the tuning parameters in such a way that as to produce satisfactory response in the case of a step input, it is only necessary to encode the elements of the tracking zone gain functions, plus the feed forward gains needed for the set point change zone, as binary strings. The binary string would be represented as

$$\{ \underset{\text{proportional}}{(K^1_p \ldots. K^4_p)} \quad \underset{\text{integral}}{(K^1_i \ldots. K^4_i)} \quad \underset{\text{derivative}}{(K^1_d \ldots. K^4_d)} \quad \underset{\text{set point change}}{(F_0 \; F_1)} \}$$

4.3.1 PLANT 1

The procedure for the tuning of genetic control systems can be conveniently illustrated by designing a genetic non-linear PID control system for the open loop single-input single-output plant with transfer function given in

$$g(z) = \frac{0.8}{z^4(z-0.9)} \qquad 4.28$$

The sampling period is 0.1 sec. The arma model for the plant is of the form

$$y_k = a_0 y_k + b_0 u_{k-5} \qquad 4.29$$

the plant variables are given by

$a_0 = 0.9$

$b_0 = 0.8$

Where the incremental PID controller is given by equation (4.17), and is governed by equations (4.21, 4.22, 4.23).

4.3.1.1 MULTI-ZONED PIN INCREMENTAL PID CONTROLLER

Initially the gain functions $\mathcal{K}_p$, $\mathcal{K}_i$, and $\mathcal{K}_d$ are chosen as functions of first order backward deference in error, error, and second order backward deference in error respectively, as given by equations, (4.21, 4.22, 4.23), i.e. multiple zoned non-linear PID controller was designed by means of genetic algorithms, such that the Integral square error (ISE)is minimised for the plant. In this case, a population of 100, a crossover probability, $P_c = 0.65$, and a mutation probability, $P_m = 0.01$, was used, the first and second order backward difference in error (i.e. Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen as the tracking maximum values for the design. the diagram in Figure (4.6) shows the transient response of the genetically designed controller, figure 4.6a, 4.6b, and 4.6c shows the resulting profiles for the proportional, integral, and derivative gains respectively.

4.3.1.2 LINEAR PID CONTROLLER

Finally to contrast the deference between linear and non-linear PID controllers, a genetically designed linear PID controller was also considered, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, K_d, are chosen by the GA to minimise the ISE for the fixed controller. In this case, a population of 100, a crossover probability, $p_c = 0.65$, and a mutation probability, $P_m = 0.01$ was used. Figure 4.7 shows the transient response of the genetically designed controller.

CONTROLLER	MULTI-ZONED PIN	LINEAR
ISE	5.005	9.16

Table (4.1)

Table (4.2) shows the gains for the multi-zoned non-linear PID controller.

Gains	Zone1	Zone2	Zone3	Zone4
$\mathcal{K}_p$	0.005	0.076	0.0.3	0.031
$\mathcal{K}_i$	1.466	1.945	0.5	0.016
$\mathcal{K}_d$	0.007	0.009	0.003	0.001

Table (4.2).

Table (4.3) shows the feed forward gains for the non-linear multi-zoned PID controller.

Feed Forward Gains	3.373	3.798

Table (4.3).

4.3.2 PLANT 2

A second plant was considered, to investigate the effectiveness of the genetic algorithms in designing PIN controllers. The plant considered has a transfer function of the form:

$$g(z) = \frac{0.03573 + 0.044625\, z}{z^{4}(0.0513423 - 1.4331\, z^{2})} \qquad 4.30$$

the sampling period is 0.1 sec.

The Arma model for the plant is of the form

$$y_k=a_0 y_{k-1}+a_1 y_{k-2}+b_0 u_{k-5}+b_1 u_{k-6} \tag{4.31}$$

The plant variables are given bellow:

a_0 = 1.4331,

a_1 =-0.51342,

b_0 = 0.044625,

b_1 = 0.03573,

Where the incremental PID controller is given by equation (4.17), and is governed by equations (4.21, 4.22, 4.23), i.e. multiple zoned PIN controller was designed by means of genetic algorithms, such that the integral square error (ISE), is minimised for the plant.

4.3.2.1 MULTI-ZONED PIN INCREMENTAL PID CONTROLLER

Once again the gain functions $\mathcal{K}_p$, $\mathcal{K}_i$, and $\mathcal{K}_d$ are chosen as functions of first order backward deference in error, error, and second order backward deference in error respectively, as given by equations, (4.21, 4.22, 4.23). Then the non-linear PID controller was designed by means of genetic algorithms, such that the Integral square error (ISE) is minimised for the plant. In this case, a population of 100, a crossover probability, P_c=0.65, and mutation probability, P_m = 0.01 was used, the first and second order backward deference in error (i.e. Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen as the tracking maximum values for the design. Figure 4.8, shows the transient response of the genetically designed non-linear controller for the above plant, Figure 4.8a, 4.8b, and 4.8c shows the resulting profiles for the proportional, integral, and derivative gains respectively.

4.3.2.3 LINEAR PID CONTROLLER

Finally to contrast the difference between linear and non-linear PID controllers, a genetically designed linear PID controller was also considered, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, K_d, are chosen by the GA to minimise the ISE for the fixed controller. In this case, a population of 100, a crossover probability, p_c = 0.65, and a mutation probability, P_m = 0.01 was used. Figure 4.9 shows the transient response of the genetically designℯd controller.

CONTROLLER	MULTI-ZONED PIN	LINEAR
ISE	6.003	10.57

Table (4.4)

Table (4.5) shows the gains for the multi-zoned non-linear PID controller.

Gains	Zone1	Zone2	Zone3	Zone4
$\mathcal{K}_p$	0.013	0.011	0.032	0.257
$\mathcal{K}_i$	7.399	6.291	1.048	0.495
$\mathcal{K}_d$	0.003	0.002	0.021	0.033

Table (4.5).

Table (4.3) shows the feed forward gains for the non-linear multi-zoned PID controller.

Feed Forward Gains	6.152	2.512

Table (4.6).

4.3.3 ROBUSTNESS TEST

This test is aimed at finding out how robust are the genetically designed dual zone PID controllers are for changes in the plant operating conditions. In this case one of the plants used in the illustrative example was modified to produce different plants . To do tl.ese tests consider plant 1 with transfer function of the form

$$g(z) = \frac{0.8}{z^4(z-0.9)} = \frac{\beta}{z^\gamma(z-\alpha)}$$

To produce the new plants the values of α, β, and Γ are changed. The tables (4.7) show the range of plants considered in this robustness test, and the respective ISEs obtained,

Plänt	Lin ISE	MZ-ISE	β	α	γ
1	9.16	5.001	0.80	0.90	5
2	11.764	6.282	0.80	0.90	6
3	13.07	7.437	1.2	0.90	6
4	14.867	12.250	1.2	0.94	6
5	9.976	7.043	1.2	0.94	4
6	13.967	12.286	1.2	0.60	4
7	22.652	17.997	0.50	0.60	4

Table (4.7)

Figure (4.10 to 4.15) show the response of the plants for the non-linear multi-zoned PID controller designed using plant 1.

Table (4.8) shows the results of the robustness test using a clipped controller as described in chapter 3.

Plant	Lin ISE	CLI-ISE	β	α	γ
1	9.16	5.001	0.80	0.90	5
2	11.764	6.282	0.80	0.90	6
3	13.07	6.838	1.2	0.90	6
4	14.867	8.168	1.2	0.94	6
5	9.976	5.263	1.2	0.94	4
6	13.967	6.517	1.2	0.60	4
7	22.652	11.993	0.50	0.60	4

Table (4.7)

Figure (4.16 to 4.21) show the response of the plants for the non-linear multi-zoned clipped PID controller designed using plant 1

4.4 CONCLUSIONS

This chapter has emphasized the fact that the use of dual zone technique employed in chapter 3, can be employed in the design any incremental PID controller. Also in this chapter the Lagrangian Polynomial interpolation technique was used to map the non-linear gain functions for the multi-zoned non-linear controller. The genetic algorithm was used to design the controllers. The results have been illustrated by genetically designing PID controllers for the linear plants. It has thus been shown that the use of Lagrangian polynomial interpolation to map the non-linear gain has improved the effectiveness of the controllers compared to the controllers designed using the straight line interpolation produced in chapter 3. The robustness test indicate that the non-linear PID controllers are more robust than the linear controllers. Furthermore the clipped non-linear PID controllers exhibits far better robust properties than the un-clipped non-linear controllers. This is because the clipping forces the controller to be rate limited.

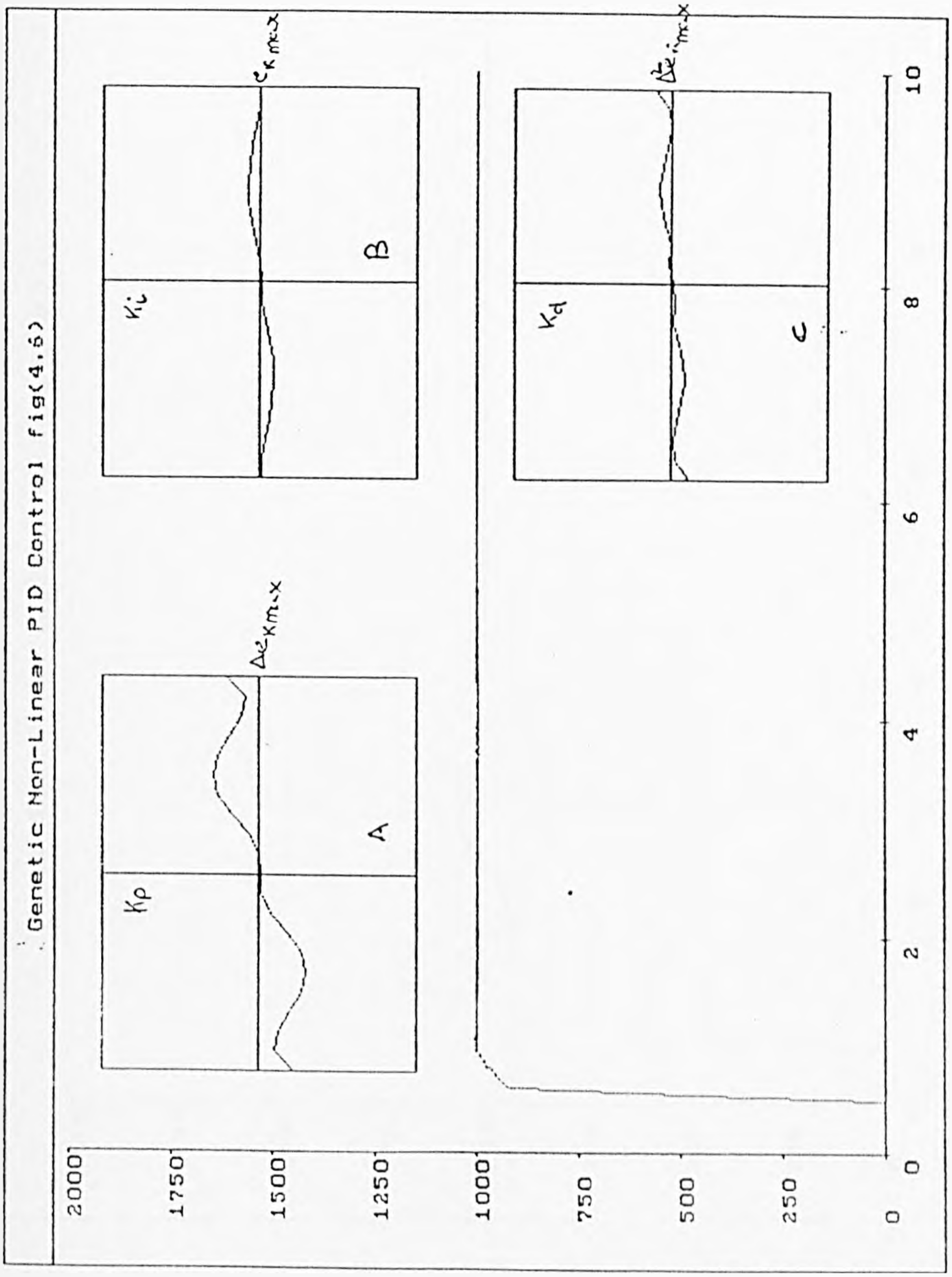


Figure (4.6) Transient response of the genetically designed multi-zoned PID controller for plant (1)

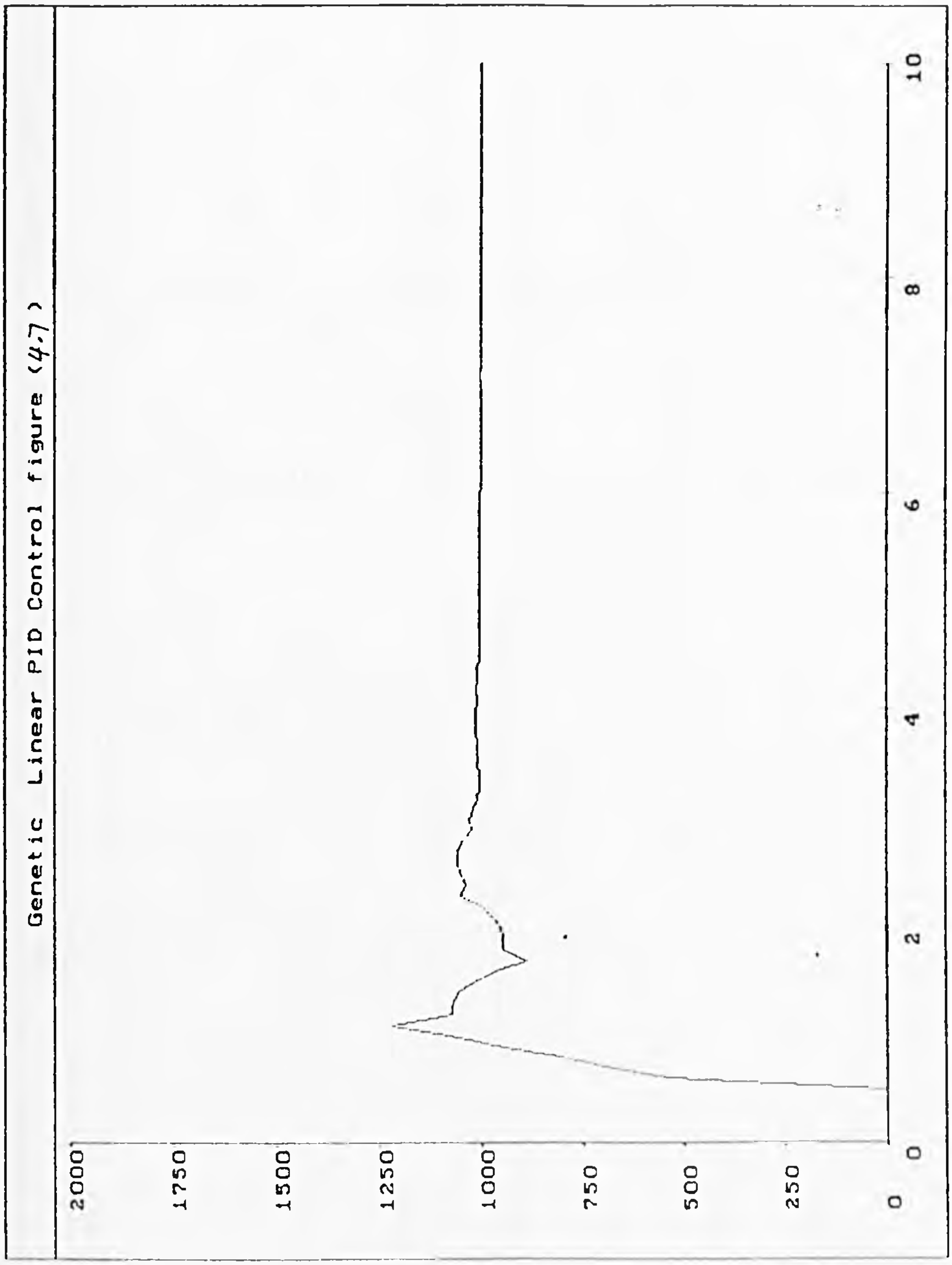


Figure (4.7) Transient response of the genetically designed linear PID controller for plant (1)

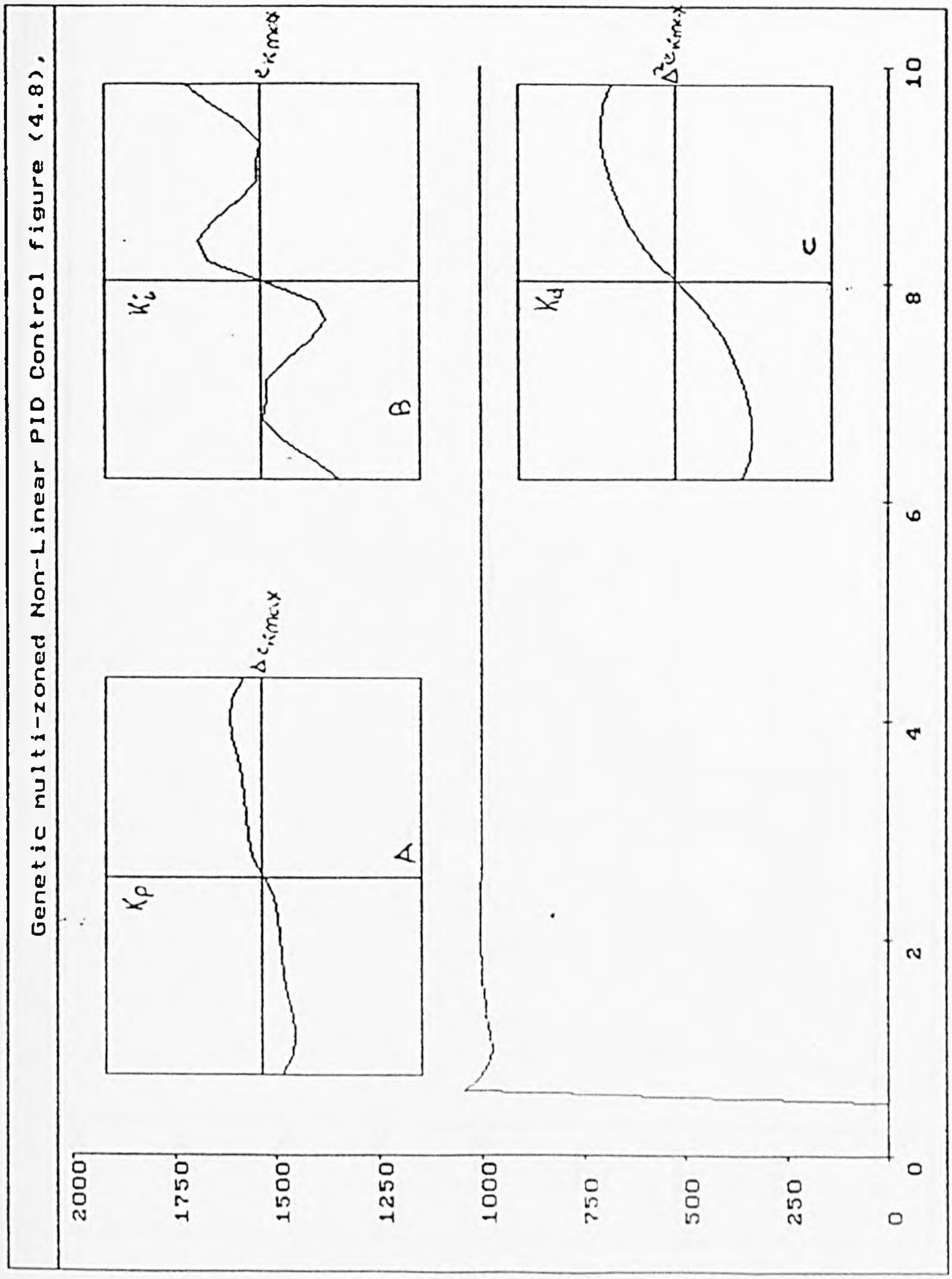


Figure (4.8) Transient response of the genetically designed multi-zoned PID controller for plant (2)

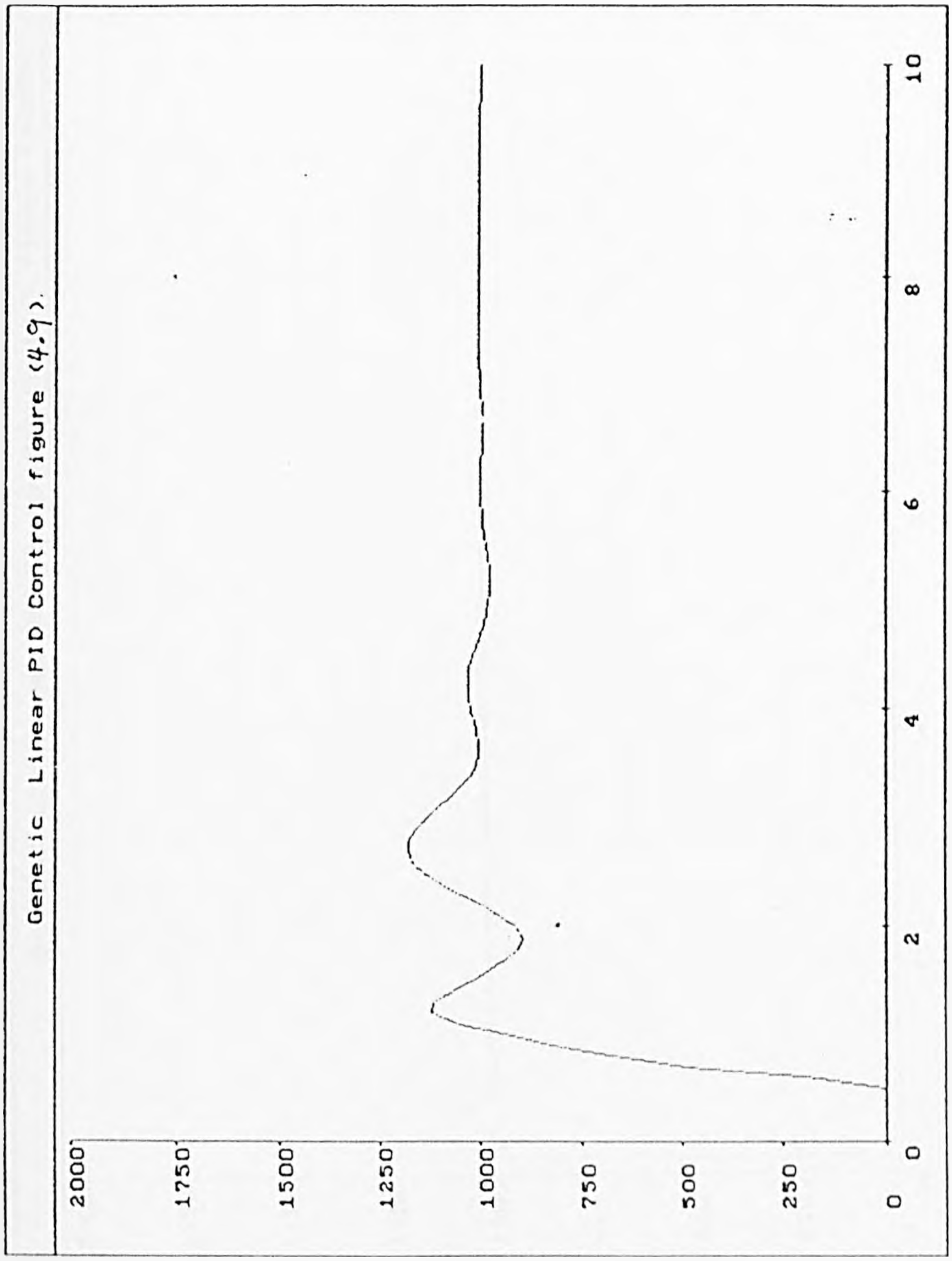


Figure (4.9) Transient response of the genetically designed linear PID controller for plant (2)

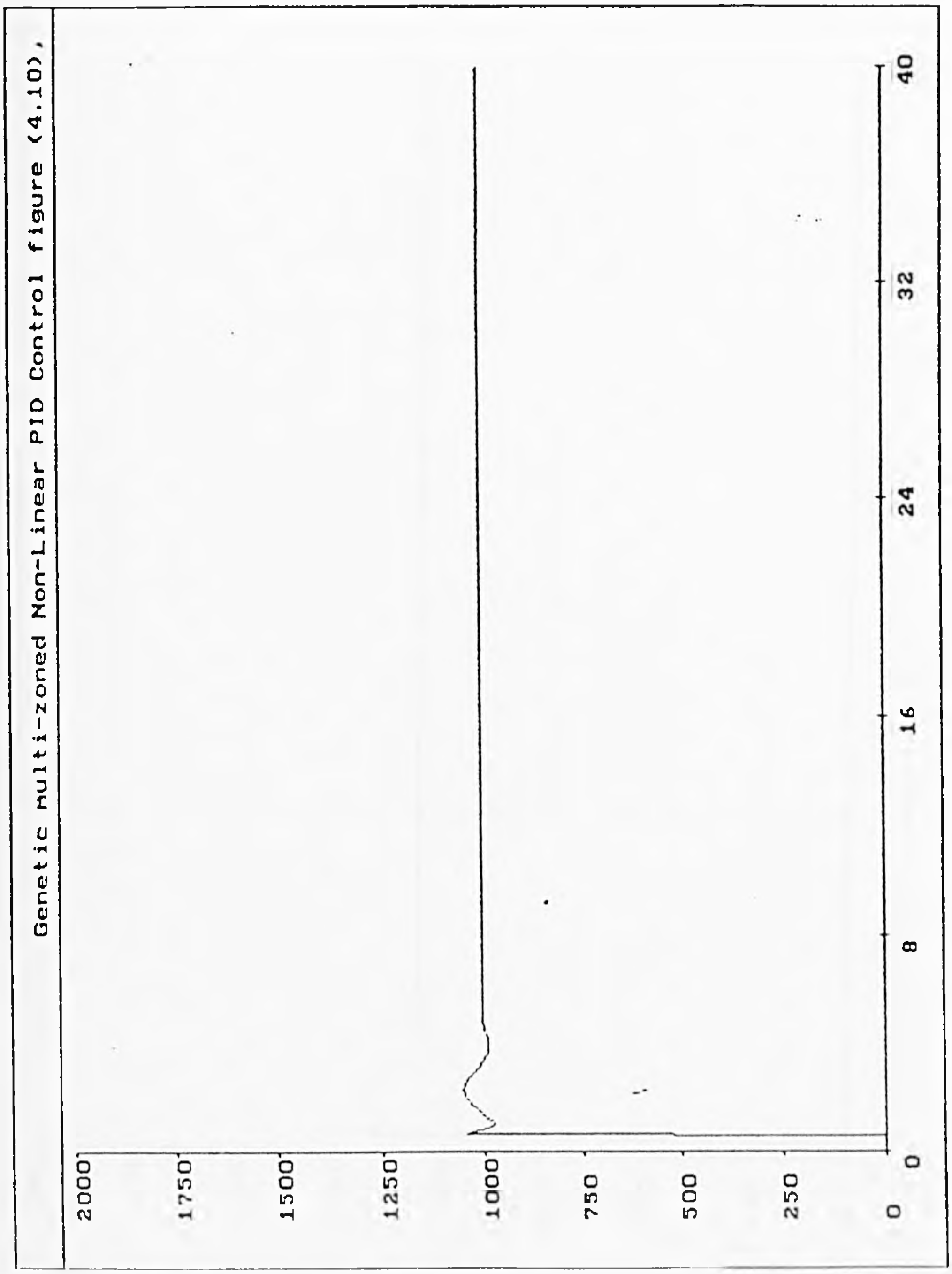


Figure (4.10) Transient response of robustness test using genetically designed linear PID controller for plant (2), table (4.7).

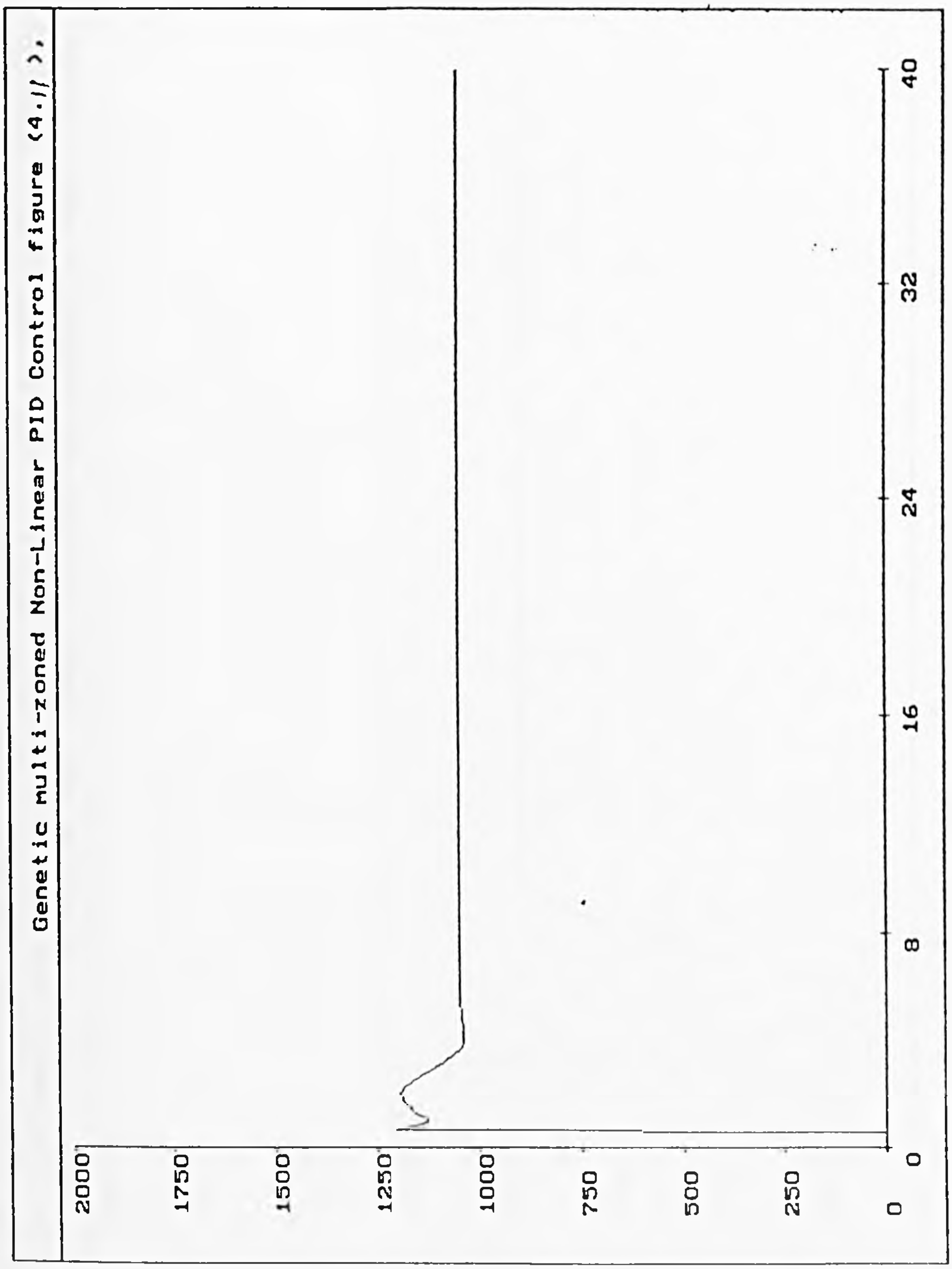


Figure (4.11) Transient response of robustness test using genetically designed linear PID controller for plant (3), table (4.7).

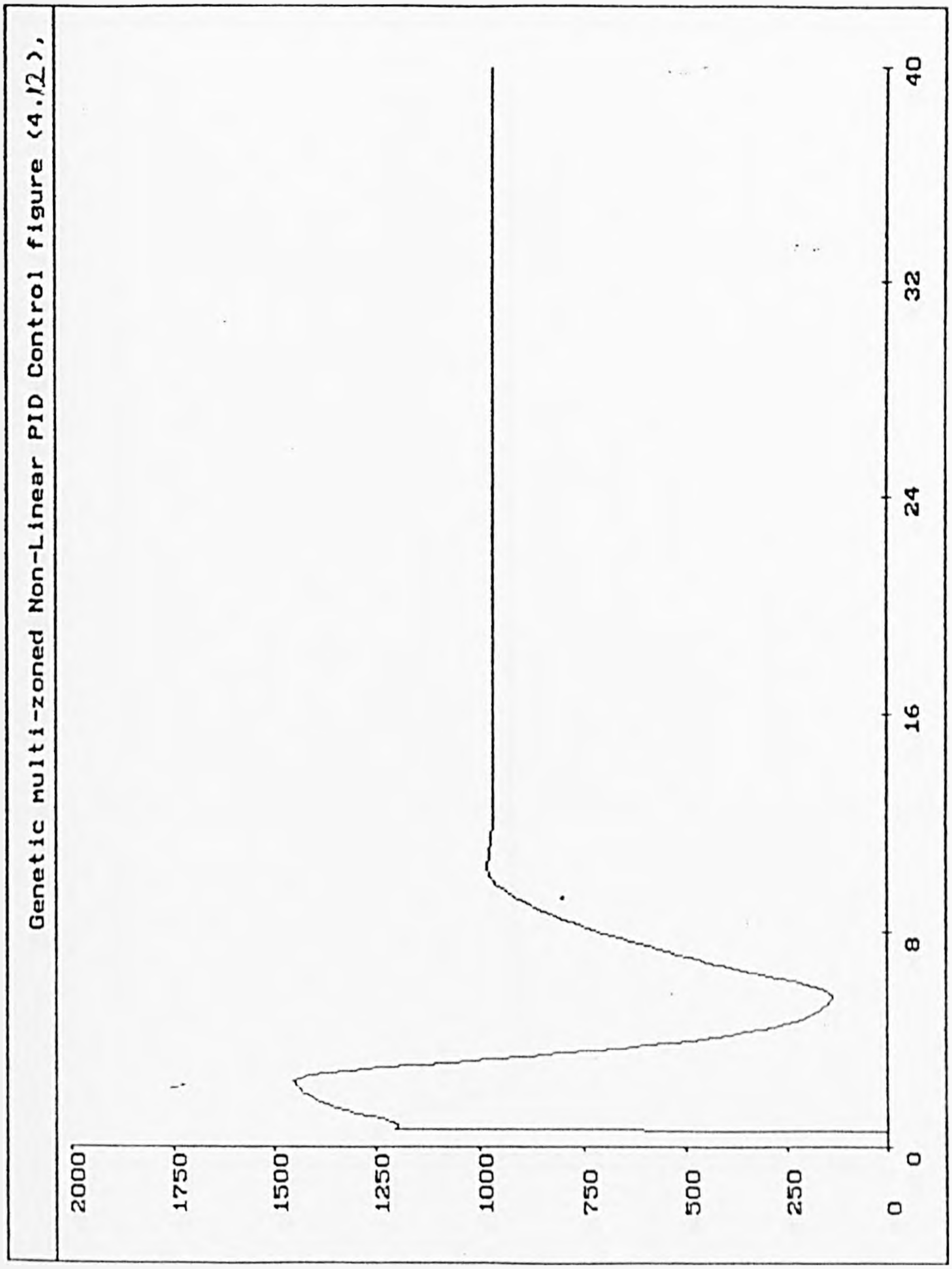


Figure (4.12) Transient response of robustness test using genetically designed linear PID controller for plant (4), table (4.7).

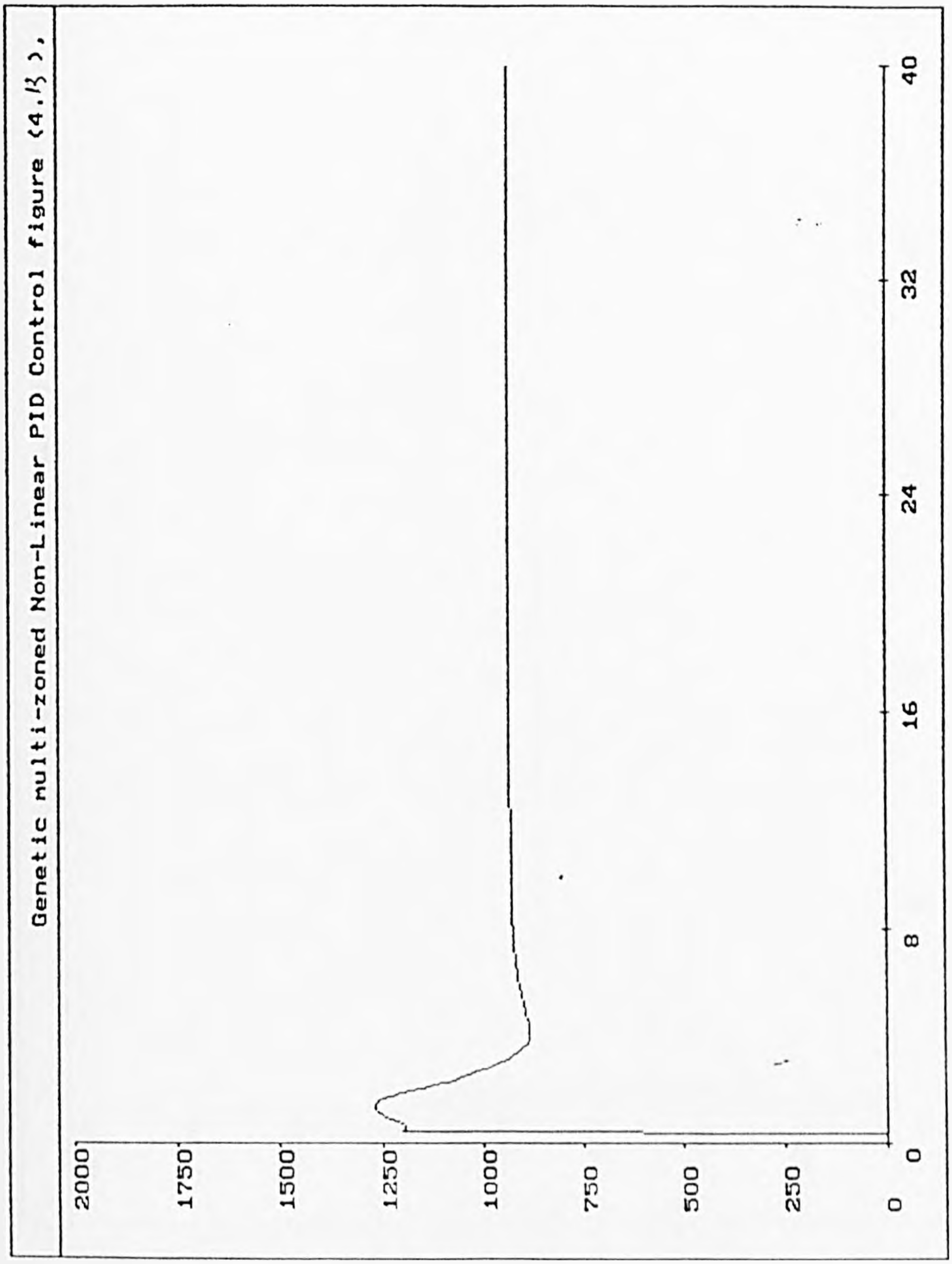


Figure (4.13) Transient response of robustness test using genetically designed linear PID controller for plant (5), table (4.7).

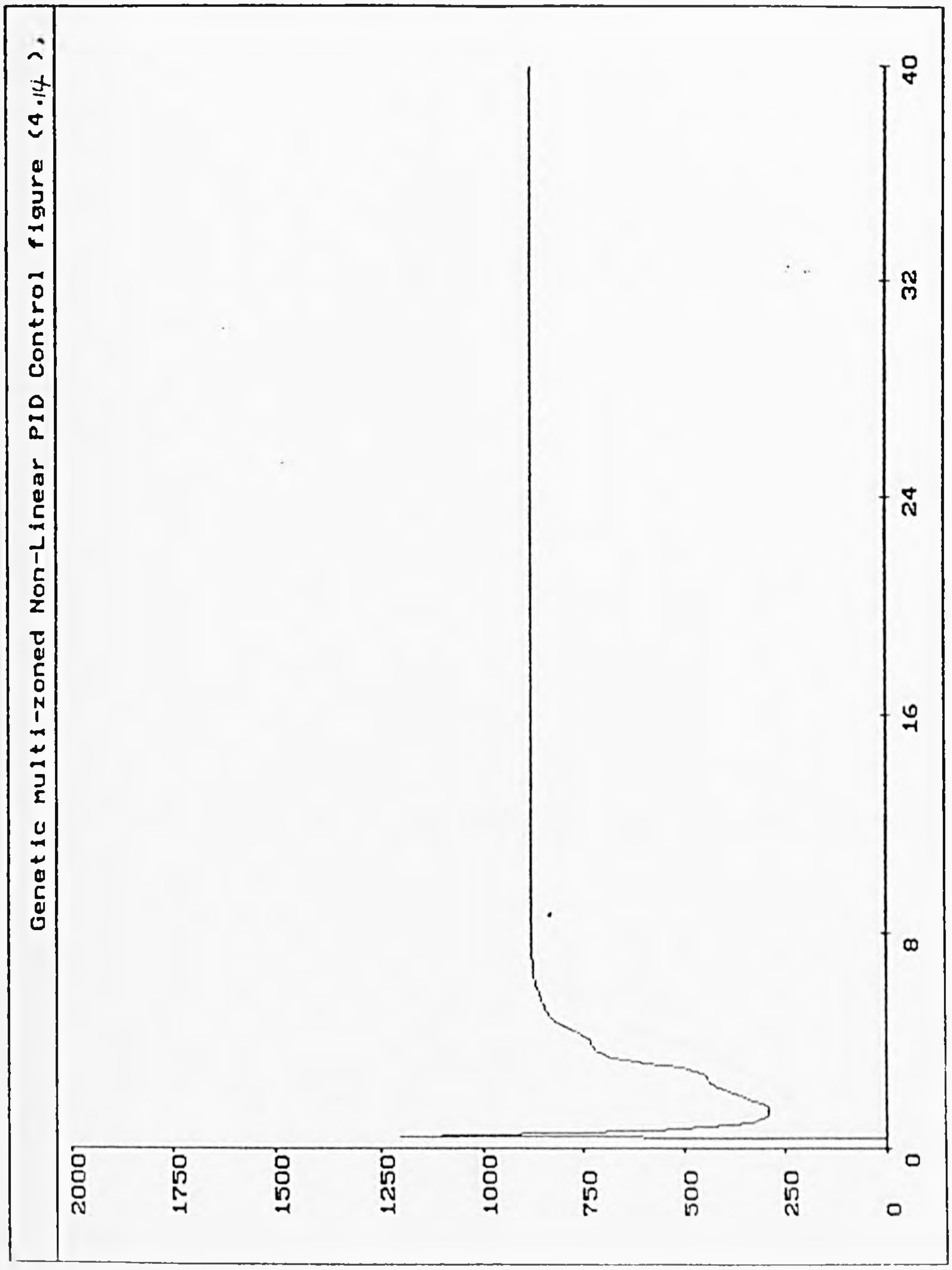


Figure (4.14) Transient response of robustness test using genetically designed linear PID controller for plant (6), table (4.7).

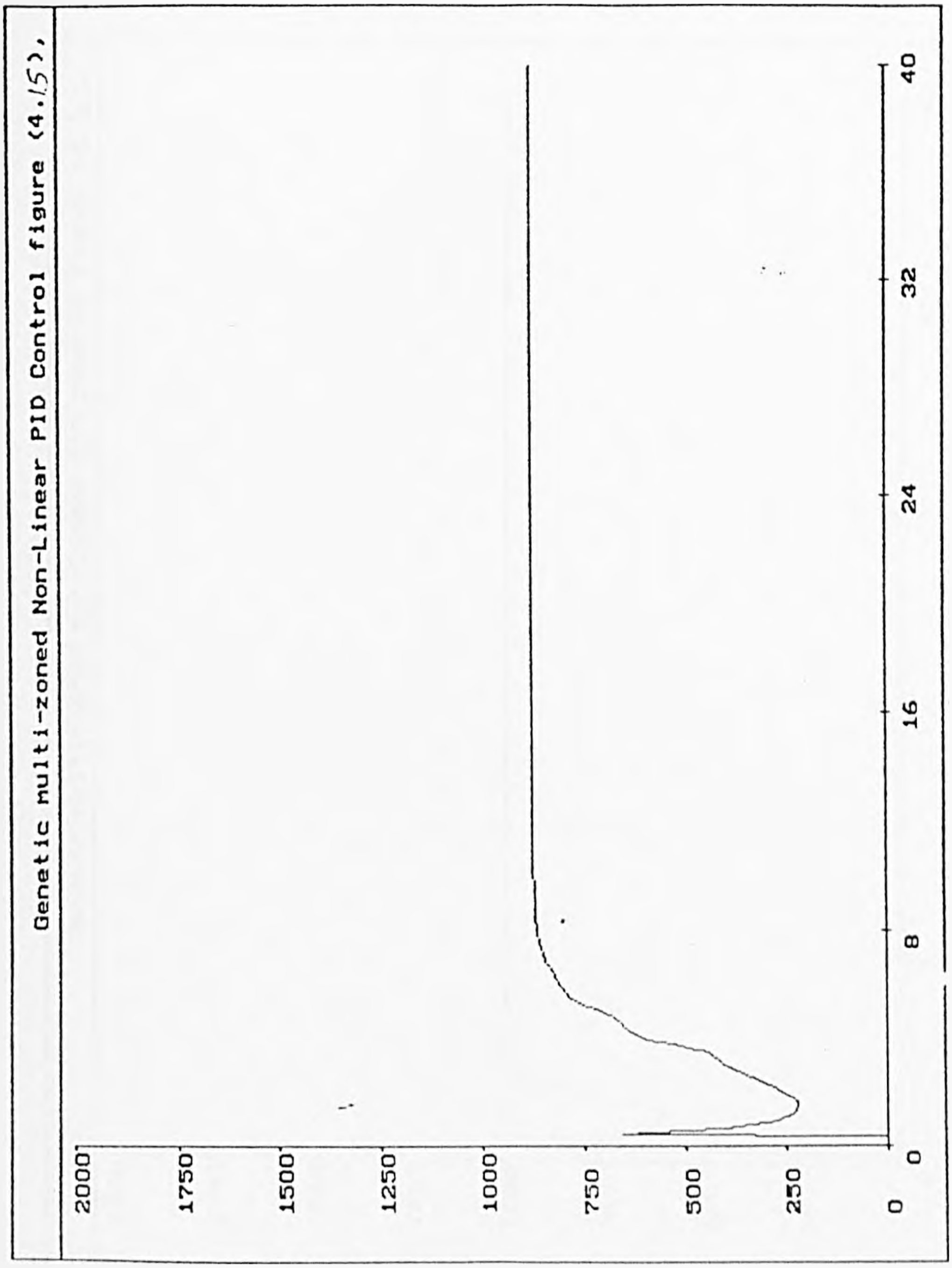


Figure (4.15) Transient response of robustness test using genetically designed linear PID controller for plant (7), table (4.7).

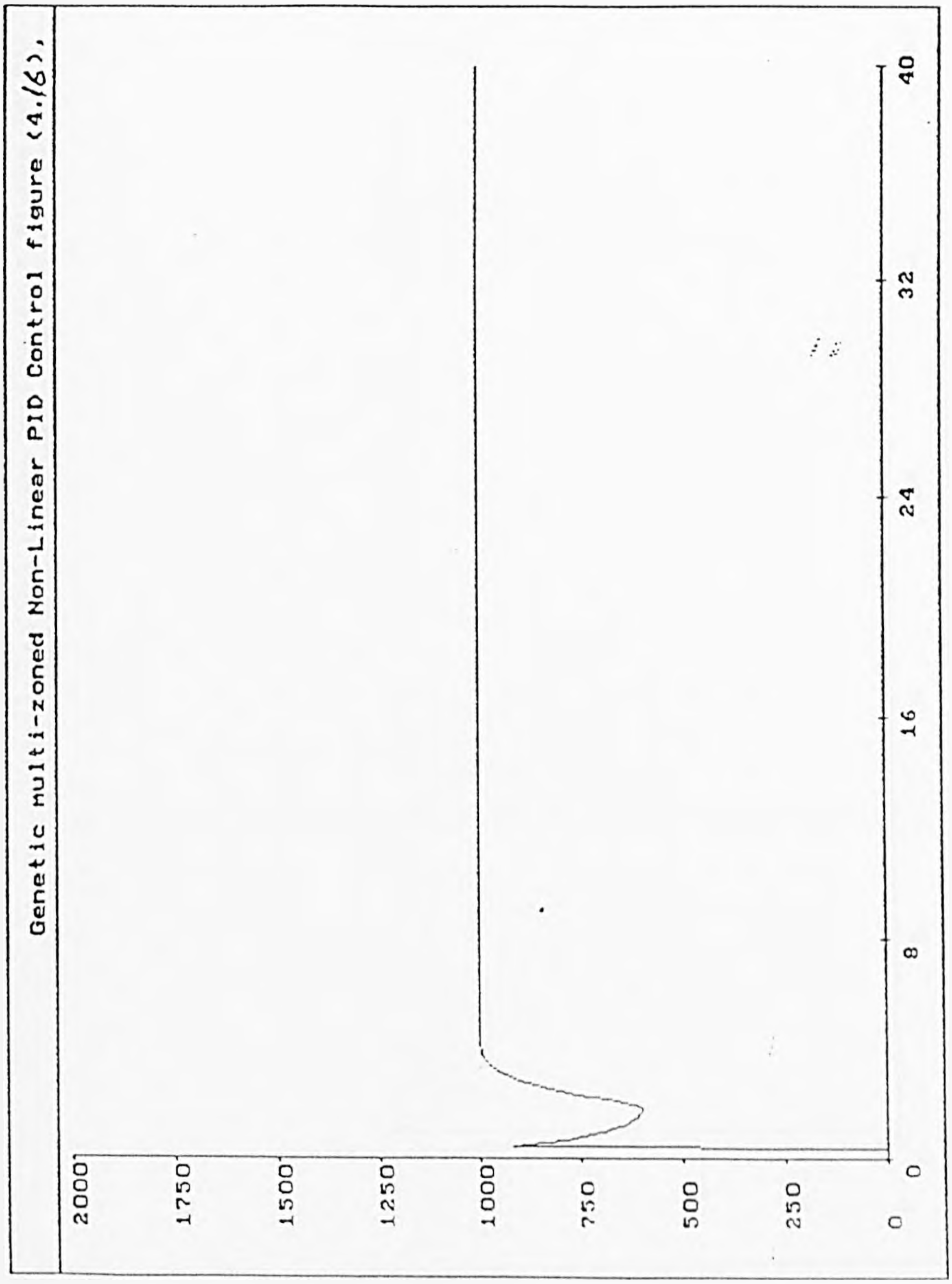


Figure (4.16) Transient response of robustness test using genetically designed non-linear multi-zoned PID controller for plant (2), table (4.8).

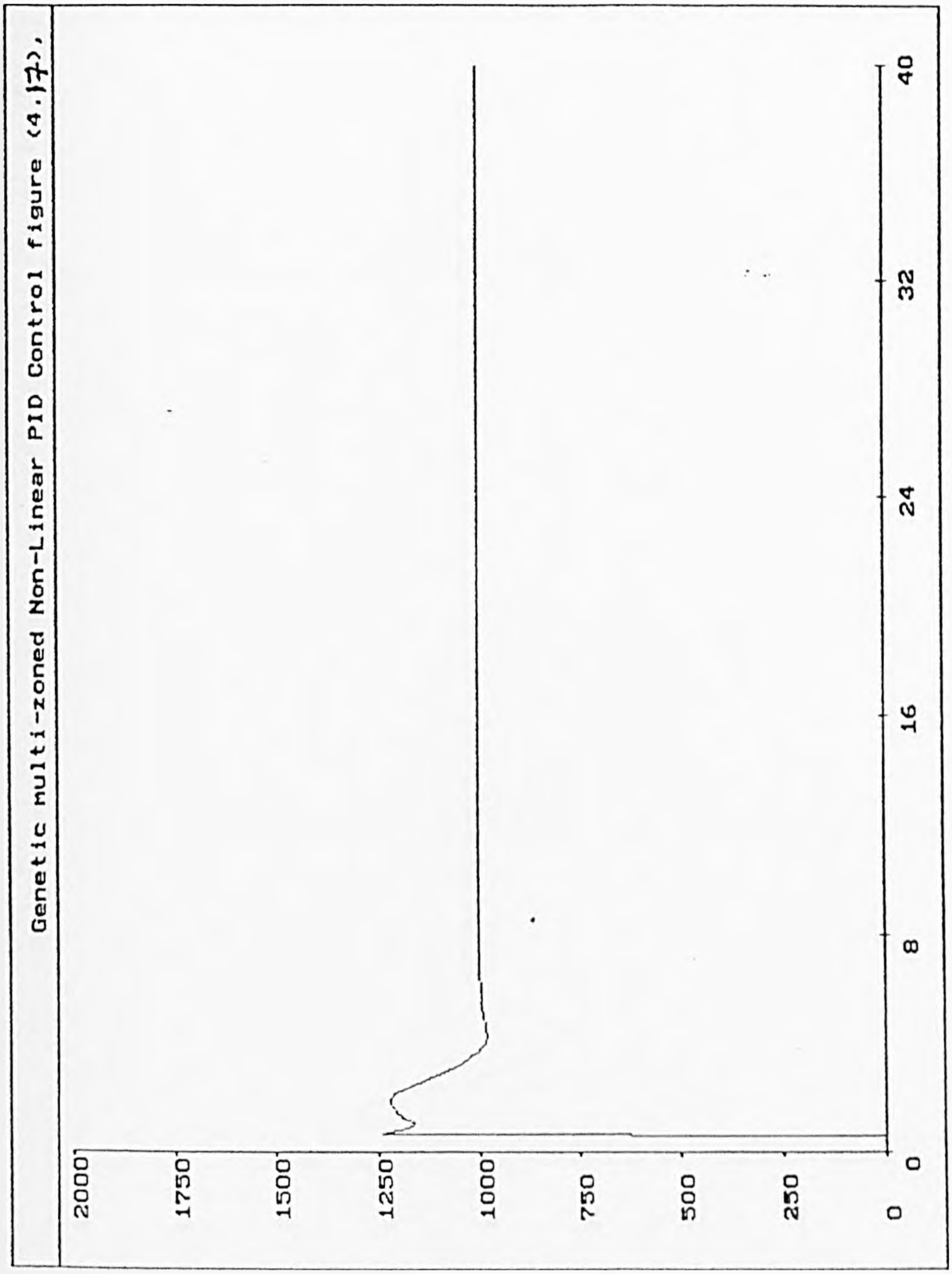


Figure (4.17) Transient response of robustness test using genetically designed non-linear multi-zoned PID controller for plant (3), table (4.8).

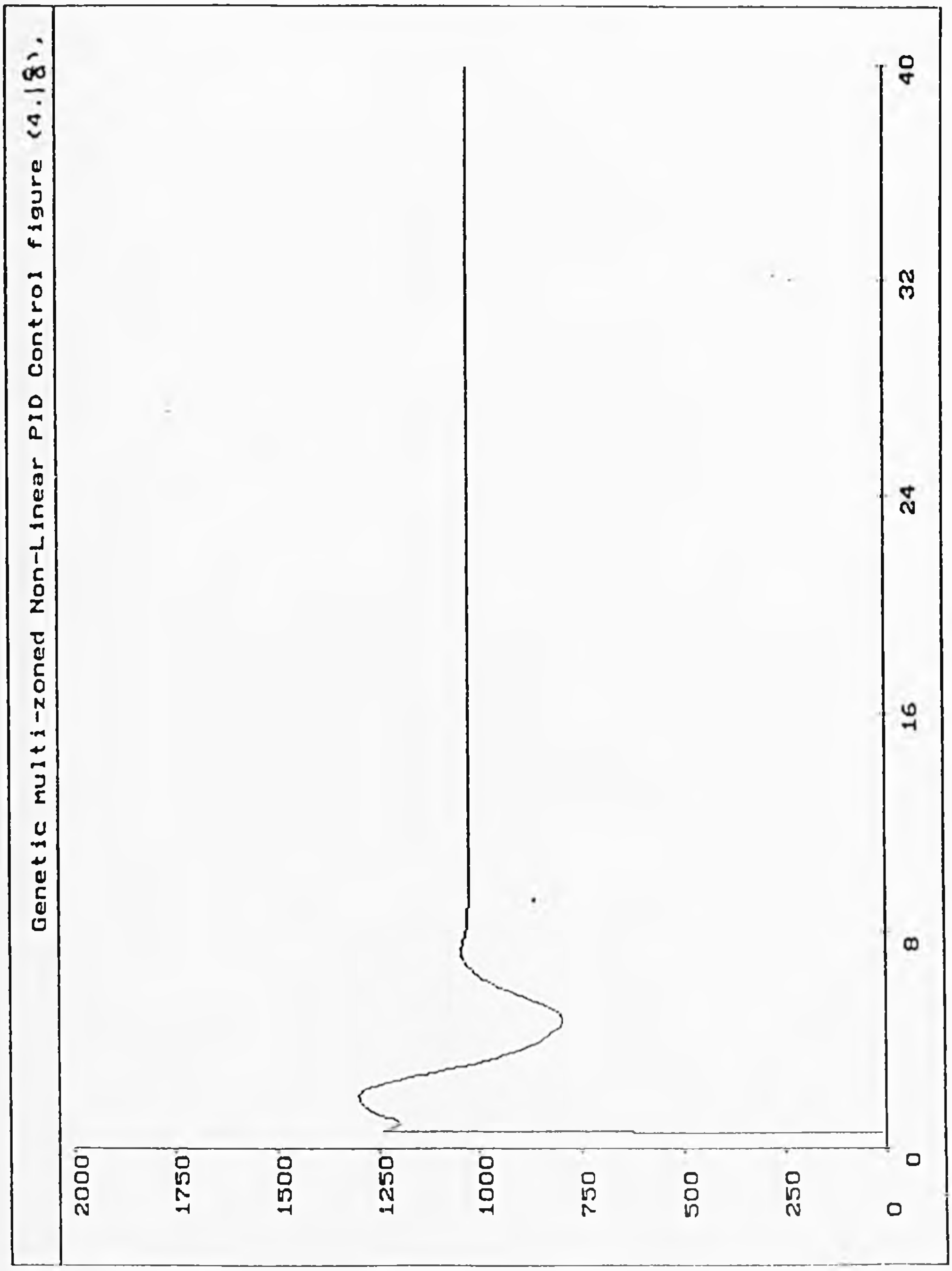


Figure (4.18) Transient response of robustness test using genetically designed non-linear multi-zoned PID controller for plant (4), table (4.8).

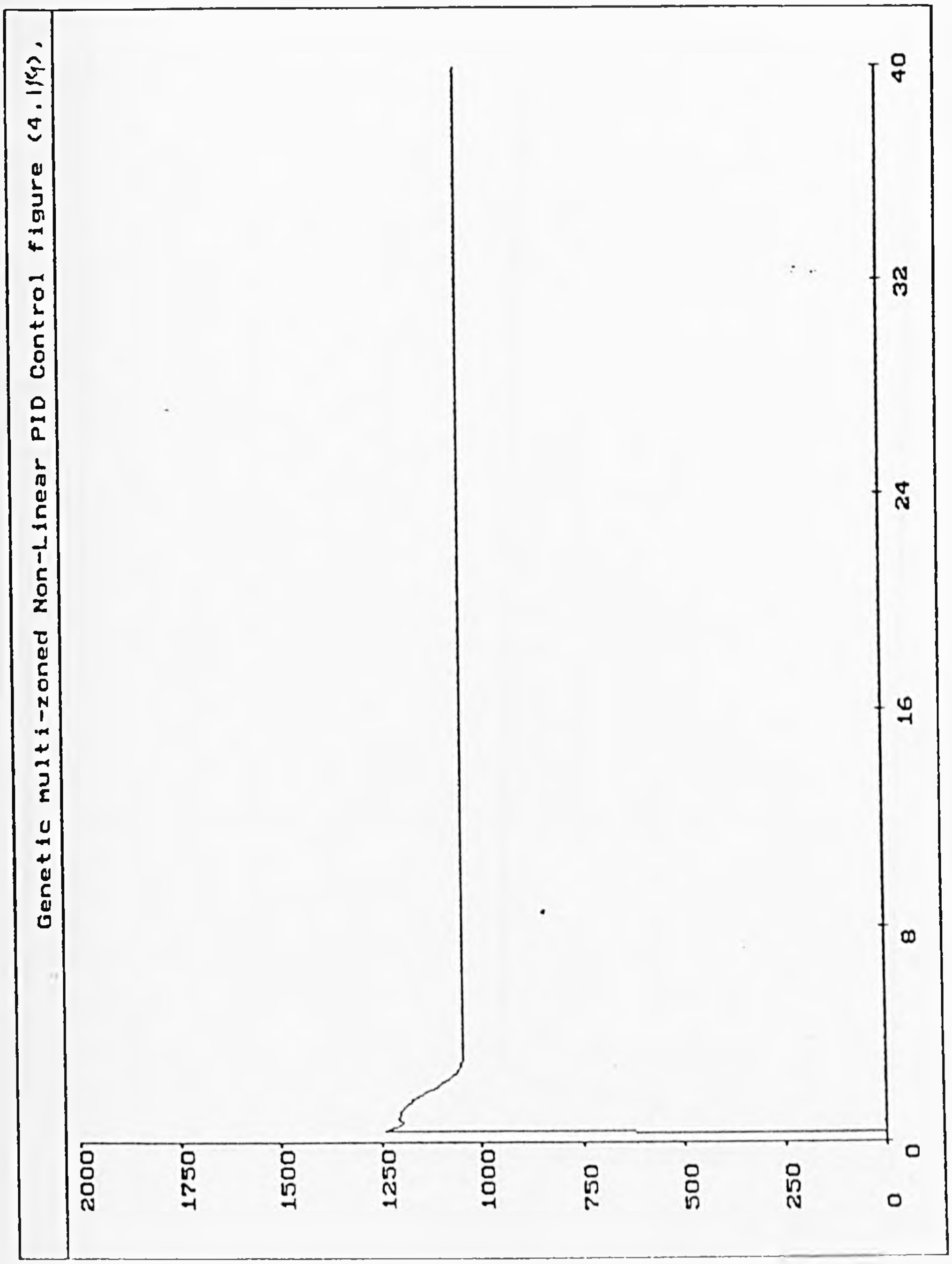


Figure (4.19) Transient response of robustness test using genetically designed non-linear multi-zoned PID controller for plant (5), table (4.8).

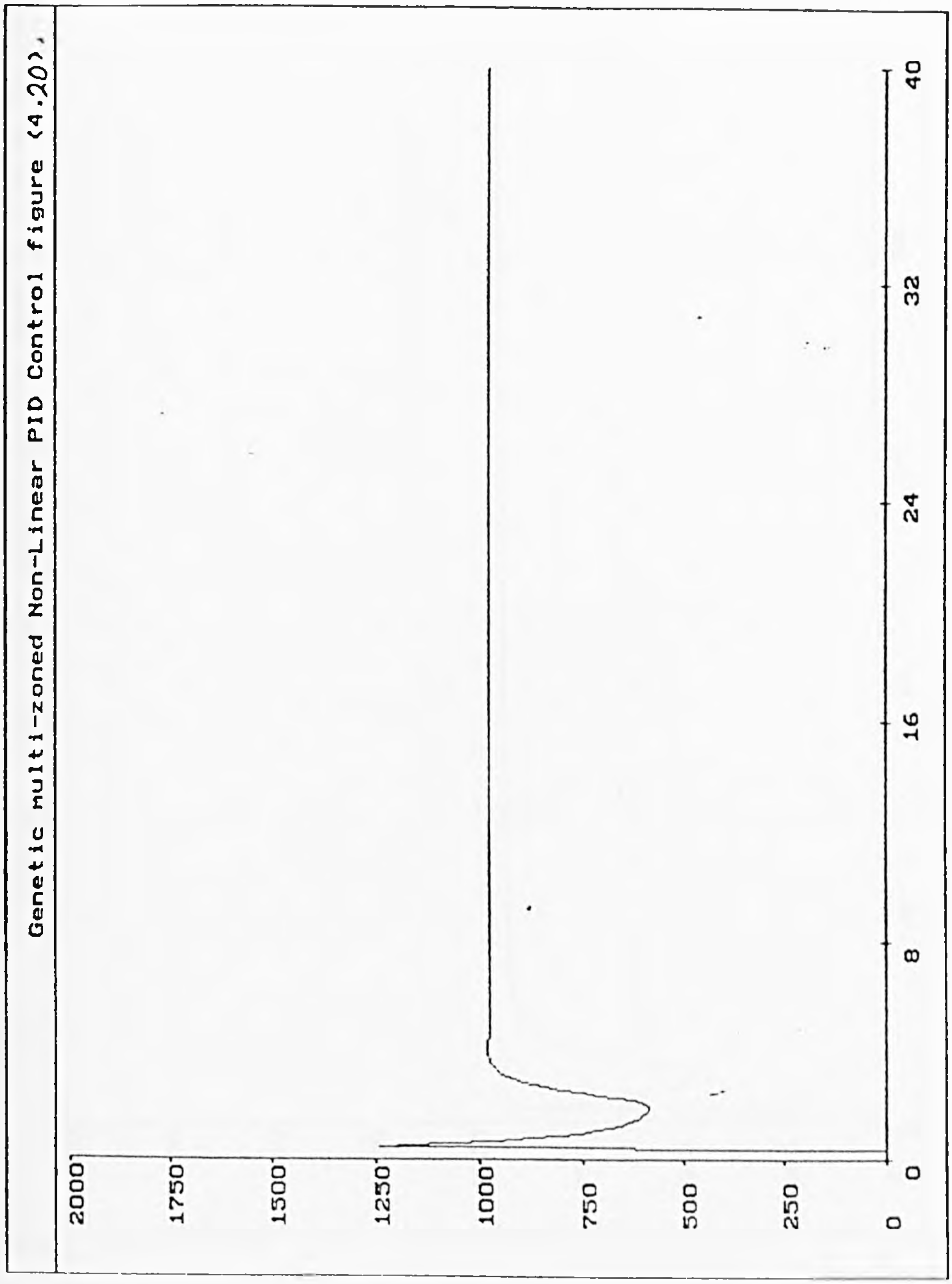


Figure (4.20) Transient response of robustness test using genetically designed non-linear multi-zoned PID controller for plant (6), table (4.8).

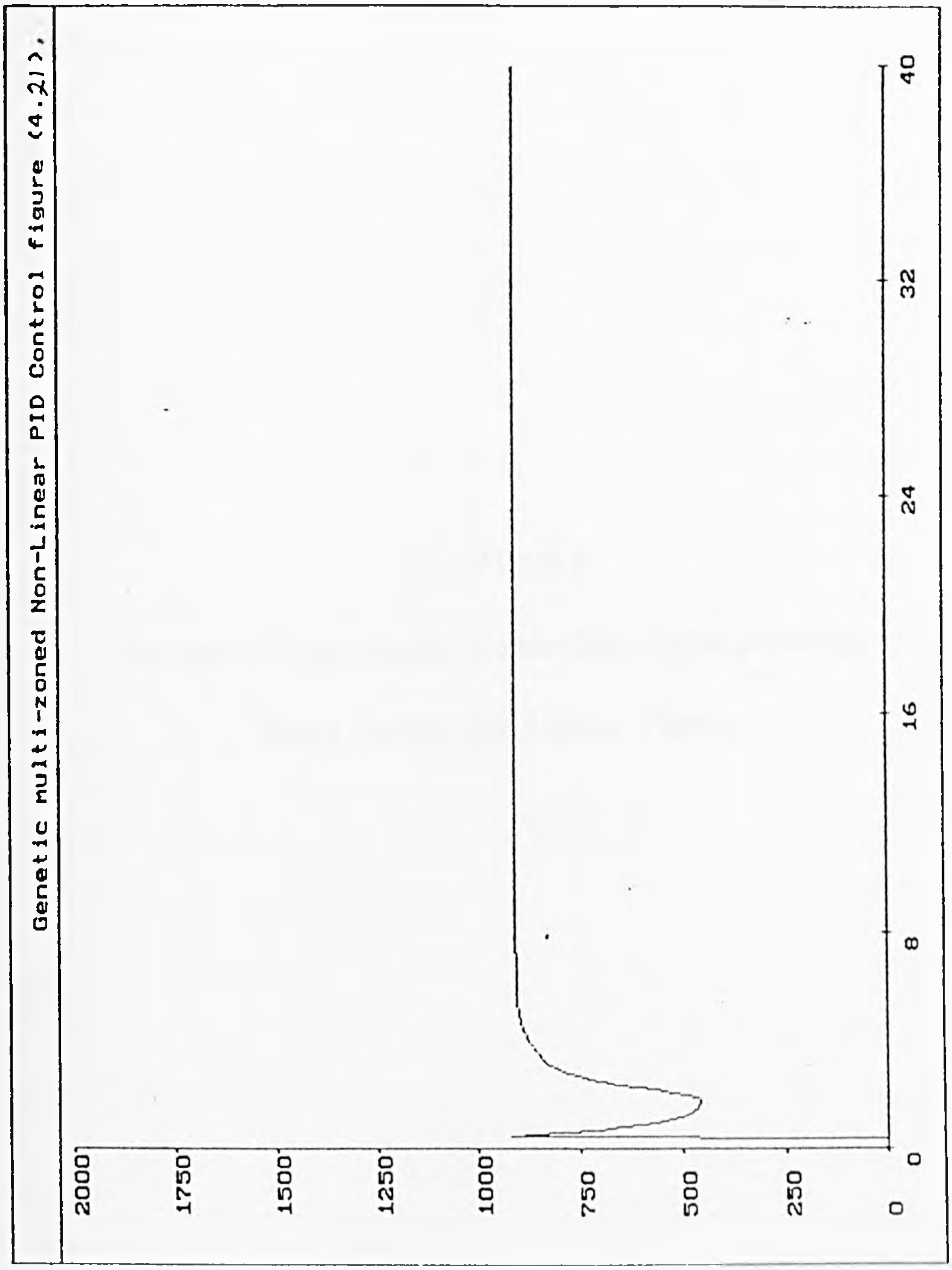


Figure (4.21) Transient response of robustness test using genetically designed non-linear multi-zoned PID controller for plant (7), table (4.8).

CHAPTER 5

Design of Non-Linear Controllers Incorporating "fuzzy Gains" for Linear Plants

Chapter 5
DESIGN OF NON-LINEAR PID CONTROLLERS INCORPORATING "FUZZY GAINS" FOR LINEAR PLANTS

5.1 INTRODUCTION

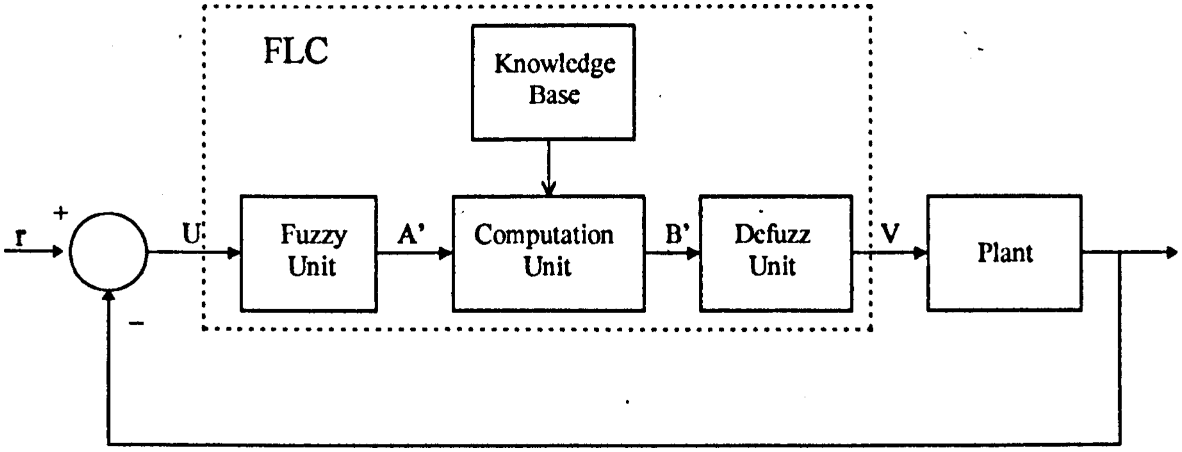
Fuzzy logic was developed by Lotfi A. Zadeh [60], of the University of California at Berkeley, it is firmly grounded in mathematical theory. The development of fuzzy theory came from the inability to describe some physical phenomena with the exact mathematical models dictated by more conventional Boolean models. Fuzziness describes event ambiguity. It measures the degree to which an event occurs, not whether it occurs. The fact that fuzziness is lacking in precision has led to its dismissal by some researchers. Others, however, see fuzzy theory as a powerful tool in the exploration of complex problems because of its ability to determine outputs for a given set of inputs without using a conventional, mathematical model. Fuzzy theory owes a great deal to human language, when people speak of temperature in terms such as "hot" or "cold" instead of in physical units such as degrees Fahrenheit or Celsius, one can see language becomes a fuzzy variable whose spatial denotation is imprecise. In this sense, fuzzy theory becomes easily understood because it can be made to resemble a high level language instead of a mathematical language. Fuzzy sets with names such as "hot" and "cold" are used to create a membership function. What determines the ranges for these fuzzy-sets values or the shape of these membership functions. In most cases, membership functions are designed by experts with a knowledge of the system being analyzed. However, human experts cannot be expected to provide optimal membership functions for a given system. Often, these are modified iteratively while trying to obtain optimality. How are these membership functions used in fuzzy controllers. In its simplest form a fuzzy logic controller is simply a set of rules describing a set of actions to be taken for a given set of inputs. It is easiest to think of these rules as if-then statements of the form IF{set of inputs} THEN{outputs}.

During the last few years researchers have concentrated on the problem of extracting control rules for fuzzy logic control (FLC). Manual extraction of rules has two major difficulties. First, experienced operators are not readily available. Second, human operators can not represents their control knowledge accurately. Thus efforts have been devoted to finding methods to extract rules automatically. For example, Procyk and Mamdani [61], have described a self-organizing FLC. Lee and Berenji [62], have reported a self-learning FLC employing reinforcement techniques to learn the required rules. Pham and Karaboga [47], produced a method for producing relation matrix for FLC this method is based on using a GA to optimise relation matrices. Jones, Kenway, and Ajlouni [39], presented a genetic design of fuzzy gain scheduled controllers for non-linear plants. Homaifar and MacCmick [49], produced a simulation design of a membership function and rule sets for fuzzy controllers using a GA. Many more researchers have worked in this field [63-67].

In chapter 3, and 4, it was shown that the GA can be used to design non-linear controllers for linear plants, by mapping the gains as non-linear functions. In this chapter, the technique of fuzzy logic is proposed to map the non-linear gains. Hence, the results of chapter 3 and 4, are extended to embrace the tuning of fuzzy non-linear PID controllers for linear plants. This technique is totally autonomous, other than choosing the number of fuzzy sets to be used in the fuzzy non-linear PID controllers. Furthermore, the resulting fuzzy controllers can be tuned to any performance measure, such as minimum rise time or minimum integral square error.

5.2 FUZZY LOGIC CONTROL

In conventional PID control, what is modeled is the system or process being controlled, whereas in fuzzy logic control the focus is on the human operator's behaviour. In general, fuzzy logic is best applied to non-linear, ill defined systems. The basic structure of a fuzzy logic controller (FLC), is conceptually shown in figure (5.1). The knowledge base of the FLC is represented by its rules for controlling the plant, using the fuzzy rules in the fuzzy inference produces a fuzzy output set B' from a fuzzy input set A'. The fuzzy output set B' is defuzzified by the defuzification unit to give the crisp output u to control the plant. The input fuzzy set A' is obtained from the fuzzification unit. The input to the later can be variables such as rate of change of error or any other variable for which fuzzy rules can be obtained.



Fgure (5.1) A block diagram showing cnventional fuzzy controller

5.3 SYNTHESIS OF FUZZY PID CONTROLLERS

In conventional fuzzy logic control the focus of design is to:

1) define reference fuzzy sets for the inputs and outputs involved;

2) define a set of control rules relates the input sets to the output sets;

3) design the membership function to obtain satisfactory control.

In this design method for a simple fuzzy control scheme is proposed, the focus of the design is shifted to choosing fuzzy membership function for the PID gains incorporated in a conventional PID controller. In this way the problems of 1, 2, and 3, are all avoided.

The linear SISO plants under consideration are governed on the continuous time set T = $[0,\infty)$ by state and output equation of respective forms

$$\dot{x}(t) = Ax(t) + bu(t) \qquad 5.1$$

and

$$y(t) = cx(t) \qquad 5.2$$

where

$x(t) \epsilon \Re^{n}$ is the state vector,

$y(t) \epsilon \Re$ is the scalar output from the plant,

$u(t) \epsilon \Re$ is the scalar input to the plant,

$A \quad \epsilon \Re^{nxn}$ is the plant matrix,

$b \quad \epsilon \Re^{nx1}$ is the input matrix,

$c \quad \epsilon \Re^{1xn}$ is the output matrix.

It is assumed that the plant is functionally controllable, so that none of the transmission zero of the plant lies at the origin in the complex plane and therefore any and all solutions of s in

$$\begin{vmatrix} sI_n-A, & -b \\ C, & 0 \end{vmatrix} = 0 \qquad 5.3$$

are non-zero [Rosenbrock (1974)]. This assumption ensures that rank M = n + 1 [Porter and Power (1970)] where the system matrix is given by

$$M = \begin{bmatrix} A, & b \\ C, & 0 \end{bmatrix} \qquad 5.4$$

In order to design a non-linear PID controllers for SISO linear plants governed by equations (5.1) and (5.2), it is convenient to consider the behaviour of such plants on the discrete-time set T_T = {0, T, 2T,....}.

This behaviour is governed by state and output equations of the respective forms [Kwakernaak and Sivan (1972)]

$$Tx_{k+1}=\Phi Tx_k+\Psi Tu_k \tag{5.5}$$

and

$$y_k=\Gamma x_k \tag{5.6}$$

where

$$\Phi=e^{AT} \tag{5.7}$$

$$\Psi=\int_0^T e^{A\tau}bd\tau \tag{5.8}$$

and

$$\Gamma = c \tag{5.9}$$

In these equations, $x_kT \in \Re^n$, $u_kT\in\Re$, $y_kT \in \Re$,, $\Phi \in \Re^{nxn}$, $\Psi \in \Re^{nx1}$, $\Gamma \in \Re^{1x}$, and T $\in \Re^+$ is the sampling period.

It is evident from chapter 3 that the linear incremental PID controller can be described as

$$\Delta u_k = T(k_p \Delta e_k + Tk_i e_k + k_d \Delta^2 e_k) \qquad \mathbf{5.10}$$

where

Δu_k is the incremental change in input,

Δe_k is the first order backward difference in error,

$\Delta^2 e_k$ is the second order backward difference in error.

K_p is the current effective value of the proportional gain,

K_i is the current effective value of the integral gain,

K_d is the current effective value of the derivative gain,

T is sampling time.

It is evident from chapter 3 that the error can be written as

$$e_k = v_k - y_k \qquad 5.11$$

It is also evident from chapter 3 that the first order backward difference in error can be written as

$$\Delta e_k = e_k - e_{k-1} \qquad 5.12$$

Further more from chapter 3 the second order backward difference in error can be expressed as

$$\Delta^2 e_k = e_k - 2e_{k-1} + e_{k-2} \qquad 5.13$$

The incremental controller given by equation (5.10), can take one of two forms, one is linear, and the other is non-linear. In this chapter the non-linear form is to be investigated. The non-linearities are a function of the plant error.

It follows from equation (5.10), that the incremental non-linear fuzzy PID controller can be described by equation of the form

$$\Delta u_k=\alpha_p(\Delta e_k)\Delta e_k+\alpha_i e_k+\alpha_d(\Delta^2 e_k)\Delta^2 e_k \qquad 5.14$$

where

$\alpha_p(\Delta e_k)$ is a fuzzy function representing non-linear proportional gain,

$\alpha_i(e_k)$ is a fuzzy function representing non-linear integral gain,

$\alpha_p(\Delta^2 e_k)$ is a fuzzy function representing non-linear derivative gain.

It is evident from equation (5.14), that the gain functions can be represented by fuzzy sets of the form

$$\mathcal{K}_p = \alpha_p(\Delta e_k) \qquad 5.15$$

$$\mathcal{K}_i = \alpha_i(e_k) \qquad 5.16$$

$$\mathcal{K}_d = \alpha_d(\Delta^2 e_k) \qquad 5.17$$

It follows from equation (5.14) that the incremental non-linear fuzzy PID controller can be conveniently be described by equation of the form

$$\Delta u_k = \mathcal{K}_p \Delta e_k + \mathcal{K}_i e_k + \mathcal{K}_d \Delta^2 e_k \qquad 5.18$$

It is important to note that the fuzzy PID controller has been implemented in the incremental form to make use of the dual zone tuning technique, and to avoid any bumpless transfer techniques associated with the integral state. The diagram in figure (5.2) shows a block diagram representing the control system using the fuzzy logic to represent the gain functions for the non-linear incremental PID controller.

The new proposed fuzzy controller in this chapter has the following feature:

1) lower number of parameters than conventional FLC;

2) higher performance than conventional PID;

3) easier to tune than conventional FLC.

This controller will also benefit from the dual zone design methodology introduced in chapter 3. To be able to achieve the above requirements, it proposed to use the fuzzy logic to map the non-linear gain functions, in the same manner as has already been introduced in the previous chapters i.e. using the fuzzy logic to map a non-linear gain functions for non-linear controllers.

In the case of non-linear controllers, where the non-linearity is a function of

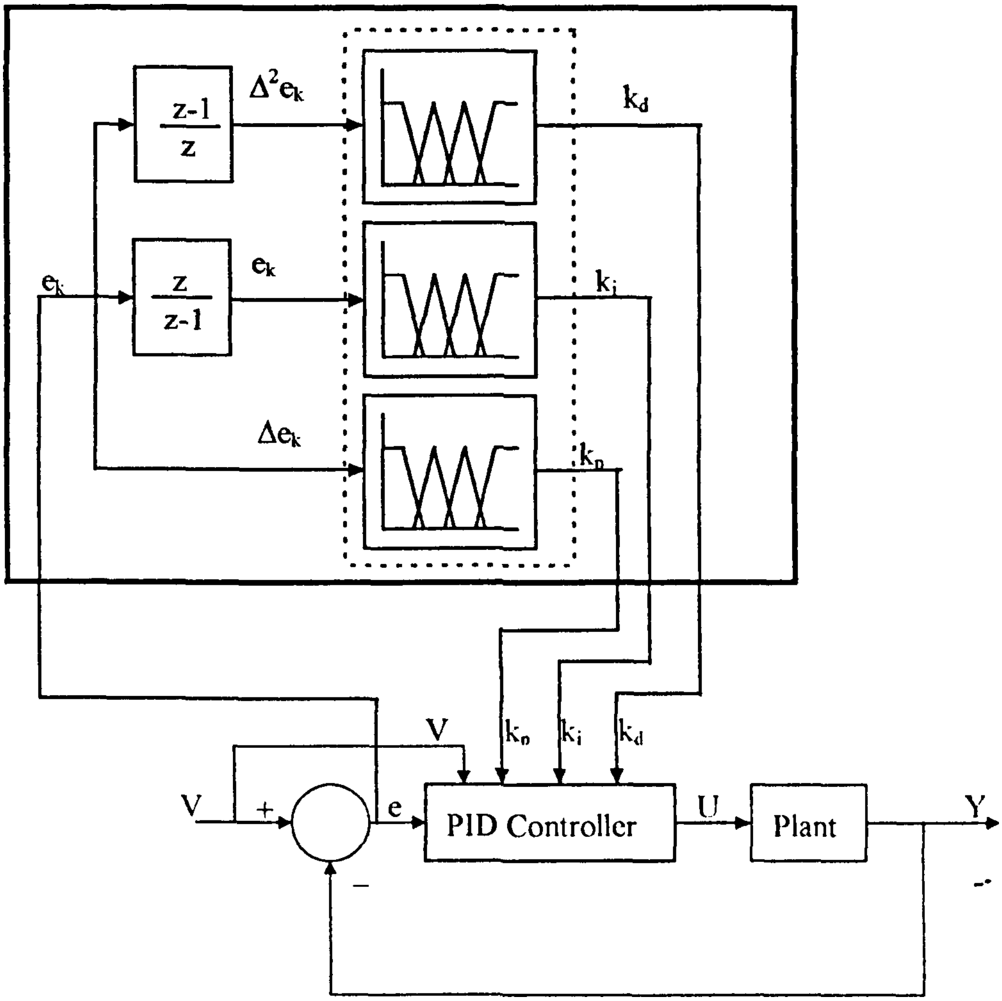


Figure 5.2 System Block Diagram

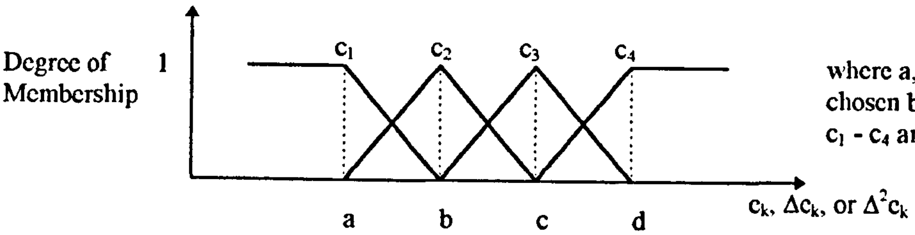
the first order backward difference in error, error, and second order backward difference in error, one strategy for achieving this goal is to choose the gains $\mathcal{K}_p$, $\mathcal{K}_i$ and $\mathcal{K}_d$ as a function of the first order backward difference in error, error, and second order backward difference in error respectively. In this chapter the gains $\mathcal{K}_p$, $\mathcal{K}_i$, and $\mathcal{K}_d$ are designed as fuzzy gain profiles for the proportional, integral, and derivative gains respectively. Therefore the gain functions of the controller given by equation (5.18), are to be mapped as fuzzy logic functions representing the proportional, integral, and derivative gains respectively.

The proposed fuzzy controller use the fuzzy methodology to map the non-linear PID controller gains. In this controller the fuzzy sets are used to represents the non-linear PID gains, each gain is considered separate in the {*fuzzification*} process .

Also the output of each of the PID gains are also found separately by the {*defuzzification*} process. Once the fuzzy gain functions are found they are introduced into the incremental PID controller to produce a fuzzy non-linear PID controller.

5.3.1 FUZZIFICATION IN THE FUZZY PID CONTROLLER

In the case of non-linear controllers, where the non-linearity is a function of the error, first order backward difference in error, and second order backward difference in error, one strategy of achieving the fuzzification process is by designing three sets of fuzzy rules which map the proportional, integral, and derivative gains against first order backward difference in error, error, and second order backward deference in error respectively. The fuzzy rules would comprise of fuzzy sets relating the first order backward difference in error, error, and second order backward difference in error to the proportional integral, and derivative gains respectively. In this case, it is necessary to map the first order backward difference in error, error, and second order backward difference in error into a collection of fuzzy sets of the form



where a,b,c, and d are centroids chosen by the GA, c_1 - c_4 are centroids

Figure 5.3 Fuzzification of gain function

5.3.2 INFERENCE ENGINE

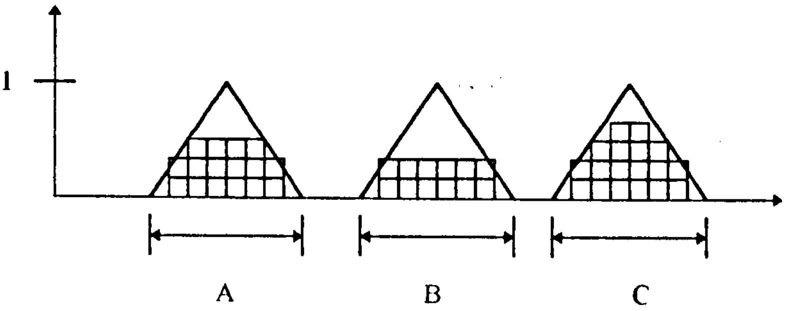
Because of the structure of the proposed fuzzy PID controller, the inference mechanism is essentially an interpolation routine, and as such it dose not require the use of the Max[Min[function]] operator.

5.3.3 DEFUZZIFICATION

The purpose of the defuzzification process is to extract quantitative value from the fuzzy output set which is then used as control signal. The centroid method has been deployed in this research, This method selects the element corresponding to the centre of the area under the curve described by the fuzzy set membership function. Mathematically this is expressed as

$$x_c = \frac{\sum\limits_i x_i m_i}{\sum\limits_i x_i} \qquad 5.19$$

where m_i is the centre of the area of the ith reference set.



where A, B, and C are equal distances

Figure 5.4 Defuzzification of fuzzy sets

5.4 GENETIC DESIGN OF A MULTIPLE ZONED FUZZY LOGIC CONTROLLER

In order to use genetic algorithms to design a dual-zoned fuzzy PID controllers, the Parameters associated with the dual-zoned fuzzy controller have to be precisely defined. It is evident from previous chapters that the behaviour of dual controller can be split into two distinct zones.

The zones are defined as the set-point zone, and the tracking zone. The parameters for the controller in the two zones are thus designed together. The performance of the dual-zoned controller will be contrasted by considering a number of plants controlled by both the genetically designed dual-zoned neural controller, and the genetically designed linear PID controller.

5.4.1 PARAMETERS FOR THE SET POINT ZONE

The parameters in this zone are chosen in exactly the same as was introduced in chapter 3 section 3.2.3.1 i.e. the GA will search the space for two feed forward gains for the set-point zone.

5.4.2 PARAMETERS FOR THE TRACKING ZONE

In the tracking zone it was found that for the proportional and derivative gains the controller operate in a very small region close to zero, and from the equations of the input parameters for the tracking zone it can be seen that the controller must have individual gain values for the proportional, integral, and derivative gains.

To design the dual zone fuzzy controller the fuzzy gain function introduced earlier in this chapter as in equation (5.19), will be used to produce gain function for the gains required to produce a non-linear incremental non-linear fuzzy PID controller. The gain functions produced will map the whole tracking zone for each of the gains (i.e. proportional, integral, and derivative). In order to use GAs to select the tuning parameters in such a way that as to produce satisfactory response in the case of a step input, it is only necessary to encode the elements of the tracking zone gain functions, plus the feed forward gains needed for the set point change zone, as binary strings. The GA will be used to generate the location of the centroid of the gain function membership functions (c_i). The binary string would be represented as

$$\{ \underset{\text{proportional}}{(c^p_1 \ldots\ldots c^p_N)} \quad \underset{\text{integral}}{(c^i_1 \ldots\ldots c^i_N)} \quad \underset{\text{derivative}}{(c^d_1 \ldots\ldots c^d_N)} \quad \underset{\text{set point change}}{(F_0 \; F_1)} \}$$

where the whole string contains the three gain functions elements, plus the feed forward gains.

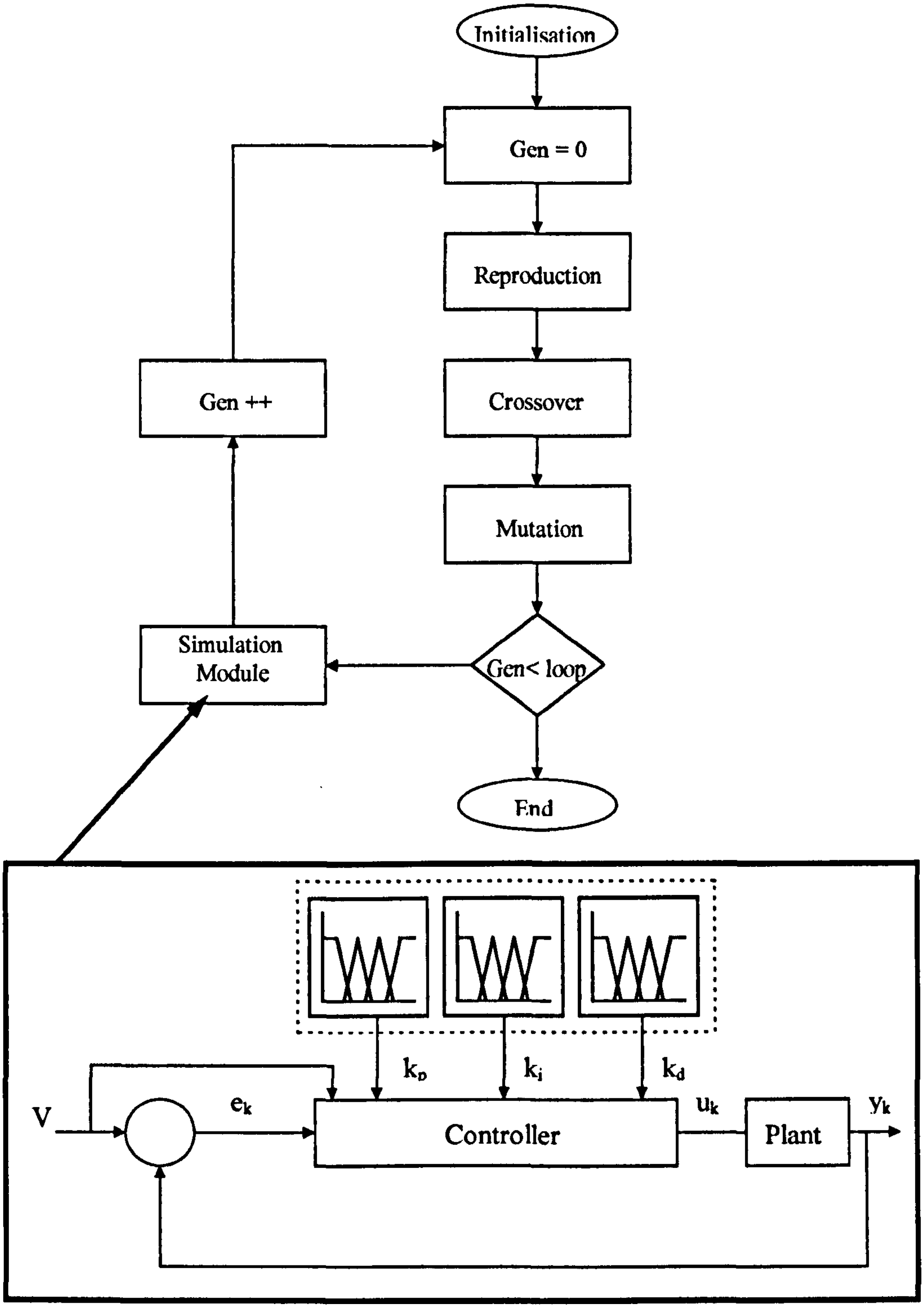


Figure 5.5 Sequence of GA's using fuzzy logic to map gain functions

Random initialization is the approach used to initialize the initial population. The system incorporates both the linear plant, and the non-linear fuzzy PID controller, the controller is designed by randomly generated sets of tracking zone gain elements, and feed forward gains by the GA's. A simulation test is then carried out on all the controllers to test the local stability. In the case of a violation of stability in any of the cases, the randomly generated set of tracking zone gains elements, and feed forward gains will not be included in the initial population. This type of initialization is essential in this type of design, since otherwise their would be a danger of creating an initial population which many of its members violate the constraints on the controllers being designed, figure (5.5) shows the sequence of genetic algorithms. Following the initialization, the objective function is introduced and its value is calculated using decoded values of the parameters in each string. Thus for example, if minimum ISE is regarded as the ultimate design requirement, genetic algorithms can be readily used to select the best set of tracking gains elements, and the feed forward gains such that the ISE is minimised.

In the genetic design of non-linear PID controllers the plant under consideration is subjected to a command input (i.e. unit step), then the performance index is computed for the plant. therefore, for each member in the population the function

$$ISE = \sum_{j=1}^{j=N} e^2{}_j \qquad 6.20$$

is evaluated, where

$$e_j = v - y_j \quad \text{and} \quad N = \frac{\tau}{T}$$

and $e_j \in \Re$, is the error signal, $y_j \in \Re$,is the output signal, T is sampling time, and τ is an appropriately chosen settling time.

Minimization of this performance index over the entire population can be rapidly obtained by using the genetic operations of reproduction, crossover, and mutation. It is interesting to note that the conditions for the existence of a non-linear PID controllers for the plant under consideration will be automatically satisfied by the genetic algorithms. This is obvious, because in the case of violation of the constrained the corresponding set of tracking zones gains elements and the feed forward gains produces large ISE and the result will be zero or low fitness, and by the action of the selection operator it would not be chosen for the next generation.

5.5 ILLUSTRATIVE EXAMPLES

In order to use the fuzzy logic to map the non-linear gain functions for the non-linear incremental PID controller, the number of fuzzy rules used in each gain function has to be defined. In this chapter it is proposed to use 4 rules for each of the gain functions. Hence it can be seen from the above analysis that the proportional gain function can be found by mapping the change in tracking error Δe_k into $\mathcal{K}_p$.

The gain functions will be used in the incremental controller given in equation (5.18) to produce a fuzzy PID controller. The centroid in the gain functions are encoded in the GA as binary strings, The GA will search and produce centroid that give the best ISE for the controller. The GA binary string is given by:

$$\{ \ \underset{\text{proportional}}{(c^p_1 \ldots\ldots c^p_4)} \quad \underset{\text{integral}}{(c^i_1 \ldots\ldots c^i_4)} \quad \underset{\text{derivative}}{(c^d_1 \ \ldots\ldots \ c^d_4)} \quad \underset{\text{set point change}}{(F_0 \ \ F_1)} \ \}$$

As it can be from the string their are 4 centroid (parameters) needed for each of the gain functions, which means that the GA will be searching for 14 Parameters in total. The parameters are 12 centroid for the three gain functions in the tracking zone, plus the two feed forward gains needed for the set-point zone.

5.5.1 PLANT 1

The procedure for the tuning of genetic control systems can be conveniently illustrated by designing a genetic non-linear PID control (GNC) system for the open loop SISO plant with transfer function of the form

$$g(z) = \frac{0.8}{z^4(z-0.9)} \tag{5.21}$$

the sampling period is 0.1 sec.

The Arma model for the plant is of the form

$$y_k = a_0 y_{k-1} + b_0 u_{k-5} \qquad 5.22$$

The plant variables are given by

$a_0 = 0.9$

$b_0 = 0.8$

where the incremental PID controller is given by equation (5.18). The controller is implemented in the incremental form so as, to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. The controller is designed using the dual zone method introduced earlier.

5.5.1.1 DUAL-ZONED FUZZY CONTROLLER

Initially the gains $\mathscr{K}_p$, $\mathscr{K}_i$ and $\mathscr{K}_d$ are chosen as a function of the plant error such as that given by equations (5.15, 5.16, 5.17), i.e. dual zoned fuzzy PID controller. Then the controller was designed by means of genetic algorithms, so as that the Integral square error is minimised for the plant.

In this case, a population of 100, a crossover probability, $P_c=0.65$, and a mutation probability, $P_m = 0.01$, was used, also this case the maximum value for both the first and second order backward diference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design. Figure 5.6 shows the transient response of the genetically designed controller.

5.5.1.2 LINEAR CONTROLLER

To contrast the deference between linear and non-linear PID controllers, a genetically designed linear PID controllers was constructed, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, and K_d are integer values chosen by the GAs to minimise the ISE for the fixed controller.

In this case, a population of 100, a crossover probability, $p_c = 0.65$, and a mutation probability, $P_m = 0.01$ was used. Figure 5.7 shows the transient response of the genetically designed controller.

CONTROLLER	Multi-Zoned Fuzzy	LINEAR PID
ISE	5.14	9..16

Table (5.1)

Table (5.2) shows the gains for the multi-zoned non-linear PID controller.

Gains	Zone1	Zone2	Zone3	Zone4
$\mathcal{K}_p$	0.561	1.139	0.682	0.469
$\mathcal{K}_i$	5.033	3.407	3.929	0.000
$\mathcal{K}_d$	0.374	0.243	1.145	0.7

Table (5.2).

Table (5.3) shows the feed forward gains for the non-linear multi-zoned PID controller.

Feed Forwad Gains	4.021	6.157

Table (3.3).

5.5.2 PLANT 2

A second plant was considered, to investigate the effectiveness of the genetic algorithms in designing linear, and non-linear controller. The plant considered has a transfer function of the form:

$$g(z) = \frac{0.03573 + 0.044625\, z}{z^{4}(0.0513423 - 1.4331\, z^{2})} \qquad 5.23$$

the sampling period is 0.1 sec.

The Arma model for the plant is of the form as

$$y_k = a_0 y_{k-1} + a_1 y_{k-2} + b_0 u_{k-5} + b_1 u_{k-6} \qquad 5.24$$

The plant variables are given by,

$a_0 = 1.4331,$

$a_1 = -0.51342,$

$b_0 = 0.044625,$

$b_1 = 0.03573,$

and the incremental PID controller is governed by equation (5.18). The plant was tested under different operating conditions as shown in the examples bellow:

5.5.2.1 DUAL-ZONED FUZZY CONTROLLER

Initially the gains $\mathcal{K}_p$, $\mathcal{K}_i$, and $\mathcal{K}_d$, where chosen as in equations (5.15, 5.16, 5.17), i.e. dual-zoned PID controller with linear plant. The controller was designed by means of genetic algorithms, such that the Integral Square Error ISE is minimised for the plant.

In this case, a population of 100, a crossover probability, $P_c = 0.65$, and mutation probability, $P_m = 0.01$ was used, also this case the maximum value for both the first and second order backward diference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design. Figure (5.8), shows the transient response of the genetically designed non-linear controller for the above plant.

5.5.2.2 LINEAR CONTROLLER

Finally to contrast the deference between linear and non-linear PID controllers, a genetically designed linear PID controller constructed, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, K_d , are integer values chosen by the GA's to minimise the ISE for the fixed controller. In this case, a population of 100, a crossover probability, $p_c = 0.65$, and a mutation probability, $P_m = 0.01$ was used. Figure 5.9 shows the transient response of the genetically designed controller response.Table (5.4) shows the ISEs for both the multi-zoned nueral and the linear PID controllers.

CONTROLLER	FUZZY PID	LINEAR PID
ISE	6.186	10.57

Table (5.4)

Table (5.5) shows the gains for the multi-zoned nueral PID controller.

Gains	Zone1	Zone2	Zone3	Zone4
$\mathcal{K}_p$	0.297	0.722	0.432	0.848
$\mathcal{K}_i$	5.852	2.837	2.374	1.175
$\mathcal{K}_d$	2.661	3.947	3.007	3.492

Table (5.5).

Table (5.6) shows the feed forward gains for the multi-zoned nueral PID controller.

Feed Forwad Gains	2.122	5.682

Table (5.6).

5.5.3 ROBUSTNESS TEST

This test is aimed at finding how robust are the genetically designed dual zone PID controllers are for changes in the plant operating conditions. In this case the plants used in the illustrative example were modified to produce different plants . To do this test consider plant 1 with transfer function of the form

$$g(z) = \frac{0.8}{z^{4}(z-0.9)} = \frac{\beta}{z^{\gamma}(z-\alpha)}$$

To produce the new plants the value of α, β, and γ are changed. The tables (5.7) show the range of plants considered in this robustnes test, and the respective ISEs obtained,

Plant	ISE	β	α	γ
1	5.242	0.8	0.90	5
2	6.363	0.8	0.90	6
3	12.615	1.2	0.90	6
4	30.412	12	0.94	6
5	4.741	1.2	0.94	4
6	10.547	1.2	0.60	4
7	24.429	0.5	0.60	4

Table (5.7)

Figure (5.10 to 5.15) show the response of the plants for the non-linear multi-zoned PID controller designed using plant 1.

Table (5.8) shows the results of the rubstness test using a cliped controller as introduced in chapter 3.

Plant	Lin ISE	CLI-ISE	β	α	γ
1	9.16	5.242	0.80	0.90	5
2	11.76	6.363	0.80	0.90	6
3	13.07	7.555	1.2	0.90	6
4	14.867	9.365	1.2	0.94	6
5	9.976	4.378	1.2	0.94	4
6	13.967	6.440	1.2	0.60	4
7	22.652	17.151	0.50	0.60	4

Table (5.8)

The transient response of the clipped controller robustness tests are shown in figure (5.16-5.21) respectively.

During this research an alternative strategy was also investigated for the fuzzy logic control. This strategy was more in-line with conventional fuzzy logic control. In this case the fuzzy sets were used to produce control outputs directly, i.e. Δu's. These results were in all cases significantly worse than the ones obtained by using the fuzzy sets to map the PID gains

5.6 CONCLUSIONS

In this chapter the dual zone technique was used to design a simple fuzzy PID controller. The fuzzy sets were used to map the non-linear PID gains to produce a genetically designed fuzzy PID controller. It has thus been shown in this chapter that a simple Fuzzy PID controller can produce very effective results. This design shows that the use of fuzzy sets to map the gain functions combined with the use of dual zone technique in a genetically designed controller can benefit from the fuzzy logic ability at a reduced number of parameters for the GA to find compared to a full fuzzy logic controller. Indeed comparing the results obtained by genetically designed dual-zoned fuzzy controller with the results obtained for the genetically designed fuzzy controller [47], it shows that the dual-zoned fuzzy controller designed in this thesis has higher performance, which indicates that the dual zoned fuzzy controller designed in this chapter produce faster tracking and smaller ISE for the plant. The robustness test indicate that the non-linear PID controllers are more robust than the linear controllers. Furthermore the clipped non-linear PID controllers exhibits far better robust properties than the un-clipped non-linear controllers. This is because the clipping forces the controller to be rate limited.

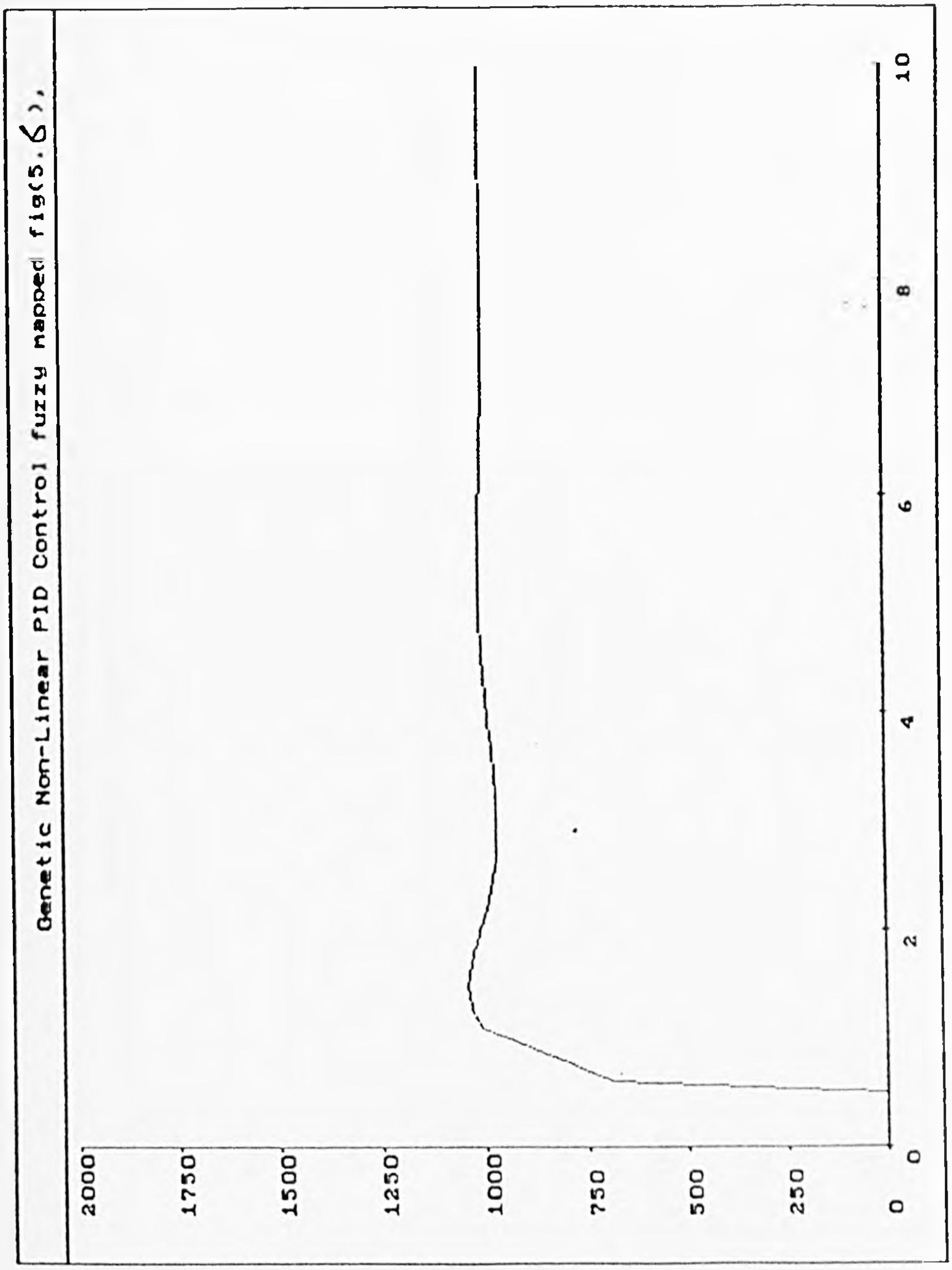


Figure (5.6) Transient response of the genetically designed dual-zoned fuzzy PID controller for plant (1)

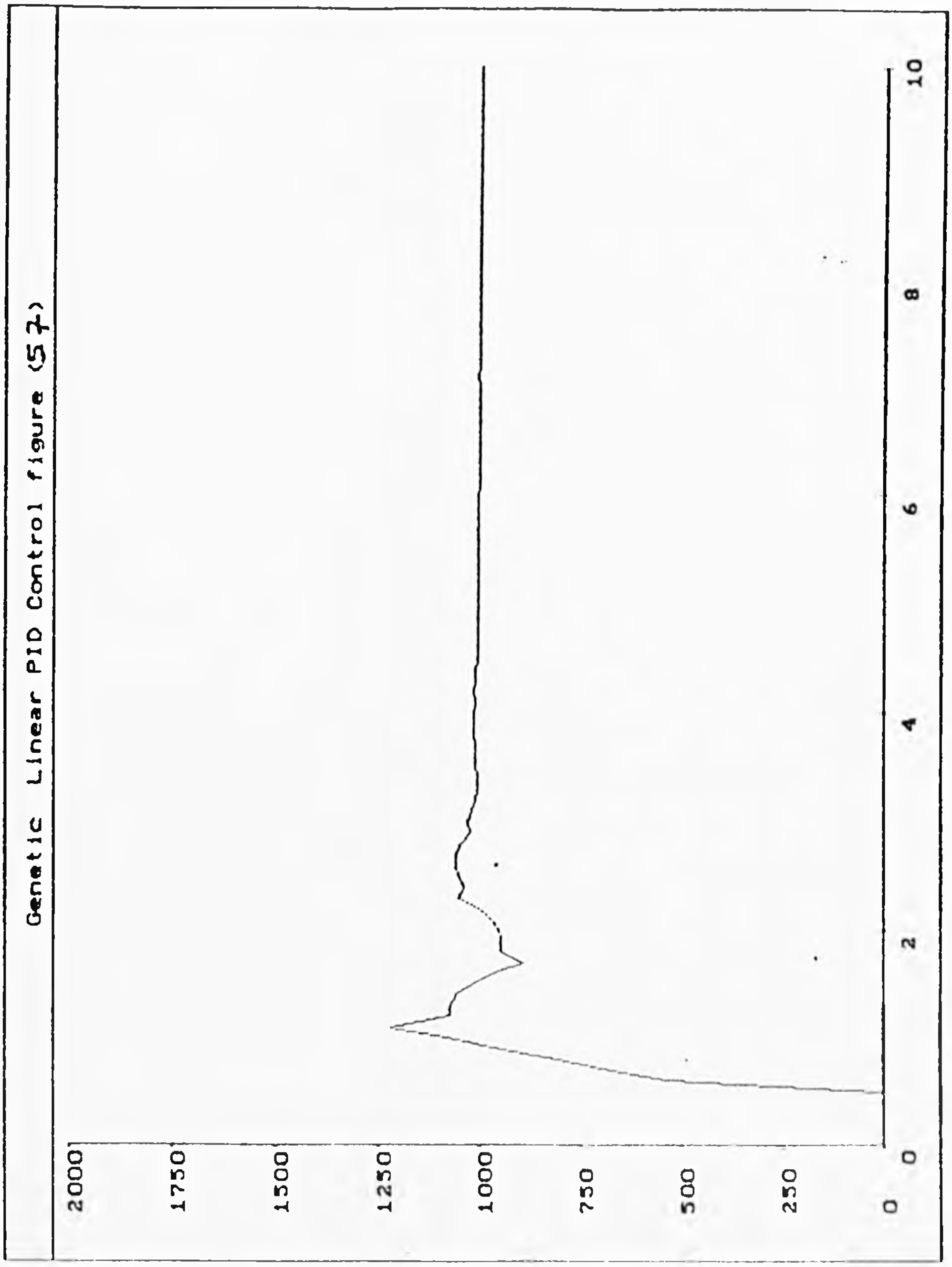


Figure (5.7) Transient response of the genetically designed linear PID controller for plant (1)

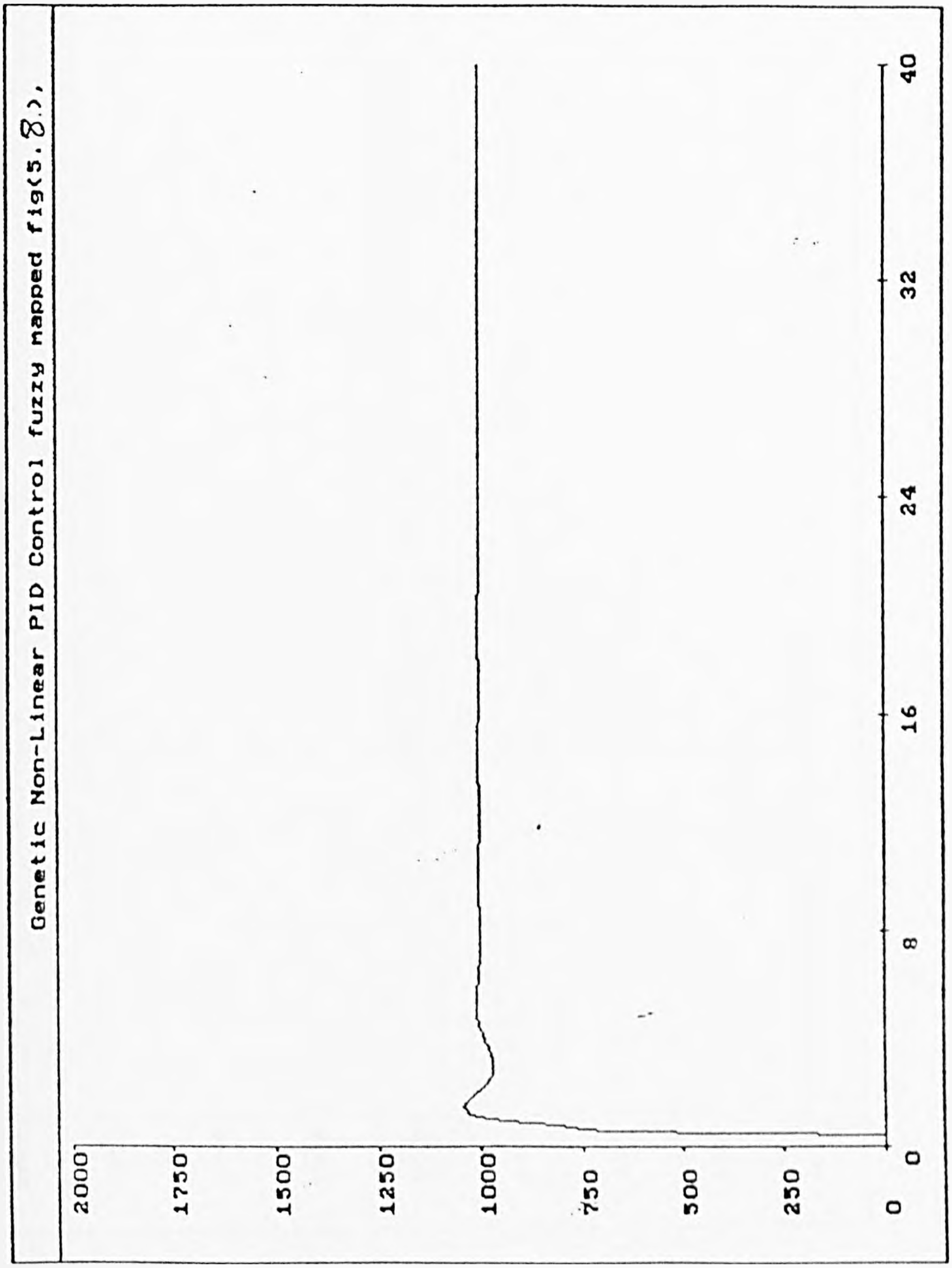


Figure (5.8) Transient response of the genetically designed dual-zoned fuzzy PID controller for plant (2)

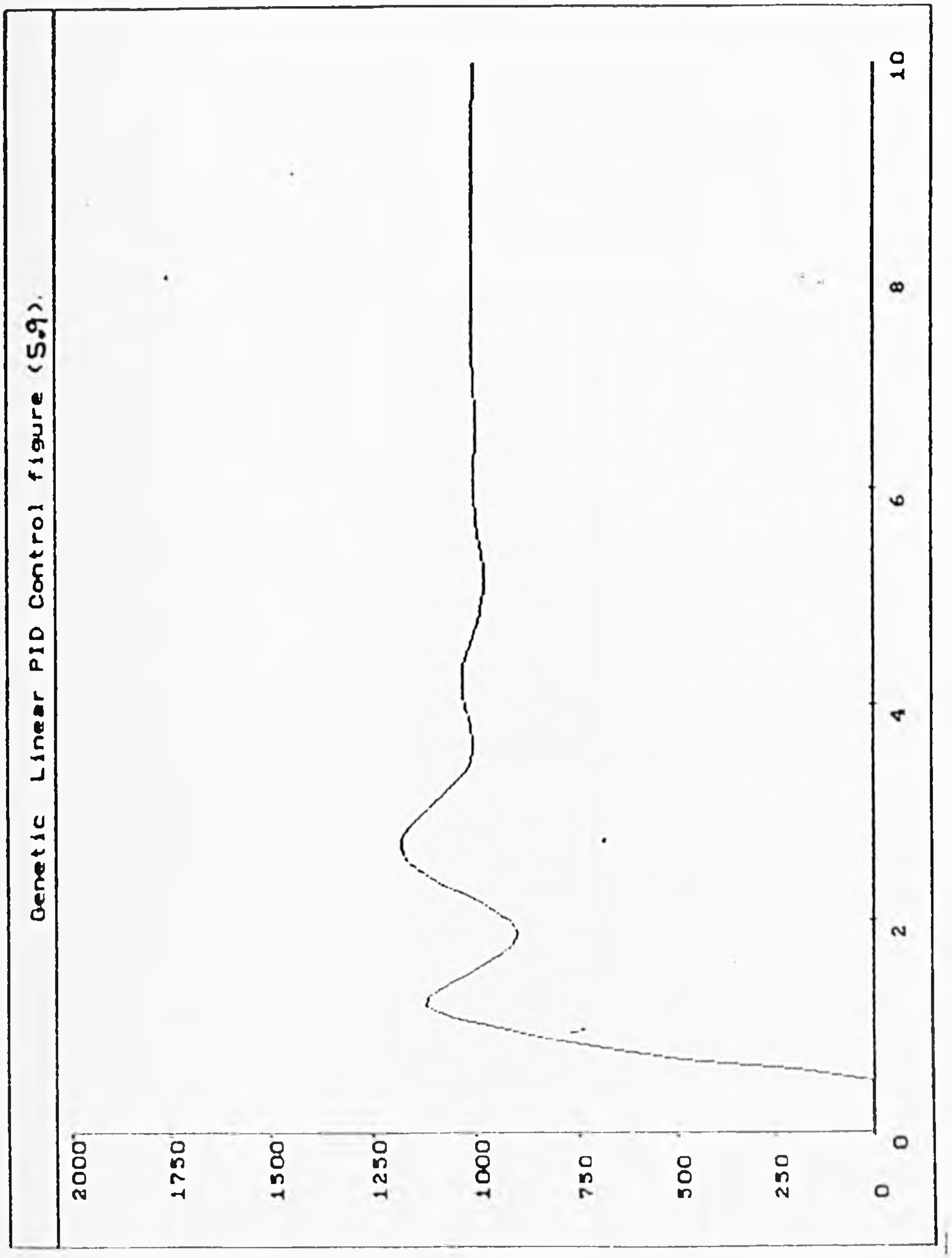


Figure (5.9) Transient response of the genetically designed linear PID controller for plant (2)

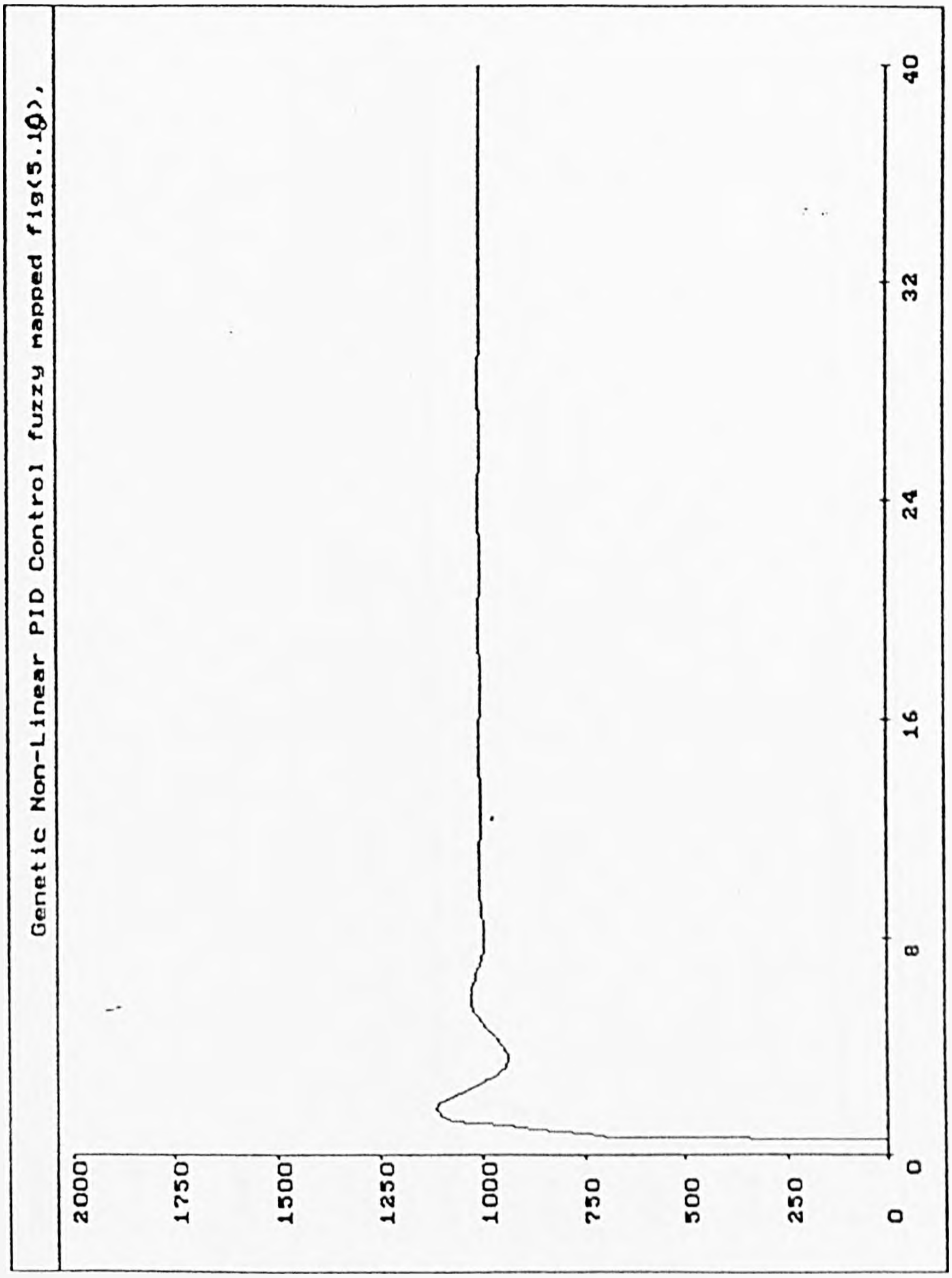


Figure (5.10) Transient response of robustness test using genetically designed non-linear PID controller for plant (2), table (5.7).

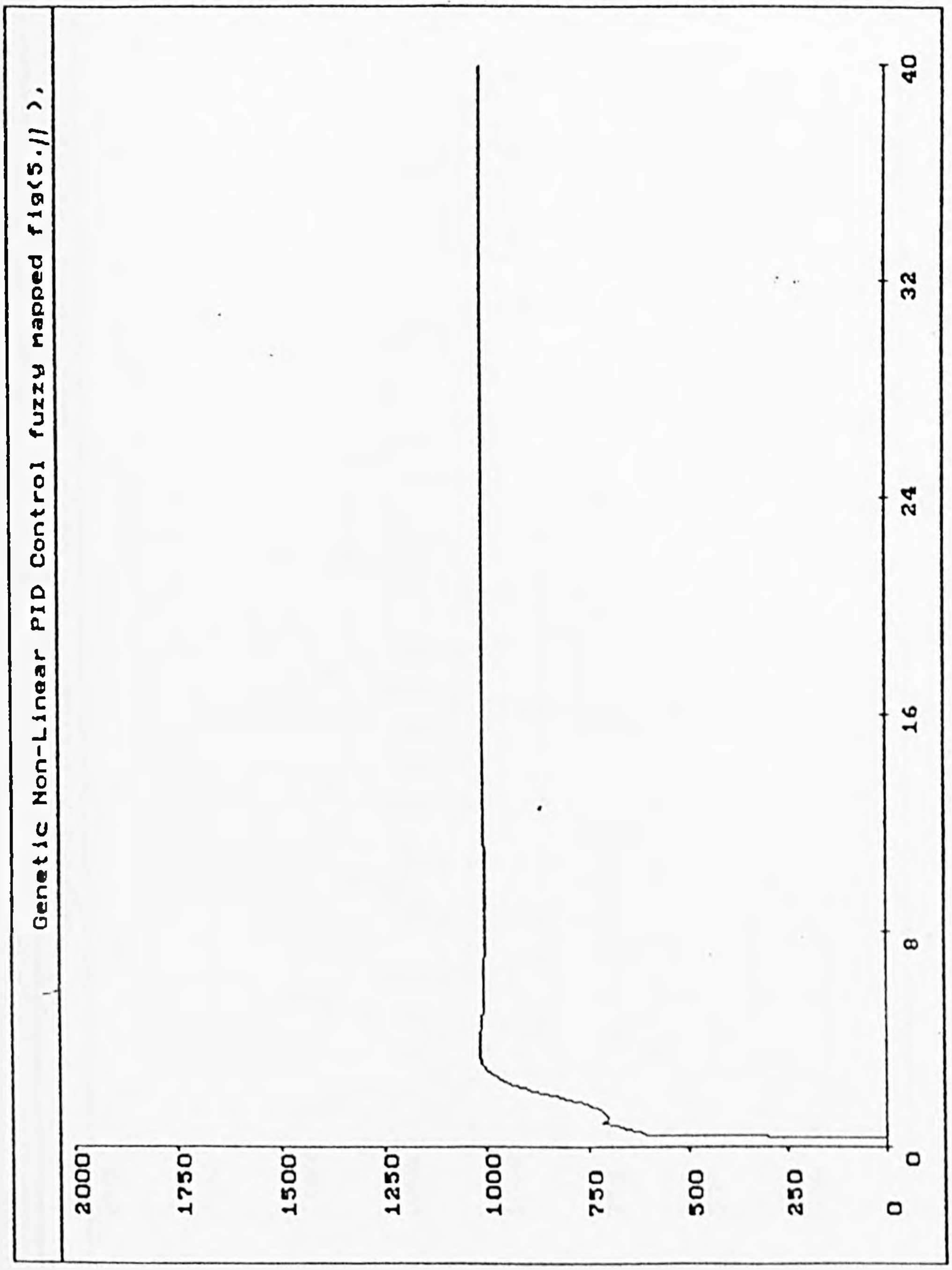


Figure (5.11) Transient response of robustness test using genetically designed non-linear PID controller for plant (3), table (5.7).

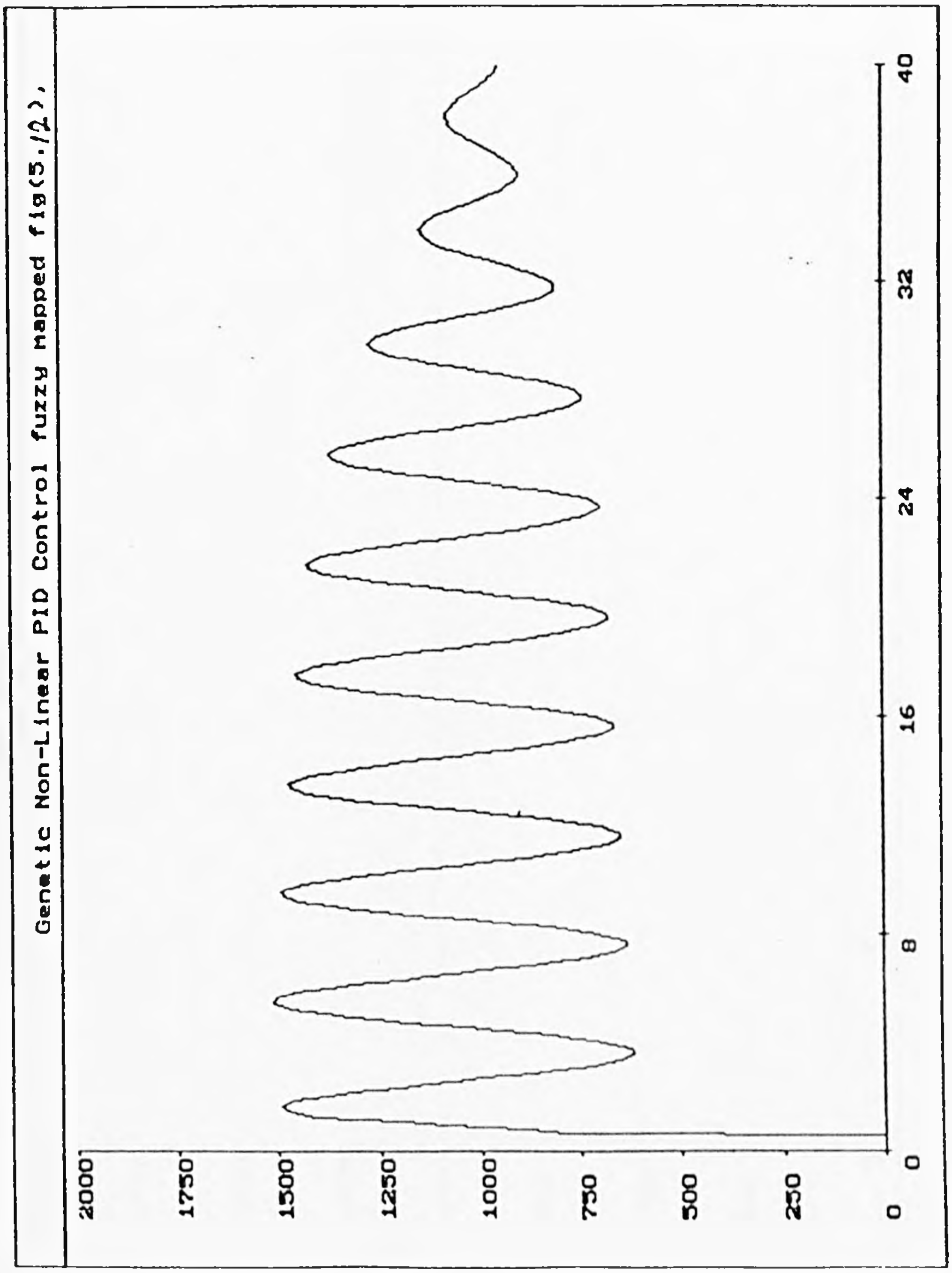


Figure (5.12) Transient response of robustness test using genetically designed non-linear PID controller for plant (4), table (5.7).

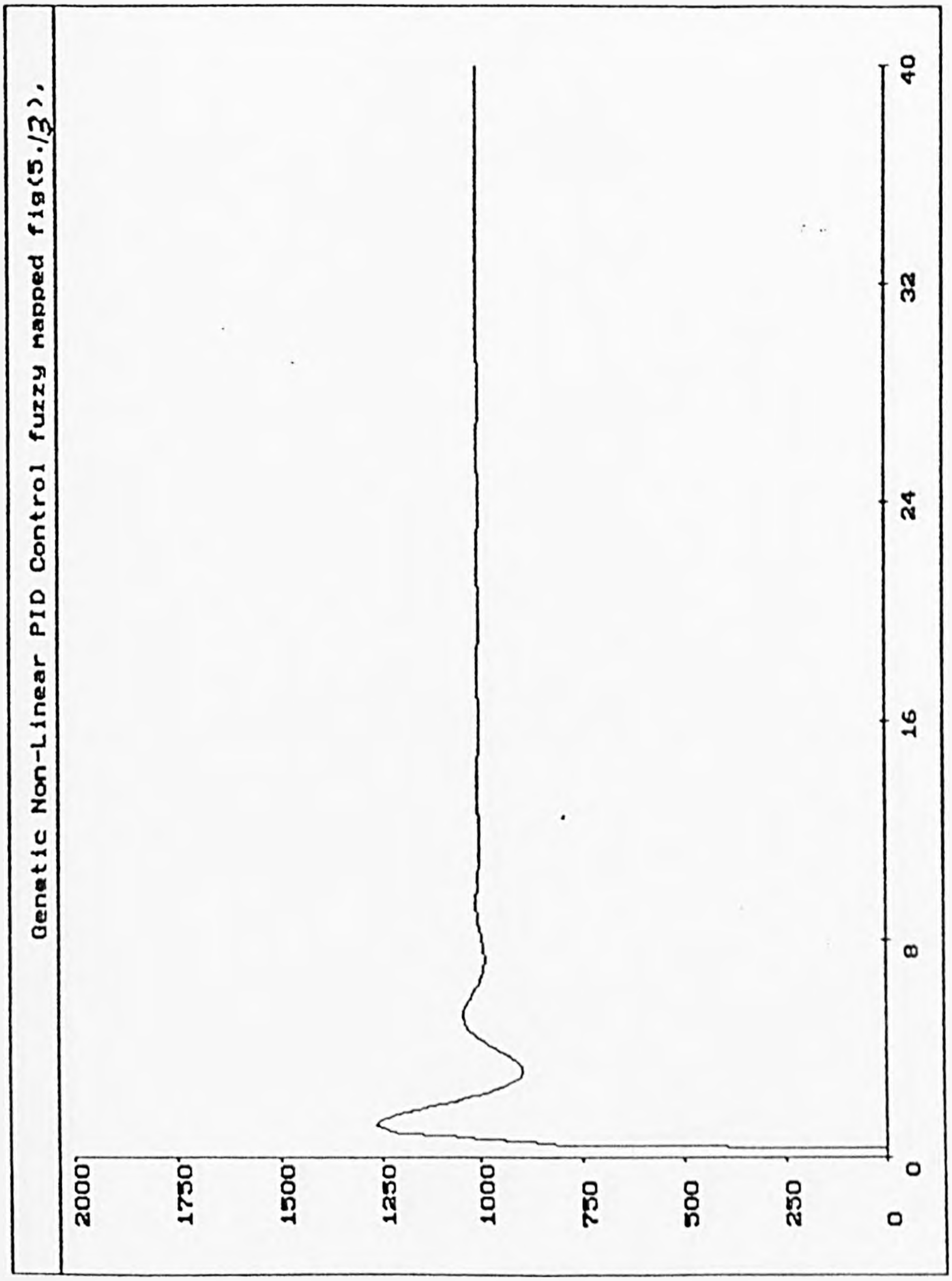


Figure (5.13) Transient response of robustness test using genetically designed non-linear PID controller for plant (5), table (5.7).

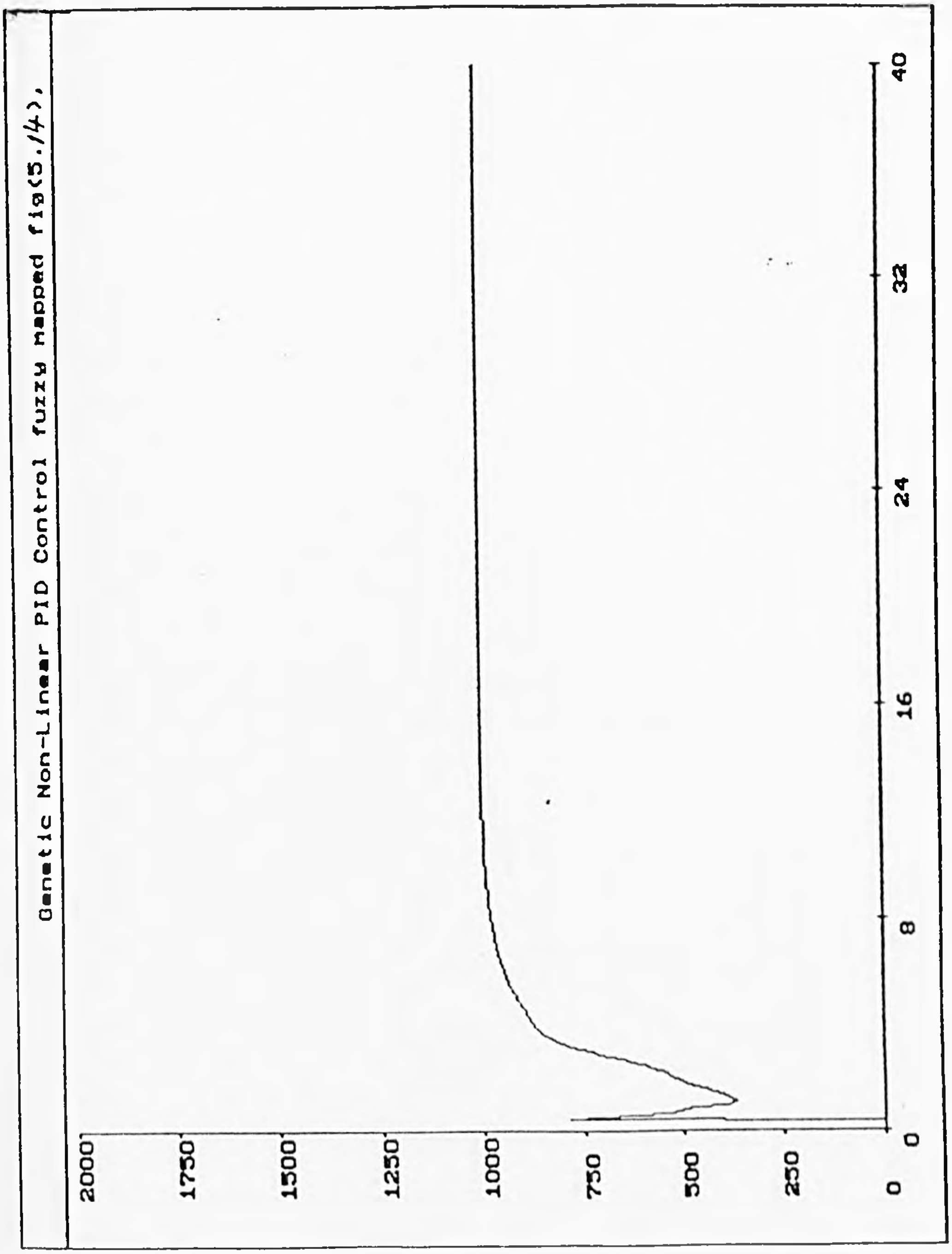


Figure (5.14) Transient response of robustness test using genetically designed non-linear PID controller for plant (6), table (5.7).

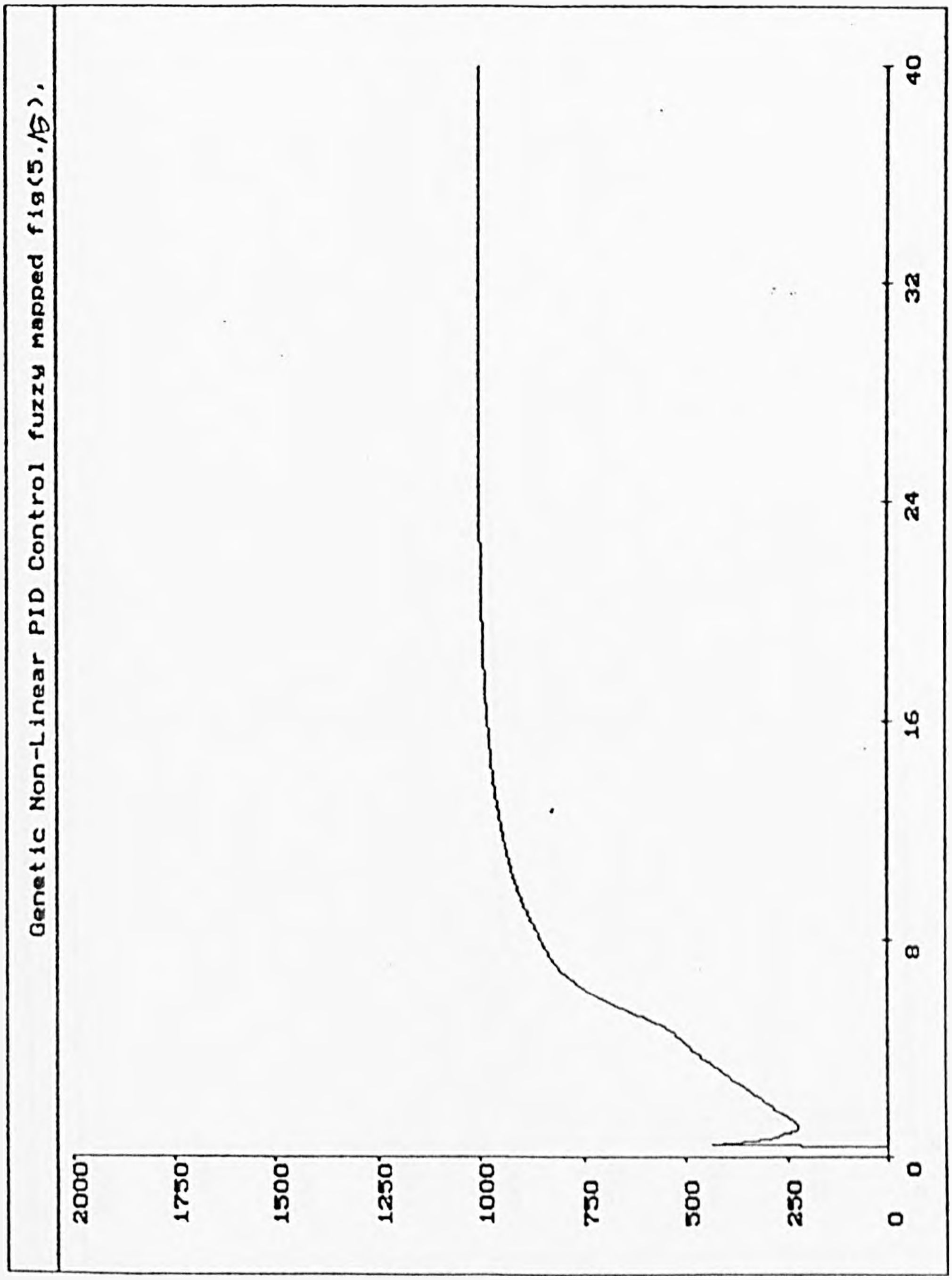


Figure (5.15) Transient response of robustness test using genetically designed non-linear PID controller for plant (7), table (5.7).

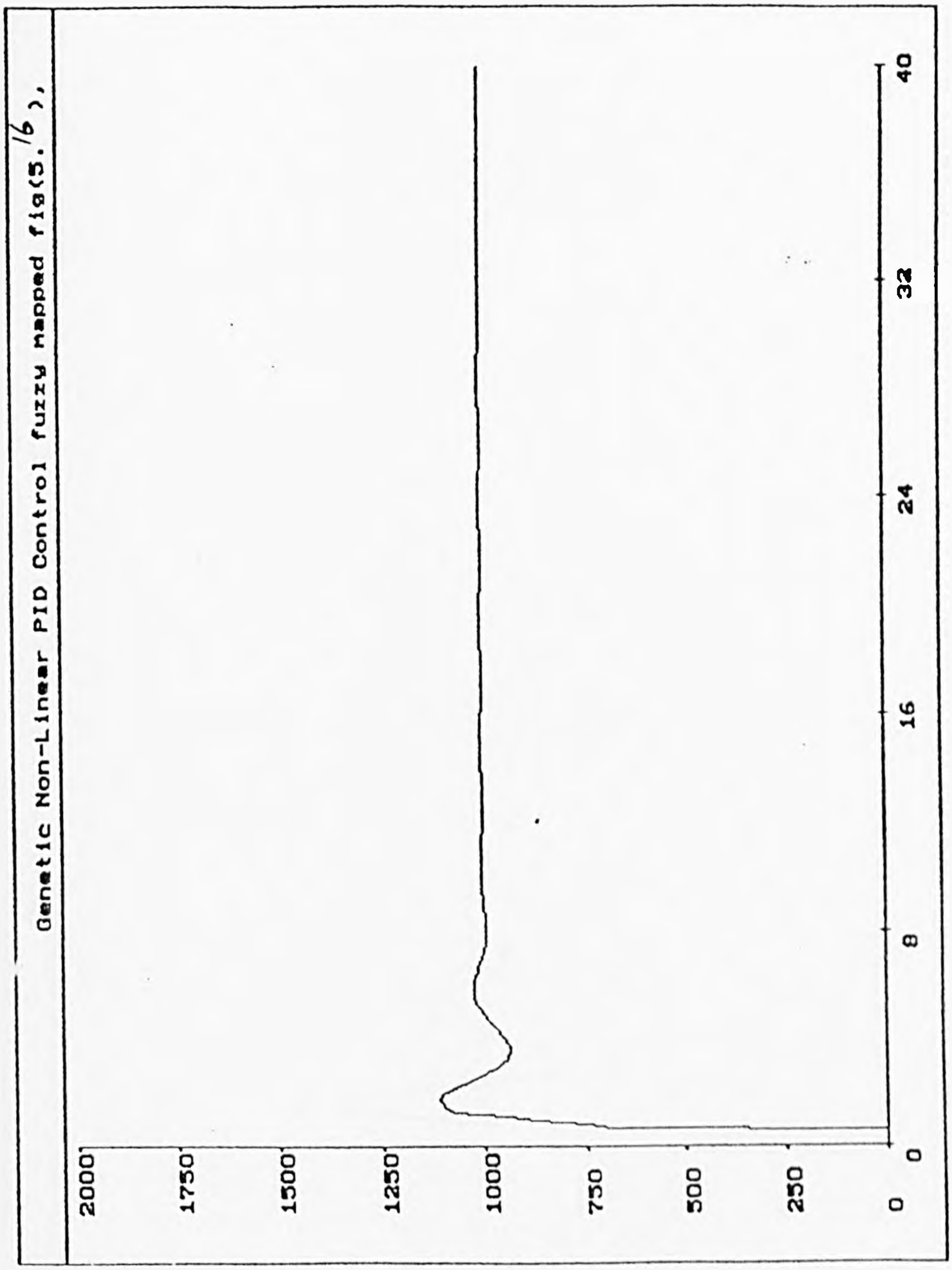


Figure (5.16) Transient response of robustness test using genetically designed non-linear clipped PID controller for plant (2), table (5.8).

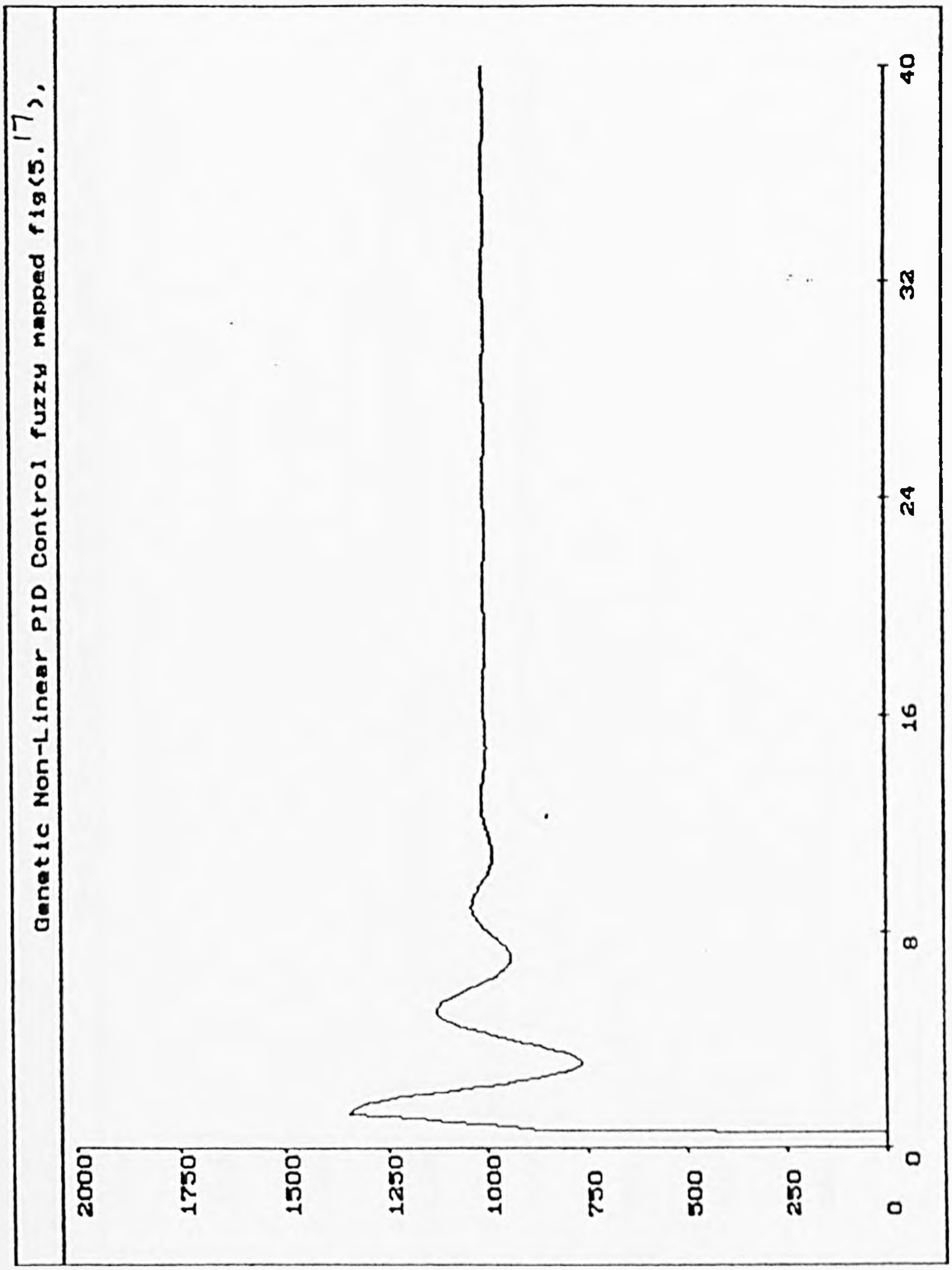


Figure (5.17) Transient response of robustness test using genetically designed non-linear clipped PID controller for plant (3), table (5.8).

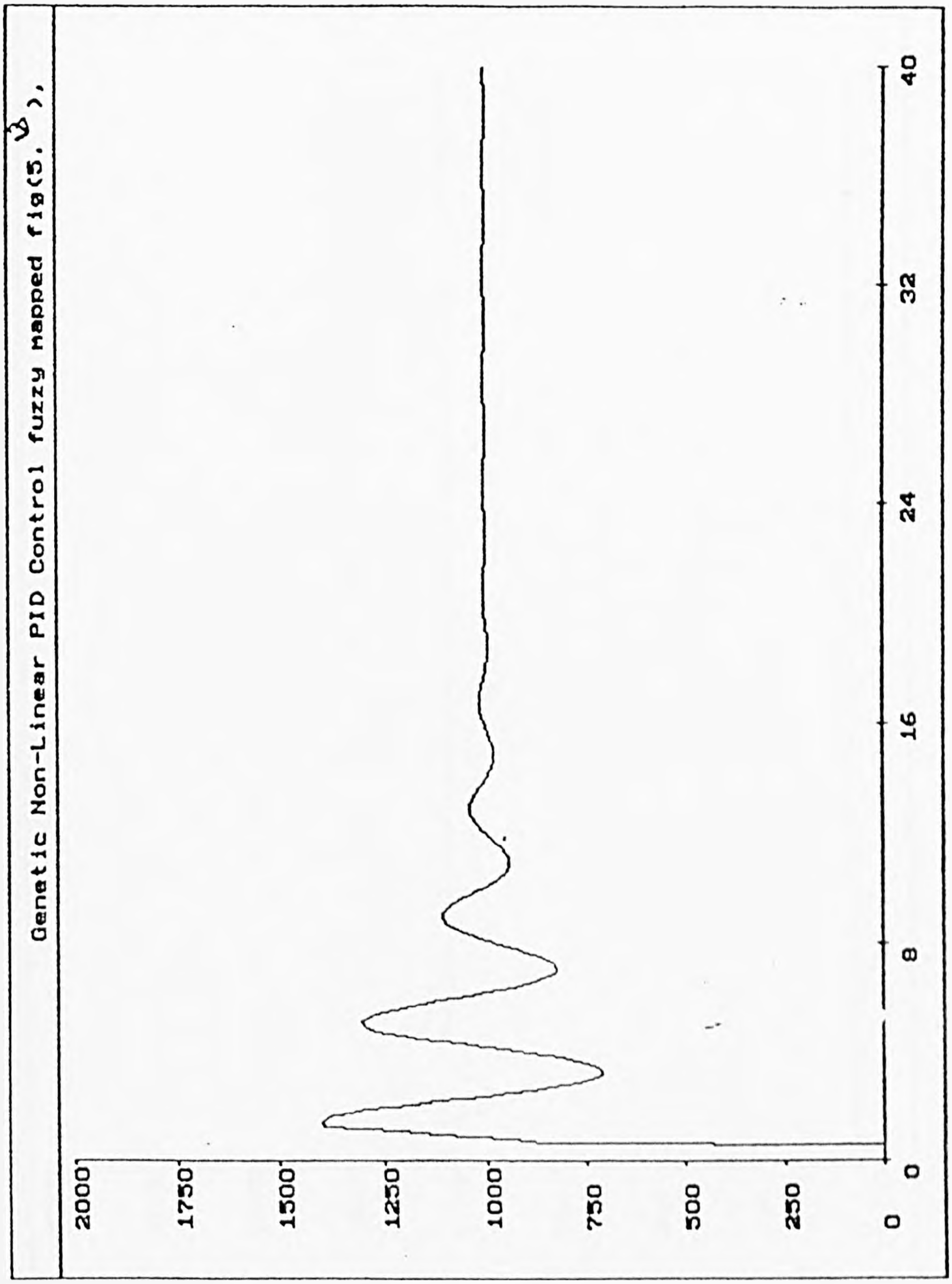


Figure (5.18) Transient response of robustness test using genetically designed non-linear clipped PID controller for plant (4), table (5.8).

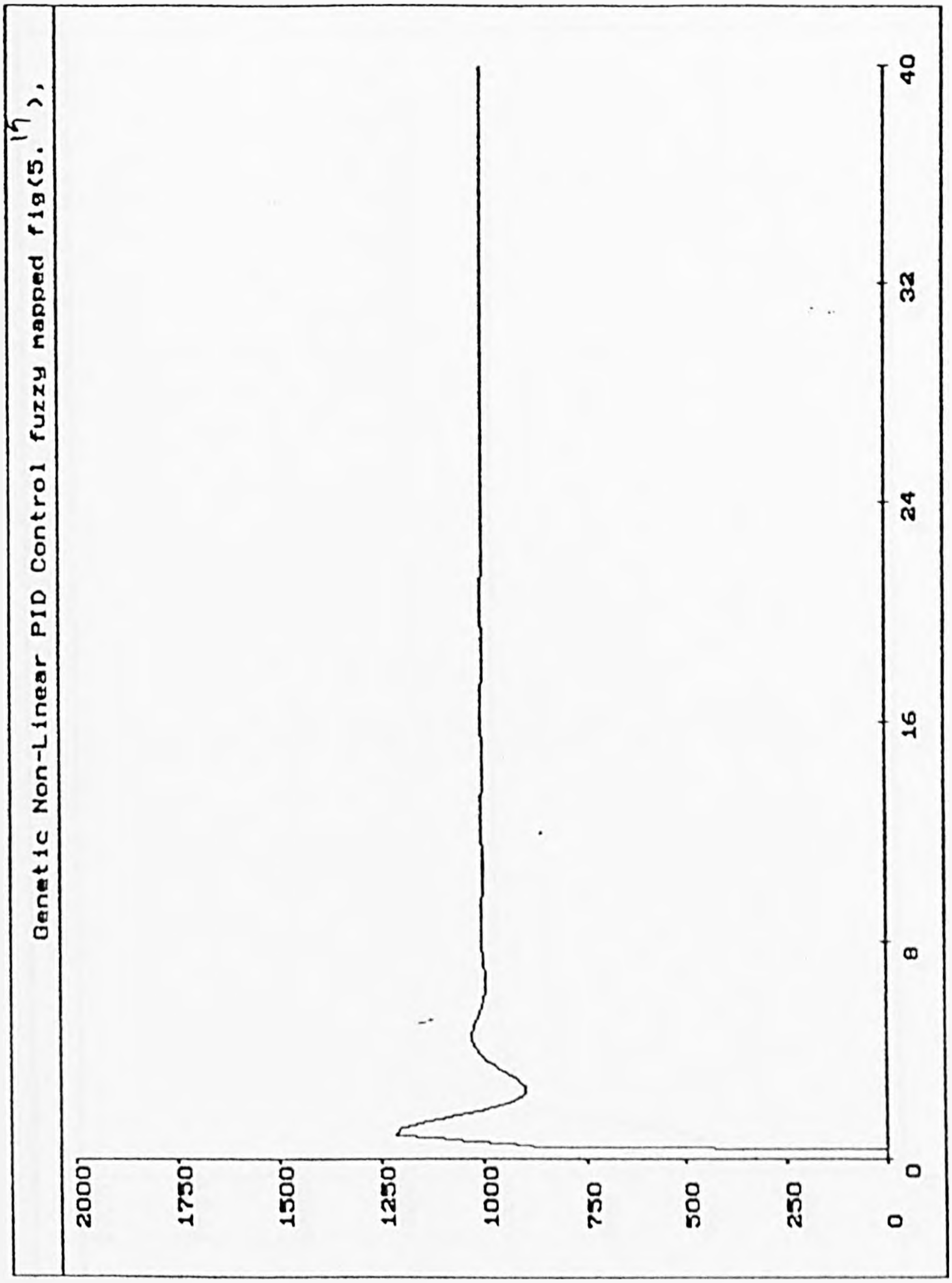


Figure (5.19) Transient response of robustness test using genetically designed non-linear clipped PID controller for plant (5), table (5.8).

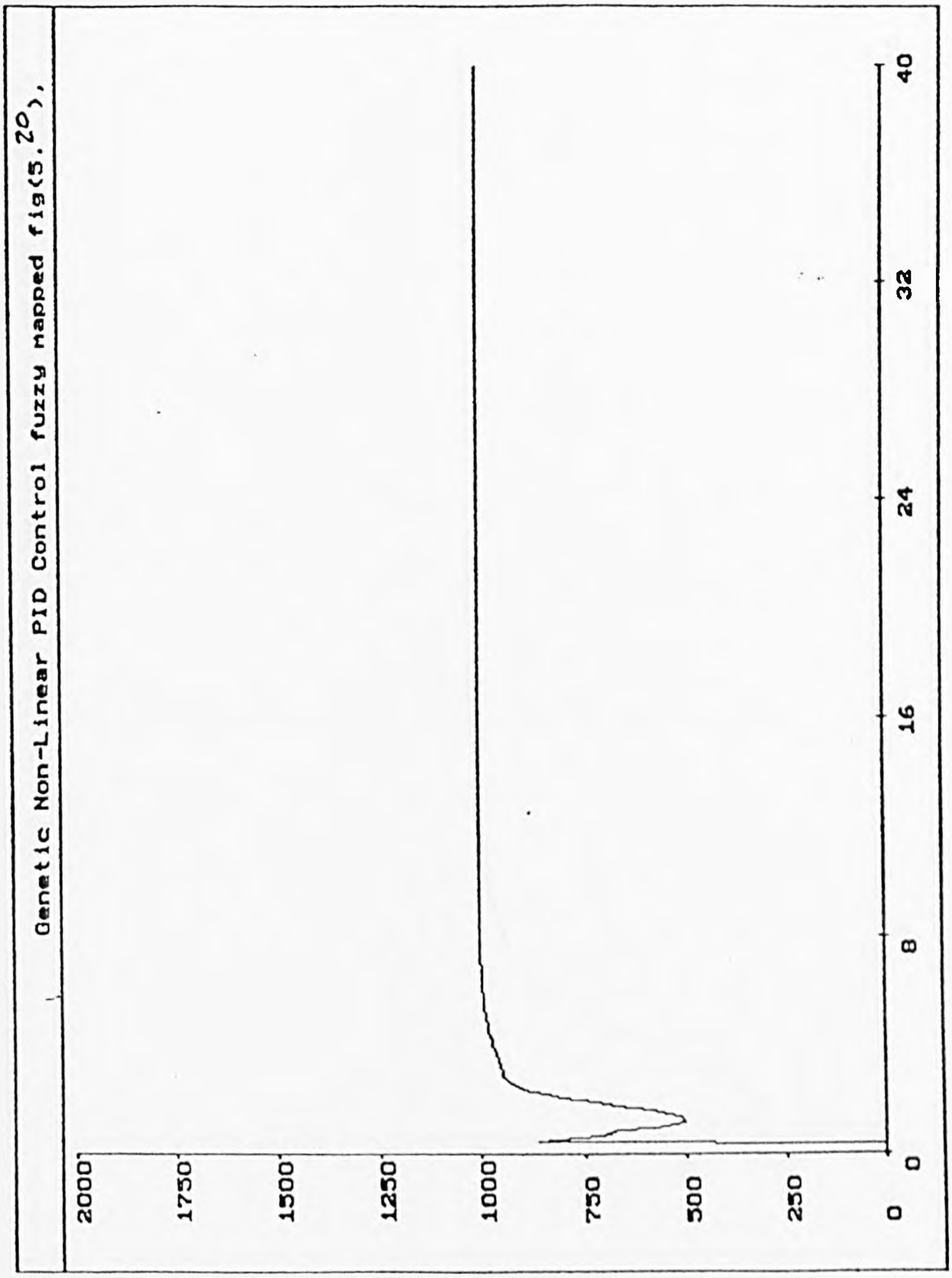


Figure (5.20) Transient response of robustness test using genetically designed non-linear clipped PID controller for plant (6), table (5.8).

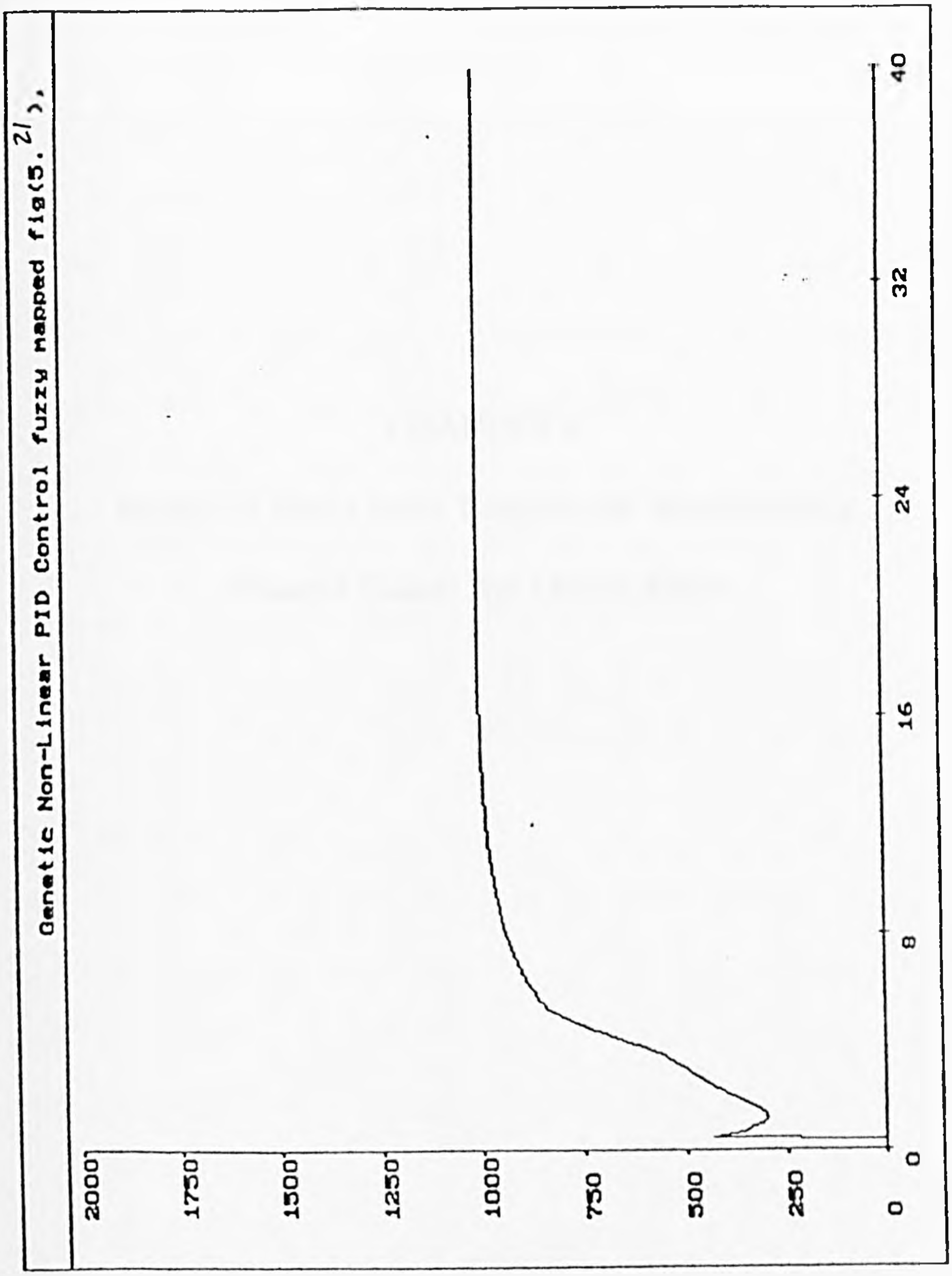


Figure (5.21) Transient response of robustness test using genetically designed non-linear clipped PID controller for plant (7), table (5.8).

CHAPTER 6

Design of Non-Linear Controllers Incorporating "Neural Gains" for Linear Plants

Chapter 6
DESIGN OF NON-LINEAR PID CONTROLLERS INCORPORATING "NEURAL GAINS" FOR LINEAR PLANTS

6.1 INTRODUCTION

The study of neural networks is an attempt to understand the functionality of the brain. In particular it is of interest to define an alternative 'artificial' computational form that attempts to mimic the brain's operation in one or a number of ways. In the last few years interest in the field of neural networks has increased considerably, due partly to a number of significant break-through in research on network types and operational characteristics, but also because of some distinct advances in the power of computer hardware which is readily available for net implementation. It is worth adding that much of the recent drive has, however, arisen because of numerous successes achieved in demonstrating the ability of neural networks to deliver simple and powerful problem solutions, particularly in the field of learning and pattern recognition, both of which have proved to be difficult areas for conventional computing.

Digital computers provide a media for well defined, numerical algorithm processing in a high performance environment. This is in direct contrast to many of the properties exhibited by biological neural systems, such as creativity, generalisation and understanding. However computer-based neural networks, both of hardware and software forms, at the present time provide a considerable move forward from digital computing in the direction of biological systems, indeed several biological neural system properties can be found in certain neural network types. This move is supported by a number of novel practical examples, even though these tend to be in fairly scientific areas, e.g. communication processing and pattern recognition. Because of its inter-displenaery bases encompassing computing, electronics, biology, neuropsychology etc, the field of neural networks attracts a variety of interested researchers and implementers from a broad range of backgrounds [41][45][46][68][69][70]. In this chapter the neural networks is used to map PID gains as non-linear function, the gains are mapped separately for each function then they are introduced into the Dual zoned incremental PID controllers.

6.1.1 NEURAL NETWOKS AND CONTROL

The use of neural network for system control and signal processing has been well accepted. The most noticeable applications are in the area of telecommunication, active noise control, pattern recognition, prediction and financial analysis, process control, speech recognition [96][97]. Such wide spread use of neural network is mainly due to its behavioral emulation to the nature of human brain and its mathematically formulated structure.

6.2 SYNTHESIS

In order to design the digital neural PID controller, it is proposed to use the dual zoned method introduced in chapter 3.

The SISO plants under consideration, are governed on the continuous time set

T=(0,∞], by state and output equation of respective forms

$$\dot{x}(t)=Ax(t)+bu(t) \qquad 6.1$$

and

$$y(t)=cx(t) \qquad 6.2$$

where

$x(t) \in \Re^n$ is the state vector,

$y(t) \in \Re$ is the scalar output from the plant,

$u(t) \in \Re$ is the scalar input to the plant,

$A \in \Re^{nxn}$ is the plant matrix,

$b \in \Re^{nx1}$ is the input matrix,

$c \in \Re^{1xn}$ is the output matrix.

It is assumed that the plant is functionally controllable, so that none of the transmission zero of the plant lies at the origin in the complex plane and therefore that any and all solutions of s in

$$\begin{vmatrix} sI_n-A, & -b \\ C, & 0 \end{vmatrix} = 0 \qquad 6.3$$

are non-zero [Rosenbrock (1974)]. This assumption ensures that rank M = n + 1

[Porter. and Power (1970)] where the system matrix is given by

$$M = \begin{bmatrix} A, & b \\ C, & 0 \end{bmatrix} \qquad 6.4$$

In order to design non-linear PID controllers for SISO linear plants governed by equations (6.1) and (6.2), it is convenient to consider the behaviour of such plants on the discrete-time set $T_T = \{0, T, 2T, \ldots.\}$.

This behaviour is governed by state and output equations of the respective forms [Kwakernaak and Sivan (1972)]

$$Tx_{k+1} = \Phi Tx_k + \Psi Tu_k \qquad 6.5$$

and

$$y_k = \Gamma x_k \qquad 6.6$$

where

$$\Phi = e^{AT} \qquad 6.7$$

$$\Psi = \int_0^T e^{A\tau} b \, d\tau \qquad 6.8$$

and

$$\Gamma = c \qquad 6.9$$

In these equations, $x_kT \in \Re^n$, $u_kT\in\Re$, $y_kT \in \Re$,, $\Phi \in \Re^{nxn}$, $\Psi \in \Re^{nx1}$, $\Gamma \in \Re^{1x}$, and T $\in \Re^+$ is the sampling period.

It is evident from the previous chapters that the linear incremental PID controller can be described as

$$\Delta u_k = T(k_p \Delta e_k + Tk_i e_k + k_d \Delta^2 e_k) \qquad 6.10$$

where

Δu_k is the incremental change in input,

Δe_k is the first order backward difference in error,

$\Delta^2 e_k$ is the second order backward difference in error.

K_p is the current effective value of the proportional gain,

K_i is the current effective value of the integral gain,

K_d is the current effective value of the derivative gain,

T is sampling time.

It is evident from the previous chapters that the error can be written as

$$e_k = v_k - y_k \qquad 6.11$$

It is also evident from the previous chapters that the first order backward difference in error can be written as

$$\Delta e_k = e_k - e_{k-1} \qquad 6.12$$

Further more from the previous chapters the second order backward difference in error can be expressed as

$$\Delta^2 e_k = e_k - 2e_{k-1} + e_{k-2} \qquad 6.13$$

5.2.1 NON-LINEAR INCREMENTAL NEURAL PID CONTROLLERS

The incremental controller given by equation (6.10), can take one of two forms, one is linear, and the other is non-linear. In this chapter the non-linear form is to be investigated. The non-linearities are a function of the plant error.

It follows from equation (6.10), that the incremental non-linear PID controller can be described by equation of the form

$$\Delta u_k = \eta_p(\Delta e_k)\,\Delta e_k + \eta_i e_k + \eta_d(\Delta^2 e_k)\,\Delta^2 e_k \qquad 6.14$$

where

$\eta_p(\Delta e_k)$ is a neural function representing non-linear proportional gain,

$\eta_i(e_k)$ is a neural function representing non-linear integral gain,

$\eta_p(\Delta^2 e_k)$ is a neural function representing non-linear derivative gain.

It is evident from equation (6.14), that the gain functions can be represented by equations of the form

$$\mathcal{K}_p = \eta_p(\Delta e_k) \qquad 6.15$$

$$\mathcal{K}_i = \eta_i(e_k) \qquad 6.16$$

$$\mathcal{K}_d = \eta_d(\Delta^2 e_k) \qquad 6.17$$

The above gain functions are to be mapped as a neural networks gain profiles, as will be illustrated later in this chapter.

It follows from equation (6.14) that the incremental non-linear neural networks PID controller can be conveniently be described by equation of the form

$$\Delta u_k = \mathcal{K}_p \Delta e_k + \mathcal{K}_i e_k + \mathcal{K}_d \Delta^2 e_k \qquad 6.18$$

It is important to note that the PID controller has been implemented in the incremental form to make use of the dual zone tuning technique, and to avoid any bumpless transfer techniques associated with the integral state. Figure (6.1) shows a block diagram representing the control system using the neural networks to represent the gain functions for the non-linear incremental PID controller.

6.2.2 NEURAL NETWORK

A fundamental aspect of artificial neural networks is the use of simple processing elements which are essentially models of neurons in the brain. These elements are then connected together in a well structured fashion, although the strength and nature of each of the connecting links dictates the overall operational characteristics for the total networks. By selecting and modifying the link strengths in an adaptive fashion, so the basis of the network learning is formed along the lines of the previous (Aleksander and Morton, 1990) definition.

One important property of a neural network is its potential to infer and induce from what might be incomplete or non-specific infoımation. This is however also coupled with an improvement in the performance due to the network learning appropriate modes of behaviour in response to problems presented, particularly where real-world data is concerned. The network can, therefore, be taught particular patterns of data presented, such that it can subsequently not only recognise such patterns when they occur again, but also recognise similar patterns by generalisation.

6.2.2.1 NEURAL NETWORK ELEMENTS

In general, neural networks consist of a number of simple node elements, which are connected together to form either a single layer or multiple layers. The relative strengths of the input connections and also the connections between layers are then decided as the network learns its specific tasks. This learning procedure can make

Chapter 6

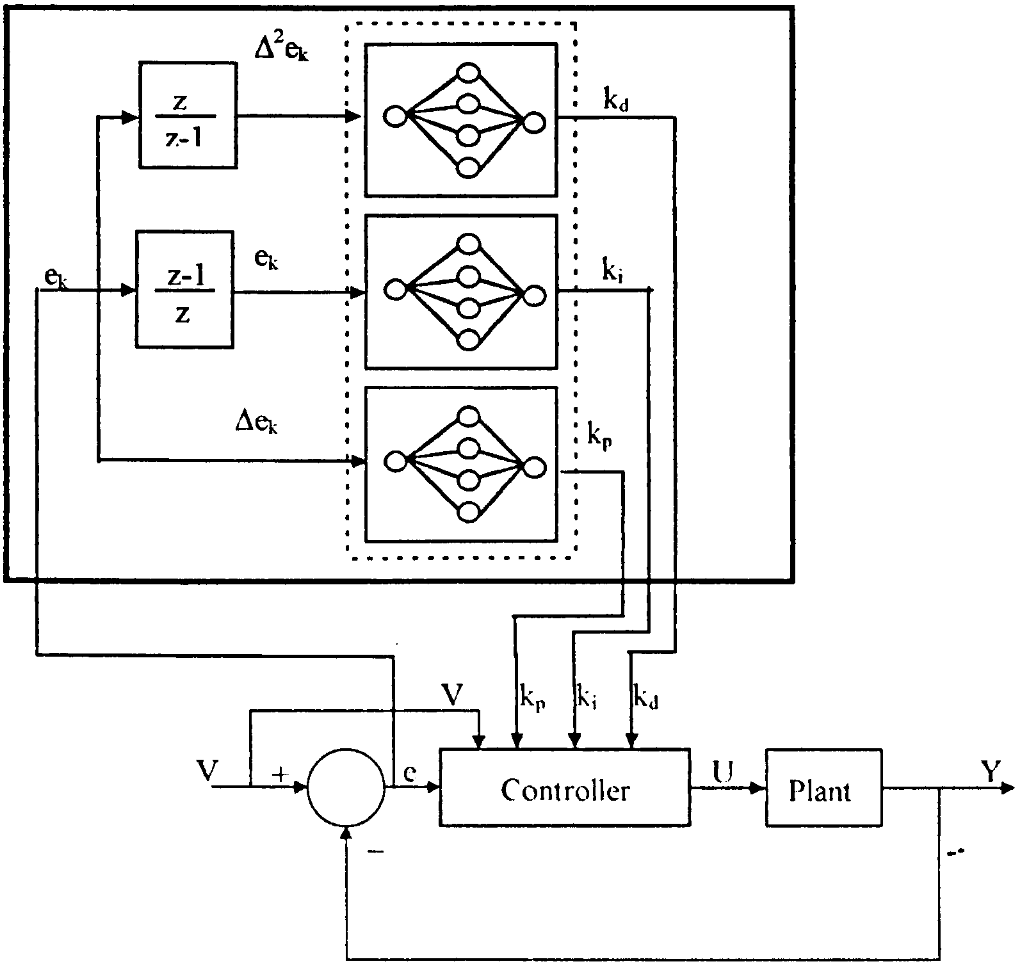


Figure 6.1 System Block Diagram

use of an individual data set, after which the strengths are fixed, or learning can continue throughout the network's lifetime, possibly in terms of a faster rate in the initial period with a limited amount of allowance in the steady state. The basic node elements employed in neural networks differ in terms of the type of network considered. However one commonly encounter model is a form of the McCulloch and Pitts neuron (Aleksander, 1991), and is shown in figure (6.2). In this the inputs to the node take the form of data items either from the real-world or from other network elements, possibly from the outputs of nodes in a previous layer.

The output of the node element is found as a function of the summed weighted strength inputs.

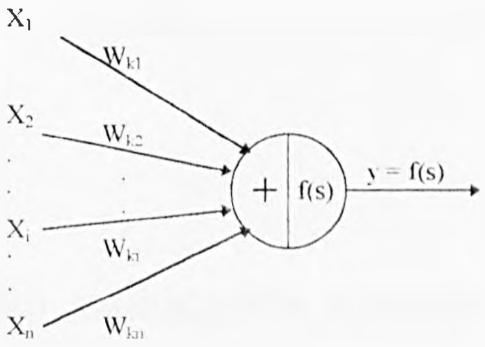


figure (6.2) Basic Neuron Model

where

x_i input value;

w_i connecting weighting;

y = f(s) output function.

The output of the neuron is determined as:

$$y = f\left(\sum_{i=1}^{n} w_i x_i\right) \qquad (i=1,2,....,n), \qquad 6.19$$

f is a defined output function that may be a sigmoid.

This output signal can then be employed directly, or it can be further processed by an appropriate thresholding or filtering action, a popular form being a sigmoid function. The above follows the forward propagation neural networks technique.

6.2.2.2 FORWARD PROPAGATION THROUGH MULTILAYER NEURAL NETWORKS

A multi layer feed forward neural network consists of a number of layers which each having an arbitrary number of neurons. Each neuron is connected to neurons in adjacent layers, as indicated in figure (6.3). However, in more general situations, it is possible for individual neurons to be connected to neurons in other layers apart from the adjacent layers. In figure (6.3), the input layer receives an input vector element through a local activation function. These activations then pass through weighted values to the next layer of the neurons where the same procedure takes place until the output layer is reached.

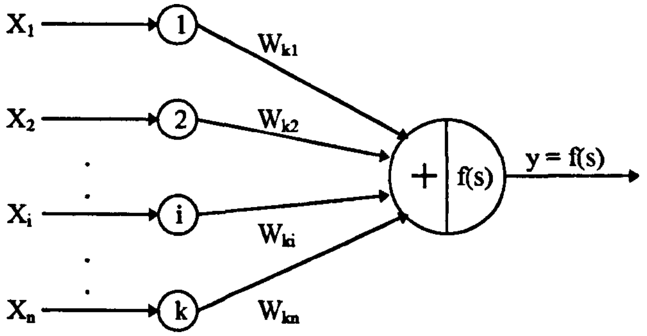


Figure (6.3) Typical Layer of four Neuron Model connected to an output Neuron

In the case of the neural network with the input vector element, x_i, this element is passed to an activation function, f_1^0, so that

$$y_i^{(0)} = f_i^{(0)}(x_i^{(0)}) \qquad 6.20$$

Here, the bracketed superscript refers to the layer and the subscript index to the particular neuron in the layer.

Thus, in the first layer,

$$y_j^{(1)} = \sum_{i=1}^{P} w_{ji}^{(1)} y_i^{(0)} \qquad 6.21$$

and .

$$y_j^{(1)} = f_j^{(1)}(y_j^{(1)}) \qquad 6.22$$

In the second layer,

$$y_k^{(2)} = \sum_{j=1}^{M} w_{kj}^{(2)} y_j^{(1)} \qquad 6.23$$

and

$$y_k^{(2)} = f_k^{(2)}(y_k^{(2)}) \qquad 6.24$$

Finally, in the output layer,

$$y_r^{(3)} = \sum_{k=1}^{N} w_{rk}^{(3)} y_k^{(2)} \qquad 6.25$$

and

$$y_r^{(3)} = f_r^{(3)}(y_r^{(3)}) \qquad 6.26$$

As can be seen from the above equation the number of neurons in any layer, and the number of layers can both be chosen by designer.

Also it is evident from the previous chapters that the sigmoid function is ideal to map the non-linear gain functions needed for the non-linear neural PID controller in this chapter.

From the above equation it can be seen that the neural network can be used to map the non-linear gain function required by the non-linear incremental PID controller.

6.3 NEURAL PID CONTROLLER GAIN FUNCTION MAPPING

In order to use a neural network to map the non-linear gain functions for the non-linear incremental PID controller, the number of layers, and the number of neurons in the layers has to be defined. In this chapter it is proposed to use 1 layer with 4 neurons, plus 1 input, and 1 input neuron. From the above analysis it can be seen that the output function for the proposed network is given by:

$$y_i^{(0)} = f_i^{(0)}(e_k^{(0)}) \qquad 6.27$$

this is the output of the input layer, thus, the input to the first layer, is given by

$$y_j^{(1)} = \sum_{i=1}^{P} w_{ji}^{(1)} y_i^{(0)} \qquad 6.28$$

and the output of the first layer is given by

$$y_j^{(1)} = f_j^{(1)}(y_j^{(1)}) \qquad 6.29$$

Finally in the output layer

$$y_k^{(2)} = \sum_{k=1}^{M} w_{kj}^{(2)} y_j^{(1)} \qquad 6.30$$

and the output of the output layer is given by

$$y_k^{(2)} = f_k^{(2)}(y_k^{(2)}) \qquad 6.31$$

From the above it can be seen that the neural network can be used to map the non-linear PID gain functions required by the non-linear PID controllers.

The above gain function is the integral gain function since the input to the network is e_k the tracking error, for the other two gain functions the input to the network is Δe_k and $\Delta^2 e_k$ for the proportional, and derivative gain function respectively. figure(6.4) shows the neural network representation of the three gain functions as they are mapped into the system.

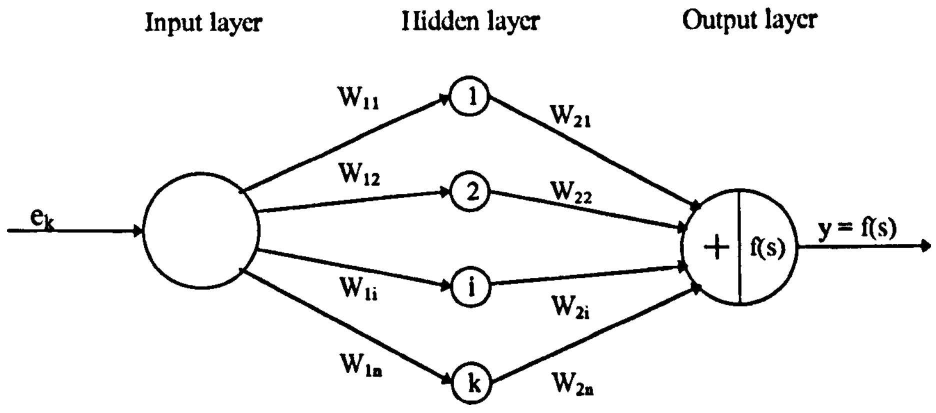


Figure (6.4) Neural network representation of a gain function

The non-linear gain function in figure (6.4) is for an integral gain function. Also as it can be seen from figure (6.4), the neural networks dose not have a bias input. This is because the gain functions are required to go through the origin. By omitting the bias and ensuring the sigmoidal function goes through the origin, there are less parameters in the neural network, and it is easier to search for the remaining parameters. The diagram in figure (6.5a) shows a typical sigmoidal function resulting from a neural networks with a bias, and figure (6.5b) shows the sigmoidal function used in this research.

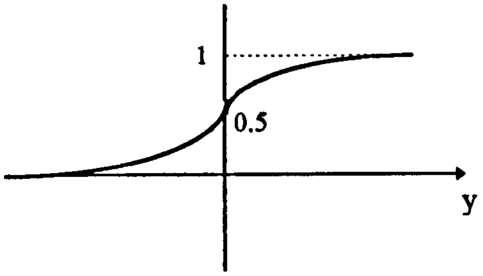


Figure (6.5a) Typical Sigmoidal function

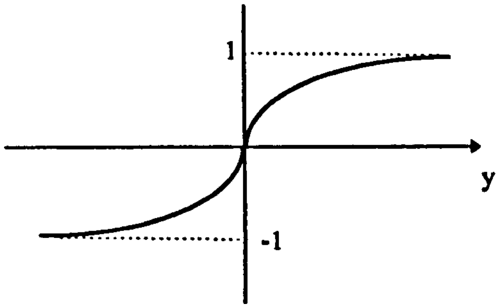


Figure (6.5b) Unbiased Sigmoidal function

6.4 GENETIC DESIGN OF A DUAL- ZONED NEURAL NON-LINEAR PID CONTROLLERS

In order to use genetic algoritms to design neural dual zoned non-linear PID controllers, the parameters associated with the dual-zoned controller have to be precisely defined. It is evident from previous chapters that the behaviour of dual controller can be split into two distinct zones. The zones are defined as set-point zone, and tracking zone.

The parameters for the controller in the two zones are thus designed together. The performance of the dual-zoned controller will be contrasted by considering a number of plants controlled by both the genetically designed dual-zoned neural non-linear controller, and the genetically designed linear PID controller.

6.4.1 PARAMETERS FOR THE SET POINT ZONE

The parameters in this zone are chosen in exactly the same as was introduced in chapter 3 section 3.2.3.1 i.e. the GA will search the space for two feed forward gains for the set-point zone.

6.4.2 PARAMETERS FOR THE TRACKING ZONE

In the tracking zone it was found that for the proportional and derivative gains the controller operate in a very small region close to zero, and from the equations of the input parameters for the tracking-zone it can be seen that the controller must have individual gain values for the proportional, integral, and derivative gains. The gains used in this zone are mapped as shown in figure (6.5).

To design the dual zone neural controller the neural gain function introduced earlier in this chapter as in equation (6.18), will now be used to produce gain function for the gains required to produce a non-linear incremental neural PID controller. The gain functions produced will map the whole tracking zone for each of the gains (i.e. proportional, integral, and derivative).

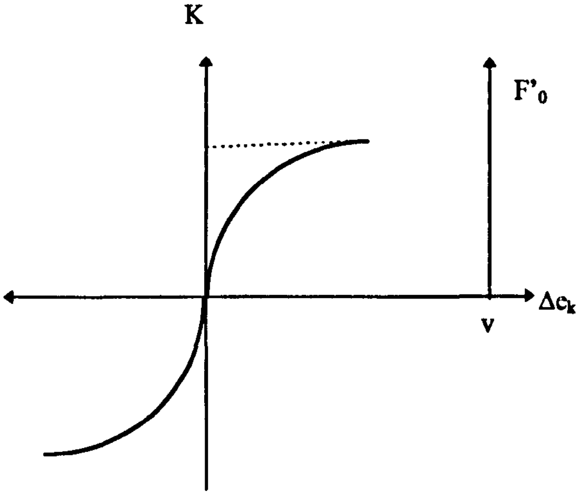


Figure (6.5) General Gain Function Mapping Using Neural Interpolation to Represents Proportional Gain

In order to use GAs to select the tuning parameters in such a way that as to produce satisfactory response in the case of a step input, it is only necessary to encode the elements of the tracking zone gain functions, plus the feed forward gains needed for the set point change zone, as binary strings. The binary string would be represented as

$$\{\ (\ w^{p}_{11} \ldots\ w^{p}_{2n})\quad (\ w^{i}_{ii} \ldots\ w^{i}_{2n})\quad (\ w^{d}_{11} \ldots\ w^{d}_{2n})\quad (\ F_0\ \ F_1)\ \}$$

proportional integral derivative set point change

where the whole string contains the neuron weights associated with the three gains, plus the feed forward gains.

Random initialization is the approach used to initialize the initial population. The process of interviewing introduced in chapter 2 is used to insure that the randomly generated variables do not initially violate any constraints on the function to be tuned. The system incorporates both the linear plant, and the non-linear PID controllers, the controllers are designed by randomly generated sets of tracking zone gains elements, and feed forward gains by the GA's. A simulation test is then carried out on all the controllers to asses local stability.

In the case of a violation of stability in any of the cases, the cases, the randomly generated set of tracking zone gains elements, and feed forward gains will not be included in the initial population. This type of initialization is essential in this type of design, since otherwise their would be a danger of creating an initial population which many of its members violate the constraints on the controllers being designed, figure (6.6) shows the sequence of genetic algorithms.

Following the initialization, the objective function is introduced and its value is calculated using decoded values of the parameters in each string. Thus for example, if minimum ISE is regarded as the ultimate design requirement, genetic algorithms can be readily used to select the best set of tracking gains elements, and the feed forward gains such that the ISE is minimised.

In the genetic design of non-linear PID controllers the plant under consideration is subjected to a command input (i.e. unit step), then the performance index is computed for the plant. Therefore, for each member in the population the function

$$ISE = \sum_{j=1}^{j=N} e^{2}_{j} \qquad 6.32$$

is evaluated, where

$$e_j = v - y \qquad\qquad N = \frac{T}{\tau}$$

and $e_j \in \Re$, is the error signal, $y_j \in \Re$,is the output signal, T is sampling time, and τ is an appropriately chosen settling time.

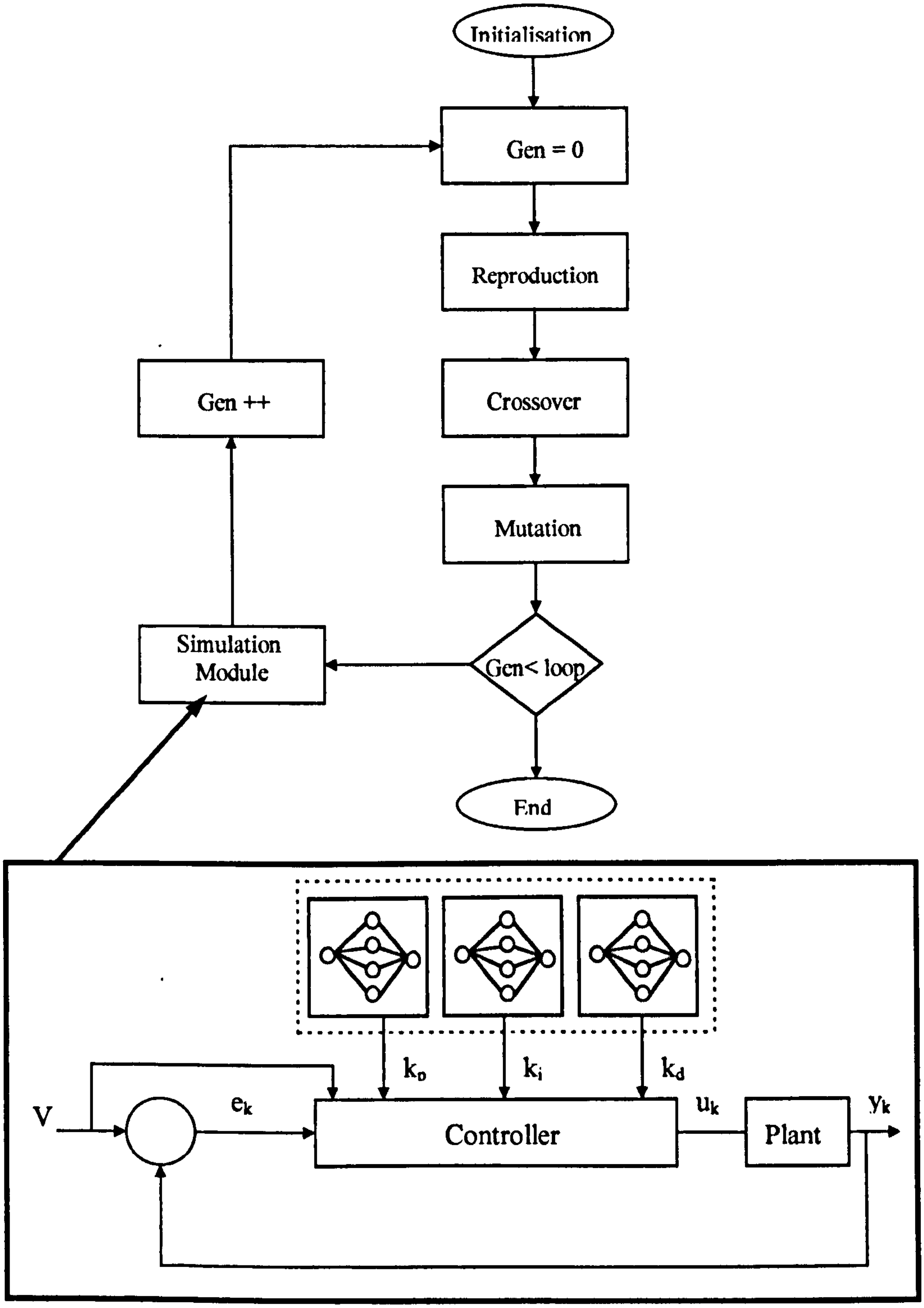


Figure 6.6 Sequence of GA's using neural networks to map gain functions

Minimization of this performance index over the entire population can be rapidly obtained by using the genetic operations of reproduction, crossover, and mutation. It is interesting to note that the conditions for the existence of a non-linear PID controllers for the plant under consideration will be automatically satisfied by the genetic algorithms. This is obvious, because in the case of violation of the constraints the corresponding set of tracking zones gains elements and the feed forward gains produces large ISE and the result will be zero or low fitness, and by the action of the selection operator it would not be chosen for the next generation.

6.5 ILLUSTRATIVE EXAMPLES

To design the dual zone neural controller the analysis done earlier in this chapter to produce gain functions using the neural network gain function mapping given by equation (6.26), will now be extended to produce gain function for the gains required to produce a non-linear incremental neural PID controller. The gain functions produced will map the whole tracking zone for each of the gains (i.e. proportional, integral, and derivative).

The gain function produced in equation (6.26) is the integral gain function since the input to the network is e_k the tracking error, for the other two gain functions the input to the network is the first order backward difference in error Δe_k, and the second order backward difference in error $\Delta^2 e_k$ for the proportional, and derivative gain function respectively.

It is evident that the proportional gain function is given by

$$y_i^{(0)} = f_i^{(0)}(\Delta e_k^{(0)}) \qquad 6.33$$

this is the output of the input layer, thus, the input to the first layer, is given by

$$y_j^{(1)} = \sum_{i=1}^{P} w_{ji}^{p(1)} y_i^{(0)} \qquad 6.34$$

and the output of the first layer is given by

$$y_j^{(1)} = f_j^{(1)}(y_j^{(1)}) \qquad 6.35$$

Finally in the output layer

$$y_k^{(2)} = \sum_{k=1}^{M} w_{kj}^{p(2)} y_j^{(1)} \qquad 6.36$$

and the function representing the proportional gain is given by

$$\mathcal{K}_p = f_k^{(2)}(y_k^{(2)}) \qquad 6.37$$

It can be seen that the above analysis can be used to produce the integral gain as

$$y_i^{(0)} = f_i^{(0)}(e_k^{(0)}) \qquad 6.38$$

this is the output of the input layer, thus, the input to the first layer, is given by

$$y_j^{(1)} = \sum_{i=1}^{P} w_{ji}^{i(1)} y_i^{(0)} \tag{6.39}$$

and the output of the first layer is given by

$$y_j^{(1)} = f_j^{(1)}(y_j^{(1)}) \tag{6.40}$$

Finally in the output layer

$$y_k^{(2)} = \sum_{k=1}^{M} w_{kj}^{i(2)} y_j^{(1)} \tag{6.41}$$

and the function representing the integral gain is given by

$$\mathcal{K}_i = f_k^{(2)}(y_k^{(2)}) \tag{6.42}$$

Finally It is evident that the derivative gain function is given by

$$y_i^{(0)} = f_i^{(0)}(\Delta^2 e_k^{(0)}) \tag{6.43}$$

this is the output of the input layer, thus, the input to the first layer, is given by

$$y_j^{(1)} = \sum_{i=1}^{P} w_{ji}^{d(1)} y_i^{(0)} \tag{6.44}$$

and the output of the first layer is given by

$$y_j{}^{(1)} = f_j{}^{(1)}(y_j{}^{(1)}) \tag{6.45}$$

Finally in the output layer

$$y_k{}^{(2)} = \sum_{k=1}^{M} w_{kj}^{d(2)} y_j{}^{(1)} \tag{6.46}$$

and the function representing the derivative gain is given by

$$\mathcal{K}_d = f_k{}^{(2)}(y_k{}^{(2)}) \tag{6.47}$$

The gain functions will be used in the incremental controller given in equation (6.18) to produce a neural PID controller. The weights in the gain functions are encoded in the GA as binary strings, The GA will search and produce weight that give the best ISE for the controller. The GA binary string is given by:

$$\{\ \underbrace{(\,w^p_{11} \;\ldots.\; w^p_{24})}_{\text{proportional}}\ \ \underbrace{(\,w^i_{11} \;\ldots.\; w^i_{24})}_{\text{integral}}\ \ \underbrace{(\,w^d_{11} \;\ldots.\; w^d_{24})}_{\text{derivative}}\ \ \underbrace{(\,F_0\ \ F_1)}_{\text{set point change}}\ \}$$

As it can be from the string their are 8 parameters (weights) needed for each of the gain functions, which means that the GA will be searching for 26 Parameters in total. The parameters are 24 weights for the three gain functions in the tracking zone, plus the two feed forward gains needed for the set-point zone.

6.5.1 PLANT 1

The procedure for the tuning of genetic control systems can be conveniently illustrated by designing a genetic non-linear PID control system for the open loop SISO plant with transfer function of the form

$$g(z) = \frac{0.8}{z^4(z-0.9)} \qquad 6.48$$

the sampling period is 0.1 sec.

The Arma model for the plant is of the form

$$y_k = a_0 y_{k-1} + b_0 u_{k-5} \qquad 6.49$$

The plant variables are given by

$a_0 = 0.9$

$b_0 = 0.8$

where the incremental PID controller is given by equation (6.18). The controller is implemented in the incremental form so as, to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. The controller is designed using the dual zone method introduced earlier.

6.5.1.1.NON-LINEAR NEURAL PID CONTROLLER

Initially the gains $\mathcal{K}_p$, $\mathcal{K}_i$ and $\mathcal{K}_d$ are chosen as a function of the first order backword deference in error, error, and second order backword deference in error respectively as that given by equations (6.37, 6.42, 6.47), i.e. dual zoned neural PID controller. Then the controller was designed by means of genetic algorithms, so as that the Integral square error is minimised for the plant.

In this case, a population of 100, a crossover probability, $P_c=0.65$, and a mutation probability, $P_m = 0.01$, was used, also this case the maximum value for both the first and second order backward difference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design .Figure (6.8) shows the transient response of the genetically designed controller.

6.5.1.2 LINEAR PID

To contrast the difference between linear and non-linear PID controllers, a genetically designed linear PID controllers was also condidered, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, and K_d are chosen by the GAs to minimise the ISE for the fixed controller.

In this case, a population of 100, a crossover probability, $p_c = 0.65$, and a mutation probability, $P_m = 0.01$ was used. Figure (6.9) shows the transient response of the genetically designed controller. The ISEs for both the multi-zoned nueral and the linear PID controllers are shown in table (6.1).

CONTROLLER	Multi-Zoned Neural	LINEAR
ISE	5.524	9.16

Table (6.1)

Table (6.2) shows the weights for the multi-zoned non-linear PID controller gains.

Gain	W11	W12	W13	W14	W21	W22	W23	W24
$\mathcal{K}_p$	8.546	4.273	0.356	5.697	9.614	6.766	3.917	5.341
$\mathcal{K}_i$	6.409	8.902	4.629	7.834	7.478	4.273	5.697	7.834
$\mathcal{K}_d$	**0.712**	**1.068**	**9.614**	**6.053**	**9.614**	**1.068**	**4.273**	**0.002**

Table (6.2).

Table (4.3) shows the feed forward gains for the multi-zoned nueral PID controller.

Feed Forwad Gains	0.712	11.039

Table (6.3).

6.5.2 PLANT 2

A second plant was considered, to investigate the effectiveness of the genetic algorithms in designing linear, and non-linear controller. The plant considered has a transfer function of the form:

$$g(z) = \frac{0.03573 + 0.044625\ z}{z^{4}(0.0513423 \ -1.4331\ z^{2})} \qquad 3.50$$

the sampling period is 0.1 sec.

The Arma model for the plant is of the form as

$$y_{k} = a_{0}y_{k-1} + a_{1}y_{k-2} + b_{0}u_{k-5} + b_{1}u_{k-6} \qquad 3.51$$

The plant variables are given by,

$a_0 = 1.4331$,

$a_1 = -0.51342$,

$b_0 = 0.044625$,

$b_1 = 0.03573$,

and the incremental PID controller is governed by equation (6.18). The plant was tested under different operating conditions as shown in the examples bellow:

6.5.2.1 NON-LINEAR NEURAL PID CONTROLLER

Initially the gains $\mathcal{K}_p$, $\mathcal{K}_i$, and $\mathcal{K}_d$, where chosen as a function of first order backward difference in error, error, and second order backward difference in error respectively, asthat given by equations (6.37, 6.42, 6.47), i.e. dual-zoned PID controller with linear plant. The controller was designed by means of genetic algorithms, such that the Integral Square Error ISE is minimised for the plant. In this case, a population of 100, a crossover probability, $P_c = 0.65$, and mutation probability, $P_m = 0.01$ was used, also this case the maximum value for both the first and second order backward difference in error (Δe_{kmax} and $\Delta^2 e_{kmax}$), were chosen for this design. Figure (6.10), shows the transient response of the genetically designed non-linear controller for the above plant, Figure 6.9a, 6.9b, and 6.9c shows the resulting profiles for the proportional, integral, and derivative gains respectively.

6.5.2.2 LINEAR PID

Finally to contrast the deference between linear and non-linear PID controllers, a genetically designed linear PID controller constructed, the genetic algorithms was used to design a fixed gain controller where the values of K_p, K_i, K_d , are integer values chosen by the GA's to minimise the ISE for the fixed controller. In this case, a population of 100, a crossover probability, $p_c = 0.65$, and a mutation probability, $P_m = 0.01$ was used. Figure 6.11 shows the transient response of the genetically designed controller. The ISEs for both the multi-zoned neural and the linear PID controllers are shown in table (6.4).

CONTROLLER	Multi-Zoned Neural	LINEAR
ISE	6.466	10.57

Table (6.4).

Table (6.5) shows the weights for the multi-zoned non-linear PID controller gains.

Gain	W11	W12	W13	W14	W21	W22	W23	W24
$\mathcal{K}_p$	3.254	0.001	0.465	11.16	2.324	3.254	7.438	4.649
$\mathcal{K}_i$	8.368	0.465	10.23	11.62	3.254	14.41	13.02	8.368
$\mathcal{K}_d$	4.649	12.55	3.254	11.62	13.95	12.55	10.69	3.719

Table (6.5).

Table (6.6) shows the feed forward gains for the multi-zoned nueral PID controller.

Feed Forwad Gains	0.930	4.649

Table (6.6).

6.5.3 ROBUSTNESS TEST

This test is aimed at finding how robust are the genetically designed dual zone PID controllers are for changes in the plant operating conditions. In this case the plants used in the illustrative example were modified to produce different plants . To do this test consider plant 1 with transfer function of the form

$$g(z) = \frac{0.8}{z^4(z-0.9)} = \frac{\beta}{z^{\gamma}(z-\alpha)}$$

To produce the new plants the value of α, β, and γ are changed. The tables (6.7) show the range of plants considered in this robustnes test, and the respective ISEs obtained,

Plant	Lin ISE	Ne-ISE	β	α	γ
1	9.16	6.28	0.89	0.90	5
2	11.76	7.894	0.80	0.90	6
3	13.07	8.959	1.2	0.90	6
4	14.867	13.221	1.2	0.94	6
5	9.976	4.986	1.2	0.94	4
6	13.967	6.066	1.2	0.60	4
7	22.652	14.623	0.5	0.60	4

Table (6.7)

Figure (6.12 to 5.17) show the response of the plants for the non-linear multi-zoned PID controller designed using plant 1.

Table (6.8) shows the results of the rubestness test using cliped controller as introduced in chapter 3.

Plant	Lin ISE	Cli-ISE	β	α	γ
1	9.16	6.28	0.89	0.90	5
2	11.764	7.894	0.80	0.90	6
3	13.07	8.959	1.2	0.90	6
4	14.867	13.221	1.2	0.94	6
5	9.976	4.986	1.2	0.94	4
6	13.967	6.066	1.2	0.60	4
7	22.652	14.623	0.5	0.60	4

Table (6.8).

The transient response of the clipped controller robustness tests is shown in figure (6.18-6.23).

6.6 CONCLUSIONS

In this chapter the dual zone technique was used to design a simple neural networks PID controllers for linear plants. The neural networks was used to map the non-linear PID gains to produce a genetically designed neural PID controller. It has thus been shown in this chapter that a simple neural PID controller can produce very effective results. This design also shows that the use of the neural networks to map the gain functions combined with the use of dual zone technique in a genetically designed controller can benefit from the neural network ability at a reduced number of parameters for the GA to find compared to a full neural network controller. The result is illustrated by genetically designed digital dual zoned neural PID controllers for linear plants. Indeed comparing the results of the genetically designed controllers from the previous chapters to that designed using the neural network it can be seen that the design method has produced very effective high performance controllers.

The robustness test indicate that the non-linear PID controllers are more robust than the linear controllers. Because of the nature of the sigmoidal function, where by it saturates for larger than normal input signal. The addition of a clipping feature did not have any effect on the performance of controllers.

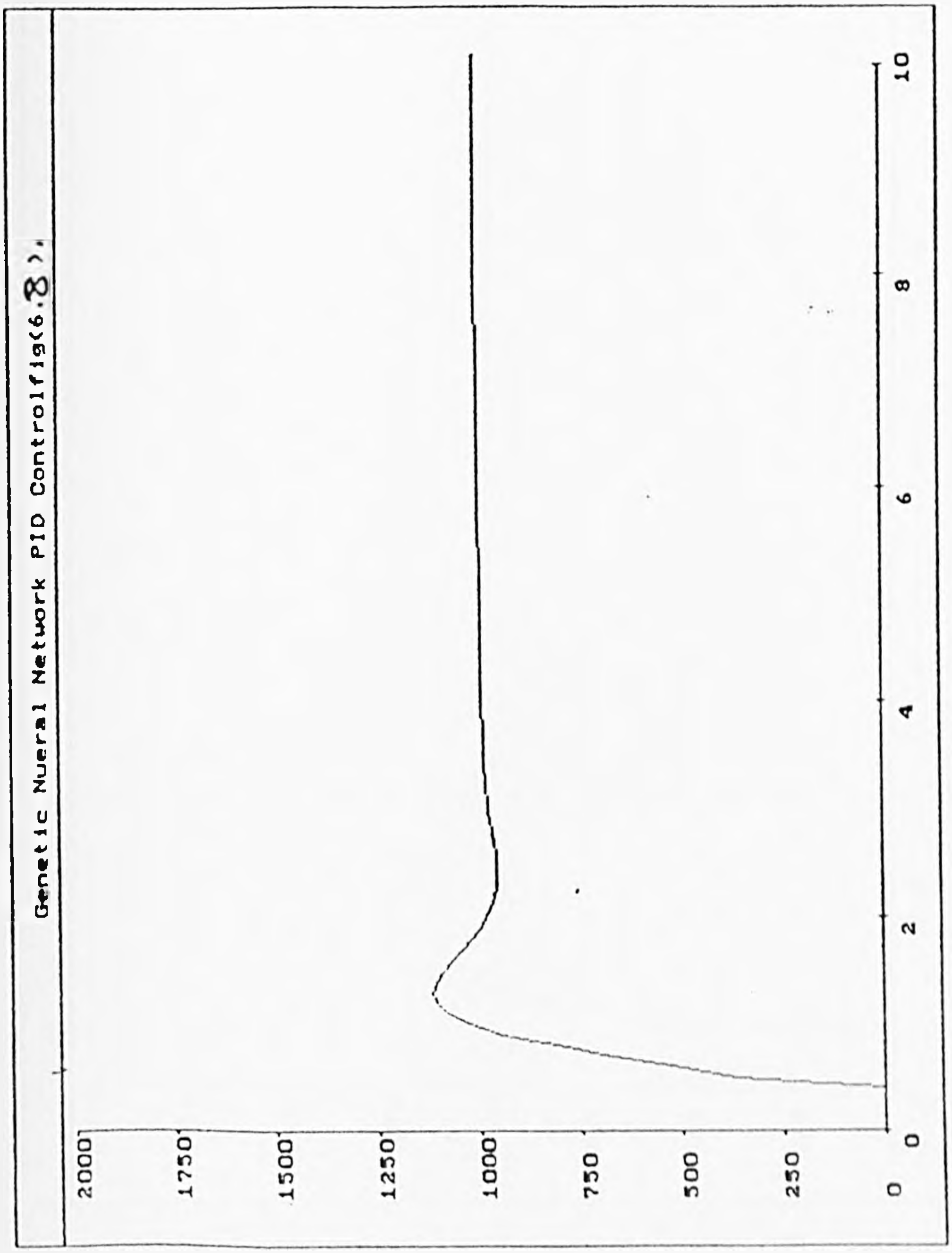


Figure (6.8) Transient response of the genetically designed dual-zoned neural PID controller for plant (1)

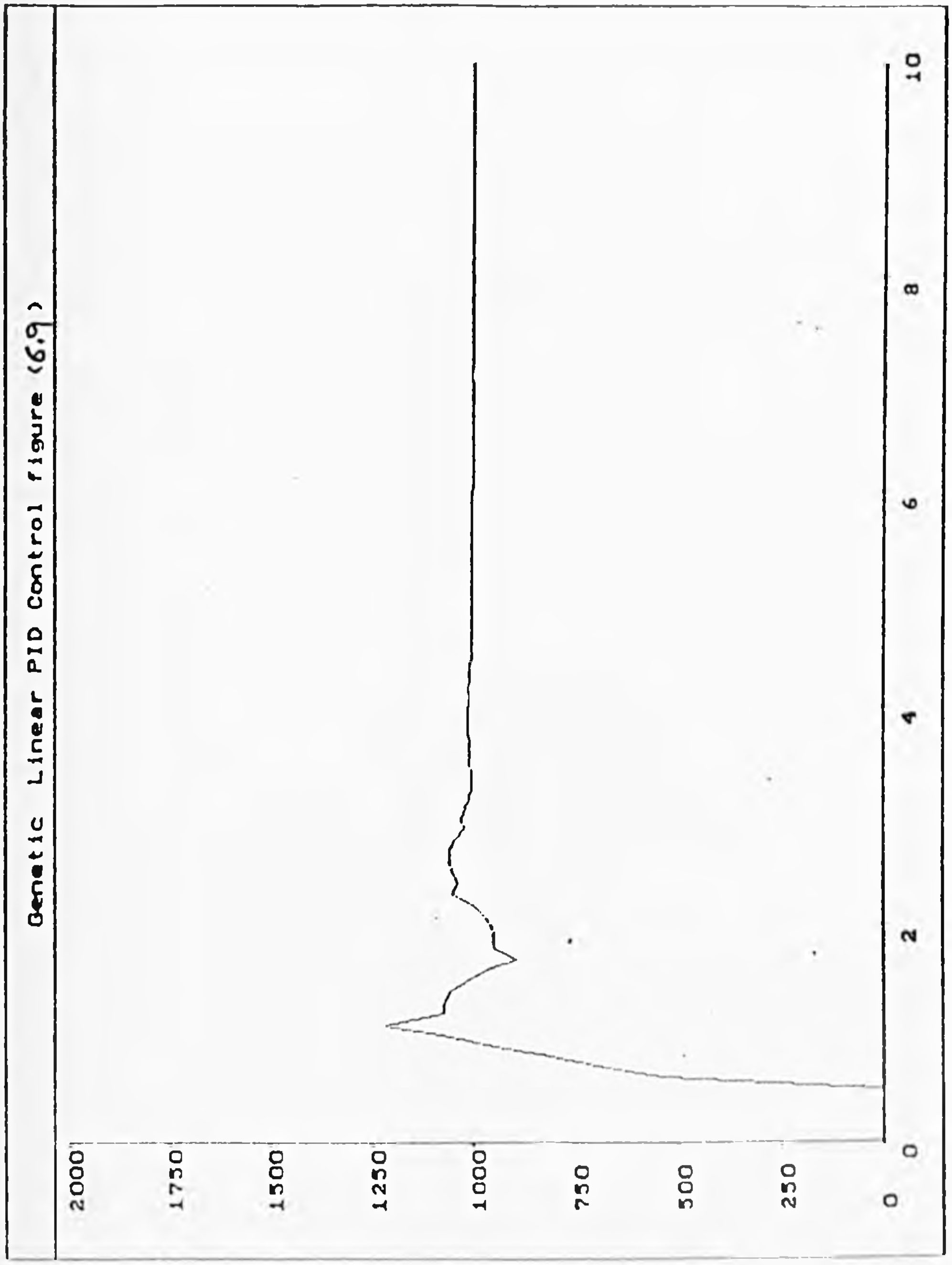


Figure (6.9) Transient response of the genetically designed linear PID controller for plant (1)

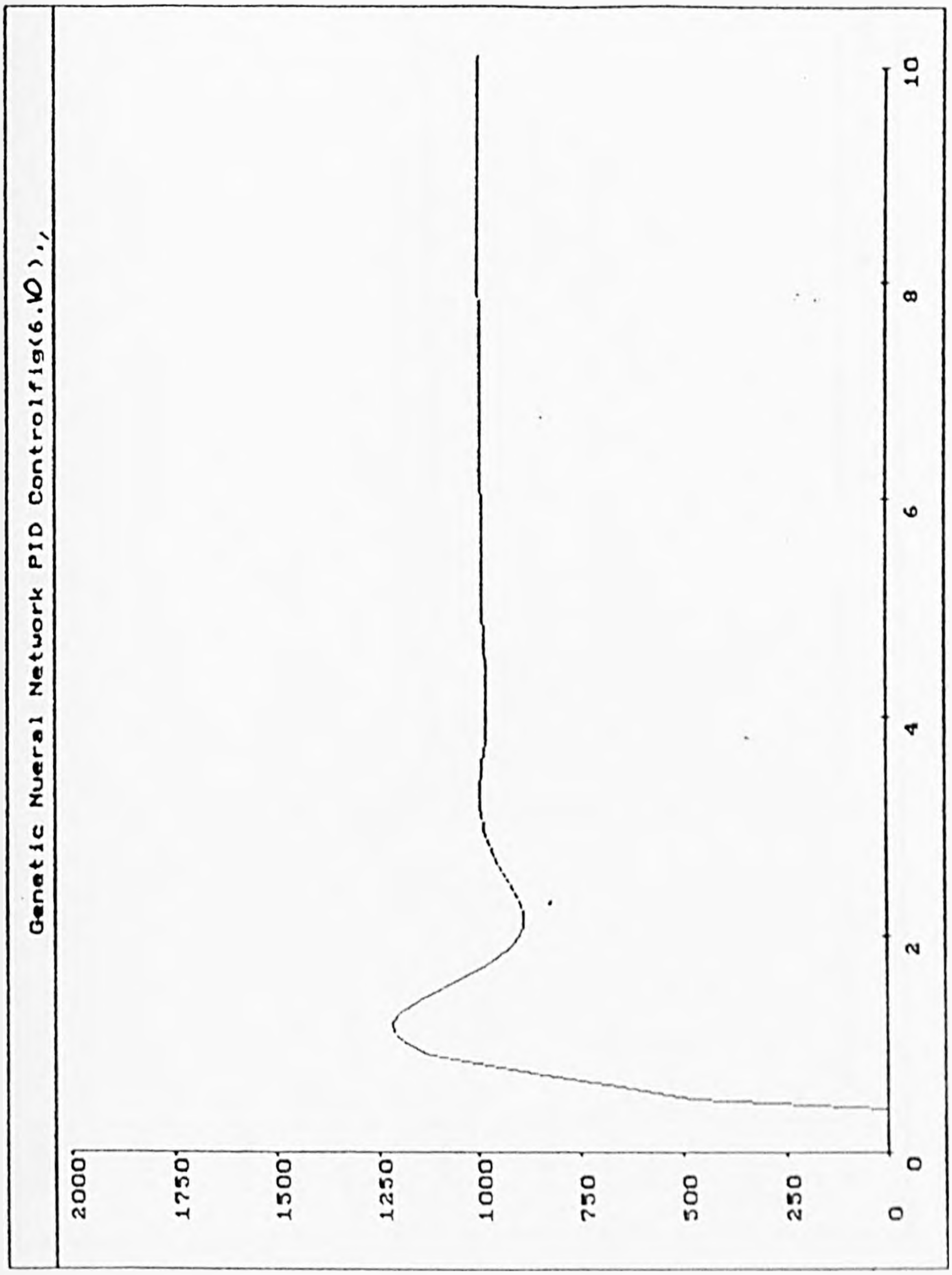


Figure (6.10) Transient response of the genetically designed dual-zoned neural PID controller for plant (2)

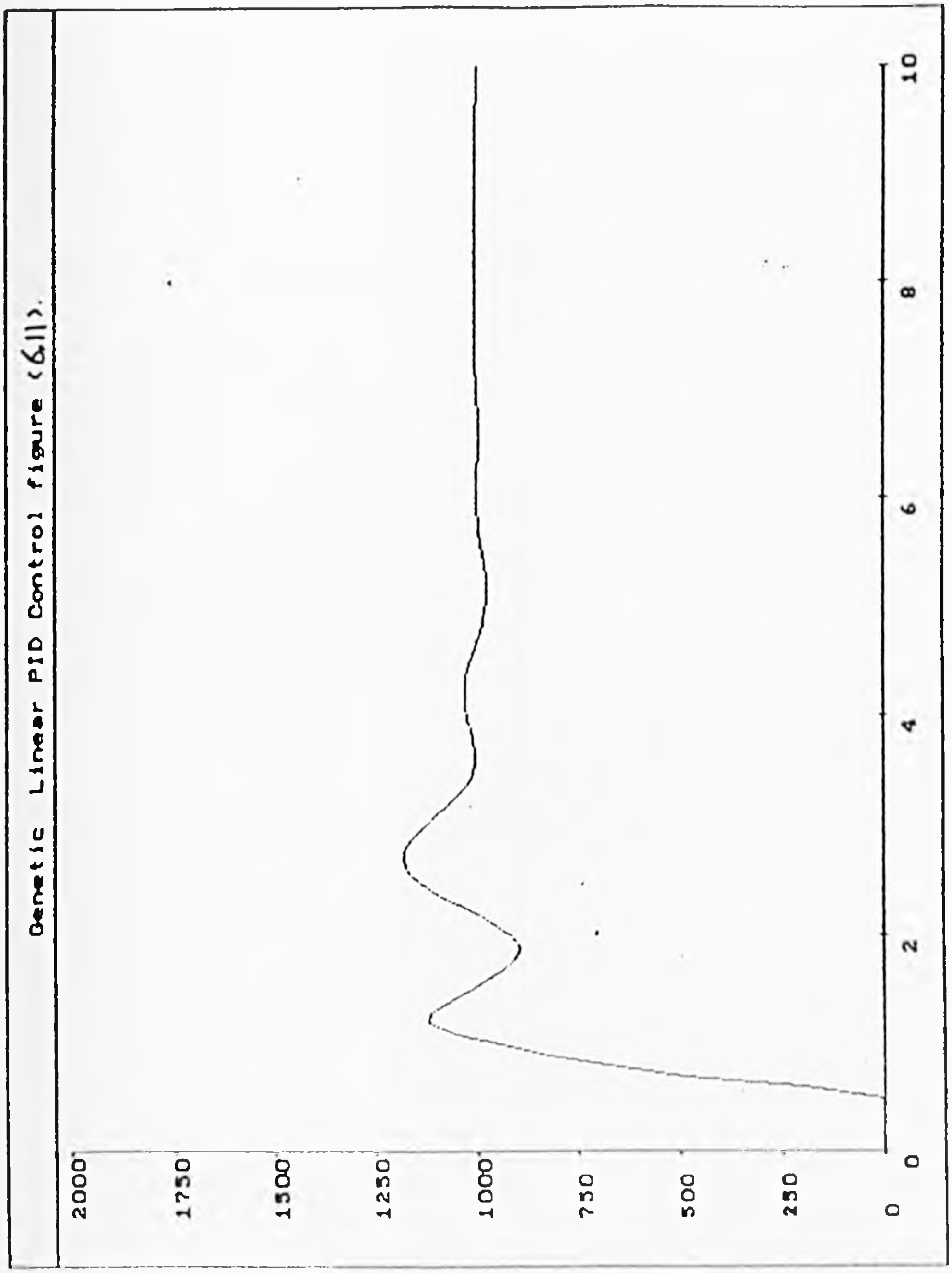


Figure (6.11) Transient response of the genetically designed linear PID controller for plant (2)

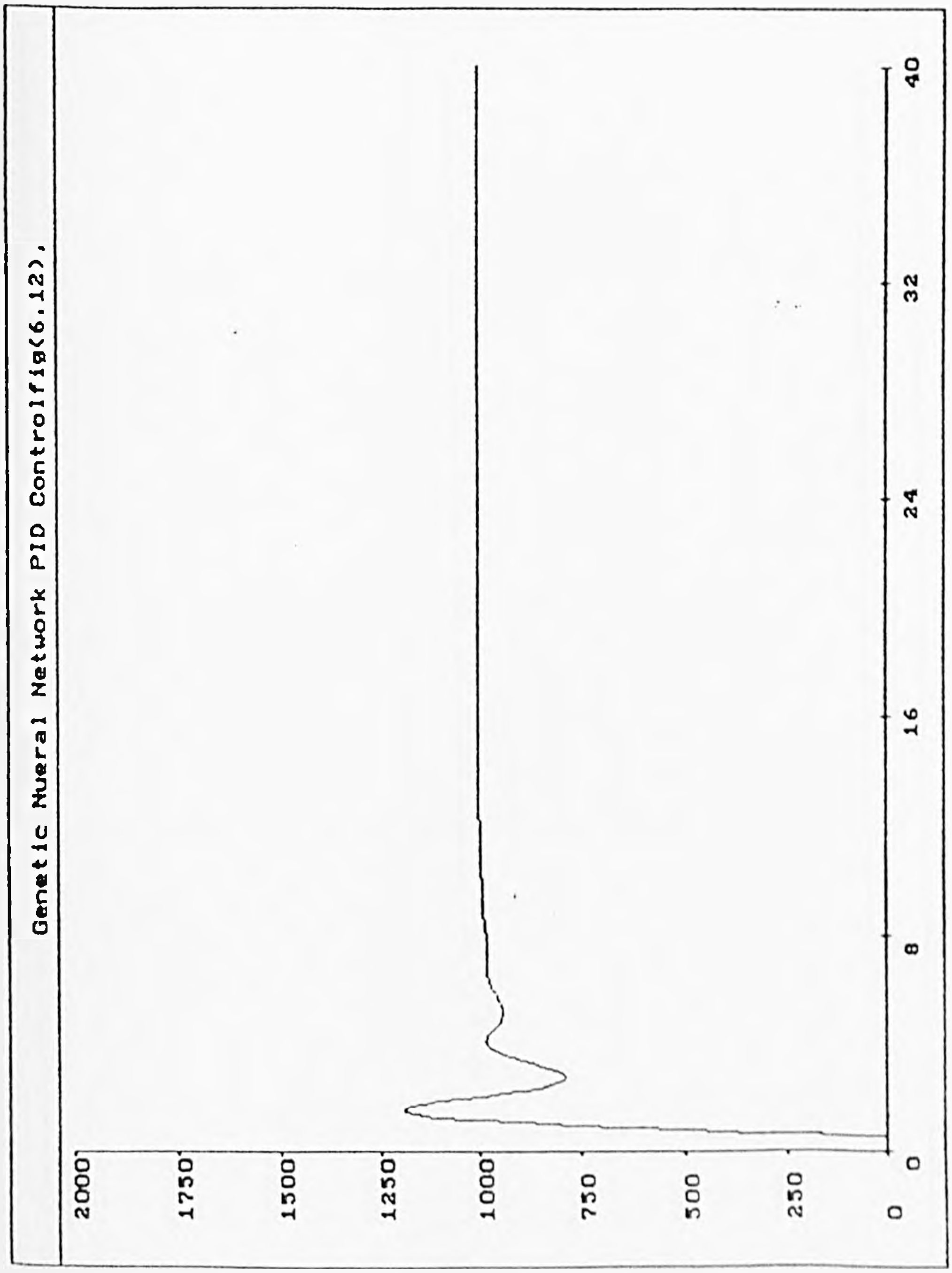


Figure (6.12) Transient response of robustness test using genetically designed non-linear PID controller for plant (2), table (6.7).

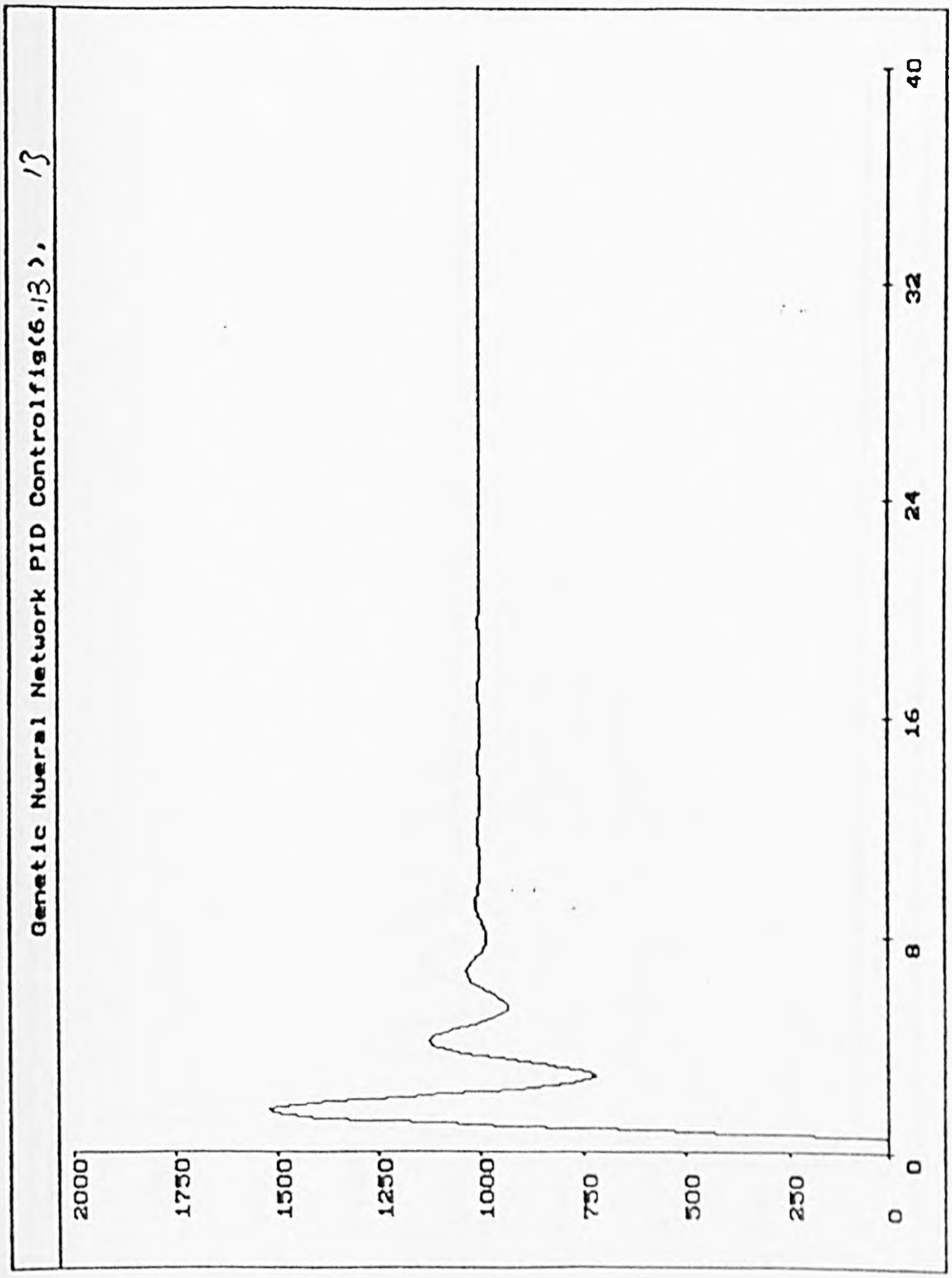


Figure (6.13) Transient response of robustness test using genetically designed non-linear PID controller for plant (3), table (6.7).

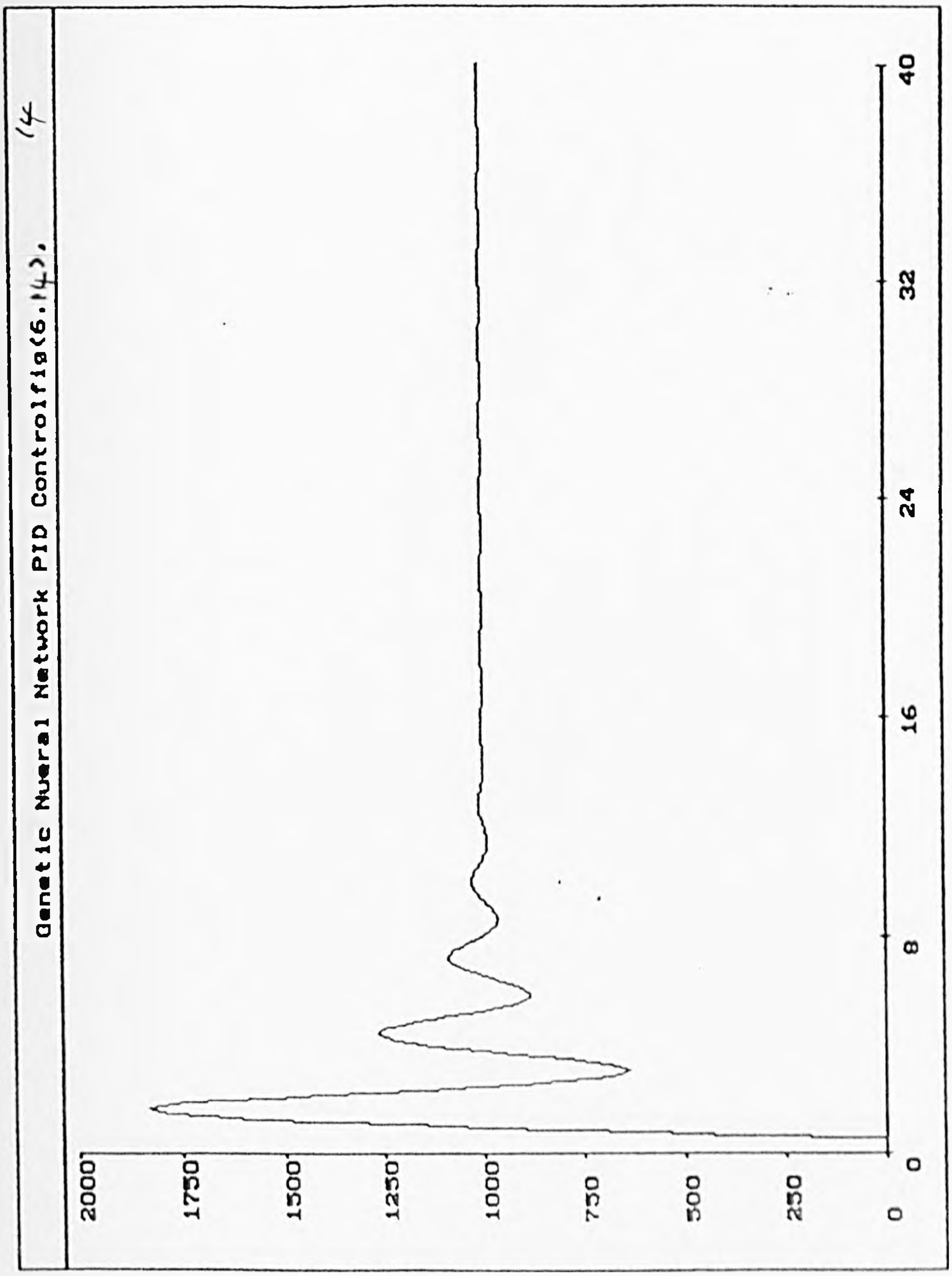


Figure (6.14) Transient response of robustness test using genetically designed non-linear PID controller for plant (4), table (6.7).

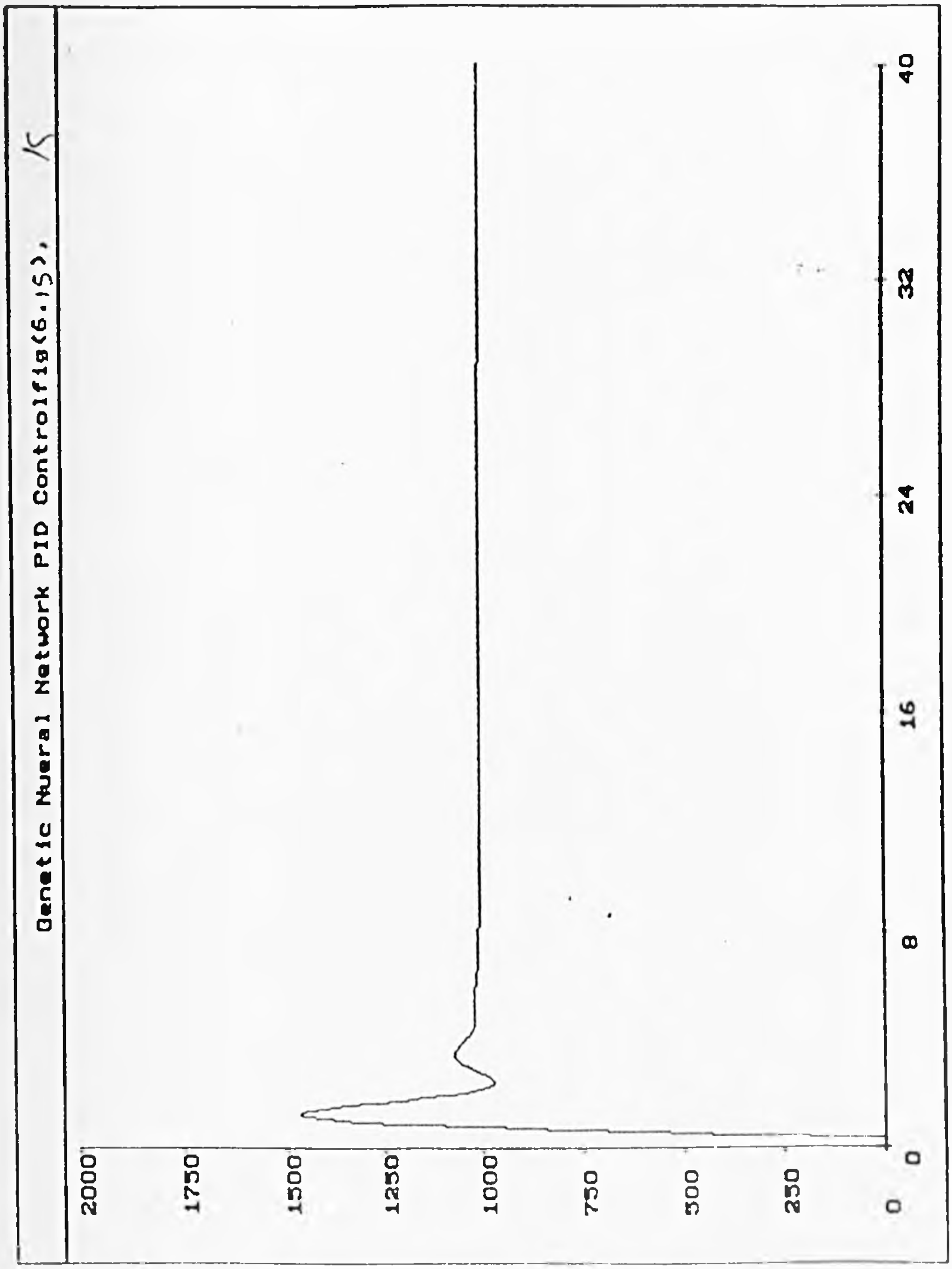


Figure (6.15) Transient response of robustness test using genetically designed non-linear PID controller for plant (5), table (6.7).

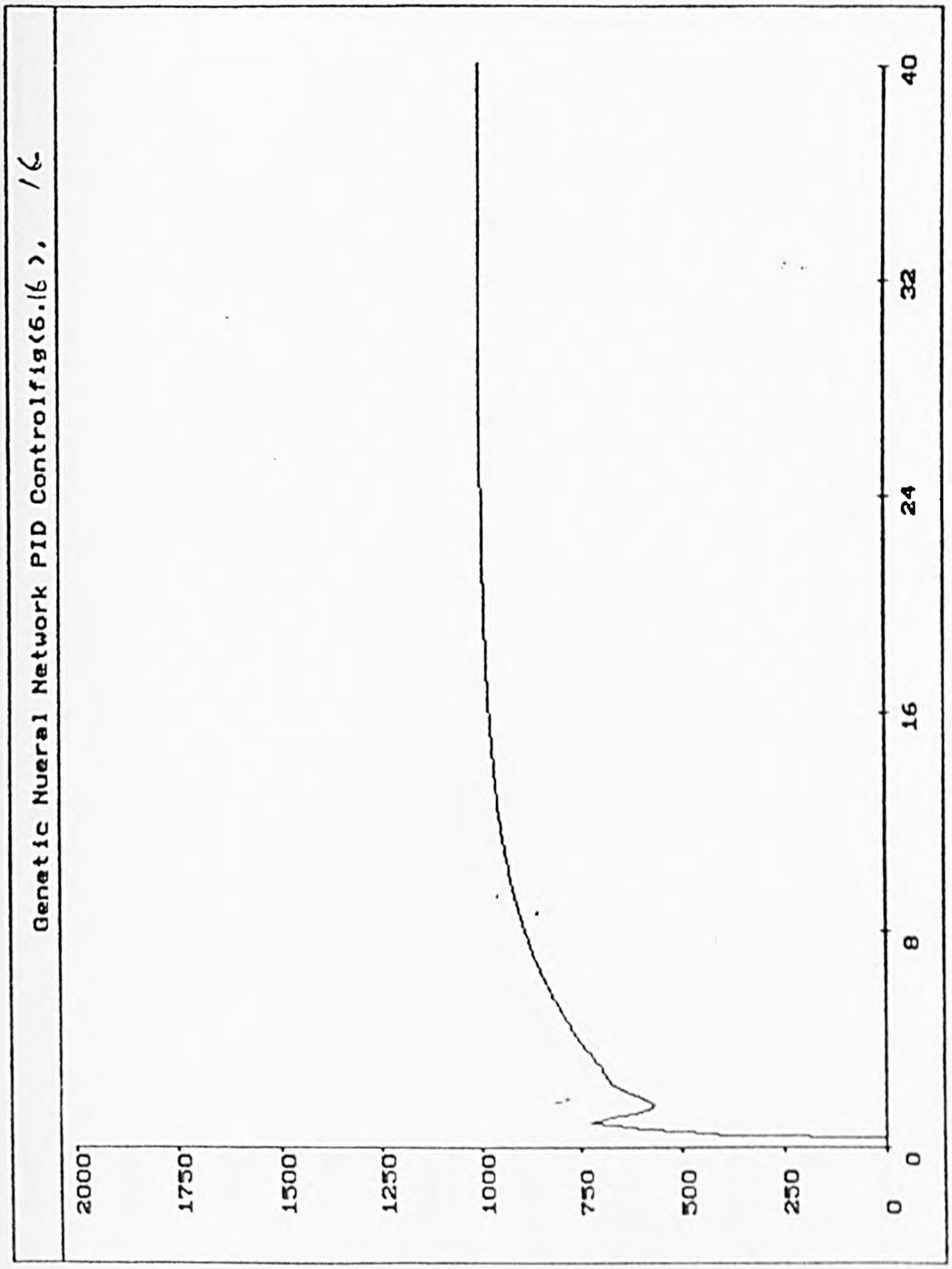


Figure (6.16) Transient response of robustness test using genetically designed non-linear PID controller for plant (6), table (6.7).

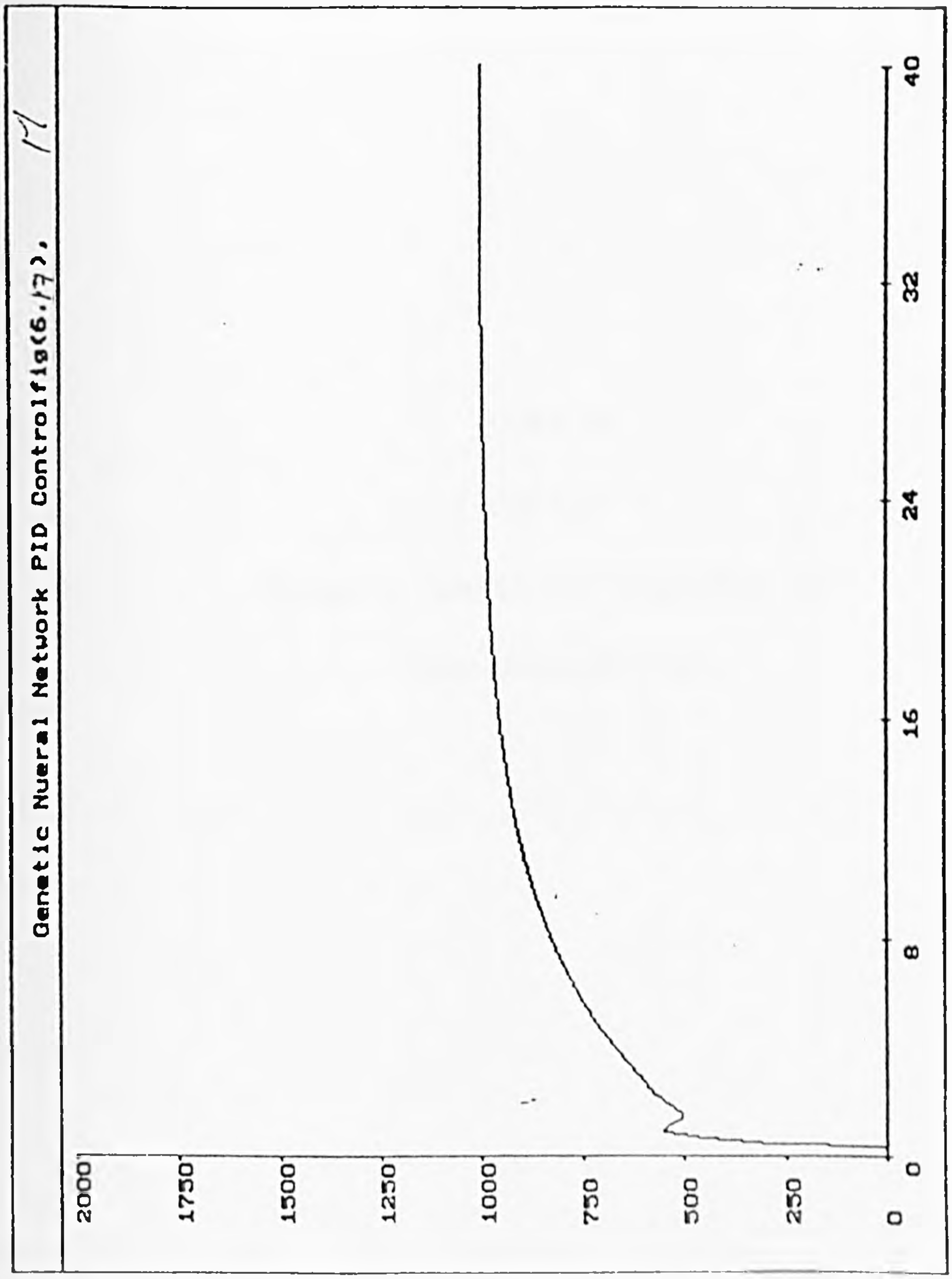


Figure (6.17) Transient response of robustness test using genetically designed non-linear PID controller for plant (7), table (6.7).

PART III

CHAPTER 7

Design of Non-Linear Controllers for Non-Linear Plants

Chapter 7
GENETIC DESIGN OF NON-LINEAR CONTROLLERS FOR NON-LINEAR PLANTS

7.1 INTRODUCTION

In this chapter it proposed to use genetic algorithms to design non-linear controllers for non-linear plants. The design of non-linear controllers for non-linear plants is a non-trivial problem. One approach which has dominated this field is the gain-scheduling technique. A typical gain scheduled design procedure for non linear plants is as follows. First, the designer selects several operating points which cover the range of plants dynamics. Then, at each of these operating points, the designer constructs a linear time-invariant approximation to the plant and designs a linear controller for each linearized plant. In between operating points, the parameters (gains) of the compensators are then interpolated, or scheduled, thus resulting in a global compensator. The design problem then becomes centred on choosing the scheduling variables, and selecting an appropriate number of operating points for designing the controller, at each of the operating points. The automation of this design procedure appears to be a daunting task. The design specification is to design a control scheme for a non-linear plant such that the controller performs optimally throughout the operating envelope. The design of a gain-scheduled controller becomes even more difficult in the case where the non-linearity is dependant on more than one variable. Furthermore, in such cases the design process is often lengthy as large number of operating points have to be explored. This problem motivates the consideration of deploying GA for searching the operating envelope and designing the appropriate locally linearized controller for incorporation into the gain-scheduled control scheme. Indeed, the technique of genetic algorithms introduced in the previous chapters, appears to be an ideal design tool for such a complex control scheme such evolutionary techniques are significantly different from traditional methods. Hence in this chapter it is proposed to design a non-linear controller for non-linear plants. Within this context it is proposed to genetically design a non-linear gain-scheduled controller for non-linear models of a water tank, and concentration tank [103].

7.2 NON-LINEAR SYSTEM

The basic definitions and general characteristics of a non-linear systems are :

i) superposition does not hold;

ii) sinusoidal inputs do not necessarily produce sinusoidal outputs;

iii) system stability may depend on input amplitude or frequency;

vi) instability may be exhibited by the presence of limit cycles, i.e. oscillation of constant amplitude and arbitrary waveform;

v) sub-harmonics can be generated.

7.2.1 METHODS OF ANALYSING NON-LINEAR SYSTEMS

There are many methods which already exist for the purpose of analysing non-linear systems, for the purpose of stability and controllability. But very little is available on the design of such non-linear control systems. The following are some of the traditional methods used to analyse non-linear systems.

7.2.1.1 PIECEWISE LINEAR APPROXIMATION

This type of approximation gives satisfactory results for systems having small size non-linearities. Many physical systems are decidedly non-linear, even within a restricted region about the operating point. For example, a dry friction element produces a damping force of the nature shown in figure (7.1a). From this figure, it is seen that no single straight line can represent such a curve through out the range of speeds usually encountered. However, the advantages of linear theory can be extended to this case by piecewise linearization as shown in figure (7.1b). In such cases, the response obeys a certain linear differential equation in one region of operation and a different one in another region. Depending upon the value of the region parameter, the differential equations are switched from one region to another, such that the end conditions of the first are carried over as the initial conditions of the other. Such systems are known as *piecewise linear systems.*

At the cost of considerably increased computational work, piecewise linearization could be applied to any non-linear system by dividing the whole region of operation into small pieces. In some situations, where the system is only slightly non-linear, piecewise linearization is cumbersome and time consuming. For such cases, two

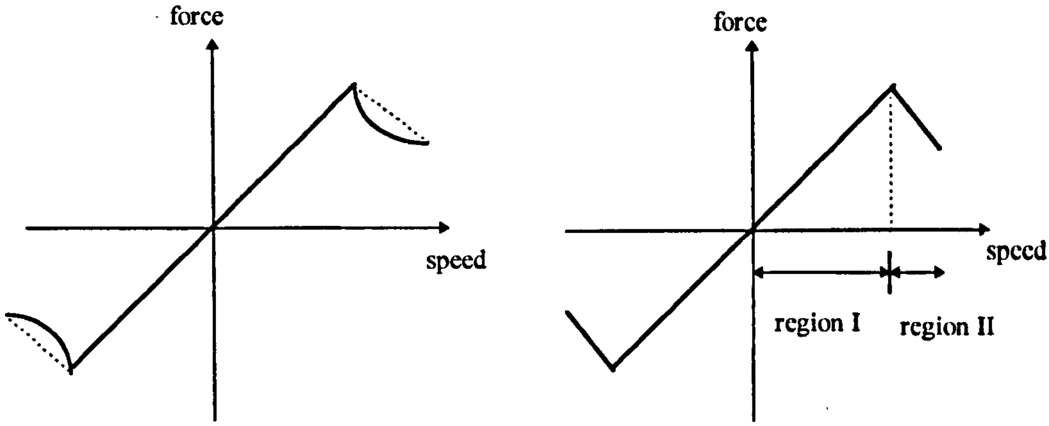


figure (7.1) Piecewise linear approximation of dry friction nonlinearity

methods of analysis are available, which have an appreciable degree of generality. One of these is the *Phase-Plane* method, the other is known as the *Describing Function* method.

7.2.1.2 DESCRIBING FUNCTION

The describing function technique is an extension of the frequency response methods to non-linear systems. It provides a way of investigating limit cycles in the class of feedback systems, which has a single non-linearity. It differs from the methods of Lure and Popov in that the non-linearity is not restricted to the first and third quadrants, and that the method is concerned only with limit cycles not with any other sort of stability. Classically, it is applicable only to non-linear elements whose outputs in response to sinusoidal inputs with period T of the Form

$$A \sin \frac{\pi t}{T}$$

are also periodic with period t, and the input amplitude A.

The output may be written as Fourier series:

$$\sum_{n=1}^{\infty} B_n \sin\ (n \omega t + \phi_n)$$

The *describing function* is the ratio of the complex Fourier coefficient

$$B_1 e^{j\phi_1}$$

which is essentially a frequency response function of an approximation of the non-linear element. In general, B_1 and ϕ_1 are functions of both the input frequency

$$\omega = \frac{2\pi}{T}$$

Once the describing function method has been applied to the non-linear element, linear frequency domain stability theory such as the Nyquist stability criteria can be applied.

7.2.1.3 PHASE-PLANE

Second-order dynamic systems are particularly well suited to graphical analysis because, with only two state variables, the state space is a plane, which is topologically simpler than higher-order spaces and has trajectories that can be represented by single lines on a two dimensional form. In this context the two-dimensional state space is known as the phase plane. Phase plan analysis, like other techniques based on state space, is applicable both to linear and non-linear systems. It is usually restricted to continuous-time systems. Nevertheless, it can lead to useful results about non-linear problems which have no closed-form solution. The basic technique of phase-plane analysis is to consider continuous-time, stationary, second-order systems under conditions where the control **u** is held constant so that the dynamic equations have the form

$$\dot{x}_1 = g_1(x_1, x_2)$$

$$\dot{x}_2 = g_2(x_1, x_2)$$

These equations give an expansion of the form

$$\frac{dx_2}{dx_1} = \frac{\dot{x}_2}{\dot{x}_1} = \frac{g_2(x_1, x_2)}{g_1(x_1, x_2)}$$

determining the slope, and thus the shape, of the trajectories at every point in phase space. Hence it can be concluded that phase-plane analysis is most useful for second- rder systems; it can be used to study transient behaviour subject only to initial conditions (i.e. no other excitation); only time-invariant system can be considered. The performance is plotted with state variables as the coordinates. In general, this technique is not applicable to systems of higher order than 2 or 3, but by extending the concept it can provide insight into the performance of the systems order.

7.2.1.4 LIAPONOV'S SECOND METHOD

The objective of Liapunov's direct method is to answer questions about stability of dynamic systems directly without solving the non-linear differential equations. It is also known as the 'second method', as opposed to the 'first method' which is to linearize about singularities and then consider eigne-values of the resulting linear equations. The first method generally can only provide information about local, rather than global, stability, although topological arguments in the phase plane can yield global, results for second-order systems. The second, or direct, method is introduced by considering stability of continuous-time autonomous systems having an equilibrium point at the origin.

$$g(0) = 0.$$

Other equilibrium points can be investigated by shifting the origin of coordinates in state space to coincide with the point in question. No control **u** appears in these equations because they would be describing 'closed-loop' behaviour of a complete state space to coincide with the point in question. No control **u** appears in these equations because they would be describing 'closed-loop' behaviour of a complete system, not the 'open-loop' behaviour of a controlled process.

The method is to consider the variation with time of the value of some, as yet specified, function V(x) of the state variables, called Liapunov function.

This method can be applied to higher-order systems if the differential equation is written in the first canonic form. However, this closed-loop system must have only one non-linear element.

As it can be seen the above analysis methods check the system for stability, but does not go into the actual design analysis of the non-linear control system itself.

7.2.1.5 PREVIOUS WORK IN NON-LINEAR CONTROL

During the last few years alot of research was done in the field of non-linear systems. For example, A S C Sinha [71], studied the controllability of the non-linear delay systems. Sharky and Oreilly [72], worked on non-linear singularly perturbed systems. Jones and Billing [73], produced a recursive algorithm for computing the frequency response of non-linear difference equation model. Many more researchers [74-90], have worked on different aspects of non-linear control systems.

7.3 SYNTHESIS OF GAIN SCHEDULED PID CONTROLLERS

In the design of a gain scheduled PID controller consider the SISO non-linear system of the respective forms

$$\dot{x}=f(x)+g(x)u \tag{7.1}$$

$$y=h(x) \tag{7.2}$$

where x $\in \Re^n$, u $\in \Re$, y $\in \Re$, f(.) and g(.) are smooth vector fields on an open set U $\subset \Re^n$, f(0)=0, and h(.) is a smooth function on U.

This behaviour is governed by non-linear state and output equations of the respective forms

$$Tx_{k+1}=\Phi(T, y_kT)\,Tx_k+\Psi(T, y_kT)\,Tu_k \tag{7.3}$$

and

$$y_k=\Gamma x_k T \qquad 7.4$$

where the state vector $x_kT \in \Re^n$, the input $u_kT\in\Re^T$,the output $y_kT \in \Re^+$,, $\Phi \in \Re^{nxn}$, $\Psi \in \Re^{nx1}$, $\Gamma \in \Re^{1x}$, $T \in \Re^+$ is the sampling period, and n is the number of plant states.

7.3.1 NON-LINEAR GAIN SCHEDULE PID CONTROLLER

In non-linear control systems the non-linearity is a function of one or more of the following:

i) system input;

ii) system output;

iii) system state.

Therefore it is very important for a designer to be able to identify which of the above is applicable to the gain-scheduled controller being designed. Hence it is evident that some identification process is needed to establish the gain-scheduling variable or variables. The identification process can be done by carrying out step response test on locally linearizied models, and then examining the change of the transfer function variables through out the operating envelope of the plant. The non-linear gain-scheduled PID controller used in this chapter is based on the linear incremental PID controller used in the previous chapters. From chapter 3 the incremental linear PID controller can be described as

$$\Delta u_k = T(k_p \Delta e_k + T k_i e_k + k_d \Delta^2 e_k) \qquad 7.5$$

where Δu_k is the incremental change in input,

Δe_k is the first order backward difference in error,

$\Delta^2 e_k$ is the second order backward difference in error.

k_p is the current value of the proportional gain,

k_i is the current value of the integral gain,

k_d is the current value of the derivative gain,

T is sampling time.

It is evident from chapter 3 that the error can be written as

$$e_k = v_k - y_k \qquad 7.6$$

Also from chapter 3 that the first order backward difference in error can be written as

$$\Delta e_k = e_k - e_{k-1} \qquad 7.7$$

Further more from chapter 3 the second order backward difference in error can be expressed as

$$\Delta^2 e_k = e_k - 2e_{k-1} + e_{k-2} \qquad 7.8$$

The incremental PID controller given by equation (7.5), can take one of two forms, one is linear, and the other is non-linear. In this chapter the non-linear form is to be investigated. In this case the non-linearities are a function of the plant output. It follows from equation (7.5), that the incremental non-linear PID controller can be described by an equation of the form

$$\Delta u_k = \pi_p(S_k)\,\Delta e_k + T\,\pi_i(S_k)\,e_k + \pi_d(S_k)\,\Delta^2 e_k \qquad 7.9$$

where $\pi_p(S_k)$ is a gain-schedule function representing proportional gain,

$\pi_i(S_k)$ is a gain-schedule function representing integral gain,

$\pi_d(S_k)$ is a gain-schedule function representing derivative gain.

It is evident from equation (7.9), that the gain functions can be represented by equation of the form

$$\mathcal{K}_p = \pi_p(S_k) \qquad 7.10$$

$$\mathcal{K}_i = \pi_i(S_k) \qquad 7.11$$

$$\mathcal{K}_d = \pi_d(S_k) \qquad 7.12$$

The issue in this case is how to design the gain functions. From previous work in chapter 4 it was found that an efficient function to represent the gain function in this way was the Lagrangian's Polynomial. Hence, from the previous finding the above gain-schedule functions are to be based on the Lagrangian's polynomial interpolation function. It follows from equation (7.5), that the incremental non-linear PID controller can be conveniently described by equation of the form

$$\Delta u_k = \mathcal{K}_p\,\Delta e_k + T\mathcal{K}_i\,e_k + \mathcal{K}_d\,\Delta^2 e_k \qquad 7.13$$

The diagram in figure (7.3) shows a block diagram representing the non-linear gain-scheduled PID control system.

7.3.1.1 GAIN SCHEDULE (INTERPOLATION) FUNCTION

The scheduling function chosen in this chapter is the smooth Lagrangian's function with overlapping to cover a minimum of 4 points. In this function the scheduling function is always working on four points i.e. if the gain being scheduled happened to be gain number 4, then the gains which the function would schedule would be 1 to 4, but when the gain is gain number 5, then the gains that the function would schedule are gains 2 to 5. The function used is an overlapping function which would give an extra smoothness to the function, i.e. no discontinuities between interpolated gains. It is also evident that the minimum number of gains in the design has to be 4 gains, the maximum only depends on the design requirement. The gain schedule function is thus given by:

$$g_i = \frac{(S_{kk} - S_{i+1})(S_{kk} - S_{i+2})(S_{kk} - S_{i+3})}{(S_i - S_{i+1})(S_i - S_{i+2})(S_i - S_{i+3})} \qquad 7.14$$

The above equation is the first element in the gain schedule function

$$g_{(i+1)} = \frac{(S_{kk} - S_i)(S_{kk} - S_{i+2})(S_{kk} - S_{i+3})}{(S_{i+1} - S_i)(S_{i+1} - S_{i+2})(S_{i+1} - S_{i+3})} \qquad 7.15$$

The above equation is the second element in the gain schedule function

$$g_{(i+2)}=\frac{(S_{kk}-S_{i})(S_{kk}-S_{i+1})(S_{kk}-S_{i+3})}{(S_{i+2}-S_{i})(S_{i+2}-S_{i+1})(S_{i+2}-S_{i+3})} \qquad 7.16$$

The above equation is the third element in the gain schedule function

$$g_{(i+3)}=\frac{(S_{kk}-S_{i})(S_{kk}-S_{i+1})(S_{kk}-S_{i+2})}{(S_{i+3}-S_{i})(S_{i+3}-S_{i+1})(S_{i+3}-S_{i+2})} \qquad 7.17$$

The above equation is the 4rth element in the gain schedule function.

By using the above equations (7.18, 7.19, 7.20, and 7.21) the Lagrangian's function is thus, given to be

$$\mathscr{K}= \; g_{i}k^{i}+g_{i+1}k^{(i+1)}+g_{i+2}k^{(i+2)}+g_{i+3}k^{i+3} \qquad 7.18$$

where i= (1,2,....n) , and n number of gains,

$\mathscr{K}$ is the effective gain-schedule function ,

k_i to k_n are the gains provided by the GA,

S_1 to y_n are the interpolation points,

S_{kk} is the actual output of the system at a given gain.

where S_k is Schedule variable

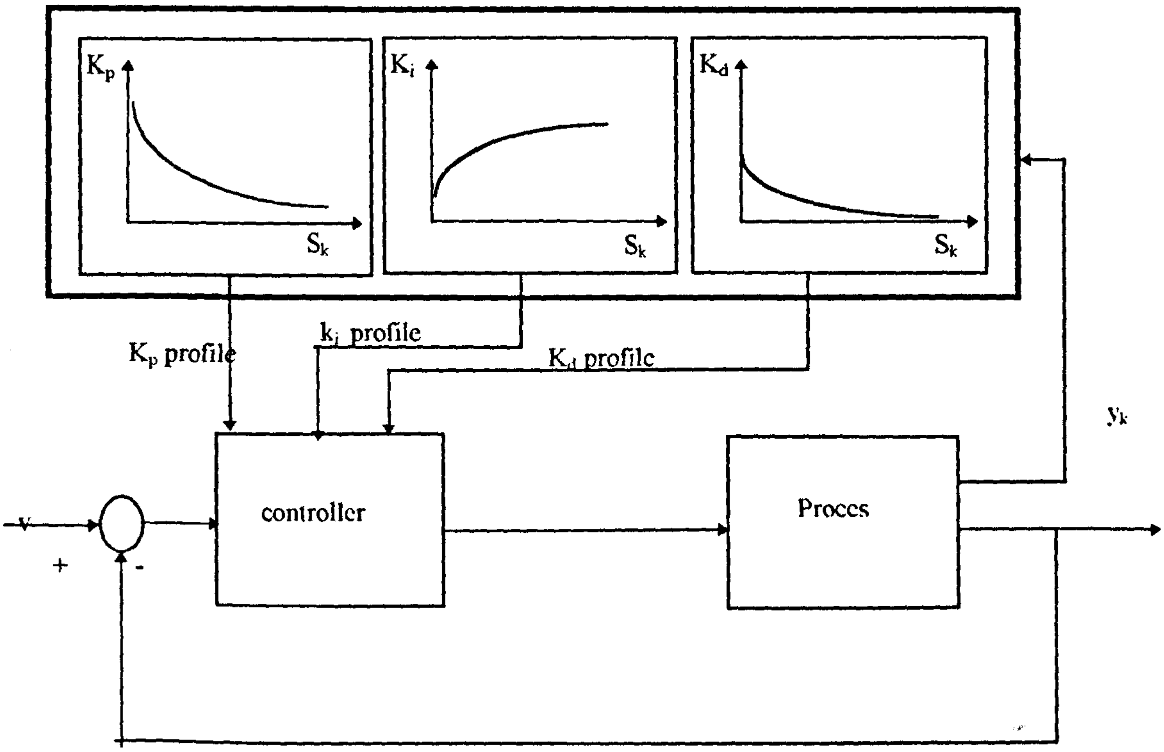


Figure 7.2 Polynomial interpolation of PID Gain scheduled Controller

The above equation gives the Lagrangian's polynomial gain-schedule function equation used in this chapter. The mapping of the gain-schedule function is shown in figure (7.4). It evident that the above method is very difficult to do by using conventional design methods. In order to overcome the difficulty, the technique of genetic algorithms is therefore proposed as a new and novel technique for designing and tuning the gain scheduled PID controller.

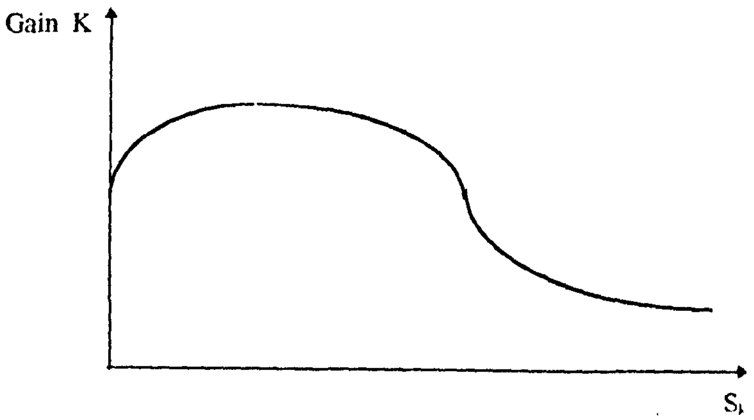


Figure 7.4 Gains mapping using Polynomial Interpolation

7.4 GENETIC DESIGN OF GAIN SCHEDULED CONTROLLER FOR NON-LINEAR PLANTS

In order to use genetic algorithms to design a gain scheduled PID controllers, the number of parameters (gains), used in the design of such controller have to be precisely defined. To assess the performance of the gain scheduled controller it will be necessary to consider a number of plants controlled by the genetically designed gain-scheduled controllers, and the genetically designed linear PID controller. In order to use the GA to select the tuning parameters (gains) in such a way that as to produce satisfactory response in the operating envelope, it is necessary to firstly chose the scheduling variables and then encode the gains required by the gain-scheduled functions i.e. (proportional, integral, and derivative gains), as binary strings in accordance with a system of concatenated, multi-parameter fixed point coding. The binary string would be represented as

$$\{ \ \underset{\text{proportional}}{(k^1_p \ldots . \ k^n_p)} \ \ \underset{\text{integral}}{(k^1_i \ldots . \ k^n_i)} \ \ \underset{\text{derivative}}{(k^1_d \ldots . \ k^n_d)} \ \}$$

where the whole string contains the three gains (proportional, integral, and derivative gains). The mapping of the string is shown in figure (7.5). Then following the initial choice of the gains $\{ k^0_p, k^1_p, \ldots , k^n_p, k^0_i, k^1_i, \ldots , k^n_i, k^0_d, k^1_d, \ldots , k^n_d \}$, entire generations of such strings can be readily obtained by using the basic genetic operators of selection, crossover, and mutation. The process of interviewing introduced in chapter 2 is used to insure that the randomly generated variables do not initially violate any constraints on the function to be tuned. The system incorporates both the non-linear plant, and the non-linear gain scheduled PID controllers, the controllers are designed by randomly generated sets of gains, by the GA. A simulation test is then carried out on all the controllers to test the local stability. In the case of a violation of stability in any of the cases, the randomly generated set of gains, will not be included in the initial population. This type of initialization is essential in this type of design, since otherwise there would be a danger of creating an initial population which many of its members violating the constraints on the controllers being designed.

Following the initialization, the objective function is introduced and its value is calculated using decoded values of the parameters in each string. Thus for example, if minimum ISE is regarded as the ultimate design requirement, genetic algorithms can be readily used to select the best set of gains, such that the ISE is minimised. In the genetic design of non-linear gain scheduled PID controllers the plant under consideration is subjected to a succession of set-point changes which span the operating envelope of the plant, and the generalised ISE is then obtained by adding the individual performance from each set-point change to obtain

$$ISE = \sum_{1}^{m} [\sum_{j=0}^{N} e^{2}_{j}] \qquad 7.19$$

where

$e_j = v_i - y_j$, and $N = \frac{\tau}{T}$ m=number of steps.

and $e_j \in \Re$, is the error signal, $y_j \in \Re$,is the output signal, T is sampling time, τ is an appropriately chosen settling time, and Δ_i (i=1,...m) are the set-point changes.

Minimization of this performance index over the entire population can be rapidly obtained by using the genetic operations of reproduction, crossover, and mutation. It is interesting to note that the conditions for the existence of a non-linear PID controllers for the plant under consideration will be automatically satisfied by the genetic algorithms. This is obvious, because in the case of violation of the constrained the corresponding set of gains elements produces large ISE and the result will be zero or low fitness, and by the action of the selection operator it would not be chosen for the next generation.

The above performance index can be modified slightly by adding a weighting factor λ. This weighting factor can be chosen to increase or decrease the performance of the plant in certain operating points within the operating envelope. The new performance index is thus given by

$$ISE = \sum_{i=1}^{m} [\sum_{j=0}^{N} \lambda_i e^2{}_j] \qquad 7.20$$

The designs done in this chapter will use both of the above performance indexes.

7.5 ILLUSTRATIVE EXAMPLES

To design the gains-scheduled controller the gain schedule functions produced earlier in this chapter will be used to produce the gain-schedule functions for the gains needed by the gain-scheduled PID controller as in equation (7.13).

The gain schedule function for the three gains (proportional, integral, and derivative gains) is thus given by

$$g_i = \frac{(S_{kk} - S_{i+1})(S_{kk} - S_{i+2})(S_{kk} - S_{i+3})}{(S_i - S_{i+1})(S_i - S_{i+2})(S_i - S_{i+3})} \qquad 7.14$$

where in the above equation the first element in the gain schedule function

$$g_{(i+1)} = \frac{(S_{kk} - S_i)(S_{kk} - S_{i+2})(S_{kk} - S_{i+3})}{(S_{i+1} - S_i)(S_{i+1} - S_{i+2})(S_{i+1} - S_{i+3})} \qquad 7.15$$

where in the above equation the second element in the gain schedule function

$$g_{(i+2)} = \frac{(S_{kk} - S_i)(S_{kk} - S_{i+1})(S_{kk} - S_{i+3})}{(S_{i+2} - S_i)(S_{i+2} - S_{i+1})(S_{i+2} - S_{i+3})} \qquad 7.16$$

where in the above equation the third element in the gain schedule function

$$g_{(i+3)} = \frac{(S_{kk} - S_i)(S_{kk} - S_{i+1})(S_{kk} - S_{i+2})}{(S_{i+3} - S_i)(S_{i+3} - S_{i+1})(S_{i+3} - S_{i+2})} \qquad 7.17$$

where above equation represents the 4rth element in the gain schedule function.

By using the above equations the resulting Lagrangian's functions for the three gain functions are of the form

$$\mathcal{K}_p = g_i k_p{}^i + g_{i+1} k_p{}^{(i+1)} + g_{i+2} k_p{}^{(i+2)} + g_{i+3} k_p{}^{(i+3)} \qquad 7.21$$

$$\mathcal{K}_p = g_i k_i{}^i + g_{i+1} k_i{}^{(i+1)} + g_{i+2} k_i{}^{(i+2)} + g_{i+3} k_i{}^{(i+3)} \qquad 7.22$$

$$\mathcal{K}_d = g_i k_d{}^i + g_{i+1} k_d{}^{(i+1)} + g_{i+2} k_d{}^{(i+2)} + g_{i+3} k_d{}^{(i+3)} \qquad 7.23$$

and (i=1,....N) where N is the number of gains used in the design.

The above equations represent the scheduling function for the three gains (proportional, integral, and derivative gains) respectively.

To be able to start the design, the GA will need some initial gains, the initial gains can be obtained by using the GA to design and tune some fixed local controllers.

A fixed gain linear controller was designed and tuned by the GA, the GA binary string is of the form

$$\{\ \underset{\text{proportional}}{k_p} \qquad \underset{\text{integral}}{k_i} \qquad \underset{\text{derivative}}{k_d}\ \}$$

The fixed gains obtained were used to initialise the gain scheduled controller, where the GA is minimising the ISE across the operating envelope using the following gain functions. In the gain-schedule PID controller the gains for the three gains(proportional, integral, and derivative gains) were encoded into the GAs binary sting as

$$\{\ \underset{\text{proportional}}{(k^0_p. \ldots k^9_p)} \quad \underset{\text{integral}}{(k^0_i. \ldots k^9_i)} \quad \underset{\text{derivative}}{(k^0_d. \ldots k^9_d)}\ \}$$

As it can be seen from the string their are 10 parameters (gains) needed for each of the gain functions, which means that the GA will be searching for 30 Parameters (gains) in total.

7.5.1 NON-LINEAR PLANT 1 (GAIN SCHEDULED OFF THE OUTPUT)

The SISO nonlinear plant used to illustrate the design of gain scheduled controller is a tank system where the cross section A varies with height *h*. The model is given by equation

$$\frac{d}{dt}(A(h)h(t)) = q_i(t-\tau) - a\sqrt{2gh}\,(t) \qquad 7.24$$

where

q_i is the input flow;

a is the cross section of the outlet pipe;

h is the output of the system;

T is the sampling time;

τ is the time delay.

The plant Arma model is given by

$$y_k = y_{k-1} - a\ \sqrt{y}_{k-1} + b\ u_{k-\frac{\tau}{T}} \qquad 7.29$$

and "a" , "b", and τ/T are constant chosen for the design to be

a = 8;

b = 0.08.

$$\frac{\tau}{T}=10$$

The gain scheduled controller is given by equation (7.13), and is governed by equations (7.21, 7.22, and 7.23). The controller is deployed in the incremental form so as to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed.

7.5.1.1 LOCALLY LINEARIZED DESIGN

A number of points were chosen to cover the whole search envelope of the plant, the gains for this controller (K_p, K_i and K_d) are chosen to be linear gains . Then at each of the points a fixed point linear PID controller for the non-linear plant was designed using the GA, such that the integral Square Error ISE is minimised for the non-linear plant. In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m = 0.01, was used. The results obtained for the set-point change design are shown in table (7.1). All ISEs in table (7.1) are multiplied by E5.

No	1	2	3	4	5	6	7	8	9	10
Size	0 - 1	1 - 2	2 - 3	3 - 4	4 - 5	5 - 6	6 - 7	7 - 8	8 - 9	9-10
ISE	1.19	1.32	1.34	1.42	1.46	2.3	5.43	5.7	7.4	10.1
k_p	1.03	0.79	0.99	1.27	1.42	1.23	1.34	1.34	1.47	1.49
k_i	1.12	0.49	0.43	0.51	0.5	0.59	0.63	0.63	0.64	0.65
k_d	0.67	0.97	0.73	0.5	0.5	0.63	0.63	0.63	0.61	0.59

Table (7.1)

The total ISE for the above design method is 3.766E6

7.5.1.2 GLOBAL NON-LINEAR DESIGN OF GAIN SCHEDULED CONTROLLERS

In this design, the fixed gains obtained from the local set-point change design done earlier are used to initialise the GA. For the purpose of this design the gain functions for the gain scheduled controller are chosen to be as given by equations (7.21, 7.22, 7.23), i.e. the gains are non-linear and they are Lagrangian's gain schedule functions. The gain-scheduled controller is implemented in the incremental form so as to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then non-linear gain scheduled PID controller for the non-linear plant was designed using the GA, such that the integral Square Error ISE is minimised for the non-linear plant. In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m = 0.01, was used. Figure 7.6 shows the transient response of the genetically designed controller using performance index 7.24, figure 7.6a, 7.6b, and 7.6c shows the resulting profile for the gain-schedule gains, proportional, integral, and derivative respectively. Figure 7.7 shows the transient response of the genetically designed controller using performance index 7.23, figure 7.7a, 7.7b, and 7.7c shows the resulting profile for the gain-schedule gains, proportional, integral, and derivative respectively.

Total ISEs in table (7.2) is 1.22E6.

zone	0 - 1	1 - 2	2 - 3	3 - 4	4 - 5	5 - 6	6 - 7	7 - 8	8 - 9	9-10
ISE	1.23	1.13	1.11	1.06	1.2	1.12	1.17	1.25	1.42	1.5
$\mathcal{K}_p$	1.29	3.84	4.9	3.85	3.12	4.88	4.19	0.44	2.29	3.1
$\mathcal{K}_i$	0.93	1.86	0.98	3.99	3.0	4.15	4.14	4.59	1.52	2.38
$\mathcal{K}_d$	3.92	2.04	1.24	1.07	4.64	1.30	1.81	4.19	3.62	1.45

Table (7.2)

Table (7.2) shows gain-schedule controller results using performance index given by equation (7.33). Table (7.3) shows gain-schedule controller results using performance index given by equation(7.34). All ISEs in table (7.3) are multiplied by 1E5, and the total ISE is 1.1356.

zone	0 - 1	1 - 2	2 - 3	3 - 4	4 - 5	5 - 6	6 - 7	7 - 8	8 - 9	9-10
ISE	1.12	1.1	1.00	1.03	0.93	0.89	1.14	1.24	1.4	1.5
$\mathcal{K}_p$	2.23	3.54	3.4	3.85	3.12	3.28	3.39	2.44	2.59	2.1
$\mathcal{K}_i$	0.93	1.36	1.48	1.99	2.2	2.35	2.14	3.20	3.22	3.38
$\mathcal{K}_d$	1.11	1.23	1.24	1.27	1.24	1.20	1.21	1.24	1.22	1.26

Table (7.3)

From the above results, it can be seen that the total ISE obtained by designing a locally linearized PID controller for the non-linear plant is a lot higher than that obtained by the gain scheduled controller design for the same non-linear controller.

7.5.2 NON-LINEAR PLANT 2 (GAIN SCHEDULED OFF THE INPUT)

The SISO nonlinear plant used to illustrate the design of gain scheduled controller is a concentration control of a tank system. Where the inlet concentration, c_{in} is changed. The model is given by equation

$$V_m \frac{dc_t}{dt} = q_t \; (c_{in}(t-\tau) - c_t) \qquad 730$$

where

$$\tau = \frac{v_d}{q_t} \quad \text{and} \quad T = \frac{V_m}{q_t}$$

if τ < T, then it is straightforward to determine a PID controller that performs well when q is constant. However it is difficult to find values of the controller parameters that will work well for a wide ranges of q (J. Astrom and B. Wittenmark [103]).

The process has a time delay, with sampling period

$$h = \frac{v_d}{dq}$$

where d is an integer, the plant Arma model is given by

$$y_k = ay_{k-1} + (1-a)\,u_{k-h} \qquad 7.31$$

where a and h are given by

$$h = int\left(\frac{K_1}{u}\right) , \qquad a = e^{-uK_2} ,$$

where K_1 and K_2 are constants.

$K_1 = 1$

$K_2 = 0.01.$

The gain scheduled controller is given by equation (7.13), and is governed by equations (7.21, 7.22, and 7.23). The controller is deployed in the incremental form so as to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed.

7.5.2.1 LOCALLY LINEARIZED DESIGN

A number of points were chosen to cover the whole search envelope of he plant, the gains for this controller (K_p, K_i and K_d) are chosen to be linear gains . Then at each of the points a fixed point linear PID controller for the non-linear plant was designed using the GA, such that the integral Square Error ISE is minimised for the non- linear plant. In this case, a population of 100, a crossover probability, $P_c=0.65$, and a mutation probability, $P_m = 0.01$, was used. The results obtained for the set-point change design are shown in table (7.4). Wher all the ISEs in table (7.4) are multiplied by E5.

No	1	2	3	4	5	6	7	8	9	10
Size	0 - 1	1 - 2	2 - 3	3 - 4	4 - 5	5 - 6	6 - 7	7 - 8	8 - 9	9-10
ISE	6.3	5.9	5.1	4.8	4.4	3.2	2.8	2.4	2.2	1.9
k_p	3.5	3.4	4.5	8.7	4.73	4.5	9.9	9.6	9.4	8.5
k_i	3.4	3.4	3.2	1.8	2.9	3.1	1.5	2.4	2.8	3.1
k_d	1.96	2.1	1.5	2.2	4.1	4.5	7.7	7.5	7.2	6.2

Table (7.4)

The total ISE for the above design method is 3.9E6.

7.5.2.2 GLOBAL NON-LINEAR DESIGN OF GAIN SCHEDULED CONTROLLERS

In this design, the fixed gains obtained from the local set-point change design done earlier are used to initialise the GA. For the purpose of this design the gain functions for the gain scheduled controller are chosen to be as given by equations (7.32, 7.33, 7.34), i.e. the gains are non-linear and they are Lagrangian's gain schedule functions. The gain-scheduled controller is implemented in the incremental form so as to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then non-linear gain scheduled PID controller for the non-linear plant was designed using the GA, such that the integral Square Error ISE is minimised for the non-linear plant. In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m = 0.01, was used. Figure 7.8 shows the transient response of the genetically designed controller using performance index 7.24, figure 7.8a, 7. 8b, and 7.8c shows the resulting profile for the gain-schedule gains, proportional, integral, and derivative respectively. Figure 7.9 shows the transient response of the genetically designed controller using performance index 7.23, figure 7.9 a, 7.9b, and 7.9c shows the resulting profile for the gain-schedule gains, proportional, integral, and derivative respectively. The result obtained from both design methods are given in table (7.5).

zone	0 - 1	1 - 2	2 - 3	3 - 4	4 - 5	5 - 6	6 - 7	7 - 8	8 - 9	9-10
ISE	1.2	1.1	1.2	1.3	1.4	1.2	1.2	1.1	1.3	1.4
$\mathcal{K}_p$	1.21	2.49	2.66	2.82	2.24	1.76	2.82	2.75	2.76	2.33
$\mathcal{K}_i$	0.57	0.34	0.40	0.42	0.46	0.56	0.32	0.22	0.25	0.45
$\mathcal{K}_d$	0.21	0.23	0.24	0.27	0.24	0.20	0.21	0.24	0.22	0.26

Table (7.5)

total ISE is 1.24E6.

Table (7.5) shows gain-schedule controller results using performance index given by equation (7.33). Table (7.6) shows gain-schedule controller results using performance index given by equation(7.34)

zone	0 - 1	1 - 2	2 - 3	3 - 4	4 - 5	5 - 6	6 - 7	7 - 8	8 - 9	9-10
ISE	1.3	1.1	1.0	1.1	1.0	1.1	1.1	1.0	1.1	1.2
$\mathcal{K}_p$	1.32	1.54	1.87	1.91	1.53	1.96	2.03	2.25	2.66	2.71
$\mathcal{K}_i$	0.74	0.21	0.14	0.12	0.26	0.38	0.32	0.18	0.15	0.1
$\mathcal{K}_d$	0.19	0.17	0.23	0.22	0.12	0.23	0.26	0.12	0.12	0.10

Table (7.6)

total ISE is 1.19E6.

From the above results, it can be seen that the total ISE obtained by designing a locally linearised PID controller for the non-linear plant is a lot higher than that obtained by the gain scheduled controller design for the same non-linear controller.

7.5.3 ROBUSTNESS TEST

This test is aimed at finding how robust are the genetically designed gain-scheduled PID controllers are for changes in the plant operating conditions. In this case the plants used in the illustrative example were modified slightly to produce two further plants, by increasing and decreasing the step-size. The resulting responses are shown in figure (7.10 and 7.11) respectively. The resulting ISEs are given in table (7.7).

Plant	1	3 (step-size 150)	4 (step-size 50)
ISE	1.19E6	1.36E6	3.3E6

table(7.7)

7.6 CONCLUSIONS

The techniques of genetic algorithms have been proposed as a means of designing non-linear PID controllers for non-linear plants. It has been shown that the use of GAs for this purpose greatly facilitates the design of such controllers such that the integral square error to set point changes across the operating envelope of the plant is minimised. The results have been illustrated by genetically designing a gain scheduled controllers for two discrete-time non-linear plants. the resulting robustness test indicates that the non-linear PID controllers designed are robust for different size set-point changes.

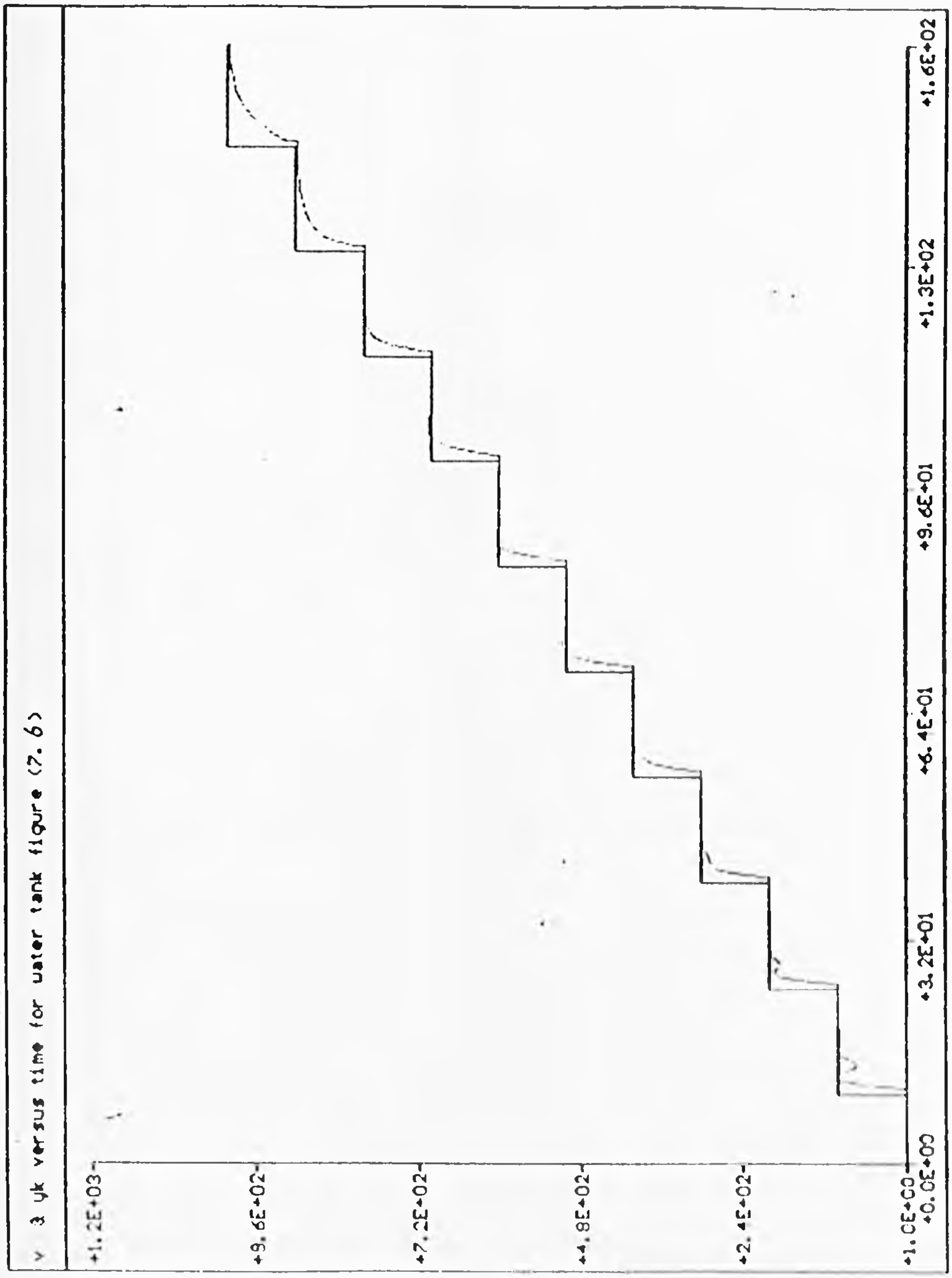


Figure (7.6) Transient response of the genetically designed controller using performance index 7.24 for plant (1).

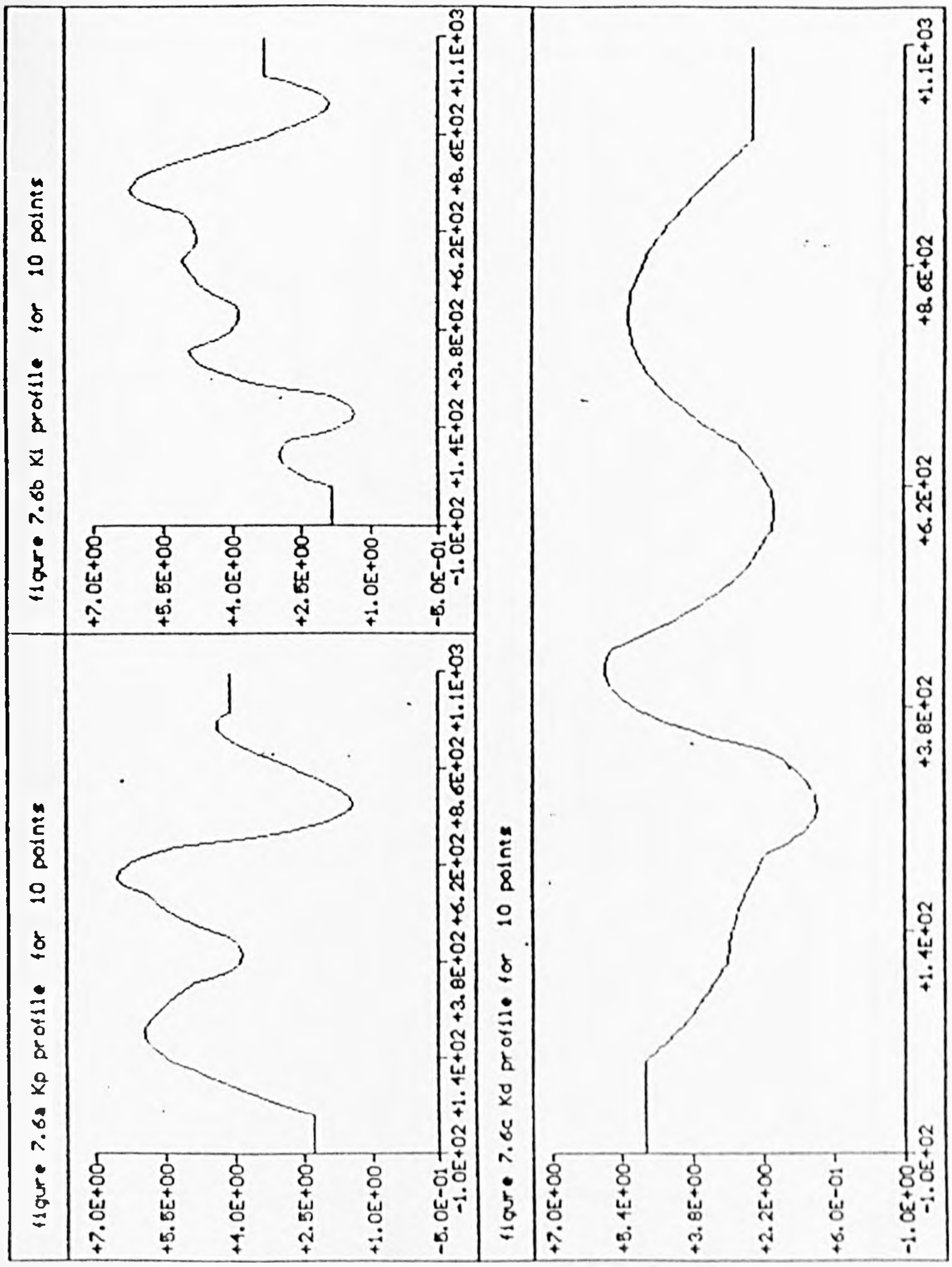


Figure 7.6a, b,and c Resulting gain profile for gain scheduled controller in fig(7.6)

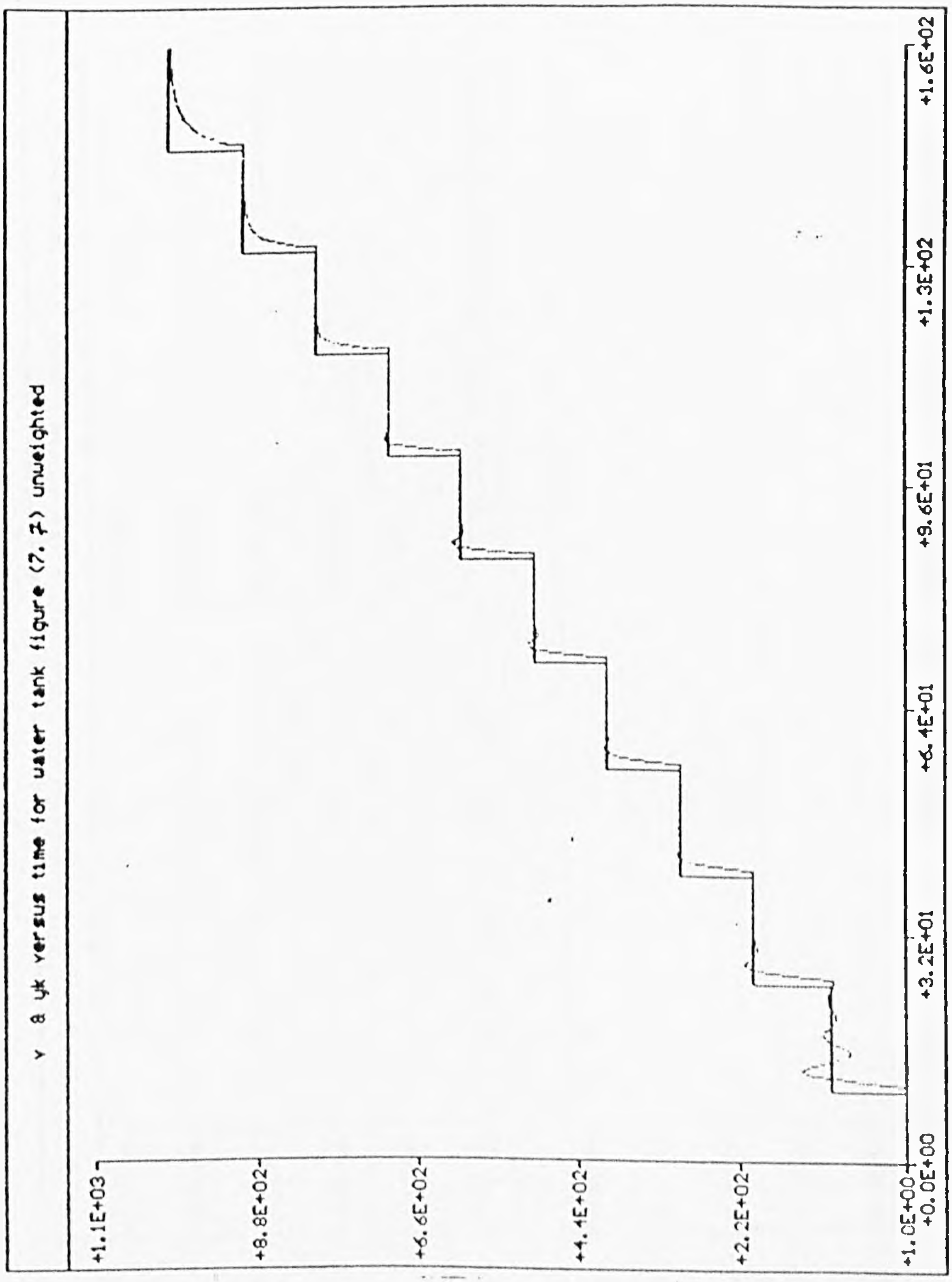


Figure (7.7) Transient response of the genetically designed controller using performance index 7.23 for plant (1).

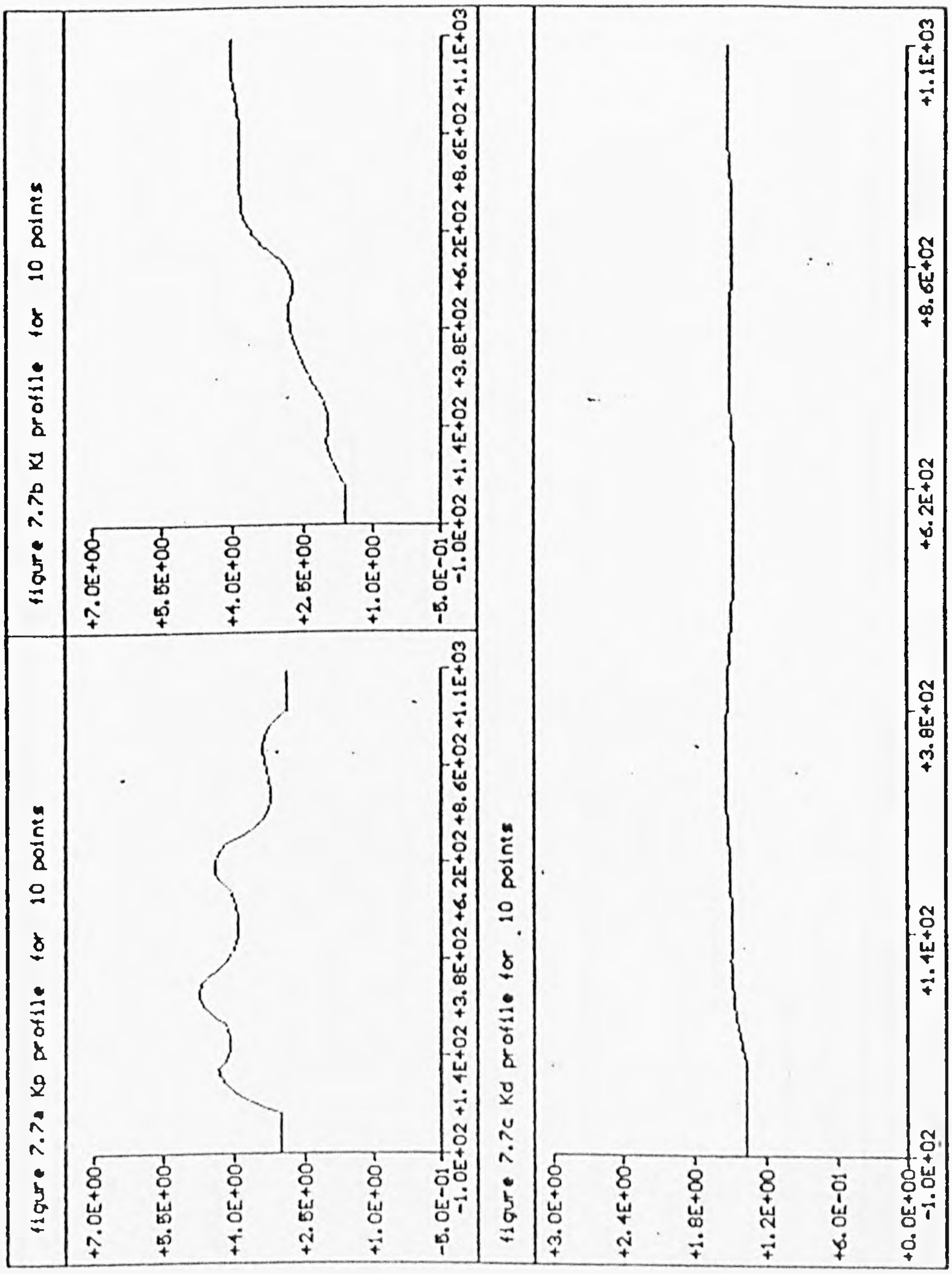


Figure 7.7a, b,and c Resulting gain profile for gain scheduled controller in fig(7.7)

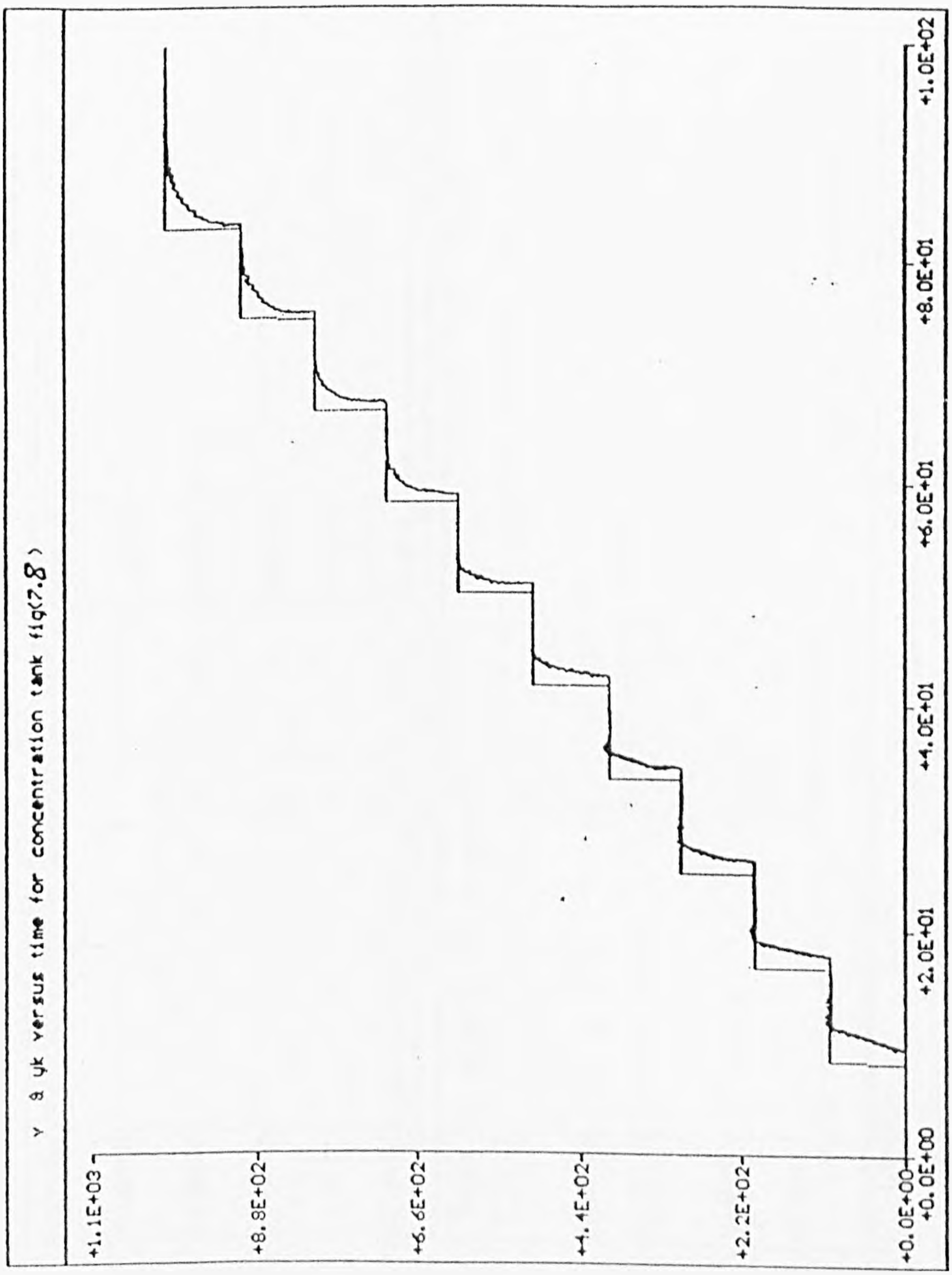


Figure (7.8) Transient response of the genetically designed controller using performance index 7.24 for plant (2).

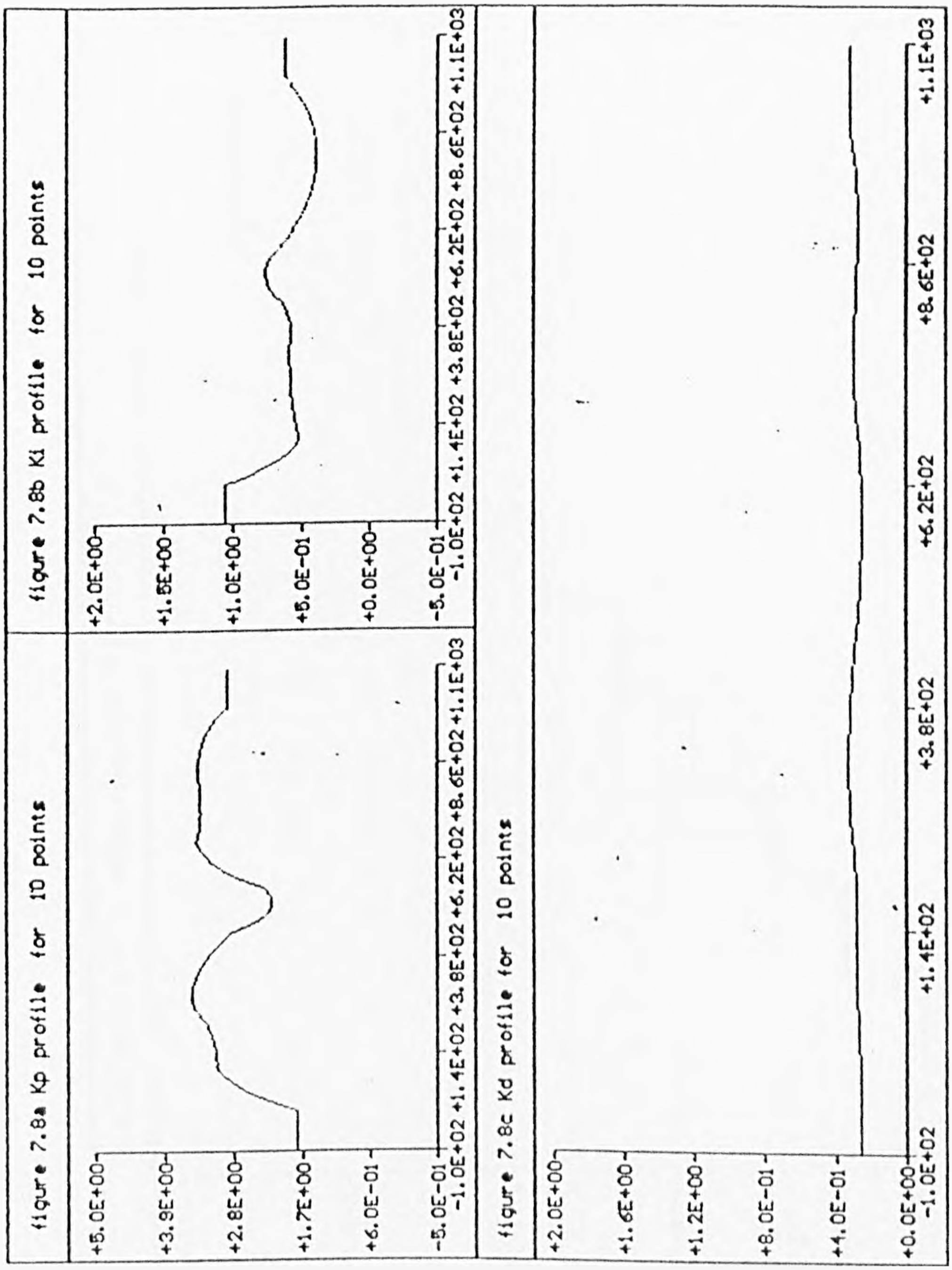


Figure 7.8a, b,and c Resulting gain profile for gain scheduled controller in fig(7.8)

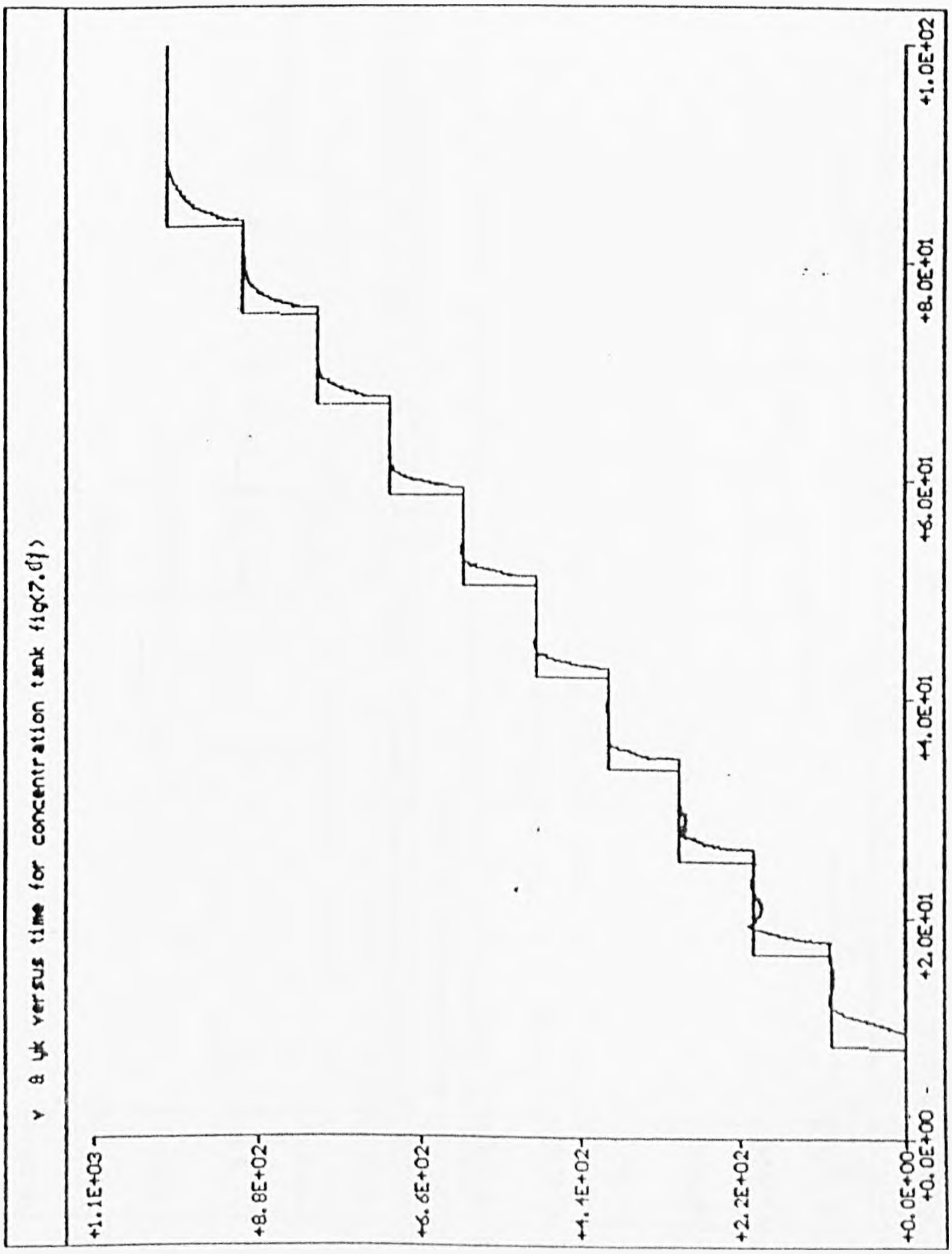


Figure (7.9) Transient response of the genetically designed controller using performance index 7.23 for plant (2).

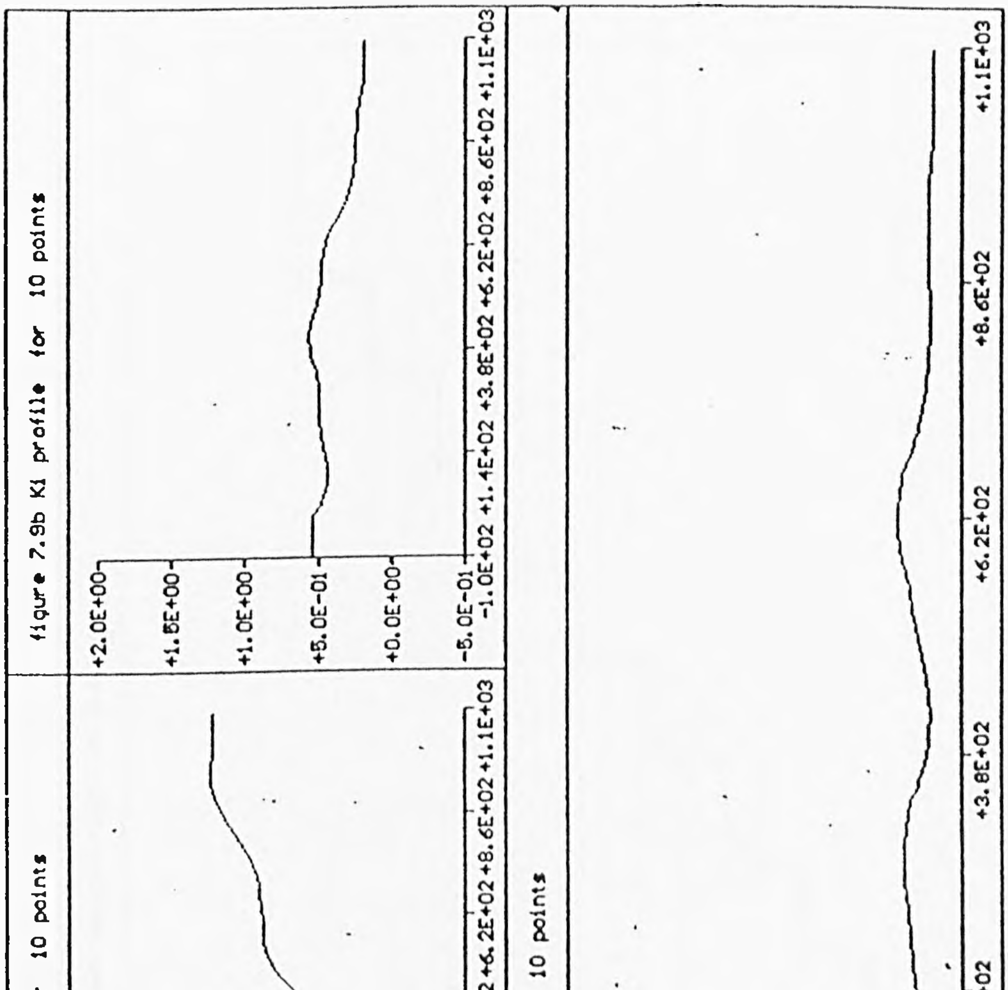


Figure 7.9a, b,and c Resulting gain profile for gain scheduled controller in fig(7.9)

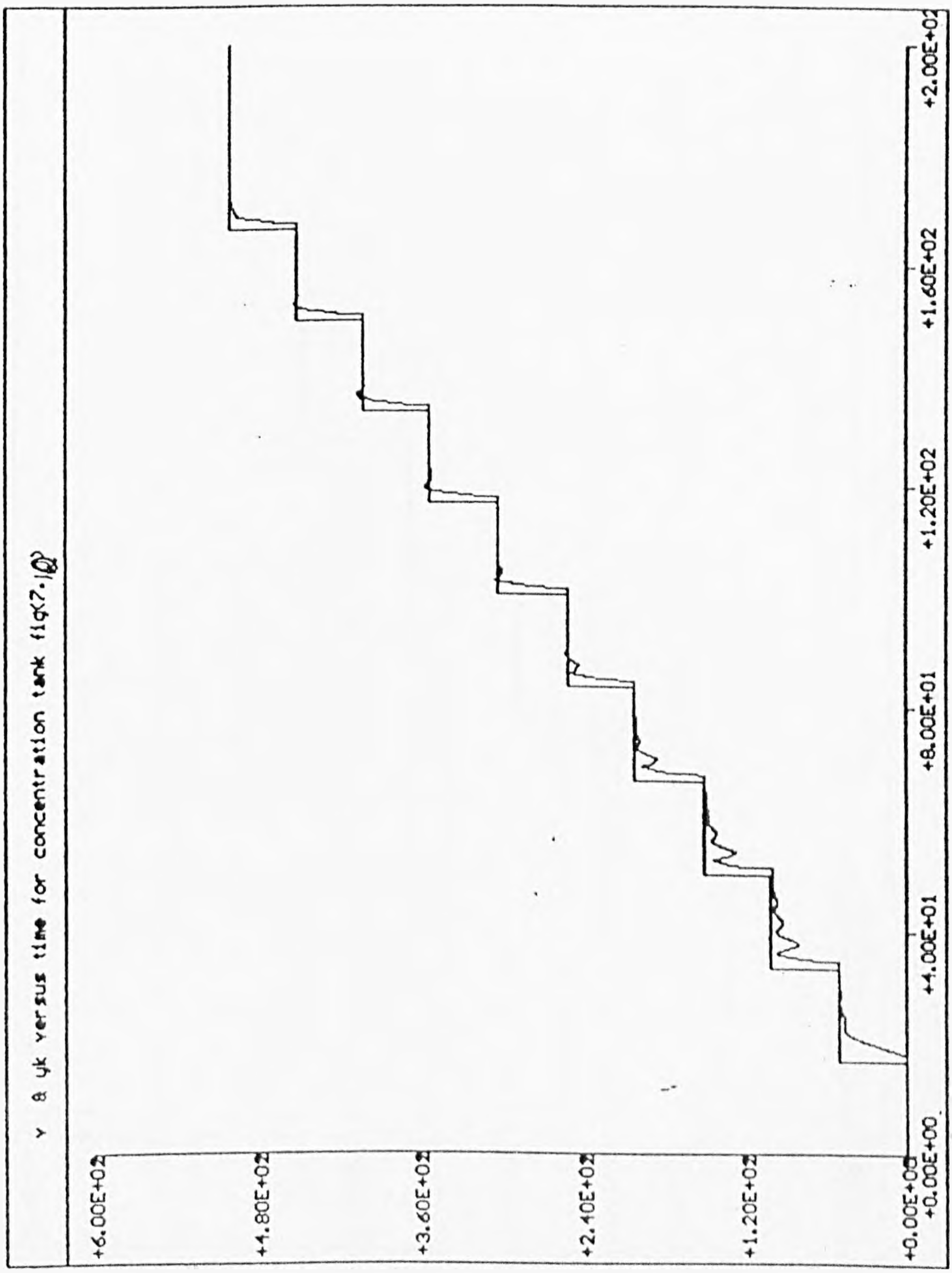


Figure (7.10) Transient response of rubstness test for step size increase.

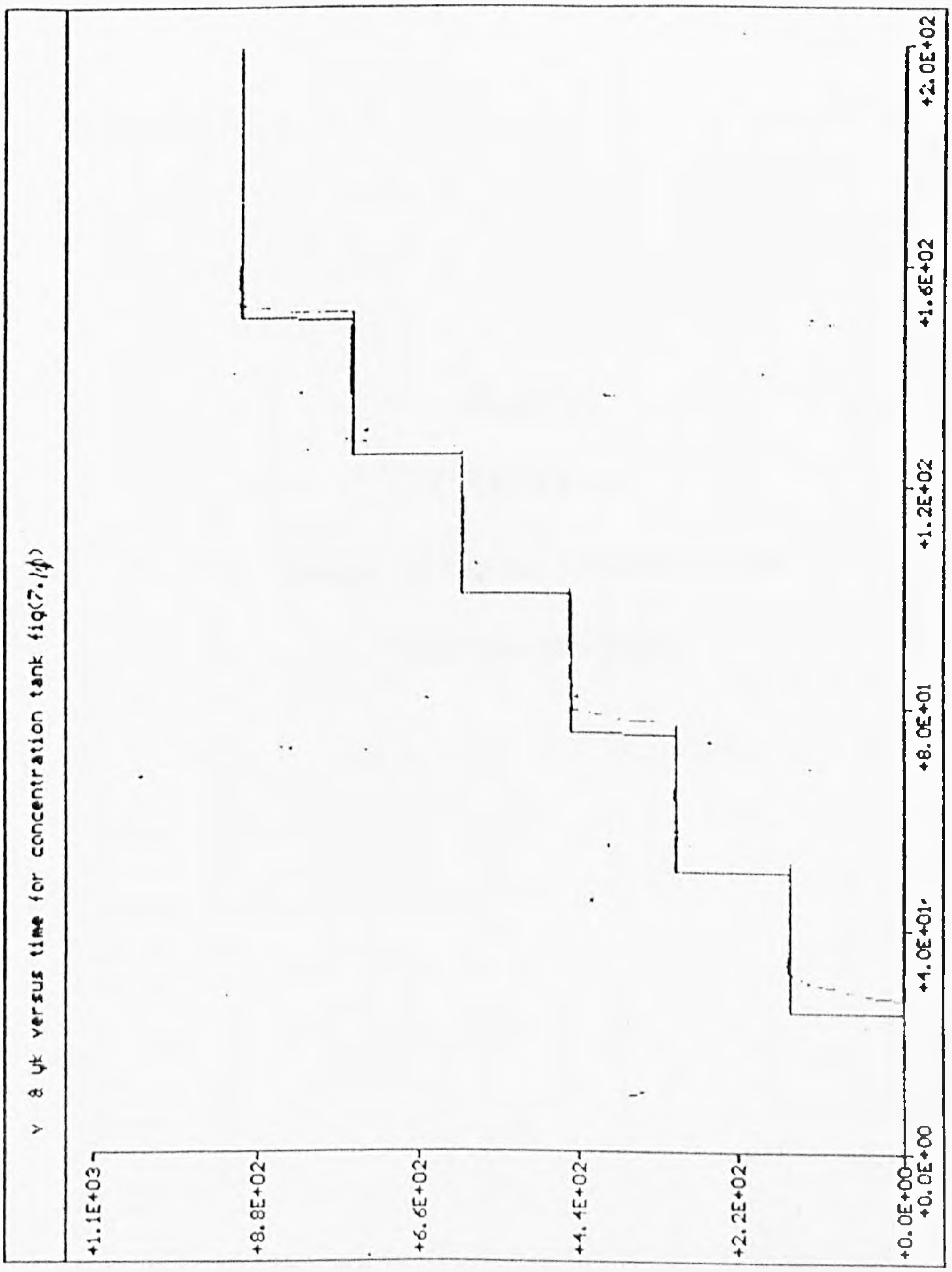


Figure (7.11) Transient response of rubstness test for decrease in step size.

PART IV

CHAPTER 8

Design of Linear Controllers for Non-Linear Plants

/

Chapter 8
GENETIC DESIGN OF A FIXED GAIN LINEAR CONTROLLERS FOR NON-LINEAR PLANTS

8.1 INTRODUCTION

The previous chapter introduced the concept of using GAs to design non-linear controllers for non-linear plants. In this chapter it is proposed to use genetic algorithms to design fixed gain linear controllers for non-linear plants. In general it is a very difficult task to design a fixed gain linear controller for a non-linear plant. This is because the gains of the controller need to be tuned to cope with the full operating envelope of non-linear plant. Hence, the design of the linear controller requires gains, that would give it the ability to control the plant satisfactorily throughout the operating envelope under all the different possible non-linear conditions . It is evident that the resulting controller can not be an optimum controller. However, it could be considered as a Locally Optimised, Worst Point, Globally Validated (LOWPGV) fixed term PID controller. Where the context of the term "optimised" refer to a controller that provides the best level of local control at the worst operating point of the non-linear system, such that the local performance of the fixed term PID controller everywhere else in the operating envelope of the non-linear system is better than the design at the worst point. Therefore in this chapter a new cost function is formulated for developing (LOWPGV) fixed term PID controllers. The cost function or performance index is defined as the minimisation of the worst local performance of the controller throughout the operating envelope. This implies that the non-linear system will be divided into a number of local operating regions which totally cover the non-linear characteristics of the system. The cost function then takes the maximum of the local cost functions as its cost. By defining the cost function in this way the design of the controller is concentrated on the difficult parts of the operating envelope. The resulting controller controls the plant at the worst operating point, and attempts to minimise this local performance.

From the definition of the cost function it follows that if the controller is used at any other points within the operating envelope of the performance of the controller is better than at the worst point. The resulting controller does not provides optimum global control in the sense of minimising a global cost function. Such a cost function could give a broad spectrum of local responses, some of which could be very good with others very bad. However, the resulting controller does provide optimum local control at the worst operating point, such that control everywhere else in the operating envelope is better.

This cost function is one that is very much sought after in the design of fixed term PID controller for non-linear plants. However, if the design was to be effected by a trial and error technique the search would have to proceed at three fronts, the first one looking for the worst operating point, the second looking to tune the controller at the worst operating point, and the third checking that the controller tuned at the worst operating point was still better at all the other operating points. Because of the multi-objective nature of the problem trial and error methods in this case would be very difficult. It is evident that the design of such controller is almost impossible to achieve manually. This motivates the consideration of using an automatic techniques for both searching the operating envelope and designing the appropriate linear controller. Indeed, the technique of genetic algorithms introduced in the previous chapters, appears to be an ideal design tool for such a complex control problem. Hence in this chapter the findings of chapter 3, are applied to design a linear controller for non-linear plants.

8.2 SYNTHESIS OF FIXED GAIN PID CONTROLLERS FOR NON-LINEAR PLANTS

In the design of a gain scheduled PID controller consider the SISO non-linear system of the respective forms

$$\dot{x} = f(x) + g(x)u \qquad 8.1$$

$$y = h(x) \qquad 8.2$$

where x ∈ $\Re^n$, u ∈ $\Re$, y ∈ $\Re$, f(.) and g(.) are smooth vector fields on an open set U ⊂ $\Re^n$, f(0)=0, and h(.) is a smooth function on U.

In order to design linear PID controllers for SISO non-linear plants governed by equations (8.1), it is convenient to consider the behaviour of such plants on the discrete-time set T_T = {0, T, 2T,....}.

This behaviour is governed by non-linear state and output equations of the respective forms

$$Tx_{k+1} = \Phi(T, y_k T)\, Tx_k + \Psi(T, y_k T)\, Tu_k \qquad 8.3$$

and

$$y_k = \Gamma x_k T \qquad 8.4$$

where the state vector $x_kT \in \Re^n$, the input $u_kT\in\Re^T$,the output $y_kT \in \Re^+$, $\Phi \in \Re^{nxn}$, $\Psi \in \Re^{nx1}$, $\Gamma \in \Re^{1x}$, $T \in \Re^+$ is the sampling period, and n is the number of plant states.

8.2.1 LINEAR INCREMENTAL PID CONTROLLER

The linear incremental PID controller introduced in chapter 3 will be used in this chapter, this controller will be used to control non-linear plants. This incremental linear PID controller can be described as

$$\Delta u_k = T(k_p \Delta e_k + T k_i e_k + k_d \Delta^2 e_k) \qquad 8.5$$

where

Δu_k is the incremental change in input,

Δe_k is the first order backward difference in error,

$\Delta^2 e_k$ is the second order backward difference in error.

k_p is the current value of the proportional gain,

k_i is the current value of the integral gain,

k_d is the current value of the derivative gain,

T is sampling time.

It is evident from chapter 3 that the error can be written as

$$e_k = v_k - y_k \qquad 8.6$$

Also from chapter 3 that the first order backward difference in error can be written as

$$\Delta e_k = e_k - e_{k-1} \qquad 8.7$$

Further more from chapter 3 the second order backward difference in error can be expressed as

$$\Delta^2 e_k = e_k - 2e_{k-1} + e_{k-2} \qquad 8.8$$

The issue in this chapter is how to design a linear robust incremental PID controller for non-linear plants . The gains used in this design are constant values chosen by the designer to give the best controller performance for the non-linear plants. But if this task is to be done manually it will be tedious and time consuming. Hence, it is proposed to use the genetic algorithm to chose the gain values, and the tune the controller to produce the best performance for the non-linear plants.

8.3 GENETIC DESIGN OF LINEAR PID CONTROLLER FOR NON-LINEAR PLANTS

The main problem in the design of linear incremental PID controllers for non-linear plants is how to define a cost function that would give the required global performance. After the cost function selection the GA can be used to select the tuning parameters (gains), in such a way that to produce satisfactory response throughout the operating envelope, in this case it is only necessary to encode the gains required by the linear controller i.e. (proportional, integral, and derivative gains), as binary strings in accordance with a system of concatenated, multi-parameter fixed point coding. The binary string would be represented as

$$\{ \; (\quad \underset{\text{proportional}}{k_p} \qquad \underset{\text{integral}}{k_i} \qquad \underset{\text{derivative}}{k_d} \quad) \; \}$$

where the whole string contains the three gains (proportional, integral, and derivative gains). Entire generations of such strings can be readily obtained by using the basic genetic operators of selection, crossover, and mutation. The system incorporates both the non-linear plant, and the linear PID controllers, the controllers are designed by randomly generated sets of gains, by the GA. A stability test is then carried out on all the controllers. In the case of a violation of stability in any of the cases, the randomly generated set of gains, will not be included in the initial population. This type of initialization is essential in this type of design, since otherwise their would be a danger of creating an initial population which many of its members violate the constraints on the controllers being designed.

8.3.1 COST FUNCTION

The cost function used in the previous design methods was

$$ISE = \sum_{i=1}^{m} \sum_{j=1}^{j=N} e^2{}_j$$

is evaluated, where

$$e_j = v_i - y_j \quad , \; N = \frac{\tau}{T} \; \text{, and } \; m = \text{number of steps}$$

and $e_j \epsilon \Re$, is the error signal, $y_j \epsilon \Re$,is the output signal, T is sampling time, and τ is an appropriately chosen settling time.

In this cost function the objective is to minimise the total ISE for the plant when it is subjected to a set-point change. This cost function however, only produces good performance for fixed gains linear controllers for linear plants, or gain scheduled controllers, since in the non-linear plant the operating condition of the plant change this cost function would results in linear gains which produces good performance in one region of the plant operating envelope. The above cost function can be modified slightly by adding a weighting factor λ. This weighting factor can be chosen to increase or decrease the performance of the plant in certain operating points within the operating envelope. The new performance index is thus given by

$$ISE = \sum_{i=1}^{m} \sum_{j=1}^{j=N} \lambda_i e^2{}_j \qquad 8.10$$

is evaluated, where

$$e_j = v_i - y \quad , \quad N = \frac{T}{\tau} \text{ and m=number of steps.}$$

and $e_j \in \Re$, is the error signal, $y_j \in \Re$,is the output signal, T is sampling time, and τ is an appropriately chosen settling time.

It was found that both of the above cost functions are not suitable for designing a linear controller for non-linear plants, because the linear gains will only produce suitable performance for the plant in a specific region within the operating envelope. Hence a different cost function is proposed which will be suitable to produce a robust controller for non-linear plants. This coast function will produce the robustness required.

8.3.1.1 COST FUNCTION FOR A LOWPGV FIXED TERM LINEAR PID CONTROLLER DESIGN

The main task in this design method is to design a cost function that is capable of producing the required global performance as was defined earlier (i.e. minimises the worst local performance of the controller throughout the operating envelope of the non-linear plant). In the genetic design of linear PID controllers for non-linear plant, the plant under consideration is subjected to a succession of set-point changes which span the operating envelope of the plant. Therefore it was decided in this design to use a cost function that would look for the maximum cost function for individual steps, and then minimise the highest of the individual ISE found by a single controller. By employing this cost function the controller will not be deigning a local step minimum ISE, but a global plant ISE. The new cost function is represented by equation of the form

$$ISE_i = \sum_{j=0}^{j=N} e^2{}_j \qquad i=(1,\ldots.m)$$

$$ISE = MAX \underset{i=1}{\overset{i=m}{[}} ISE_i] \qquad 8.11$$

is evaluated, where

$$e_j = v_i - y_j \ , \ N = \frac{\tau}{T} \ , \ \text{m=number of steps.}$$

and $e_j \in \Re$, is the error signal, $y_j \in \Re$,is the output signal, T is sampling time, and τ is an appropriately chosen settling time.

This performance index produced similar ISE for every set-point change in the plant operating envelope. This of course means that the performance of the controller everywhere in the plant operating envelope is similar.

8.4 ILLUSTRATIVE EXAMPLES

To be able to start the design, the GA will need some initial gains, the initial gains can be obtained by using the GA to design and tune two fixed local controllers the first at the initial set-point change, and the second at the last set-point change then choosing the initial gains to be in between these two gains.

A fixed gain linear controller was designed and tuned by the GA, the GA binary string is of the form

$$\{ \underset{\text{proportional}}{k_p} \qquad \underset{\text{integral}}{k_i} \qquad \underset{\text{derivative}}{k_d} \}$$

The fixed gains obtained were used to initialise the global linear controller, where the GA is minimising the ISE across the operating envelope. In the linear PID controller the gains for the three gains(proportional, integral, and derivative gains) were encoded into the GAs binary sting as shown above.

8.4.1 NON-LINEAR PLANT 1 (WATER TANK)

The SISO nonlinear plant used to illustrate the design of gain scheduled controller is a tank system where the cross section A varies with height *h* [91] . The model is given by equation

$$\frac{d}{dt}(A(h)\,h(t)) = q_i(t-\tau) - a\sqrt{2}gh(t) \qquad 8.12$$

where

q_i is the input flow;

a is the cross section of the outlet pipe;

h is the output of the system;

T is the sampling time;

τ is the time delay.

the plant Arma model is given by

$$y_k = y_{k-1} - a\ \sqrt{}y_{k-1} + b\ u_{k-\frac{\tau}{T}} \qquad 8.13$$

and "a" , "b", and τ/T are constant chosen for the design to be

$$a = 8.0;$$

$$b = 0.08.$$

$$\frac{\tau}{T}=10$$

The controller is deployed in the incremental form so as to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed.

8.4.1.1 GLOBAL LINEAR PID CONTROLLER USING UNWEIGHED PERFORMANCE INDEX

In this design, the fixed gains obtained from the local set-point change designs were used to initialise the GA. The linear PID controller is implemented in the incremental form so as to avoid any bumpless transfer technique associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then the linear PID controller for non-linear plants was designed using the GA, such that the integral square error ISE is minimised for the non-linear plant. In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m = 0.01, was used. Figure 8.1 shows transient response of the genetically designed controller. The results obtained for this controller are shown in table (8.1). All the ISEs in table (8.1) are multiplied by 1E5.

zone	0-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
ISE	1.60	1.32	1.23	1.37	1.32	1.45	1.50	1.65	1.73	1.75

table (8.1).

The gains and the total ISE for the controller are given in table (8.2).

k_p	0.45
k_i	7.91
k_d	0.71
Total ISE	1.532E6

table (8.2).

8.4.1.2 GLOBAL LINEAR PID CONTROLLER USING WEIGHTED PERFORMANCE INDEX

In this design, the fixed gains obtained from the local set-point change designs were used to initialise the GA. The linear PID controller is implemented in the incremental form so as to avoid any bumpless transfer technique associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then the linear PID controller for non-linear plants was designed using the GA, such that the integral square error ISE is used for the non-linear plant. In this case the performance index is modified slightly by adding a weighting factor λ. This weighting factor can be chosen to increase or decrease the performance of the plant in certain operating points within the operating envelope. The new performance index is thus given by equation (8.15). In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability,

P_m= 0.01, was used. Figure 8.2 shows the transient response for the genetically designed controller. The results obtained the controller are shown in table (8.3).

zone	0-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
ISE	1.73	1.55	1.53	1.50	1.65	1.58	1.75	1.82	1.68	1.80

Table (8.3).

The gains and the total ISE for the controller are given in table (8.2).

k_p	0.45
k_i	7.9
k_d	0.71
Total ISE	1.669E6

Table (8.4).

8.4.1.3 LOWPGV FIXED TERM LINEAR PID CONTROLLER

In this design, the fixed gains obtained from the local set-point change designs were used to initialise the GA. The linear PID controller is implemented in the incremental form so as to avoid any bumpless transfer technique associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then the linear PID controller for non-linear plants was designed using the GA, such that the integral square error ISE is minimised for the non-linear plant. In this case the performance index used is the sum of minimum of the maximum of the individual set-point change. The new performance index is thus given by equation (8.16). In this case, a population of 100, a crossover probability, $P_c=0.65$, and a mutation probability, $P_m = 0.01$, was used. Figure 8.3 shows the transient response for the genetically designed controller. The results obtained the controller are shown in table (8.5).

zone	0-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
ISE	1.90	1.76	1.77	1.87	1.90	2.10	2.28	2.25	2.26	1.99

Table (8.5)

The gains and the total ISE for the controller are given in table (8.6).

k_p	7.87
k_i	0.02
k_d	4.66
Total ISE	2.0081E6

Table (8.6).

8.4.2 NON-LINEAR PLANT 2 (Concentration of Water Tank)

The SISO nonlinear plant used to illustrate the design of gain scheduled controller is a concentration control of a tank system. Where the inlet concentration, c_{in} is changed.

The model is given by equation

$$V_m \frac{dc_t}{dt} = q_t \ (c_{in}(t - \tau) - c_t) \qquad 8.14$$

where

$$\tau = \frac{v_d}{q_t} \qquad \text{and} \qquad T = \frac{V_m}{q_t}$$

if τ < T, then it is straightforward to determine a PID controller that perform well when q is constant. However it is difficult to find values of the controller parameters that will work well for a wide ranges of q (J.Astrom and B.Wittenmark),[103].

The process has a time delay, with sampling period

$$h = \frac{v_d}{dq}$$

where d is an integer, the plant Arma model is given by

$$y_k = a y_{k-1} + (1 - a) u_{k-h} \qquad 8.15$$

where a and h are given by

$$h = i\,nt\,(\frac{K_1}{u})\ , \qquad a = e^{-uK_2}\ ,$$

where K_1 and K_2 are constants.

$$K_1 = 1, \qquad K_2 = 0.01.$$

The controller is deployed in the incremental form so as to avoid any bumpless transfer techniques associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed.

8.4.2.1 GLOBAL LINEAR PID CONTROLLER USING UNWEIGHED PERFORMANCE INDEX

In this design, the fixed gains obtained from the local set-point change designs were used to initialise the GA. The linear PID controller is implemented in the incremental form so as to avoid any bumpless transfer technique associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then the linear PID controller for non-linear plants was designed using the GA, such that the integral square error ISE is minimised for the non-linear plant. In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m = 0.01, was used. Figure 8.4 shows the transient response for the genetically designed controller. The results obtained for this controller are shown in table (8.7), which is multiplied by E5.

zone	0-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
ISE	1.32	1.40	1.60	1.68	1.75	1.78	1.73	1.83	1.90	2.38

table (8.7).

The gains and the total ISE for the controller are given in table (8.8).

k_p	3.9
k_i	2.3
k_d	2.5
Total ISE	1.737E6

Table (8.8).

8.4.2.2 GLOBAL LINEAR PID CONTROLLER USING WEIGHTED PERFORMANCE INDEX

In this design, the fixed gains obtained from the local set-point change designs were used to initialise the GA. The linear PID controller is implemented in the incremental form so as to avoid any bumpless transfer technique associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then the linear PID controller for non-linear plants was designed using the GA, such that the integral square error ISE is minimised for the non-linear plant. In this case the performance index is modified slightly by adding a weighting factor λ. This weighting factor can be chosen to increase or decrease the performance of the plant in certain operating points within the operating envelope. The new performance index is thus given by equation (8.15). In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m= 0.01, was used. Figure 8.5 shows the transient response for the genetically designed controller. The results obtained the controller are shown in table (8.9), which should be multiplied by E5.

zone	0-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
ISE	4.6	0.16	0.19	0.22	0.33	1.25	4.1	9.85	12.1	12.9

Table (8.9).

The gains and the total ISE for the controller are given in table (8.10).

k_p	0.15
k_i	0.17
k_d	3.23
Total ISE	2.531E6

Table (8.10)

8.4.2.3 LOWPGV FIXED TERM LINEAR PID CONTROLLER

In this design, the fixed gains obtained from the local set-point change designs were used to initialise the GA. The linear PID controller is implemented in the incremental form so as to avoid any bumpless transfer technique associated with the integral state, because the integral state would require bumpless transfer every time the integral gain is changed. Then the linear PID controller for non-linear plants was designed using the GA, such that the integral square error ISE is minimised for the non-linear plant. In this case the performance index used is the sum of the minimum of the maximum of the individual set-point change. The new performance index is thus given by equation (8.16). In this case, a population of 100, a crossover probability, P_c=0.65, and a mutation probability, P_m = 0.01, was used. Figure 8.6 shows the transient response for the genetically designed controller. The results obtained the controller are shown in table (8.11), which is multiplied by E5.

zone	0-1	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
ISE	4	3.62	3.33	3.1	2.84	2.64	2.44	2.3	2.13	2.14

Table (8.11).

The gains and the total ISE for the controller are given in table (8.12).

k_p	4.61
k_i	3.04
k_d	0.23
Total ISE	2.854E6

Table (8.12).

8.5 CONCLUSIONS

In this chapter genetic algorithms were used to design a globally optimised Linear PID controller for a non-linear plant. The design shows that the cost function employed in this design is effective for the design of global controllers it also guaranties some minimum level of performance through out the operating envelope. Indeed comparing the results of this design with that obtained using the gain scheduling design for the two plants indicates that the robust design has produced a highly effective global controller. It is evident that the resulting controller does provide near optimum local control at the worst operating points, such that the control everywhere else in the operating envelope is better. This shows the controller performs in the same manner as a robust controller.

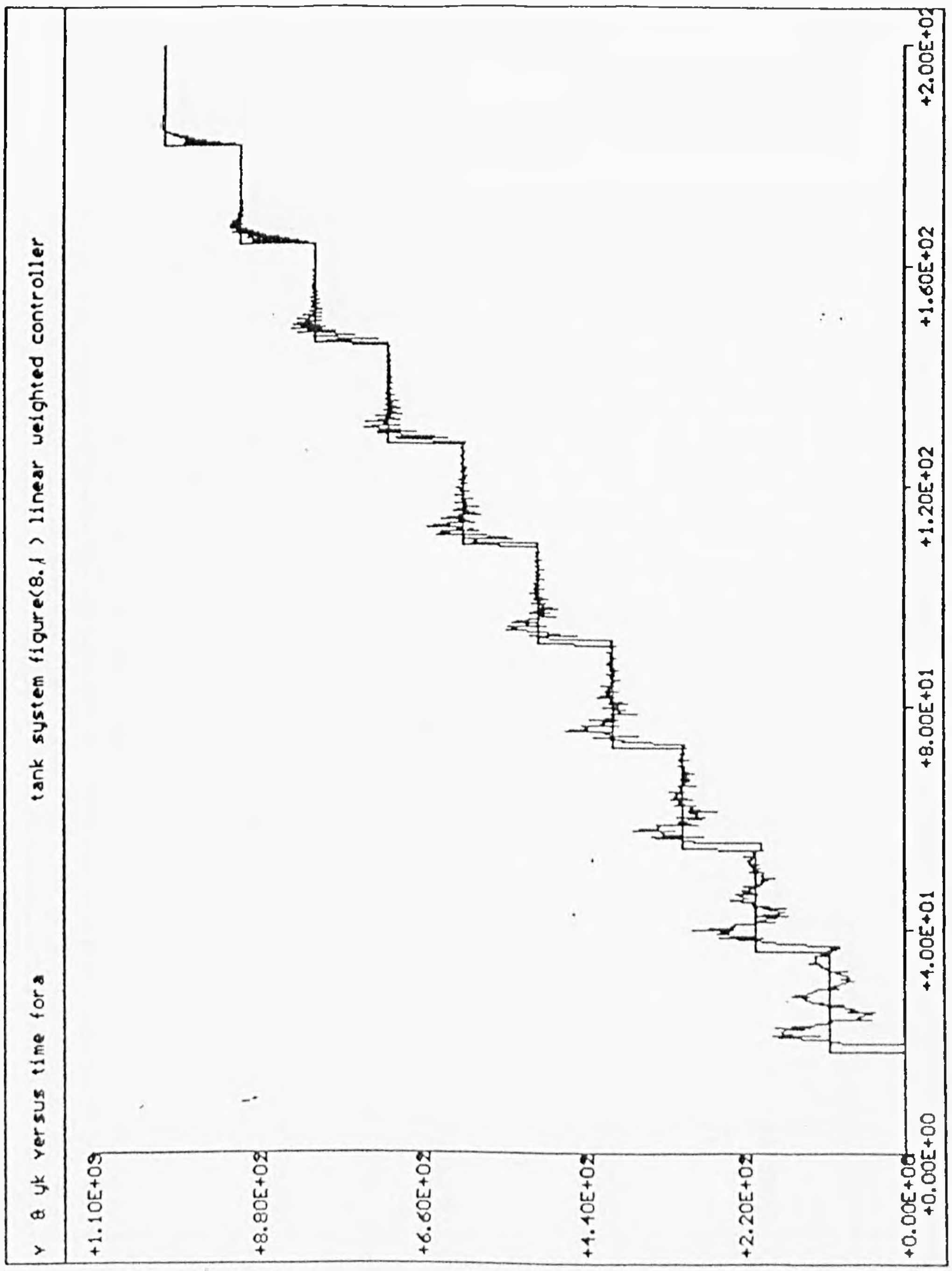


Figure (8.1) Transient response of the genetically designed global linear PID controller for plant (1), example (8.4.1.1)

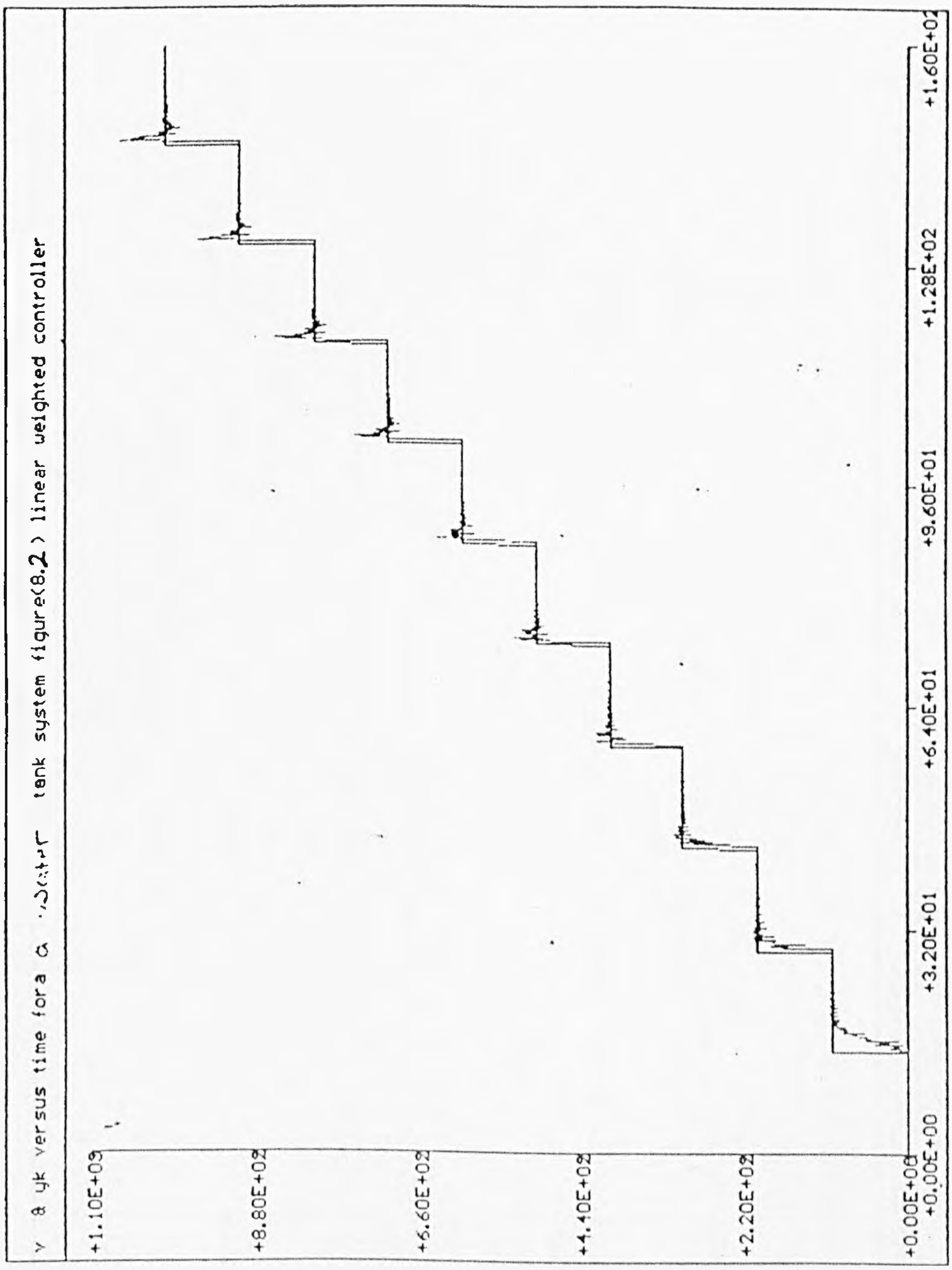


Figure (8.2) Transient response of the genetically designed global linear PID controller for plant (1), example (8.4.1.2)

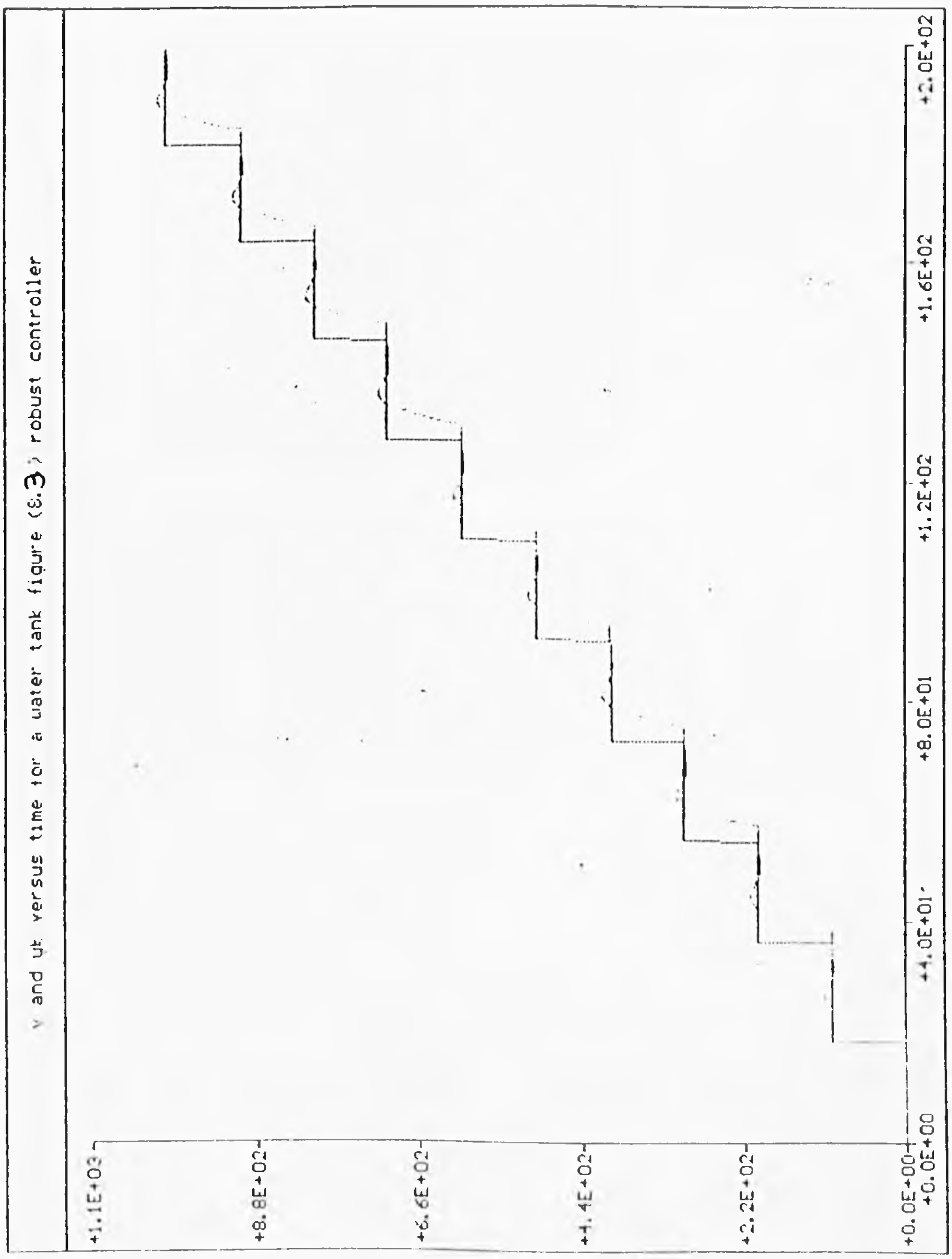


Figure (8.3) Transient response of the genetically designed locally optimised PID controller for plant (1), example (8.4.1.3).

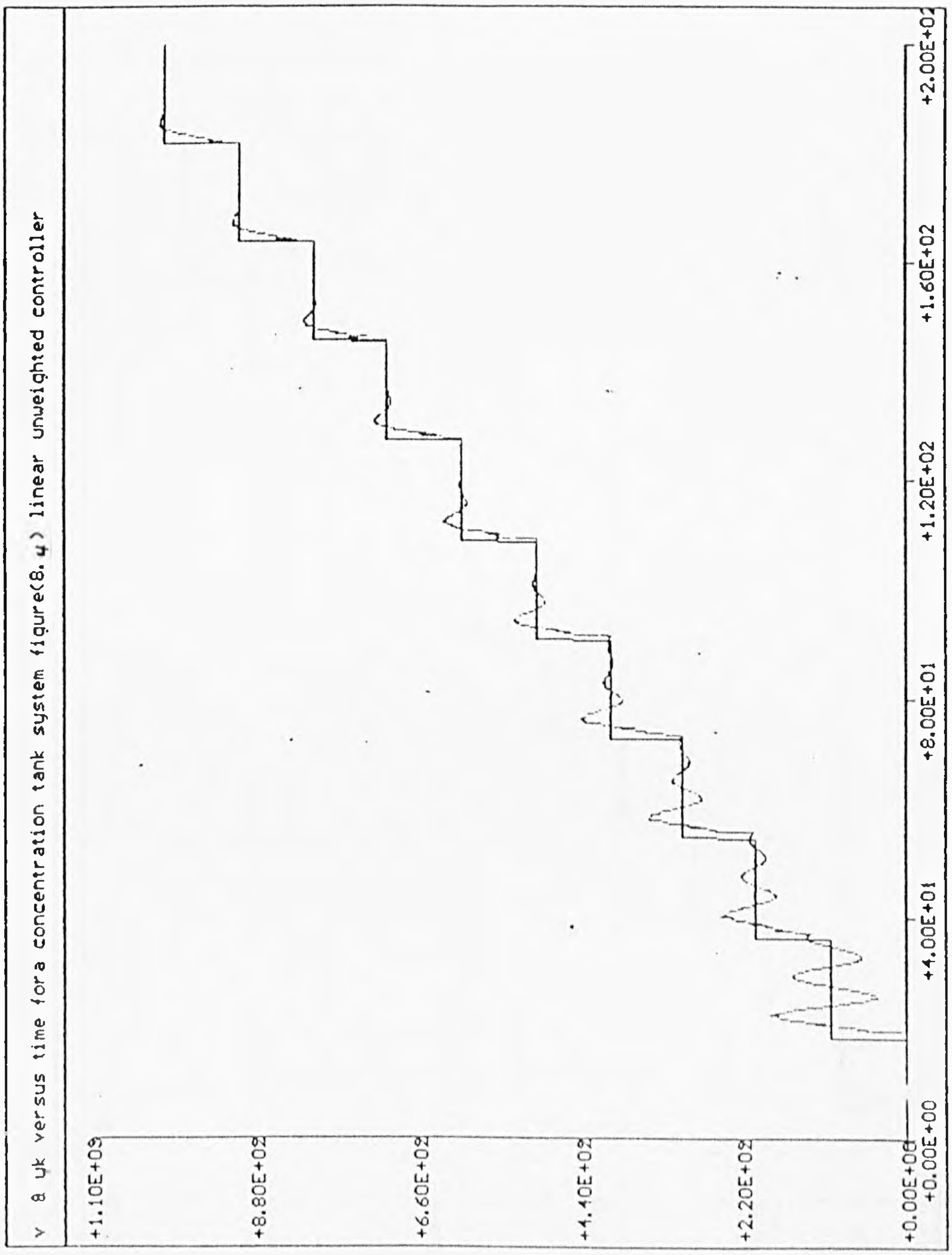


Figure (8.4) Transient response of the genetically designed global linear PID controller for plant (2), example (8.4.2.1)

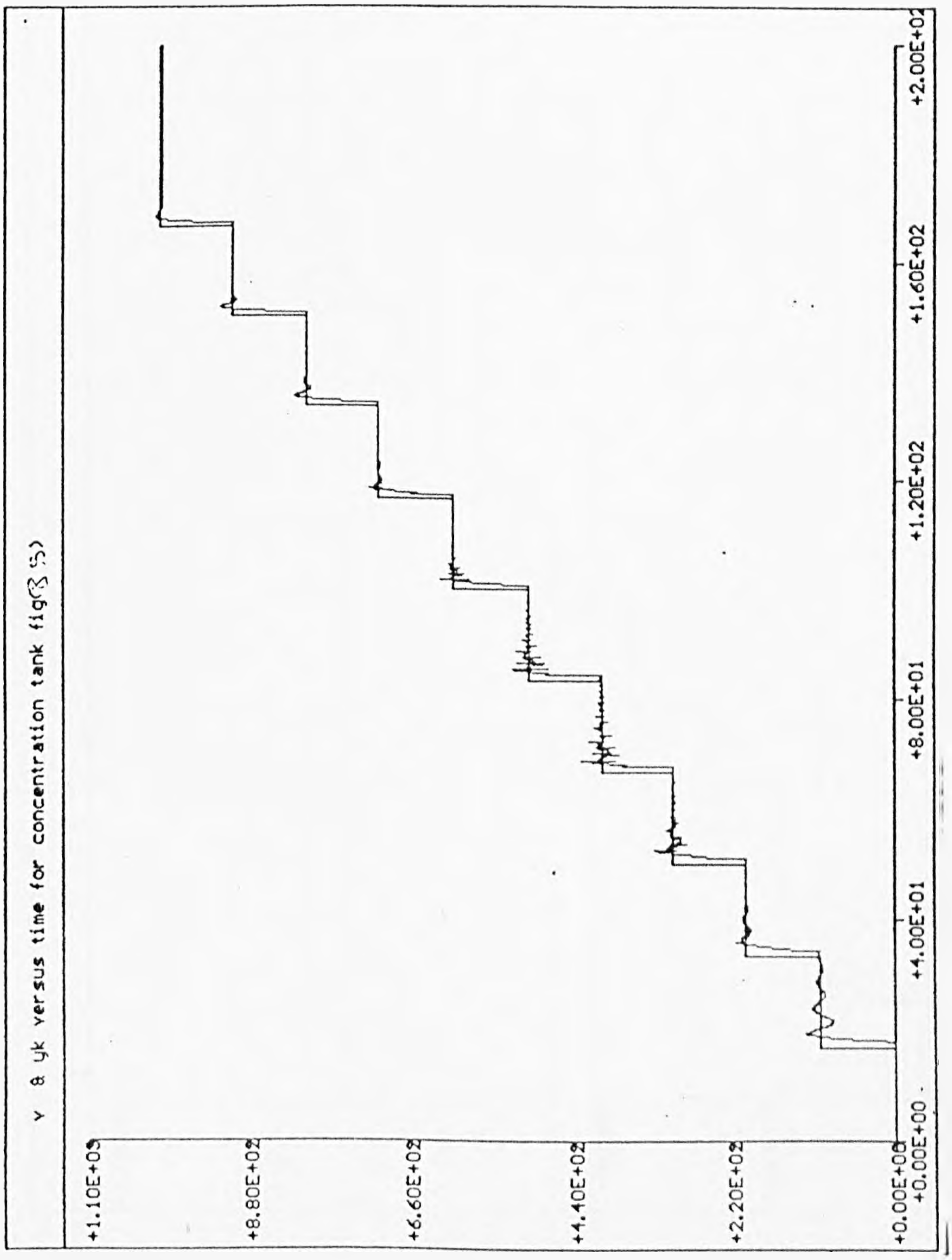


Figure (8.5) Transient response of robustness test using genetically designed global PID controller for plant (2), example (8.4.2.2).

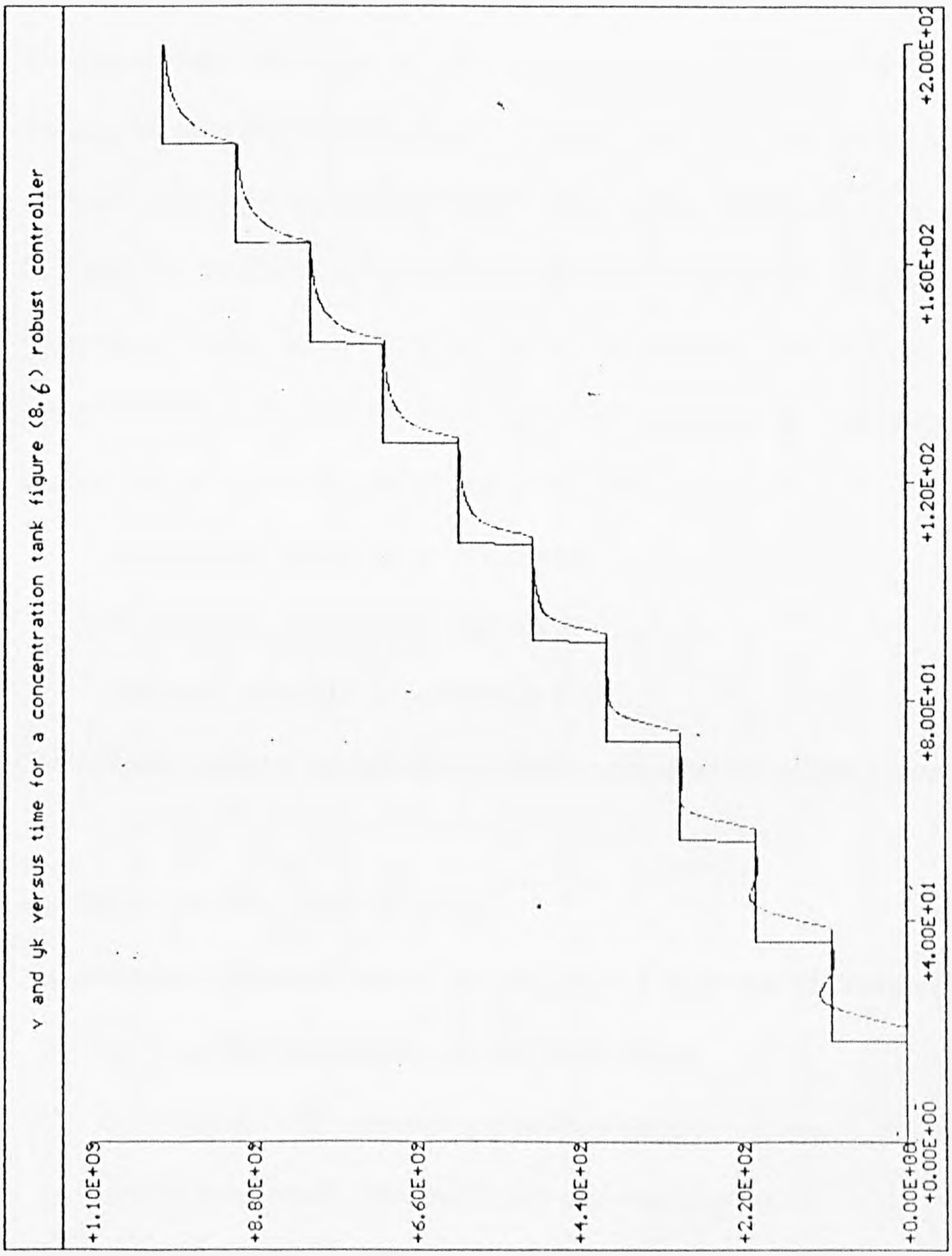


Figure (8.6) Transient response of the genetically designed global linear PID controller for plant (2), example (8.4.2.3)

CONCLUSIONS

9.1 CONCLUSIONS

The concepts, and design procedures for linear control systems has been studied by numerous people. But this is not the case for the non-linear control system. This is because of the lack of a technique or method. Therefore this thesis attempts to develop a new technique for the design of non-linear control systems. In achieving this goal the technique of genetic algorithms has been proposed as a means of effecting the design. It was found that the GA can provide a new design method that is significantly different from the mathematical approach. In general non-linear control systems can be defined as one of the following types:

i) *non-linear controller for linear plant*;

ii) *non-linear controller for nonlinear plant*;

iii) *linear controller for non-linear plant.*

The GA was deployed successfully to design controllers for all the above cases.

non-linear controllers for linear plants

The non-linear controllers can be implemented in a number of different ways such:

i) *by using functions to map non-linear gains*;

ii) *by using fuzzy sets to map non-linear gains*;

iii) *by using neural networks to map non-linear gains.*

The GA was used to successfully design non-linear control systems using all the above techniques.

Indeed, it is evident that in principle GAs could be used to design any controller involving PID, fuzzy or neural architectures. During the analysis of the non-linear incremental controller it was found that the controller operates in two distinct region within the operating envelope of the plant. Hence, the non-linear gain mapping was done so as the controller can operate in the two zones, this implementation resulted in an increase in the performance of the controllers being designed. The principle reason for this being that the search space for the GA could then be chosen precisely. From the results obtained it was found that the GA could design a high performance controller. Indeed, in some of the cases the resulting controller was a dead beat controller. The most interesting point of this study was that the polynomial, fuzzy and neural controllers all performed to a similar level of performance. However, the neural controller had 26 parameters, the fuzzy and the polynomial both had 14 parameters each. This is of interest as it suggest that polynomial functions may be better than fuzzy or neural method for implementing non-linear PID controllers.

non-linear controllers non-linear plant

In the third section of the thesis the GA was used to design non-linear controllers for non-linear plants (gain scheduled controllers). In this case polynomial interpolation was used to map the gains of the gain scheduled controllers. Using this implementation the GA produced high performance controllers over the operating envelope of the plant. Since in the non-linear control systems, the non-linearity is a function of either input, output or state, in this work two illustrative examples were considered one with input and one with output non-linearities. The resulting gain scheduled control systems could not have been designed autonomously using any other technique.

linear controller for non-linear plants

In the fourth section of the thesis the GA was used to design linear controller for non linear plants. When designing fixed linear PID controllers for a non-linear plant the question of local and global performance is addressed. In this context the GA was used to design a globally optimised fixed term linear PID controller for a non-linear plant. The robustness of the control systems has long interested control engineers, with the most recent advances being made by the H^{∞} design method. In this section of the thesis a simple cost function involving the minimisation of the worst performance of the controller throughout the operating envelope was proposed. This min-max performance index provides a time-domain robustness measure if used to design a linear controller for a set of linear plants which may rival the H^{∞} robustness concepts.

Finally it can be said that the future of the GA as a general control systems design tool is very bright. And from the results obtained using the GA it can be postulated that in the future all computer aided control system design (CACSD) will be done using GAs, where the controller can either be linear or non-linear.

9.2 RECOMMENDATION FOR FURTHER WORK

The new concepts, and design methods presented in this thesis can be readily extended for application in the fields of:

i) *aerospace*, to design gain scheduled aircraft controllers;

ii) *process control*, to design high performance dual zoned controllers;

iii) *robotics*, to design high performance dual zoned controllers;

iv) *green approach*, move away from "output optimisation" to "input optimisation" i.e. minimise sum of input for set-point changes, this could provide energy saving hence, the term "green".

The robustness studies could be further enhanced by using the principle of co-evolution and fitness sharing. With the additional concepts the robustness design method of such control system could be developed to rival the H^∞ design method. The only problem with using GA is they essentially really on the ability to simulate the process being controlled. This requirement makes the issue of process identification very important. GAs and genetic programming GPs have been used by other researchers to model non-linear process [102]. This identification procedure enables the evolutionary techniques to be applied. Therefore further research in the identification of models of non-linear systems is required.

REFERENCE TABLE

[1] P S Laplace, "Exposition du system du mode", de L' Impr.dr Cerecl-Social, 1795-1796.

[2] H Nyquist, "Regeneration Theory", Bell System Technical Journal, Vol. II, pp 126-147, 1932.

[3] H Black, "Stabilized feedback amplifier", Bell System Tech Journal, 1934.

[4] H W Bode, "Network analysis and feedback amplifier design", New York: Van Nostrand, pp451-488, 1945.

[5] W R Evans, "Graphical analysis of control Systems", AIEE Trans, Vol. 67, pp 547-551, 1948.

[6] L S Pontryagin, V C Boltyanski, R V Gomkrelidze, and E F Mischenko, "The mathematical theory of optimal process", John Wiley, New York, 1962.

[7] H H Rosenbrock, "Design of multvariable control systems using inverse Nyquist array", proc IEE, Vol. 116, pp 1929-1936, 1969.

[8] H H Rosenbrock, 2state space and multivariable theory", Nelson London, 1970.

[9] H H Rosenbrock, "Progress in the design of multivariable control system", Trans . Inst. Meas. Control, Vol4, pp 9-11,1971.

[10] P D McMorran, "Extension of inverse Nyquist arry method", Electron. Lett, Vol. 6, pp 800-801, 1970.

[11] A G J MacFarlane and J J Belletruti, " The characteristic Locus design method", Automatica, Vol. 9, pp 575-588, 1973.

[12] P A Cook, "Modified multivariable Circle Theorems", In recent mathematical developments in control, ed. by D J Bell, New York: Academic press, pp 367-372,1973.

[13] N Munro, "Design of controllers for open-loop unstable multivariable systems using inverse Nyquist arry", proc IEE, Vol. 119, No. 9, pp 1377-1382, 1972.

[14] H H Rosenbrock, "Multivariable circle theorems", In recent mathematical developments in control, ed. by D J Bell, New York: Academic press, pp 345-346, 1973.

[15] L F Yeung and G F Bryant, "Robust stability of diagonal dominant system", proc IEE, Vol. 131, No. 6, pp 253-260, 1984.

REFERENCE TABLE

[16] B Van Der pol, 1934, Proc. IRE 22, pp 1051-1081.

[17] N M Krylov and N NBogoliubov, " Introduction to non-linear mechanics", Moscow, 1937.

[18] Goldberg E.D. (1989), Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley P.C., ISBN 0-201-1567-5.

[19] J H Holland, "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor, 1975.

[20] D E Goldberg, " Computer-Aided Gas pipeline operation using Genetic Algorithms and rule learning" PhD Dissertation, University of Michigan, Ann Arbor, 1983.

[21] Z Zhang. and P D Roberts (1992), "Use of genetic algorithms in training diagnostic rules for process fault diagnosis", Vol 5, No 4, pp 277-288, December.

[22] K A Dejong, "Analysis of the behaviour of a class of genetic adaptive systems", PhD Dissertation, University of Michigan, Ann Arbor, 1975.

[23] J J Grefenstette, "Genetic Algorithms", IEEE Expert Systems, Vol. 8, No. 5, Oct. 1993, pp 6-8.

[24] S S Roa, T S Pan, and B Venkayya, " Optimal placement of actuators in actively controlled structures using GAs", AIAA, Journal, Vol. 29, No. 6, pp 942-943, 1991.

[25] J Onoda and Hanawa, " Actuator placement optimization by genetic and improved simulated annealing algorithms", AIAA Journal, Vol. 31, No. 6, pp 1167-1169, 1993.

[26] M Avriel, "Non-linear programming: Analysis and methods", Prentice Hall, 1976.

[27] D G Luenberger, "Introduction to linear and non-linear programming", Addison Wesley, 1973.

[28] E Polak, "Computation methods in optimization", Academic press, New York, 1971.

[29] W Gesing and E J Davison, "An exact penality function algorithm for solving general constrained parameter optimization problems", Automatica,15, pp175-188, 1979.

[30] B Porter and M Barairi, "Genetic design of Active controllers for flexible space structures", AIAA Guidance, Navigation, and Control Conference, pp 196-213, 1993.

[31] Porter B. and Jones A.H. (1992)," Genetic tuning of digital PID controllers", Electron Lett, Vol 28, pp 843-844.

[32] E Polak and D Q Mayne, "An algorithm for optimisation problems with functional inequality constraints", IEEE Trans Auto Control, Vol AC-21, pp184-193, 1976.

[33] Porter B. S S Mohamed, and A H Jones, (1993), "Genetic tuning of multivariable PID controllers", Proc ECC, Groningen Netherlands.

[34] E Polak and D Q Mayne, "An algorithm for optimisation problems with functional inequality constraints", IEEE Trans Auto Control, Vol AC-21, pp184-193, 1976.

[35] P J Angeline, G M Saunders, and J B Pollack, "An Evolutionary Algorithm that Constructs Recurrent Neural Networks", IEEE Trans. Neural Networks, Vol. %, No. 1, pp39-53, Jan 1994.

[36] P R Weller, R Summers, A C Thompson, " Using a genetic algorithm to evolve an optimum input set for a predictive neural network", IEE/IEEE International Conference Publication No. 414, pp 256-258.

[37] A Kelemen,M Imecs, C Rusu, and Z Kis, "Run-time auto-tuning of a robot controller using a genetic based machine learning control system, IEE/IEEE International Conference Publication No. 414, pp 307-312.

[38] A H Jones and P B De Moura Olivera, (1995), " Genetic Auto-Tuning of PID Controllers", IEE/IEEE International Conference Publication No. 414, pp 141-145.

[39] A H Jones, S B Kenway, and N Ajlouni (1995), "Genetic Design of Fuzzy Gain-Scheduled Controllers for Non-Linear Plants", Proceeding of the international ICSC symposium 0n fuzzy logic.

[40] A H Jones and N Ajlouni (1994), " Genetic Tuning of Gain-Scheduled Controllers for Non-Linear Plants", Proc IASTED Conference on System and Control, Lugano.

[41] J Kim, Y Moon, and B P Zeigler, (1995), "Designing Fuzzy Ne Controllers using Genetic Algorithms" IEEE control systems, June, pp 66-72.

[42] A H Jones and B Porter, "Expert tuners for PID Controllers", Proc IASTED on Computer-Aided Design and Applications, Paris, 1985.

[43] W Gensing and E J Davidson, "An exact penalty algorithm for solving general constrained parameter optimisation problems", Automatica, Vol 15, pp 175-188, 1979.

[44] R E Kalman. (1958), " Design of self-optimising control systems ", Trans ASME, Vol 80, pp 468-478.

[45] A. H. Jones (1995), Genetic Tuning Of Neural Non-Linear PID Controllers, International Conference on Artificial Neural Networks and Genetic Algorithms.

[46] K S Tang, C Y Chan, K F Man, and S Kwong, "Genetic Structure for NN Topology and Weights Optimization", IEE/IEEE International Conference Publication No. 414, pp 250-255.

[47] D T Pham and D Karaboga, "New Method to Obtaining the Relation Matrix for Fuzzy Logic Controllers", 6th International Pr. for AI in Engineering, No. IBSN 1/851 66/678/8, pp 567-581.

[48] J J Grefenstette . (1993), Genetic Algorithms, IEEE Expert Journal, October, pp 6-7.

[49] A Homaifar and Ed McCormick, (1995), "Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controllers using Genetic Algorithms.

[50] S Shenoi, K Ashenayi, and M Timmerman, (1995), " Implementation of a Learning Fuzzy Controller", IEEE control systems, June, pp 73-80.

[51] D S Reay, M Mirkazemi-Moud, T C Green, and B W Williams, "Switched Reluctance Motor Control Via Fuzzy Adaptive Systems", (1995), IEEE control systems, June, pp 8-14.

[52] M Salami and G Cain, (1995), " An Adaptive PID Controller Based on Genetic Algorithms Processor", IEE/IEEE International Conference Publication No. 414, pp 88-93.

[53] L Davis, "Handbook of Genetic Algorithms", Van Nostrand Reinhold, New York, (1991).

[54] A J Chipperfield, P J Fleming, and C M Fonseca, (1994), " Genetic algorithm tool for control systems engineering", Proceeding of Adaptive Computing in Engineering Design and Control, 21-22 September, Polmmouth, UK.

[55] J J Grefenstette, (1986), "Optimization of control parameters for genetic algorithms", IEEE Trans. Systems, Man and Cybernetics, SMC-16, No. 1, January/February, pp 122-128.

[56] A H Jones and M L Tatnall, " Automated frequency domain identification and genetic identification of transfer functions", Proc SYSID'94, Copenhagen, 1994.

[58] W F Punch, R C Averill, E D Goodman, Shyh-Chang lin, and Ying Ding, (1995), " Using Genetic Algorithms to Design Laminated Composite Structures", IEEE Expert, February, pp 42-50.

[59] S Kwong, A CLNg and K F Man, " Improving local search in genetic algorithms for numerical global optimization using modified GRID-point search technique" IEE/IEEE International Conference Publication No. 414, pp 419-423.

[60] L Zadeh, (1973) "Outline of a new approach to the analysis of complex systems and desion processes", IEEE Trans. on systems, Man and Cybernetics, Vol. SMC-3, pp 28-44.

[61] T J Procyk and E H Mamdani, "A self-organizing linguistic process controller", Automatica, 15, pp 15-30, 1979.

[62] C C Lee and H R Berenji, (1989), "An intelligent controller based on approximate reasoning and reinforcement learning, Proc. IEEE Int. Symp. on intelligent control, Albany, NY, pp 200-205.

[63] C C Lee. (1990), "Fuzzy logic in control systems", IEEE Trans on Systems, Man and Cybernetics, SMC, Vol 20, pp 404-435.

[64] D G Schwatz, (1992), "Fuzzy logic flowers in Japan", IEEE Spectrum, July, pp 32-35.

[65] R N Lea and Y Jani (1991), "Fuzzy logic captures human skills", Aerospace America, October, pp 25-28.

[66] W Pedrycz, "An approach to the analysis of fuzzy systems", International Jurnal of Control, Vol. 34, No. 3, Sep. 1981, pp 403-421.

[67] J R Rao and N Roy, "Fuzzy set theoretic approach of assigning weights to objectives in multicriteria decision making", International Journal of Systems Science, Vol. 20, No. 8, Aug. 1989 pp 1381-1386.

[68] T Masters, (1993), Practical Neural Network Recipes in C++, Academic Press, Inc., ISBN 0-12-479040-2.

[69] W Godenthal and J Farrel, (1990), "Applications of neural network to automatic control", Proc AIAA Guidance, Navigation, and Control Conference, pp 1108-1112, Portland, USA.

[70] D E Rumelhart, B Widrow, and M A Lehr, "The basic ideas in neural networks", Communications of the ACM, Vol. 37, No. 3, pp 87-105, March 1994.

[71] S N Singh and A A Schy, "Output feedback non-linear decoupled control synthesis and observer design for manoeuvring aircraft", International Journal of Control, Vol. 31, No. 4, April 1980, pp 781-806.

[72] P M Sharkey and J Oreilly, "Composite control of non-linear singularly perturbed systems: a geometric approach", International Journal of Control, Vol. 48, No. 6, Dec.1988, pp 2491-2506.

[73] J C P Jones and S A Billings, "Recursive algorithm for computing the frequency response of a class of non-linear difference equation model", International Journal of Control, Vol 50, No 5, Nov. 1989, pp 1925-1940

[74] T J Gordan, "Infinite gain margins in non-linear regulators", International Journal of Control, Vol 54, No 3, Sep 1991, pp 547-560.

[75] A Glamineau and C H Moog, "Essential orders and non-linear decoupling problem", International Journal of Control, Vol 50, No 50, Nov 1989, pp 1825-1834.

[76] J C Kantor, "Non-linear sliding-mode controller and objective function for surge tanks", International Journal of Control, Vol 50, No 5, Nov. 1989, pp 2025-2047.

[77] K Balachandran, "Existence of optimal control for non-linear multiple-delay systems" International Journal of Control, Vol 49, No 3, March 1989, pp 769-775.

[78] P E Crouch and I S Ighneiwa, "Stabilization of non-linear control systems: the role of Newton diagrams", International Journal of Control, Vol 49, No 3, Nov. 1989, pp 1055-1071.

[79] J Klamka, "Controllability of non-linear systems with distributed delays in control", International Journal of Control, Vol 31, No 5, May. 1980, pp 811-819.

[80] S K P Wong and D E Seborg, "Control strategy for SISO non-linear systems with time delay", International Journal of Control, Vol 48, No 6, Dec. 1988, pp 2303-2327.

[81] L E Faibusovich, "Explicit solvable non-linear optimal control problems", International Journal of Control,

[82] J Hammer, "Fraction representations of non-linear systems and non-additive state feedback", International Journal of Control, Vol 50, No 5, Nov. 1990, pp 1981-1990.

[83] S Behtash, "Robust output tracking for non-linear systems", International Journal of Control, Vol 51, No 6, June 1990, pp 1381-1407.

[84] A K Majumdar and A K Choudhury, "On the decoupling of non-linear control", International Journal of Control, Vol. 16, No. 4, Oct. 1972, pp 705-718.

[85] J C P Jones and S A Billing, "Describing functions volterra series, and analysis of non-linear systems in the frequency domain", International Journal of Control, Vol 53, No 4, April 1991, pp 871-887.

[86] S P Banks, "Generalization of the Lyapunov equation to non-linear systems", International Journal of System Science, Vol 19, No 4, April. 1988, pp 629-636.

[87] J F Frankena and R Sivan, "A non-linear optimal control law for linear systems", International Journal of Control, Vol 30, No 1, July 1979, pp 159-178.

[88] J Hammer, "Robust stabilization of non-linear systems", International Journal of Control, Vol 49, No 2, Feb. 1989, pp 629-653.

[89] D Meizel and J C Gentina, "New aspects on linear and non-linear SISO systems", International Journal of Control, Vol 30, No 6, Dec. 1979, pp 1043-1060

[90] S Nicosia, P Tomel, and A Tornambe, "Observer-based control law for a class of non-linear systems", International Journal of Control, Vol 51, No 3, March. 1990, pp 553-566.

[91] Mingwu Chen and A M S Zalzala, (1995), "Safety consideration in the optimisation of paths for mobile robots using GAs", IEE/IEEE International Conference Publication No. 414, pp 299-306.

[92] Y Davidor, H P Schwefel, and R Manner (eds.), "Parallel problem solving from Nature ", Springer Verlag, Berlin, 1994.

[93] A V Sebald and L J Fogel (eds.), Proceeding of the third Annual Conf. on EP, World Scientific, River Edge, NJ, 1994.

[94] S Forrest (ed), Proc. of the fifth Int. Conf. on GAs, Morgan Kaufmann, San Mateo, CA, 1993.

[95] D E Goldberg, K Deb, and J H Clark, "GAs, noise and the population sizing" Complex Systems, 6, 8, (Aug. 1992), pp 333-362.

[96] Bernard Widrow, D E Rumelhart, and M A Lehr, " Neural networks: Applications in industry, Business and Science", Communication of the ACM, Vol.37, N0.3, pp93-105, March 1994.

[97] K A Sakawa and H Takagi, " Neural networks in Japan", Communication of the ACM, Vol.37, N0.3, pp 106-112, March 1994.