EVOLUTIONARY DESIGN OF DIGITAL TRAJECTORY-TRACKING CONTROLLERS FOR ROBOTIC MANIPULATORS

by

Chafik ALLAOUI

A Thesis Submitted for the degree of

Doctor of Philosophy

of the University of Salford

Department of Aeronautical, Mechanical,

and Manufacturing Engineering

November 1998

ORIGINAL COPY TIGHTLY BOUND

PAGE NUMBERING AS ORIGINAL

To My Mum, Late Father (1929-1969), Family and Friends

ACKNOWLEDGEMENTS

I would like to express my deep appreciation to my supervisor Professor Brian Porter for his unending patience, guidance and inspiration throughout the entire course of this academic adventure.

A special thank-you is due to all my flat-mates who have accepted to spend time in the kitchen cooking while I was facing my monitor. My thanks also go to all my friends who cannot be cited individually by name for their moral support when things were not going the way I wanted them to.

I also thank my brother Karim, my sister Nassima and most importantly my mother whose continuous encouragement enable me to deal with the pressure and turn things my own way.

My thanks also go to those who have helped me at any stage of my research and I may have forgotten their names. Had they not been such helpful people, this piece of work would not have been possible.

Last but not least, I wish to thank The British Council/FCO and The Algerian Foreign Office for co-sponsoring this research. To all of you, from the deepest of my heart, I thank you.

SUMMARY

The design of digital trajectory-tracking controllers for robotic manipulators is a challenging task, since such manipulators are multivariable non-linear plants. In addition, in many applications of robotic manipulators, it is required that very high-accuracy trajectory-tracking performance be achievable even in the presence of unpredictable payload variations. These requirements can all be met to some extent by application of the previously developed fast-sampling digital PID controllers to robotic manipulators. Indeed, for such controllers, it is possible to prove a series of very reassuring robustness results using only the Markov parameters associated with locally linearised representations of robotic manipulators.

However, these theoretical optimisation results for digital PID controllers are only valid as sampling periods become vanishingly small. In practice, of course, the sampling periods of digital controllers remain non-zero; but, in such cases, no theoretical optimisation results are available. There is, therefore, a great need for some alternative optimisation procedure that will facilitate the non-asymptotic design of digital PID controllers for robotic manipulators.

This design need is addressed in this thesis. In particular, the following evolutionary optimisation techniques are used to design digital trajectory-tracking controllers for robotic manipulators:

(i) genetic algorithms,

(ii) non-adaptive evolution strategies,

(iii) adaptive evolution strategies.

It is shown that, with increasing effectiveness, these techniques are very useful in the design of high-accuracy digital PID controllers. These techniques are illustrated by the presentation of simulation results for a typical three-link robotic manipulator performing a range of demanding trajectory-tracking tasks in the presence of unpredictable payload variations. In addition, these evolutionary optimisation techniques are also used in the design of unconstrained digital PID controllers, in which all elements of the controller matrices are used as the design parameters.

In order to validate these evolutionary design techniques in practice, an experimental laboratory investigation is also undertaken. This involves the practical implementation, in the case of a direct-drive two-link robotic manipulator, of digital PID trajectory-tracking controllers designed using evolutionary techniques. The results thus obtained indicate that such optimisation techniques greatly facilitate the tuning of digital PID controllers for robotic manipulators under practical non-asymptotic conditions.

CONTENTS

Page

PERTURBATION METHODS

PART I

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In a world where industry is constantly pushing for optimal performance, control engineers are being compelled to strive hard when designing or tuning the many thousands of controllers implemented in different forms world-wide in vital sectors of the economy, e.g. in the growing numbers of industrial robots. The development of control system design techniques which can be enhanced by incorporating simple optimisation techniques is a pressing necessity. The development of new theoretical and computational methods for the design of machines which, in some sense are controlled so as to exhibit 'intelligent' behaviour by emulating the behaviour of biological systems, is therefore considered to be a big step in the exploration of fundamental problems in control theory and also in the solution of a number of practical design problems. [*Porter* (1989)]

1.2 ROBOT DYNAMICS

It is commonly claimed in much literature concerning robot control that the dynamics of robotic manipulators are highly non-linear and that there is strong

dynamic coupling between joints. Two formulation methods are used to model the dynamics of such robotic manipulators, namely, the Euler-Lagrange formulation (see Appendices B and C) and the recursive Newton-Euler formulation [*Luh et al* (1980)] [*Paul* (1981)]. The former facilitates the modelling of complex systems, whilst the latter is computationally more efficient and better suited for real-time control applications. A simple description of the motion of robotic manipulators was derived using vectorial mechanics in [*Mc Innis* and *Liu* (1986)]. By using this vectorial approach, dynamical models are obtained with a direct physical interpretation.

The use of the mechanical/electrical analogy known as the 'bond graph' was proposed to describe linear mechanical systems via electrical networks and was initially launched by *Paynter* (1960) and *Rosenberg* (1968). This technique was later extended to cope with non-linear mechanical systems by *Rosenberg* and *Karnopp* (1983). However, limitations remained since bond graphs do not directly express the energy flows involved in mechanical systems. Another expression of non-linear mechanical systems via a network was proposed by *Anderson* (1995) on the basis of general Hamiltonian systems, which did not describe internal energy flows between elementary operational blocks such as resistors, gravity capacitor blocks, kinetic inductors, etc.

The attempt at expressing the Lagrangian dynamics of a robot manipulator and analysing its control via a non-linear position-dependent circuit was presented

by *Arimoto* (1994) as another language for describing robot dynamics. The first work explicitly describing the idea of introducing a class of non-linear position-dependent circuits is found in *Arimoto* (1995).

In addition to the motion of the robotic manipulator, the dynamics of robotic systems includes actuators dynamics and sensors dynamics. In fact, electric motors are the actuators usually employed in robotic manipulators to provide the necessary forces and torques for motion. These electrical actuators are normally voltage-controlled devices and, therefore, possess armature inductances that should be incorporated into the system dynamics. In that case, a complete dynamical model including both electrical and mechanical parts would be a third-order differential equation. However, the response of a third-order system can be accurately emulated by a second-order model when the electrical time constant is significantly small compared with the mechanical time constant [*D'Azzo* and *Houpis* (1995)]. The electrical time constant is a function of motor inductance which is very small in most practical situations, so that the complete system model can then be accurately represented by a second-order system. Besides the frequently neglected dynamics of actuators, there are many other sources of dynamics in a given robotic system: for example, dynamics of power amplifiers for actuators and sensor dynamics. Such dynamics are usually neglected and thus classified as unmodeled dynamics.

1.3 MODERN CONTROL THEORY AND ROBOT CONTROL

1.3.1 REVIEW OF MODERN CONTROL THEORY

In the early 1930s, *Nyquist* (1932) and *Black* (1934) used their theoretical work to establish control engineering as a discipline in its own right. These foundations were later strengthened by the contributions of *Bode* (1945) and *Wiener* (1949), with the frequency-domain approach, and *Evans* (1948) (1950), with the introduction of the root-locus method. However, since all these methods were mainly dedicated to linear Single Input-Single Output (SISO) systems, there were difficulties in applying them in the case of complex industrial plants such as robotic manipulators which are non-linear Multiple Input-Multiple Output (MIMO) systems, with possibly time-varying characteristics. It is, therefore, evident that methods for the design of control systems for complex industrial plants should provide multivariable control laws. Such methods should guarantee good performance in the face of the uncertainties inherent in complex, non-linear, and time-varying plants such as robotic manipulators.

The control theory developed through the late 1950s may be categorised as *conventional* control theory and this has been effectively applied to many control-design problems, especially for SISO systems. Since then, control theory has been developed for the design of more complicated systems and

more notably for MIMO systems. In the late 1950s, several authors, including *Bellman* (1957) and *Kalman* (1960) in the United States and *Pontryagin* (1963) in the U.S.S.R, began again to consider the ordinary differential equation (ODE) as a model for control systems. Much of this work was stimulated by the new field of control of artificial earth satellites, in which the ODE is the natural form for writing the model. This endeavour was supported by digital computers, which could be used to carry out calculations unthinkable 10 years before. The work of *Lyapunov* (1893) was translated into the language of control at about this time, and the study of optimal controls, begun by *Wiener* (1949) and *Phillips* (1951), was extended to the optimisation of trajectories of non-linear systems based on the calculus of variations [*Kalman* (1963)]. Much of this work was presented at the first conference of the newly formed International Federation of Automatic Control held in Moscow in 1960. This work did not use the frequency response or the characteristic equation but worked directly with the ODE in "normal" or "state" form and typically called for the extensive use of computers. Space travel had become possible only because of the advent of *modern* control theory. Indeed, areas such as trajectory optimisation and minimum-time and/or minimum-fuel problems, which are very important in space travel, can be readily handled by modern multivariable control theory. The introduction of the digital computer and its compatibility with multivariable control theory was crucial in solving important problems associated with the manoeuvring, guidance, and tracking of aircraft, missiles, and space vehicles.

This approach, which is now often called modern control to distinguish it from the classical control methods of *Bode* and others, was given considerable attention by engineers and yielded comprehensive design procedures for linear multivariable plants as proposed by *Kalman* (1963) in order to solve the linear optimal control problem. Indeed, the optimal control problem, sometimes referred as the Linear Quadratic Regulator (LQR) or Linear Quadratic Gaussian (LQG) problem, for control system design is concerned with the minimisation of a cost function involving error and control signals. However, this procedure suffers from some drawbacks since the choice of quadratic performance criteria for non-aerospace problems is not easy and also because the use of a state-feedback law causes severe difficulties when not all the states are available for measurement and feedback. These LQR/LQG systems therefore incorporate the use of observers to 'recover' inaccessible states by the introduction of an observer [(*Luenberger* (1966)] which increases the complexity of the feedback loop. Moreover, these methods were also complicated by the problem of observer transients, ie, the period before the estimated state variables converge to their true values. As a result, attempts were made to avoid state feedback by using output feedback in the solution of optimal control problems [*Levine* and *Athans* (1970)]. In the same period, as linear multivariable optimal controllers development was underway, a great deal of effort was also directed towards gaining a deeper insight into the structure of linear systems.

In this new approach, the structural concepts of controllability and observability

due to *Kalman* (1960) were exploited by *Gilbert* (1963) to produce the relationship between state-space and transfer-function descriptions. Since most physical systems exhibit MIMO behaviour with highly interactive dynamics, the realistic representation of many systems calls for high-order dynamics equations. This undoubtedly leads to complexity in the system's analysis and control. Thus, the idea of working with the simplest possible representation of systems became attractive and compelled several authors to work on obtaining minimal realisations and reduced-order models. However, it was clear that minimal realisations must be achieved in such a way as to preserve all the important characteristics of the system or process. Therefore, algorithms were soon developed for obtaining minimal-order state-space realisations from the transfer-function matrices [*Ho* and *Kalman* (1966), *Davidson* et *al* (1978)]. Thus, as a result of the theoretical work of *Falb and Wolovich* (1967) and *Rosenbrock* (1968, 1970), and others, a basis was formed on which numerous new design techniques were built. However, some of these methods suffered from their inability readily to guarantee the tracking accuracy and decoupling capability of the various channels in closed-loop systems incorporating multivariable plants.

Since the introduction of the multivariable H_∞ control theory by *Zames* (1981) to address the problem of designing a controller which minimises the H_∞ closed-loop transfer-function matrix, this methodology has received much attention [*Francis* and *Doyle* (1986), *Glover* and *MacFarlane* (1989)] in recent years. However, one major problem in this design procedure is that in most cases a

model of the whole plant is needed, and the controllers resulting from H_∞ design methodology can sometimes be mathematically complicated [*Carr* and *Grimble* (1992)]. Further development resulted in the formulation of the mixed H_2 and H_∞ control approach [*Bernstein et al* (1989) and *Zhou et al* (1990)]. It was thus clear that H_2 and H_∞ control designs are quite useful for robust performance design in the presence of parameter perturbations and uncertain disturbances. However, conventional output feedback designs of mixed H_2 and H_∞ optimal control turned out to be rather complicated and not easily implemented for practical industrial applications [*Chen et al* (1995)]. The extension of H_∞ control theory to non-linear dynamical systems was effected by Van Der *Schaft* (1991, 1992) and *Ball et* al (1993).

The Quantitative Feedback Theory (QFT) developed by [*Horowitz* (1979)] is an example of another multivariable control design technique which uses feedback control to achieve desired system performance. This is a very powerful design method when plant parameters vary over a broad range of operating conditions [*D'azzo* and *Houpis* (1995)]. However, it involves a lengthy procedural design process even for the simplest of systems. It is thus important to note, for example, that during the development and flight testing of a QFT flight control system for an unmanned research vehicle [*Sheldon* and *Rasmussen* (1994)], the controller had to be re-designed at several stages of development. However, this inconvenience has not prevented control engineers from using this powerful design methodology which was even developed in the case of

Multiple Input Single Output (MISO) systems as a CAD package by *Yaniv* (1992) or as a multivariable control toolbox for MATLAB/SIMULINK (MathWorks 1994).

The previously mentioned 'curse' of dimensionality associated with realistic representations of physical systems posed formidable computational problems for the analysis and control of these systems. Indeed, the presence of some 'parasitic' parameters such as small time constants, masses, moments of inertia is often the cause of the increased order and 'stiffness' of these systems. This stiffness, attributed to the simultaneous occurrence of 'slow' and 'fast' phenomena, gives rise to different time scales. The singular perturbation and time-scale methods, 'gifted' with the two remedial features of dimensional reduction and stiffness relief, are considered as a boon to control engineers. These techniques have attained a certain level of maturity in the theory of continuous control systems described by ordinary differential equations. The idea of singular perturbations in differential equations was first introduced to control theory by *Kokotovic* and *Sannuti* (1968), and further progress in this direction was achieved by *Kokotovic* and *Perkins* (1972). Some of the particular problems considered are multivariable systems with state feedback [*Porter* (1974, 1977, 1982), *Grujic* (1979) and *Bradshaw* and *Porter* (1979, 1981)], and output feedback [*Porter* and *Bradshaw* (1981)]. Other related problems are concerned with eigenvalues as proposed by *Porter* and *Shenton* (1975a), *Moore* (1976) and *Chow* (1978), transfer function matrices as investigated by

Porter and *Shenton* (1975b), and frequency domain characteristics as studied by *Luse* and *Khalil* (1985). The fact that the theory of difference equations is in most respects akin to that of ordinary differential equations leads to the singular perturbation analysis being successfully extended to discrete systems [*Locatelli* and *Schiavoni* (1976)]. In the beginning, attempts to model discrete systems with slow and fast behaviour in strictly perturbed structure faced some stability problems [*Comstock* and *Hsiao* (1976), and *Reinhardt* (1979)]. However, with the formulation of some other types of singularity pertubed structures, there has been considerable interest in the analysis of two-time-scale discrete systems [*Phillips* 1980, *Naidu* and *Rao* (1981, 1985), and *Mahmoud* (1982, 1986).

In other relevant work, *Porter* and his associates at the University of Salford [*Porter* and *Shenton* (1975a, b), *Porter* and *Bradshaw* (1979a ,b), *Bradshaw* and *Porter* (1980a, b), *Porter* (1982), *Porter et al* (1985), *Porter* and *Manganas* (1985, 1987)] have developed design techniques for high-performance set-point tracking systems which are conceptually and computationally easy to implement. These controllers, which ensure 'tight' and non-interacting set-point tracking together with disturbance rejection, are robust in the face of plant-parameter variations [*Porter* and *Othman* (1990), *Porter* and *Abidin* (1990a)]. Furthermore, the design of such controllers is remarkably simple since, even in the case of complex and highly interactive multivariable plants only the step-response matrices of such plant are required in the synthesis of control laws. These remarkable features proved to be essential when tackling non-linear

time-varying plants such as robotic manipulators [*Porter* and *Abidin* (1991a)]. In addition, following the guidelines advocated by *Porter* (1989) for designing practical intelligent digital controllers for multivariable plants, the application of powerful optimisation techniques using genetic and fuzzy-genetic methodologies in the design of robotic controllers and the associated identification processes was presented by *Porter, Sangolola* and *Zadeh* (1994) and *Porter* and *Zadeh* (1995a, b).

In the aftermath of the development of the feedback-amplifier design procedure described by *Bode* (1945), feedback control of industrial processes was becoming standard. This field, characterised by processes that are highly complex and also non-linear and subject to relatively long time delays between actuator and sensor, developed Proportional-Integral-Derivative (PID) control. PID controllers, the bread and butter of control engineering practice, are found in large numbers in all industries. They come in many different forms and are also embedded in various kinds of special purpose control systems. Most feedback loops are controlled by this most common control algorithm or some minor variation of it. However, although PID controllers are common and well known, they are often poorly tuned. Evidence of this can be found in the control rooms of most industrial plants. Thus, the derivative action in PID controllers is frequently switched off for the simple reason that it is difficult to tune properly. Nevertheless, enormous progress has been made since PID controllers were first described by *Callender et al* (1936). The association of PID controllers with

the different SISO and MIMO design techniques led to the pressing need to provide systematic tuning of such controllers so that the full and optimal performance of the application of these controllers could be achieved [*Astr□m* and *H□gglund* (1988)]. The ongoing process of acquiring computer power offers interesting possibilities to provide automatic tuning as well as adaptation to changing operating conditions as is commonly the case in real industrial process. The development and use of autotuning devices offered several advantages. Since automatic tuning is faster than manual tuning, the commissioning time for installation of new processes was decreased. Automatic tuning also means that the tuning is made systematically, even for the simplest control loops. The simplicity is an advantage for those loops that do not require any sophisticated controller. However, there are some drawbacks. The simple structure gives the controller limited behaviour. Indeed, control loops with large dead-time or other kinds of complex dynamics are hard to control efficiently with traditional PID controllers. On the other hand, controllers based on methods that require much *a priori* information often have a pretune phase that helps the operator in the choice of such information. Hence, the autotuning devices available on the market today differ with respect to the amount of prior knowledge needed. For more difficult control problems, the controllers are often tailor-made. Simple PID controllers are then not appropriate, but rather robust PID controllers relying on automatic tuning methods with more sophisticated properties such as intelligence and adaptation are required.

1.3.2 ROBOT CONTROL

The role of robot control is to integrate all the outputs supplied from the robot sensing system into the of control inputs that are able to realise the movements required in performing the tasks imposed on robots. In the early generations of robot control, when roboticists were enthusiastically dreaming of intelligent robots that could undertake humans tasks, many people took it for granted that robot control would soon be accomplished by simply applying the outstanding achievements of modern control theory to robots. However, despite these achievements, the difficulties in understanding human motor control and vision have greatly upset the dream of producing human-like intelligent robots before entering the new millennium. Despite this temporary failure to produce what sci-fi movies describe as "humanoids", a great deal of work has been accomplished on the control of complex non-linear systems such as robotic manipulators.

One of the major tasks assigned to robotic manipulators is to track given positional trajectories with high accuracy. In order to achieve accurate control of a robot during the execution of such tasks, feedback control is used. The simplest and most common controllers, which are extensively used in current industrial robotic manipulators, are proportional-plus-derivative (PD) controllers [*Craig*(1989), *Fu et al* (1987), and *An* and *Atkeson* (1989)]. These PD controllers contain no dynamical models of manipulators. In fact, these simple

PD control laws are used in decentralised control schemes in which the robotic system is considered as a set of independent sub-systems which can be stabilised by local servos [*Vukobratvic* and *Stokic* (1985)]. Therefore, PD controllers can usually achieve satisfactory performance for robotic manipulators which are indirectly driven by geared systems because coupling effects become negligible due to the associated high gear ratios. However, in recent years, direct-drive manipulators have begun to play an increasingly important role in many industries. There is no gearing between motors and arm links in direct-drive robotic manipulators, and coupling effects therefore become significant. Nevertheless, the usefulness and effectiveness of such PD controllers were comprehensively discussed in [*Kawamura et al* (1988)]. The asymptotic stability of such linear PD controllers for highly non-linear robotic manipulators was also discussed in [*Asada* and *Slotine* (1986)] using Lyapunov's method. It was argued that such linear PD controllers are useful not only for positioning but also for trajectory tracking [*Kawamura et al* (1988)] [*Koditshek* (1987)]. However, PD controllers ensure the asymptotic stability of position control if the gravity term can be compensated carefully by installing weight-balancers at some manipulators links or feedforward control on the basis of real computation for the gravity term. This result was first obtained by *Takegaki* and *Arimoto* (1981). But, without compensating for the gravity term, any PD controller for set-point positional control results in a steady-state error as long as the dynamics of the robot manipulator are subjected to gravity forces. In order to remedy this flaw, an integral term of positional errors can be

introduced together with PD terms and thus leads to asymptotic stability of positioning in a local sense. Indeed, most stability and tuning analyses have been carried out by assuming local PD or PID control of robotic manipulators which have been modelled by second-order linear systems.

However, this tuning procedure becomes only approximate when there are large dynamical interactions and significant non-linearities within dynamical models. In such cases, the tuning procedure is usually performed by a trial-and-error process [*An et al* (1988), *Desilva* (1995)] . Recently, a tuning procedure for PID control of robotic manipulators has been proposed [*Kelly* (1995)] in which lower bounds on proportional and derivative matrices have been obtained by invoking Lyapunov's direct method for set-point control. In addition, Kelly (1995) has also reported some other relevant work on the stability of such controllers for robotic manipulators and has emphasised that it is very hard to extract an explicit tuning policy for the gains of such linear controllers from these results. However, Kelly (1995) has not proposed any tuning procedure for finding optimal gains such that a specified cost function can be optimised. Indeed, it is very difficult to find optimal gains for such linear controllers when they are applied to the non-linear and coupled dynamical models of robotic manipulators. Moreover, the problem of finding optimal gains becomes even more complex when multivariable (centralised) linear PD or PID controllers are used for robotic manipulators instead of monovariable (decentralised) controllers.

Another approach for robot control is linear perturbation adaptive control in which a perturbed control effort is used to correct the nominal control effort based on the nominal model of a robotic manipulator. A MRAC Lyapunov-based design technique was used in [*Takegaki* and *Arimoto* (1981)] in order to derive the adaptation law for the perturbed control effort. Similarly, a self-tuning design technique based on the discrete-time model of the desired trajectory was used in *Lee* and *Chung* (1982. 1984).

Most of the research in adaptive control for robotic manipulators has proceeded by using the structure of the manipulator dynamics, and these schemes were accordingly called model-based adaptive controllers in [*Colaugh et al* (1995)]. The first globally convergent, non-linear adaptive control law for these schemes was developed by *Craig et al* (1986), in which the structure of the computed-torque method was maintained whilst the unknown regrouped dynamical parameters of the manipulator were modified adaptively in a Lyapunov sense. In this approach, the inertia matrix of the robotic manipulator needs to remain bounded during adaptation of the dynamical parameters. Similarly, *Slotine* and *Li* (1987,1988) proposed an adaptive control algorithm which consists of a PD feedback part and a full dynamical feedforward compensation part with the unknown parameters being estimated adaptively. In this adaptive control algorithm, zero steady-state joint velocity errors, rather than zero steady-state joint position errors, are guaranteed. In order to modify the algorithm, a sliding surface was used such that zero steady-state errors for joint position were also

guaranteed. The use of inverse dynamics of robotic manipulators for adaptive control schemes was also reported by *Spong* and *Ortega* (1990) and *Dawson* and *Lewis* (1991) in which the approach of Craig *et al* [1986] was modified so as to avoid the need for inverting the inertia matrix of a robotic manipulator during adaptation. A summary description of all these adaptive controllers for robotic manipulators can be found in [*Hsia* (1986)] and [*Ortega* and *Spong* (1988)]. However, the main drawback of these controllers is the computational complexity of the adaptive schemes [*Sadegh* and *Horowitz* (1990)], which require the on-line computation of many non-linear functions of joint positions and velocities. Digital implementation of these schemes may therefore require a slow sampling rate, which in turn may degrade the performance of the controller. Such adaptive control schemes are accordingly rather too complicated for routine industrial use and have therefore failed to replace the relatively simple PD or computed-torque/PD controllers frequently used in practical applications of industrial robots.

In a different approach, another control algorithm that was used for robotic manipulators introduced sliding mode control [*Young* (1978), *Slotine* (1985) ,and *Su* and *Leung* (1993)]. The control law in this case is generally a switching controller in which the controller is designed such that the sliding mode occurs on the predefined sliding surface. However, these types of controller frequently suffer from chattering as a result of the use of high switching frequencies and the presence of unmodelled dynamics of the system.

The extension of H_∞ control theory to non-linear dynamical systems due to Van Der *Schaft* (19921, 1992) and others has promoted the use of H_∞ control for robotic manipulators in line with the general theoretical approaches mentioned by *Astolfi* and *Lanari* (1994a, b). In addition to this early promotion of the application of the H_∞ control theory to robotic manipulators, *Arimoto* (1995d, 1996) proposed a simplification of the treatment of H_∞ control theory for robotic manipulators on the basis of physical interpretations of energy conservation and dissipation in mechanical systems. However, the analytical complexity of the H_∞ -tuning problem for highly non-linear electromechanical systems such as robotic manipulators favours the use of simpler alternative control methodologies. More recently, a new optimal control approach to robust control of robotic manipulators in the framework of *Lin* (1997) has been proposed by *Lin* and *Brandt* (1998). This solves the robust control problem by solving the optimal control problem, which is much easier to accomplish in many cases. However, although this most recent approach is fundamentally different from all the previous approaches, it bears resemblance to the robust saturation approach in the sense that some kind of Lyapunov argument is used. Among other approaches, it is also relevant to refer to the learning control approach introduced in the work of *Arimoto et al* (1984) and *Arimoto* (1990) which was subsequently further extended by *Gorinevsky et al* (1997). It is also relevant to note that the use of the powerful QFT method might be effective for time-varying plants such as robotic manipulators which parameters vary over a broad

range of operating conditions [*D'azzo* and *Houpis* (1995)]. However, as previously mentioned, the lengthy procedural design process involved in the case of complex systems such as robotic manipulators would ultimately favour straightforward simpler design methodologies.

1.4 EVOLUTIONARY COMPUTATION

Although the origin of evolutionary computation (EC) can be traced back to the late 1950's and early 1960's [*Box* (1957) and *Bremermaan* (1962)], this field remained relatively unknown to the broader scientific community for almost three decades. This was largely due to the lack of available powerful computer platforms at that time, but also due to some methodological shortcomings of the early approaches [*Fogel* (1995)]. However, the fundamental work of *Holland* (1962), *Rechenberg* (1965), *Schwefel* (1968), and *Fogel* (1962) served gradually to change this picture during the 1970's. There is currently a remarkable and steady increase of interest of the scientific, academic, and industrial communities in EC as a clear demonstration of the scientific and economic relevance of this field.

In view of the significant effort invested in the area of evolutionary computation, it was necessary for EC advocates to justify this effort by displaying the benefit of EC compared to other approaches. *B□ck et al* (1997) therefore argued that

the most significant advantage of using evolutionary search methods lies in the gain of flexibility and adaptability to the task at hand, in combination with robust performance (although this depends on the problem class) and global search characteristics. In fact, EC should be understood as a general adaptable technique which is especially well suited to solving difficult multi-dimensional optimisation problems. However, as pointed out by *Porter* (1997), it is necessary to be careful not to confuse mere algorithmic novelty with real practical utility.

The majority of current implementations of evolutionary algorithms descend from the two strongly related but independently developed techniques:

i) Genetic Algorithms (GA)

ii) Evolution Strategies (ES)

GA and ES, which were originally developed respectively by *Holland* (1975) and, *Rechenberg* (1973) and *Schwefel* (1975), are search procedures based on natural evolution. Both GA and ES are, in fact, strongly influenced by classic Darwinian evolutionary theory, and mimic the mechanics of natural selection and the natural genetics of biological organisms, in which the "*survival of the fittest* " is the basis for the evolution of natural populations during many generations. Indeed, all natural species survive by adapting themselves to the environment, which is the basic underlying theme of both GA and ES. There is

competition in nature between individuals in a population such that those which have unfit characteristics will be eliminated in the subsequent generations. However, those individuals which are most successful in surviving will attract mates and will therefore have the largest numbers of offspring: each offspring inherits some characteristics from each parent. GA and ES use this analogy with natural behaviour and are thus able to evolve solutions to real-world problems by exploring and finding solutions through the search in either a multidimensional Euclidean space or various discrete spaces. Indeed, in many industrial optimisation (control, job-machine assignment, scheduling or planning) problems, the cost function could be non-differentiable and/or discontinuous with various forms of constraints. This global random search approach has been recognised as an attractive future direction in the optimisation of industrial systems.

GA and ES provide an iterative procedure, which works with populations of individuals. Each individual may represent a solution to a problem and has a 'score' showing its fitness as a solution. During the next iteration, called a *generation*, an objective performance measure is used to assess the fitness of each solution. Then, on the basis of these fitnesses, a selection mechanism determines which solution should be maintained as "parents" of the subsequent generation. Since both GA and ES are inherently parallel, all strings or individuals evolve simultaneously without central co-ordination. Thus, the population gets more chance to converge toward an optimal solution of the

problem. However, the differences between GA and ES are characterised by the type of alteration that are imposed on solutions to create "offspring", the methods employed for selecting new parents, and the data structures that are used to represent solutions. More details on GA and ES are given in Appendix A.

It is evident that GA and ES are different from other optimisation and search methods in the following principal respects:

i) By working with a population of individuals (potential solutions) rather than a single individual, GA and ES reduce the chance of getting stuck on a false optimum.

ii) GA and ES offer the flexibility of using either real or binary representation of encoded search space, thus adding efficiency to the search procedure.

iii) GA and ES require values only of the relevant objective function, rather than values of its derivative, thus broadening the applications of GA and ES.

iv) Both GA and ES use probabilistic transition rules, rather than deterministic transition rules.

The increased interest of the scientific community in the EC approach during the last decade has considerably helped to promote this new trend of artificial intelligence as an optimisation technique by highlighting the advantages of the EC techniques over their non-evolutionary counterparts. In addition, this decade of development for EC has also served as a fertile ground for the cultivation of diverse opinions within the EC community on the relative merits of individual EC techniques. EC effectiveness extends to a broad field of applications, but of course with a corresponding loss of efficiency when applied to the classes of simple problems classical for which traditional procedures have been specifically devised.

The advantages of using evolutionary techniques compared with other optimisation techniques for real complex optimisation problems were addressed in *Goldberg* (1989) and *Beasly et al* (1993) in the case of GA and by *Schwefel* (1997) in the general case of EA. In addition, there are some explanatory comparisons of GA, hill climbing, and simulated annealing [*Michalewicz* (1992)] that clearly show the uniqueness and superiority of genetic algorithms. Moreover, *Porter* (1995) has shown that genetic design techniques are frequently superior to non-genetic techniques in the following principal respects in control problems:

(i) there is no conceptual difference between solving multivariable and

monovariable control problems,

(ii) practical constraints such as hard amplitude and rate limits on inputs and outputs can be readily satisfied,

(iii) robustness requirements can be explicitly introduced with great ease.

In his survey of the use of EC for control engineering, *Dracopoulos* (1997) reviewed the most successful applications of GA in control engineering in either their pure or hybrid form (mostly used with neural networks and fuzzy logic). In order to demonstrate the power of these techniques *Dracopoulos* (1997) describes how a hybrid genetic controller can be successfully applied to the control of complex (even chaotic) dynamic systems in the case of the detumbling and attitude control of a satellite.

Since the first engineering application of GA to pipeline optimisation by Goldberg in 1983, there has been an increasing usage of GA in many different search and optimisation problems. In fact, the range of applications of GA for practical real world problems is vast. Thus, for example, a wide range of application case studies was collected in [*Davis* (1991)] comprising GA applications in aircraft design, telecommunications, robot trajectory generation, neural network synthesis, fault diagnosis, scheduling, and some other engineering applications. GA were also used to optimise the performance of GA

on a given of function optimisation problem in a two-level genetic algorithm design [*Grefenstette* (1986)]. In addition, it was shown that GA can be effectively used for automatic path planning and collision-free path generation for robotic systems [*Davidor* (1990)] [*Solano* and *Jones* (1994)]. In other work, *Porter* and *Allaoui* (1996) successfully proposed the use of GA in the synthesis of optimal control policies for manufacturing systems.

In the use of GA for control system optimisation, some early researches were reported in *Porter* and *Jones* (1992) and *Krishnakumar* and *Goldberg* (1992). Indeed, *Porter* and *Jones* (1992) used GA to design optimal PID controllers for the same plant as that investigated non-genetically by *Polak* and *Mayne* (1976). It was shown by *Porter* and *Jones* [1992] that the genetic approach is much simpler than that of the rather complicated non-genetic optimisation algorithms previously proposed by *Polak* and *Mayne* (1976). *Hunt* (1992) has used GA to tackle the problem of synthesising LGQ and H_∞ optimal controllers. GA were also used successfully in the area of system modelling and identification [*Kristinson* and *Dumont* (1992)] [*Maclay* and *Dorey* (1992)]. In this area, GA were used effectively in a parallel structure to estimate a hydraulic system parameters in a heavy-duty hydraulic-actuated manipulator by *Sepehri et al* (1994). Furthermore, GA were also used to identify the dynamical model of a direct-drive robotic manipulator in a way that is very suitable for practical use [*Porter et al* (1994)].

On the other hand, ES which were used by *Schwefel* (1965, 1968) and *Lichtfuß* (1965) in their initial version were subsequently improved. These algorithms were used to deal with applications like the travelling salesman problem or TSP [*Herdy* (1991)], in training neural networks [*Salomon* [1991)], in the design of a truss bridge [*Rechenberg* (1995)]. However, besides these non control engineering applications, ES were also used successfully in control system optimisation of industrial problem such as production planning [*Porter* (1997)], in the synthesis of optimal control policies for manufacturing systems [*Porter* and *Merzougui* (1997)], in effectively solving job-scheduling problems [*Porter* and *Zadeh* (1998)] and in the synthesis of robust digital trajectory-tracking controllers for robotic manipulators [*Porter* and *Allaoui* (1998)].

It is thus evident that there is increasing interest in GA and ES and their applications. This indicates that both the academic and industrial communities are seriously considering the potential of the EC techniques as a future direction adapted to the challenges and requirements of the new millennium.

CHAPTER 2

2.1 SCOPE OF THE THESIS

The design of tracking systems, in which the plant output is required to track or follow accurately the command input, has been for many years an important issue in control engineering and has therefore been extensively investigated. This requirement of accurate tracking is of paramount importance in many industrial processes that are automated by means of robotic technology. Indeed, a robotic manipulator is a typical system for accurately tracking specific trajectories while moving parts and tools in its work-space.

In order to achieve the required high-accuracy performance of robotic manipulators, it is necessary to choose suitable control algorithms which exhibit excellent tracking behaviour. The importance of such control algorithms becomes evident when it is considered that the dynamical equations governing the motion of robotic manipulators are highly coupled and non-linear. Even more importantly, the unavailability of an accurate and precise linearised model will ultimately result in failure in the majority of currently available control design methodologies. These contentions led to the choice of the fast-sampling PID digital controllers of *Porter* and *Abidin* (1990b) which overcome the problem of obtaining an accurate linearised model by utilising data directly obtained from input/output measurements in the time-domain.

However, although the theoretical results of *Porter* and *Abidin* (1990) greatly facilitate the design of digital trajectory-tracking controllers, these results are valid only asymptotically (i.e., as the sampling frequency of the digital PID controllers becomes infinite). In practice, of course, the sampling frequencies of such digital controllers must remain finite; but it has so far proved impossible to obtain theoretical non-asymptotic results.

The core problem in robot control is that the optimal values of the control parameters are not known and there is no straightforward algorithm to discover them. In the case of fast-sampling digital PID controllers, the design engineer must often resort to the manual tuning of important parameters to achieve increasingly high performances. Indeed, it was mentioned by *Porter* (1995) that the selection of a suitable set of tuning parameters for that type of controller is frequently difficult, even if expert systems are used for this purpose [*Porter* (1989)]. The obtained performance will, therefore, in most cases lack optimality. Ideally, powerful optimisation algorithms should be implemented in order to automate this tuning process. One aim of this thesis is accordingly the extension to robotic-control systems from flight-control systems of the work by *Porter* and *Hicks* (1994) on optimal design using genetic algorithms. The resulting automated optimisation technique is intended to determine the various controller tuning parameters under non-asymptotic conditions. The procedures presented in this thesis are thus the natural extension of the previous non-robotic studies of *Porter*

and *Jones* (1992) and *Porter, Mohamed* and *Jones* (1993). In these studies, it was shown that evolution-like mechanisms can automate the process of controller tuning without requiring accurate and complex knowledge of the control environment.

The use of evolutionary computational techniques, rather than traditional search and optimisation procedures, is stimulated by the limitations in terms of 'efficiency' or 'robustness' of traditional optimisation procedures. These used techniques such as hill-climbing (gradient method), simulated annealing, exhaustive search, linear and non-linear programming [*Hicks* (1994)]. Indeed, whilst these traditional techniques have been successfully used in various fields, they often exhibit limitations. Thus, they are frequently not robust to changes in the domain of application and often fail to find global optimal in functions when they are surrounded by local optimal. In particular, they do not offer an escape from the curse of dimensionality inherent in very complex systems such as robotic manipulators. The success of such traditional techniques using conventional procedures occurs either in situations where the systems can be modelled with accuracy, or when the number of plausible parameter values is small enough to be tested exhaustively [*Davidor* (1991)]. As a result of these limitations the use of traditional optimisation techniques when designing a fast-sampling digital PID controller for complex systems such as robotic manipulators is therefore likely to produce poorly tuned controllers.

In recent years, the availability of fast processors and large memory PCs, in addition to parallel processing mainframe facilities, has promoted the use of evolutionary computation (which encompasses techniques such as genetic algorithms, evolution strategies and evolutionary programming) to solve engineering design problems. However, as pointed out by *Porter* (1997), it is necessary to be very careful not to confuse mere algorithmic novelty with real practical utility. In addition, it is argued by *Bäck et al* (1997) that the most significant advantage of using evolutionary search —in contrast to classical or standard methods— lies in the gain of flexibility and adaptability, in combination with robust performance and global search characteristics.

In this thesis, it is shown that the automated tuning of optimal controller parameters can be effectively achieved using strongly related evolutionary approaches such as genetic algorithms (GA) (Appendix A.1) and evolution strategies (ES) with non-adaptive and adaptive variants (Appendix A.2). Genetic algorithms, introduced by *Holland* (1975) and subsequently described by *Goldberg* (1989) and others, were originally proposed as a general model of adaptive processes; but by far the largest application of the technique has occurred in the domain of optimisation. Evolution strategies, as developed by *Rechenberg* (1973) and extended by Schewfel (1991) and others, were initially designed with the objective of solving difficult discrete and continuous parameter optimisation problems.

In fact, both genetic algorithms and evolution strategies are strongly influenced by classic Darwinian evolutionary theory, combined with the selectionism of Weismann and the genetics of Mendel (*Fogel* (1997)). These epochal results culminated, following the work of *Fisher* (1930), *Haldane* (1932) and *Wright* (1932), in the neo-Darwinism synthesis. In their application to practical problem solving ,genetic algorithms and evolution strategies begin with populations of competing alternative solutions. New solutions are then created by randomly altering the existing solutions, and an objective measure of performance is then used to assess the "fitness" of each solution. On the basis of these "fitnesses", a selection mechanism determines which solutions should be maintained as "parents" for the subsequent generation. The differences between genetic algorithms and evolution strategies are characterised by the type of alterations that are imposed on solutions to create "offspring", the methods employed for selecting new parents, and the data structures that are used to represent solutions.

It is convenient to begin the investigations presented in this thesis by describing the previously established techniques for the non-evolutionary design of fast-sampling digital PID controllers for robotic manipulators of *Porter* and *Abidin* (1990b). This desciption is followed by the use of genetic algorithms to achieve the optimal robustified design of digital fast-sampling trajectory-tracking PID controllers for robotic manipulators under non-asymptotic conditions. Then, evolution strategies with both non-adaptive and adaptive variants are used as an

alternative technique in the design of optimal robustified digital fast-sampling PID trajectory-tracking controllers for robotic manipulators, for both the constrained and unconstrained designs. In each design case presented in this thesis, initial comparisons are made between the performance of 'evolutionarily' tuned controllers and that of their 'non-evolutionarily ' tuned counterparts. This is followed by comparisons between the performances of different variants of the 'evolutionarily' tuned controllers. These results demonstrate the effectiveness of powerful but simple genetic algorithms, non-adaptive evolution strategies or adaptive evolution strategies in the selection of controller parameters so as to provide the required high-accuracy tracking performance of robotic manipulators. This demonstration of the powers of evolutionary design methodologies is proffered in the belief that, as mentioned by *Porter* (1995), control engineering has been revolutionised by an increasing interest in with the emulation of biological processes such as those involved in evolution.

2.2 OUTLINE OF THE THESIS

This thesis is divided into six parts. In Part I, Chapter 1 briefly describes the challenge facing control engineers who strive for optimal performance by resorting to new theoritical and computational methods in an industrial world invaded by complex machines such as robots, gives a brief survey of the robot dynamics analysis, presents and discusses previous approaches and research contributions

in the field of modern control and robot control, introduces genetic algorithms and evolution strategies as evolutionary computation optimisation techniques, and surveys the applications of such techniques to non-robotic and robotic control problems. Chapter 2 presents an outline of the objectives of this thesis.

In Part II, the non-evolutionary design of digital trajectory-tracking controllers is considered. Indeed, Chapter 3 describes the asymptotic design of PID trajectory-tracking controllers using non-evolutionary fast-sampling techniques.

In Part III (Chapters 4, 5, 6 and 7), the constrained genetic design of fast-sampling error-actuated digital trajectory-tracking PID controllers is considered. Chapter 4 presents the genetic design of digital trajectory-tracking controllers for robotic manipulator in the presence of sudden changes of payload. Chapter 5 introduces the design of digital trajectory-tracking PID controllers for robotic manipulators using both non-adaptive and adaptive evolution strategies. The use of different performance measures in the evolutionary design of digital controllers for robotic manipulators for typical trajectory-tracking tasks is investigated in Chapter 6. Chapter 7 presents an evaluation of the performance of the evolutionary designed digital tracking-trajectory controllers for robotic manipulators.

In Part IV, Chapter 8 the unconstrained genetic design of fast-sampling error-actuated digital trajectory-tracking PID controllers is considered. In such cases, the designer has the ability freely to choose values of the elements of the

multivariable controllers matrices without being constrained by formal design equations.

In Part V, Chapter 9 demonstrates the practical utility of the developed evolutionary design method for digital controllers for robotic manipulators. The use of evolutionary design procedures is illustrated by the implementation of digital PID trajectory-tracking controllers for a two-link direct-drive robotic manipulator.

In Part VI (Chapter 10), the conclusions that emerge from the use of these evolutionary design procedures are presented and recommendations for further work are suggested.

Finally, Appendix A gives a brief introduction to the genetic algorithms and evolution strategies used as evolutionary optimisation techniques in this thesis. In Appendix B, the dynamical model of the three-link robotic manipulator used in Parts II, III and IV is derived. In Appendix C, the dynamical model of the two-link direct-drive robotic manipulator used in the practical implementation of Part V is derived. Appendix D describes the practical two-link direct-drive robotic manipulator used for experimental purposes.

PART II

CONSTRAINED DESIGN OF DIGITAL TRAJECTORY-TRACKING CONTROLLERS USING SINGULAR PERTURBATION METHODS

CHAPTER 3

DESIGN OF FAST-SAMPLING DIGITAL CONTROLLERS FOR ROBOTIC MANIPULATORS USING SINGULAR PERTURBATION METHODS

3.1 INTRODUCTION

The design of effective digital trajectory-tracking controllers for robotic manipulators constitutes a difficult problem in control engineering, because such manipulators are non-linear multivariable plants with time-varying parameters. However, it was shown by *Porter and Abidin* (1990a) that the design of digital trajectory-tracking controllers for completely irregular multivariable plants (i.e., plants like most robotic manipulators) can be readily effected by using the singular perturbation methodologies of *Porter et al* (1985) for the design of fast-sampling error-actuated digital multivariable PID controllers.

These design methodologies are characterised by the following very important features:

1) only the first and second Markov parameters of the linear components of such plants are required in the controller design;

2) the design can be further simplified by introducing the step-response matrices of linear multivariable plants instead of the first and second Markov parameters;

3) the robustness characteristics of the controller can be expressed in terms of the step-response matrices of the nominal and actual plants

This chapter summarises the prominent features of the methodology for the design of fast-sampling digital PID controller for completely irregular linear multivariable plants developed by *Abidin* (1991) using the singular perturbation methodologies of *Porter et al* (1985).

These general results are illustrated by the design of a fast-sampling error-actuated digital PID controller for the three-degree-of-freedom robotic manipulator previously investigated by *Petropoulakis (1986).*

3.2 SYSTEM CONFIGURATION

It was shown by *Abidin* (1991) that the design of digital trajectory-tracking controllers for completely irregular linear multivariable plants can be readily effected by using the methodologies of *Porter et al* (1985).

The digital trajectory-tracking systems under consideration, as shown in Figure 3.1, consist of linear multivariable plants together with digital PID controllers. These multivariable plants are assumed to exhibit minimum-phase

characteristics and therefore to be amenable to fast-sampling error-actuated digital control [*Bradshaw* and *Porter* (1980a)]. Such plants are governed on the continuous-time set $T = [0, +\infty)$ by state and output equations of the respective forms [*Manganas* (1985)]

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

and

$$y(t) = Cx(t), \tag{3.2}$$

where

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \in \Re^{nxn}, \tag{3.3}$$

$$B = \begin{bmatrix} O_{n-1} \\ B_2 \end{bmatrix} \in \Re^{nxl}, \tag{3.4}$$

$$C = [C_1 \; C_2] \in \Re^{lxn}, \tag{3.5}$$

and

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \in \Re^n \, . \tag{3.6}$$

In these equations, $x(t) \in \Re^n$ is the plant state vector, $u(t) \in \Re^l$ is the plant input vector, $y(t) \in \Re^l$ is the plant output vector, A is the plant matrix, B is the input matrix, C is the output matrix, $x_1(t) \in \Re^{n-l}$, $x_2(t) \in \Re^l$, $A_{11} \in \Re^{(n-l)x(n-l)}$, $A_{12} \in \Re^{(n-l)xl}$, $A_{21} \in \Re^{lx(n-l)}$, $A_{22} \in \Re^{lxl}$, $B_2 \in \Re^{lxl}$, $C_1 \in \Re^{lx(n-l)}$, $C_2 \in \Re^{lxl}$. Furthermore, in view of the assumed complete irregularity of the plants, the first Markov parameter, $CB \in \Re^{lxl}$, of the continuous-time open-loop plant [*Abidin* (1991)] is such that

$$rank\, CB = 0 \quad , \tag{3.7}$$

and the full rank second Markov parameter, $CAB \in \Re^{lxl}$, is such that

$$rank\, CAB = l \quad . \tag{3.8}$$

In order to design digital trajectory-tracking controllers for such plants, it is convenient to model these plants on the discrete-time set $T_T = \{0, T, 2T, \ldots\}$ by the respective state and output equations

$$x\{(k+1)T\} = \Phi x(kT) + \Psi u(kT) \tag{3.9}$$

and

$$y(kT) = \Gamma x(kT) \quad , \tag{3.10}$$

where

$$\Phi = e^{AT} = I_n + TA + \frac{T^2}{2!}A^2 + \ldots + \frac{T^i}{i!}A^i + \ldots \in \Re^{nxn} \quad , \tag{3.11}$$

$$\Psi = \int_0^T e^{At} B dt = TB + \frac{T^2}{2!}AB + \ldots + \frac{T^i}{i!}A^{i-1}B + \ldots \in \Re^{nxl} \quad , \tag{3.12}$$

and

$$\Gamma = C \in \Re^{lxn} \quad . \tag{3.13}$$

In equations (3.9), (3.10), (3.11) and (3.12), $I_n \in \Re^{nxn}$ is the identity matrix, $T \in \Re^{+}$ is the sampling time period, and $k \in \{0,1,2,\ldots\}$.

The fast-sampling digital PID controllers incorporated in such trajectory-tracking systems are governed on the discrete-time set $T_T = \{0,T, 2T,\ldots\}$ by control-law equations of the form [*Porter et al* (1985)]

$$u(kT) = K_1 r(kT) + K_2 z(kT) \quad . \tag{3.14}$$

In equation (3.14), $K_1 \in \Re^{lxl}$ and $K_2 \in \Re^{lxl}$ are the proportional and integral

controller matrices, and the vectors $r(kT) \in \Re^l$ and $z(kT) \in \Re^l$ are generated in accordance with the difference equations [*Porter* (1987)]

$$s\{(kT + 1)T\} = -\alpha s(kT) + e(kT), \tag{3.15}$$

$$r(kT) = -\frac{2}{T}(1 + \alpha)Ds(kT) + \left(I_l + \frac{2}{T}D\right)e(kT), \tag{3.16}$$

and

$$z\{(k+1)T\} = z(kT) + Tr(kT). \tag{3.17}$$

Moreover, in equations (3.15), $\alpha \in (-1, +1]$, $s(kT) \in \Re^l$, $e(kT) = v(kT) - y(kT) \in \Re^l$ is the error vector, $v(kT)$ is the set-point command vector, and the derivative matrix $D \in \Re^{lxl}$ is such that

$$rank\, D = l. \tag{3.18}$$

Now, it is convenient to express the design equations of *Porter et al* (1985) for the proportional and integral controller matrices in equation (3.14) in the forms [*Porter* (1987)]

$$K_1 = TH^{-1}(T)\Sigma(TI_l + 2D)^{-1} \in \Re^{lxl} \tag{3.19}$$

and

$$K_2 = \rho T H^{-1}(T)\Sigma(TIl + 2D)^{-1} \in \Re^{lxl} \quad . \tag{3.20}$$

In equations (3.19) and (3.20),

$$\Sigma = \sigma I_l (\sigma \in \Re^+) \tag{3.21}$$

is the positive diagonal tuning matrix,

$$D = \delta I_l (\delta \in \Re^+) \tag{3.22}$$

is the positive diagonal derivative matrix, and ρ is the positive tuning parameter. Furthermore, in equation (3.20),

$$H(T) = \int_0^T C e^{A t} B \, dt \tag{3.23}$$

is the step-response matrix of the open-loop plant with state-space triple (A,B,C) governed by the state-space equations (3.1) and (3.2).

It is evident from equations (3.9), (3.10), (3.11), (3.12), (3.13), (3.14), (3.15), and (3.16) that such discrete-time closed-loop tracking systems are governed on T_T by state and output equations of the respective forms [*Abidin* (1991)]

$$\bar{x}\{(K+1)\}T = \bar{A}\bar{x}(KT) + \bar{B}u(KT) \tag{3.24}$$

and

$$y(t) = \bar{C}\bar{x}(KT) \quad . \tag{3.25}$$

In equations (3.24) and (3.25),

$$\bar{x}(KT) = \begin{bmatrix} z(KT) \\ x(KT) \end{bmatrix} = \begin{bmatrix} \bar{x}_1(KT) \\ \bar{x}_2(KT) \end{bmatrix} \in \Re^{n+2l} \ , \tag{3.26}$$

$$\bar{A} = \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{bmatrix} \in \Re^{n+2lxn+2l} \ , \tag{3.27}$$

$$\bar{B} = \begin{bmatrix} \bar{B}_1 \\ \bar{B}_2 \end{bmatrix} \in \Re^{n+2lxl} \ , \tag{3.28}$$

and

$$\bar{C} = \left[0_{l,2l}, \bar{C}_1\right] \in \Re^{lxn+2l} \ , \tag{3.29}$$

where

$$\bar{A}_{11} = \begin{bmatrix} I_l \ , & -2((1+\alpha)D \\ 0_l \ , & -\alpha I_l \end{bmatrix} \in \Re^{2lx2l} \quad , \tag{3.30}$$

$$\bar{A}_{12} = \begin{bmatrix} -(TI_l + 2D)C \\ -C \end{bmatrix} \in \Re^{2lxn} \quad , \tag{3.31}$$

$$\overline{A}_{21} = \left[\Psi K_2 \quad -\Psi K_1 \frac{2}{T}(1+\alpha) D \right] \in \Re^{nx2l} \quad , \tag{3.32}$$

$$\overline{A}_{22} = \left[\Phi - \Psi K_1 (I_l + \frac{2}{T} D) C \right] \in \Re^{nxn} \quad , \tag{3.33}$$

$$\overline{B}_1 = \begin{bmatrix} (TI_l + 2D) \\ I_l \end{bmatrix} \in \Re^{2lxl} \quad , \tag{3.34}$$

$$\overline{B}_2 = \Psi K_1 (I_l + \frac{2}{T} D) \in \Re^{nxl} \quad , \tag{3.35}$$

$$\overline{C}_1 = [C1 \quad C2] \in \Re^{lxn} \quad . \tag{3.36}$$

The transfer-function matrix relating the plant output vector to the command input vector of the closed-loop discrete-time tracking system governed by equations (3.24) and (3.25) evidently has the form [*Abidin* (1991)]

$$G(z) = [0_{l,2l} , \overline{C}_l] \begin{bmatrix} zI_{2l} - \overline{A}_{11} , & \overline{A}_{12} \\ \overline{A}_{21} , & zI_n - \overline{A}_{22} \end{bmatrix}^{-1} \begin{bmatrix} \overline{B}_1 \\ \overline{B}_2 \end{bmatrix} \tag{3.37}$$

3.3 ASYMPTOTIC PROPERTIES OF CLOSED-LOOP SYSTEM AS $T \rightarrow 0$

3.3.1 ANALYSIS

In order to study the asymptotic properties of the closed-loop system as $T \rightarrow 0$, it is essential to express the following matrices in their partitioned forms to facilitate the block-diagonalisation of the closed-loop plant matrix [*Abidin* (1991)]:

$$A_{11} = \begin{bmatrix} \Lambda_1 & \Lambda_2 \\ \Lambda_3 & \Lambda_4 \end{bmatrix} , \tag{3.38}$$

$$A_{12} = \begin{bmatrix} \Xi_1 \\ \Xi_2 \end{bmatrix} , \tag{3.39}$$

$$A_{21} = \begin{bmatrix} \Theta_1 & \Theta_2 \end{bmatrix} , \tag{3.40}$$

$$C_1 = \begin{bmatrix} \Delta_1 & \Delta_2 \end{bmatrix} , \tag{3.41}$$

$$C_2 = \begin{bmatrix} 0_{l,l} \end{bmatrix} , \tag{3.42}$$

where

$\Lambda_1 \in \Re^{n-2l \times n-2l}$, $\Lambda_2 \in \Re^{n-2l \times l}$, $\Lambda_3 \in \Re^{l \times n-2l}$, $\Lambda_4 \in \Re^{l \times l}$, $\Xi_1 \in \Re^{n-2l \times l}$, $\Xi_2 \in \Re^{l \times l}$,

$\Theta_1 \in \Re^{lxn-2l}$, $\Theta_2 \in \Re^{lxl}$, $\Delta_1 \in \Re^{lxn-2l}$, $\Delta_2 \in \Re^{lxl}$, and rank $\Delta_2 = l$.

Indeed, by introducing the matrices [*Abidin* (1991)]

$$\Omega = \Delta_1 \Xi_1 + \Delta_2 \Xi_2 \in \Re^{lxl} \text{ with rank } \Omega = l, \tag{3.43}$$

$$\Pi = \Delta_1 \Lambda_1 + \Delta_2 \Lambda_2 \in \Re^{lxn-2l} \quad , \tag{3.44}$$

$$\Upsilon = \Delta_1 \Lambda_3 + \Delta_2 \Lambda_4 \in \Re^{lxl} \text{ with rank } \Upsilon = l, \tag{3.45}$$

and by using the explicit expression for K_1 and K_2 given by equations (3.19) and (3.20) in equation (3.37), equations (3.38), (3.39), (3.40), (3.41), (3.42), (3.43), (3.44) and (3.45) facilitate the block-diagonalisation procedure. This, using the results of *Porter* and *Shenton* (1975a), indicates that the closed-loop transfer function *G(z)* assumes, as $T \to 0$, the asymptotic form [*Abidin* (1991)]

$$\Gamma(z) = \left[(z-1)(z+\alpha)I_l + \sigma(z+1)I_l\right]^{-1} \sigma(z+1)I_l \ . \tag{3.46}$$

In addition, block diagonalisation [*Abidin* (1991)] indicates that the poles of the closed-loop transfer function matrix assume the asymptotic values given by the elements of the set $Z_1 \cup Z_2 \cup Z_3 \cup Z_4$ as $T \to 0$. This block diagonalisation procedure also reveals the following characteristics [*Abidin* (1991)]:

1. The 'slow' modes Z_{s1} of the closed-loop system which correspond as $T \to 0$ to the poles $Z_1 \cup Z_2$, where

$$Z_1 = \{z \in \mathrm{C}: |zI_l - I_l + T\rho I_l| = 0\} \tag{3.47}$$

and

$$Z_2 = \left\{ z \in \mathrm{C}: \left| zI_l - I_l + T\left(\frac{(1+\alpha)}{2\delta}\right) I_l \right| = 0 \right\} \quad , \tag{3.48}$$

become asymptotically uncontrollable but remain observable as $T \to 0$.

2. The 'slow' modes Z_{s2} of the closed-loop system which correspond as $T \to 0$ to the poles Z_3, where

$$Z_3 = \{z \in \mathrm{C}: |zI_{n-2l} - I_{n-2l} + T\{\Lambda_1 - \Lambda_2\Delta_2^2\Delta_1 - \Xi_1\Omega^{-1}(\Pi - \mathrm{Y}\Delta_2^{-1}\Delta_1| = 0\}, \tag{3.49}$$

become asymptotically unobservable but remain controllable as $T \to 0$.

3. The 'fast' modes Z_f of the closed-loop system which correspond as $T \to 0$ to the poles Z_4, where

$$Z_4 = \{z \in \mathrm{C}: \left|(z-1)(z+\alpha)I_1 + \sigma(z+1)I_1\right| = 0\} \ , \qquad (3.50)$$

remain both controllable and observable as $T \rightarrow 0$.

Thus, as stated by *Manganas* (1985), the increasingly fast closed-loop behaviour as $T \rightarrow 0$ is a consequence of the fact that the slow modes corresponding to the transmissions zeros $Z_1 \cup Z_2$ of the PID controller become asymptotically unobservable, whilst the slow modes corresponding to the transmission zeros Z_3 of the open-loop plant become asymptotically uncontrollable. Furthermore, only the fast modes corresponding to the poles Z_4 remain both controllable and observable as $T \rightarrow 0$ and therefore appear in the outputs of the system.

In view of the fact that robotic manipulators are non-linear systems with time-varying plant parameters, it is in most cases a very difficult and time- consuming process to try to obtain an accurate model. The major problem is that the values of the parameters in the model are often not known accurately: this is particularly true of frictional effects [*Craig* (1986)]. Therefore, it is relevant to consider the robustness characteristics of fast- sampling digital PID controllers for completely irregular plants established by *Porter* and *Abidin* (1990a). In this context, robustness is the ability of such controllers to cope with the disparities between the actual and nominal plant parameters without causing instabilities . The nominal plant parameters are those parameters used in the actual design of the digital PID controllers.

Now, in order to investigate these robustness characteristics of the fast-sampling digital PID controllers, it is convenient to express the design equations (3.19) and (3.20) of *Porter et al* (1985) for the proportional and integral controller matrices in the forms [*Porter* (1987)]

$$K_1 = T\tilde{H}^{-1}(T)\Sigma(TIl + 2D)^{-1} \in \Re^{lxl} \tag{3.51}$$

and

$$K_2 = \rho T\tilde{H}^{-1}(T)\Sigma(TIl + 2D)^{-1} \in \Re^{lxl} \tag{3.52}$$

In equations (3.51) and (3.52),

$$\tilde{H}(T) = \int_0^T \tilde{C}\, e^{\tilde{A}t}\, \tilde{B}\, dt \tag{3.53}$$

is the step-response matrix of the nominal open-loop plant with state-space triple $(\tilde{A}, \tilde{B}, \tilde{C})$ which is used for design purposes in obtaining the controller for the actual open-loop plant with state-space triple (A,B,C) governed by equations (3.1) and (3.2). Then, by using the explicit expressions for K_1 and K_2 given by equations (3.51) and (3.52) in equation (3.37), the results of *Porter* and *Shenton* (1975a) from the singular perturbation analysis of transfer function matrices can be used to elucidate the required robustness characteristics.

Thus, it follows from equation (3.37) that the transfer function matrix of the closed-loop digital tracking system assumes as $T \rightarrow 0$ the asymptotic form [*Abidin* (1991)]

$$\Gamma(z) = \left[(z-1)(z+\alpha)I_l + \sigma(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1}\right]^{-1} \sigma(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1} . \qquad (3.54)$$

In addition, the poles of the closed-loop transfer function matrix assume the asymptotic values given by the elements of the set $Z_1 \cup Z_2 \cup Z_3 \cup Z_4$ as $T \rightarrow 0$ where

$$Z_1 = \{z \in \mathrm{C}: |zI_l - I_l + T\rho I_l| = 0\} \quad , \qquad (3.55)$$

$$Z_2 = \left\{z \in \mathrm{C}: \left|zI_l - I_l + T\left(\frac{(1+\alpha)}{2\delta}\right)I_l\right| = 0\right\} \quad , \qquad (3.56)$$

$$Z_3 = \{z \in \mathrm{C}: |zI_{n-2l} - I_{n-2l} + T\{\Lambda_1 - \Lambda_2\Delta_2^2\Delta_1 - \Xi_1\Omega^{-1}(\Pi - \mathrm{Y}\Delta_2^{-1}\Delta_1| = 0\}, \qquad (3.57)$$

and

$$Z_4 = \{z \in \mathrm{C}: |(z-1)(z+\alpha)I_l + \sigma(z+1)(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1}| = 0\} . \qquad (3.58)$$

It is important to note that only equations (3.54) and (3.58) are affected by the

disparity between the nominal plant with second Markov parameter $\tilde{C}\tilde{A}\tilde{B}$ and the actual plant with the second Markov parameter CAB. Indeed, equation (3.58) indicates that this disparity will affect the values of the elements of Z, and therefore tend to alter the stability characteristics of the closed-loop system; whilst equation (3.54) indicates that it will also destroy the diagonality of $\Gamma(z)$ and therefore tend to alter the non-interacting response characteristics of the closed- loop system. However, as expected, these equations reduce to the previously presented equations (3.46) and (3.50) in case the actual and nominal plants coincide.

3.3.2 SYNTHESIS

It was shown by *Abidin* (1991) that a sufficient condition for tracking to occur is that the closed-loop poles defined by equations (3.55), (3.56), (3.57) and (3.58) satisfy the stability condition

$$Z_1 \cup Z_2 \cup Z_3 \cup Z_4 \subset D^- \quad , \tag{3.59}$$

where D^- is the open unit disc.

Therefore, in view of equations (3,55), (3.56), (3.57) and (3.58), the stability requirement (3.59) will be satisfied for sufficiently fast sampling frequencies if the controller tuning parameters α, δ, σ and ρ are chosen such that both $Z_1 \subset D^-$

and $Z_2 \subset D^-$ for sufficiently small sampling periods since $Z_3 \subset D^-$ in the case of minimum-phase plants [*Manganas* (1987)]. Indeed, since the set of transmission zeros [*Porter* and *D'Azzo (1977)*]

$$Z_t = \{s \in C : |sI_{n-2l} - \{\Lambda_1 - \Lambda_2\Delta_2^2\Delta_1 - \Xi_1\Omega^{-1}(\Pi - Y\Delta_2^{-1}\Delta_1| = 0\} \subset C^- \qquad (3.60)$$

where C^- is the open left half-plane, it follows [*Abidin* (1991)] in view of equation (3.57) that $Z_3 \subset D^-$ for sufficiently small sampling periods in the case of minimum-phase plants.

Furthermore, in view of equations (3.55) and (3.56), both $Z_1 \subset D^-$ and $Z_2 \subset D^-$ for sufficiently small sampling periods for any choice of positive diagonal tuning parameters $\sigma \in \Re^+$ and $\rho \in \Re^+$ and any choice of the controller parameter $\alpha \in \Re^+$ within the permissible interval (-1 , +1]. [*Manganas* (1987)]

It follows [*Abidin* (1991)] that the crucial requirement for set-point tracking in the case of minimum-phase plants is that $Z_4 \subset D^-$. These considerations thus lead to the following main results of *Porter* and *Abidin* (1990a) in establishing the robustness of fast-sampling digital controllers for completely irregular plants:

Robustness Theorem [*Porter* and *Abidin* (1990a)]

In the case of a completely irregular minimum-phase linear multivariable plant,

there exists a fast-sampling digital PID controller that causes set-point tracking to occur for any plant parameter variation such that all the numbers $\hat{\pi}_i (i = 1,2,3,...,2l)$ *lie within the unit disc, where* $\{\hat{\pi}_1, \hat{\pi}_2, ..., \hat{\pi}_{2l}\}$ *is the spectrum of the matrix*

$$\hat{\Pi} = \begin{bmatrix} 0_l & I_l \\ \alpha I_l - \sigma(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1} & (1-\alpha)I_l - \sigma(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1} \end{bmatrix}. \qquad (3.61)$$

In interpreting this robustness theorem, it is important to note that $\tilde{C}\tilde{A}\tilde{B}$ is the second continuous-time Markov parameter of the nominal plant and CAB is the second continuous-time Markov parameter of the actual plant. In addition, it is evident that

$$(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1} = \lim_{T \to 0} H(T)\tilde{H}^{-1}(T) \quad , \qquad (3.62)$$

where $\tilde{H}(T)$ is the open-loop step-response matrix of the nominal plant given by equations (3.53) and

$$H(T) = \int_0^T C e^{At} B \, dt \qquad (3.63)$$

is the open-loop step-response matrix of the actual governed by equation (3.1) and (3.2). This means that [*Porter* and *Abidin* (1990)] the robustness theorem

can be conveniently applied in order to assess robustness characteristics in the absence of explicit state-space models of plants provided that step-response matrices are available from input/output tests.

Moreover, the following typical interesting corollary of the robustness theorem can be readily derived:

Robustness Corollary [*Porter* and *Abidin* (1990a)]

In the case of a completely irregular linear multivariable plant, there exists a fast-sampling error-actuated digital PID controller that causes set-point tracking to occur for any plant-parameter variation such that all the numbers ω_i *(i=1,2,...,l) are real and lie on the open interval* $(0,(1+\alpha)/\sigma)$ *for any admissible value of* $\alpha \in$ *(-1, +1] where* $\{\omega_1, \omega_2, \ldots, \omega_l\}$ *is the spectrum of the matrix* $(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1}$.

3.4 DESIGN PROCEDURE FOR ROBOTIC CONTROLLERS

In order to demonstrate the effectiveness of the digital PID trajectory-tracking controllers proposed by *Porter* and *Abidin* (1990a), the use of such a controller is considered in detail in this section in the case of a three-link direct-drive robotic manipulator. It will be made clear later in this section that such robotic

manipulators are completely irregular multivariable plants, and that the design of controllers for such devices is therefore amenable to the design method presented in section 3.2.

3.4.1 MANIPULATOR DYNAMICS

Since any robotic manipulator is basically a set of links connected at joints, the configuration of such a device at any instant is uniquely described by the joint-angle vector

$$\theta = [\theta_1, \theta_2, \ldots, \theta_l]^T \quad , \tag{3.64}$$

where l is the number of degrees of freedom and θ_i (i=1,2,...,l) are the joint angles.

The kinetic energy of such a manipulator can then, accordingly, be expressed in the form

$$T(\theta, \dot{\theta}) = \frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta} \quad , \tag{3.65}$$

where $\dot{\theta} = \left[\dot{\theta}_1, \dot{\theta}_2, \ldots, \dot{\theta}_l\right]^T$ is the angular-velocity vector and $M(\theta)$ is the

manipulator inertia matrix which depends on the joint-angle vector $\theta = [\theta_1, \theta_2, \ldots, \theta_l]^T$.

Furthermore, in terms of the potential energy, $V(\theta)$, of the manipulator, the Lagrangian of the system is given by [*Bejczy* (1974), *Mackiewicz* (1973)]

$$L = T\left(\theta, \dot{\theta}\right) - V(\theta) \quad . \tag{3.66}$$

Once the Lagrangian has been obtained, the manipulator dynamic equations for the joints (k=1,2, ,*l*) can be easily derived using the Lagrange-Euler formulation [*Bejczy* (1979)]

$$\tau = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} \quad , \tag{3.67}$$

where $\tau \in \Re^l$ is the input torque vector applied at the joints (k=1,2,...,*l*).

Thus, by substituting equation (3.65) into equation (3.66) and using equation (3.67), the dynamic equations of a manipulator for the joints (k=1,2,...,*l*) can be expressed in the vector-matrix form [*Craig (1986), Bejczy (1974)*]

$$\tau = M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) \quad . \tag{3.68}$$

In equation (3.68), $c(\theta,\dot{\theta})$ is the vector of coriolis and centrifugal torques, $g(\theta)$ is the vector of gravity torques, and $\tau=[\tau_1,\tau_2,...,\tau_l]^T$ is the input torque vector. The non-linear coriolis, centrifugal and gravity torque vectors are often combined as the vector $n(\theta,\dot{\theta})$. Equation (3.68) can then accordingly be expressed in the form

$$\tau = M(\theta)\ddot{\theta}+n(\theta,\dot{\theta}) \quad , \tag{3.69}$$

where $n(\theta,\dot{\theta})=c(\theta,\dot{\theta})+g(\theta)$ is the entire non-linear torque vector.

3.4.2 CONTROL SCHEME

Due to the non-linear dynamical characteristics of robotic manipulators, there is no 'standard method ' by which to represent their equations in state-space forms. However, the method adopted in this thesis consists in treating the non-linear terms in equation (3.69) as disturbances [*Li* (1986*)*, *Petropoulakis* (1986*)*, *Choi* and *Bien* (1988*)* and *Abidin* (1991)]. In fact, in the singular perturbation design technique, the resulting controller matrices are the same whichever method is chosen to represent the manipulator dynamics,.

The control scheme commonly used in robot-control systems is depicted in Figure 3.1. In this scheme, the task description, expressed in terms of the sequence of desired end-effector coordinates in cartesian space, is transformed using inverse kinematics to a series of angular positions in the joint space. The angular positions thus obtained can then be used as reference inputs to the closed-loop system. The control action, therefore, involves the joint-torque vector and is generated from the error vector in joint space. This method is popular since the inverse kinematic transformation can be performed off-line.

The discrete-time tracking system under consideration, as shown in Figure 3.2, consists of a robotic manipulator, together with a digital error-actuated trajectory-tracking PID controller. This robotic manipulator is governed on the continuous-time set $T = [0, +\infty)$ by state and output equations of the respective forms [*Bradshaw* and *Porter* (1980a)]

$$\dot{x}(t) = Ax(t) + Bu(t) + Ed(t) \tag{3.70}$$

and

$$y(t) = Cx(t) \quad . \tag{3.71}$$

In equation (3.70) and (3.71), $x(t) \in \Re^n$ is the plant state vector, $u(t) \in \Re^l$ is the plant input vector, $d(t) \in \Re^l$ is the disturbance input vector, $y(t) \in \Re^l$ is the

plant output vector, $A \in \Re^{nxn}$ is the plant matrix, $B \in \Re^{nxl}$ is the input matrix, $E \in \Re^{nxl}$ is the disturbance input matrix, and $C \in \Re^{lxn}$ is the output matrix.

In the case of robotic manipulators, these equations have the explicit forms [*Petropoulakis* (1986)]

$$\dot{x}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \begin{bmatrix} 0_l, & I_l \\ 0_l, & 0_l \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 0_l \\ B_2 \end{bmatrix} u(t) + \begin{bmatrix} 0_l \\ E_2 \end{bmatrix} d(t) \tag{3.72}$$

and

$$y(t) = \begin{bmatrix} I_l, & 0_l \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \quad , \tag{3.73}$$

where $x_1 = [\theta_1, \theta_2, \ldots, \theta_l]^T \in \Re^l$ is the joint-angle vector, $x_2 = \left[\dot{\theta}_1, \dot{\theta}_2, \ldots, \dot{\theta}_l\right]^T \in \Re^l$ is the joint angular-velocity vector, $u = [\tau_1, \tau_2, \ldots, \tau_l]^T \in \Re^l$ is the input torque vector, $0_l \in \Re^{lxl}$ is the zero matrix, $I_l \in \Re^{lxl}$ is the identity matrix, $B_2 = M^{-1}(x_1) \in \Re^{lxl}$, $E_2 = M^{-1}(x_1)\, n(x_1, x_2) \in \Re^l$, $M(x_1) \in \Re^{lxl}$ is the manipulator inertia matrix, and $n(x_1, x_2) \in \Re^l$ is the vector of centrifugal, coriolis, and gravitational torques. It is important to note that the dimension of the plant state vector is $n=2l$.

It is also clear from equations (3.72) and (3.73) that, in the terminology of equations (3.70) and (3.71),

$$\text{rank } CB = 0 \quad , \tag{3.74}$$

and

$$\text{rank } CAB = l \quad . \tag{3.75}$$

In view of equations (3.74) and (3.75), such robotic manipulators are completely irregular multivariable plants and are, therefore, amenable to control by the trajectory-tracking PID controllers described in section 3.2.

Furthermore, by using the definition of the matrix exponential in the form

$$e^{(AT)} = I_n + AT + \frac{T}{2!}A^2 + \ldots + \frac{T^{i-1}}{i!}A^i + \ldots \tag{3.76}$$

the step-response matrix of the nominal open-loop plant can be calculated substituting equation (3.76) into equation (3.53). This yields [*Petropoulakis* (1986)]

$$H(T) = TCB + \frac{T^2}{2!}CAB + \frac{T^3}{3!}CA^2B + \ldots = \frac{T^2}{2!}CAB = \frac{T^2}{2}M^{-1}(x_{1d}), \tag{3.77}$$

where $M(x_{1d})$ is the inertia matrix chosen at a design position $x_{1d} = [\theta_{1d}, \theta_{2d}, \ldots, \theta_{ld}]^T$ and the rest of the variables are as defined in section 3.2.

Similarly, by using equations (3.51) and (3.52), the proportional and integral controller matrices can be expressed in the forms [*Abidin* (1991)]

$$K_1 = \frac{2}{T} M(x_{1d}) \Sigma (TI_m + 2D)^{-1} \quad , \tag{3.78}$$

and

$$K_2 = \rho \frac{2}{T} M(x_{1d}) \Sigma (TI_l + 2D)^{-1} \quad . \tag{3.79}$$

It thus follows [*Abidin* (1991)] that the corresponding closed-loop transfer-function of a manipulator in the neighbourhood of the design position, x_{1d}, under fast-sampling digital PID controller assumes, as $T \rightarrow 0$, the asymptotic form

$$\Gamma = \left[(z-1)(z-\alpha) I_l + \sigma(z+1) M(x_1)^{-1} M(x_{1d})\right]^{-1} \sigma(z+1) M(x_1)^{-1} M(x_{1d}). \tag{3.80}$$

In addition, the poles of the closed-loop transfer function matrix assume the asymptotic values given by the elements of set $Z_1 \cup Z_2 \cup Z_4$ as $T \rightarrow 0$, where

$$Z_1 = \{z \in \mathrm{C}: |zI_l - I_l + T\rho I_l| = 0\} \quad , \tag{3.81}$$

$$Z_2 = \left\{ z \in \mathrm{C}: \left| zI_l - I_l + T\left(\frac{(1+\alpha)}{2\delta}\right) I_l \right| = 0 \right\} , \tag{3.82}$$

and

$$Z_4 = \{z \in \mathrm{C}: |(z-1)(z+\alpha)I_l + \sigma(z+1)M^{-1}(x_1)M(x_{1d})| = 0\} . \tag{3.83}$$

Furthermore, using a result from *Rosenbrock* (1970), it is easily shown that this plant has no transmission zeros. Indeed, the closed-loop poles corresponding to Z_3 given by equation (3.57) are non-existent for robotic manipulators since $n - 2l = 0$. Thus, the set Z_3 reduces to

$$Z_3 = \varnothing \quad . \tag{3.84}$$

Now, in order to ensure that the closed-loop system is stable it is required that all the poles of the transfer function matrix lie within the open unit disc D^-. Moreover, since it is clear from equations (3.81) and (3.82) that $Z_1 \subset \mathrm{D}^-$ and $Z_2 \subset \mathrm{D}^-$ for sufficiently small sampling periods for any choice of positive controller parameters $\delta \in \Re^+$ and $\rho \in \Re^+$ and any choice of the controller parameter $\alpha \in \Re$ within the permissible interval (-1, +1], the crucial requirement for stability of the robotic manipulator control is that $Z_4 \subset \mathrm{D}^-$. In view of the Robustness

Corollary (see page 19), this stability condition is equivalent to the requirement that all the eigenvalues of $M(x_1)^{-1} M(x_{1d})$ are real and lie on the the open interval $(0,(1+\alpha)/\sigma)$.

3.5 ILLUSTRATIVE EXAMPLE

The design of fast-sampling digital PID controllers for trajectory-tracking systems incorporating robotic manipulators can be conveniently illustrated by considering the three-degrees-of-freedom manipulator previously considered by [*Petropoulakis* (1986)] (See Figure 3.3 and Appendix B) . In this case, the manipulator is governed on T = [0, +∞) by state and output equations of the respective forms [*Petropoulakis* (1986)]

$$M(\theta)\ddot{\theta} + n(\theta,\dot{\theta}) = \tau \tag{3.85}$$

and

$$y = f(\theta) \quad , \tag{3.86}$$

where $\theta = [\theta_1, \theta_2, \theta_3]^T \in \Re^3$ is the joint-angle vector, $\tau = [\tau_1, \tau_2, \tau_3]^T \in \Re^3$ is the joint-torque vector, $y = [x_e, y_e, z_e]^T \in \Re^3$ is the position vector of the end-

effector in cartesian space. In addition,

$$M(\theta)=\begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & m_{23} \\ 0 & m_{32} & m_{33} \end{bmatrix} \in \Re^{3x3} \tag{3.87}$$

with

$$m_{11} = (m_1 a^2 + m_2 r_1^2)\sin^2\theta_2 + I_{a1} + m_2 b^2 \sin^2\theta_3 - 2m_2 b r_1 \sin\theta_2 \sin\theta_3 \quad , \tag{3.88}$$

$$m_{22} = m_1 a^2 + m_2 r_1^2 + I_{a2} \quad , \tag{3.89}$$

$$m_{23} = m_{32} = -m_2 b r_1 \cos(\theta_2 - \theta_3) \quad , \tag{3.90}$$

$$m_{33} = m_2 b^2 + I_{a3} \quad , \tag{3.91}$$

is the inertia matrix,

$$n(\theta,\dot{\theta})=\begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix} \in \Re^3 \tag{3.92}$$

with

$$\begin{aligned} n_1 &= 2(m_1 a^2 + m_2 r_1^2)\sin\theta_2 \cos\theta_2\, \dot{\theta}_1 \dot{\theta}_2 - 2m_2 b r_1 \cos\theta_2 \sin\theta_3\, \dot{\theta}_1 \dot{\theta}_2 \\ &+ 2m_2 b(b\sin\theta_3 - r_1 \sin\theta_2)\cos\theta_1\, \dot{\theta}_3 \end{aligned} \tag{3.93}$$

$$n_2 = -m_2 b r_1 \sin(\theta_2 - \theta_3)\dot{\theta}_3^2 - (m_1 a^2 + m_2 r_1^2)\sin\theta_2 \cos\theta_2 \dot{\theta}_1^2 + m_2 b r_1 \cos\theta_2 \sin\theta_3 \dot{\theta}_1^2 - (m_1 a + m_2 r_1) g \sin\theta_2 \quad (3.94)$$

$$n_3 = m_2 r_1 b \sin(\theta_2 - \theta_3)\dot{\theta}_2^2 + m_2 b(r_1 \sin\theta_2 - b\sin\theta_3)\cos\theta_3 \dot{\theta}_1^2 + m_2 g b \sin\theta_3 \quad (3.95)$$

is the vector of centrifugal, coriolis, and gravitational torques, and

$$f(\theta) = \begin{bmatrix} (r_1 \sin\theta_2 - r_2 \sin\theta_3)\cos\theta_1 \\ (r_1 \sin\theta_2 - r_2 \sin\theta_3)\sin\theta_1 \\ r_1 \cos\theta_2 - r_2 \sin\theta_3 \end{bmatrix} \quad (3.96)$$

is the vector of direct kinematic relationships. The numerical values of the inertial and kinematic parameters of the typical three-degree-of-freedom robotic manipulator under consideration are given by *Petropoulakis* (1986).

In order to illustrate the robust performance of fast-sampling digital PID controllers, it is instructive to design such a controller corresponding to the arbitrarily selected initial end-effector position (0, 0.45, 0) m corresponding to the joint space coordinates ($\pi/2$, 0.27π, -0.27π). This controller is then used while the end-effector of the manipulator is caused to track straight-line trajectories between the following points:

I. (-0.5,0,-0.2) m ,

II. (-0.4,0.3,0) m ,

III. (0.3,0.3,0.3) m ,

IV. (0.3,0.3,0.3) m ,

V. (-0.45,0.35,0) m ,

VI. (-0.45,0.35,0) m .

In this task, a fifth-order trajectory generator is used to interpolate between two successive points so that the generated torque vector produces smooth motion (ie, no jerk occurs). These transitions are effected with appropriate acceleration, cruise, and deceleration profiles in the following times [*Petropoulakis* (1986)]:

I → II . 1.5 s ,

II → III . 2.0 s ,

III → IV . 0.5 s ,

IV → V . 2.5 s ,

V → VI . 0.5 s .

In addition, after the initial transition I → II, the manipulator grasps an additional payload of 5 Kg. In view of the intrinsic non-linearity of the manipulator and sudden variation in payload, this sequence of tracking constitutes a formidable test of robustness and performance for the non-adaptive controllers.

Nevertheless, when the high-accuracy trajectory-tracking digital PID controller

parameters are tuned such that σ = 0.7, α = 0.1, δ = 0.01, ρ = 1 [*Porter* and *Abidin* (1990b)], the time variations of the real values of the spectrum of the matrix $M(x_1)^{-1}M(x_{1d})$ during this sequence of tracking tasks are shown in Fig 3.4. Clearly, the fluctuations of the spectrum of the matrix $M(x_1)^{-1}M(x_{1d})$ always lie on the interval $(0,(1+\alpha)/\sigma)=(0,1.571)$ permitted by the robustness corollary [*Porter* and *Abidin* (1990b)] and thus indicate that this controller remains stable during this sequence of tracking tasks for an appropriately small sampling period. This prediction is confirmed by the plots of the trajectory tracking errors e1 and e2 and the joint torques shown in Figure 3.5 (a) and Figure 3.5 (b) for a PID controller with a sampling time of 0.01s. The tracking error is defined as the distance between the actual and desired position at the same instant of time: its projections along and perpendicular to the desired trajectory are e_1 and e_2, respectively. These graphs indicate that the controller produces accurate tracking performance, with a maximum trajectory-tracking error of about 1mm except for a brief transient period when the sudden variation of payload occurs at 1.5s causes a transient error of 6mm. The torques generated by the controller during the sequence of tracking tasks are shown in Figure 3.5 (b), and exhibit no practically undesirable characteristics.

3.6 CONCLUSION

The design of digital trajectory-tracking controllers for robotic manipulators

using non-evolutionary singular perturbation techniques has been presented in this chapter. It has been shown that the design method of *Porter* and *Abidin* (1990a) for completely irregular linear multivariable plants is directly applicable to robotic manipulators. In particular, the robustness theorem is very useful in the case of robotic manipulators due to the time-varying terms characteristics of such systems. Finally, it is evident from the illustrative example presented in this chapter that, despite the occurrence of non-linear time-varying terms in robotic manipulators, the trajectory-tracking performance of such systems under digital PID control is remarkably good.

However, it is important to acknowledge the lack of an adequate method for computing the optimal quadruple $\{\alpha, \delta, \sigma, \rho\}$ of design parameters for fast-sampling digital PID controllers capable of delivering the best possible trajectory-tracking behaviour. It is therefore necessary to develop a generally applicable optimisation technique for this purpose. This is a central task of this thesis, using various genetic and evolutionary algorithms.

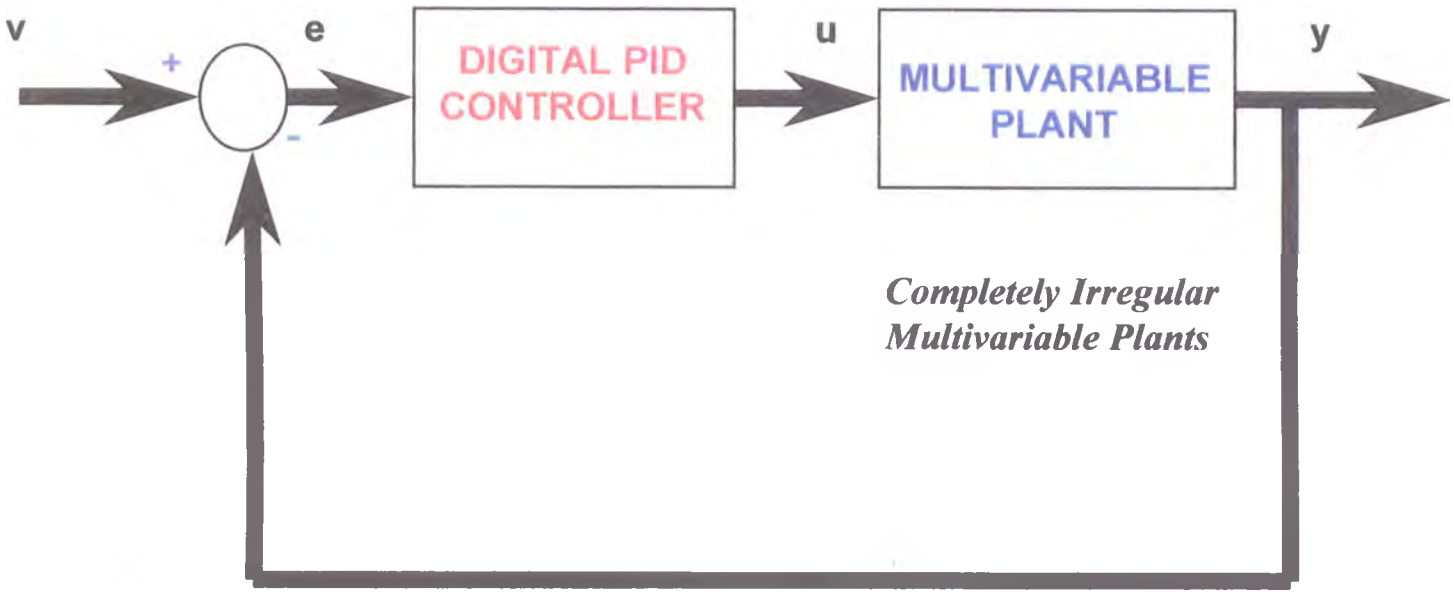

Figure 3.1: Block diagram of fast-sampling controller for completely irregular linear multivariable plants

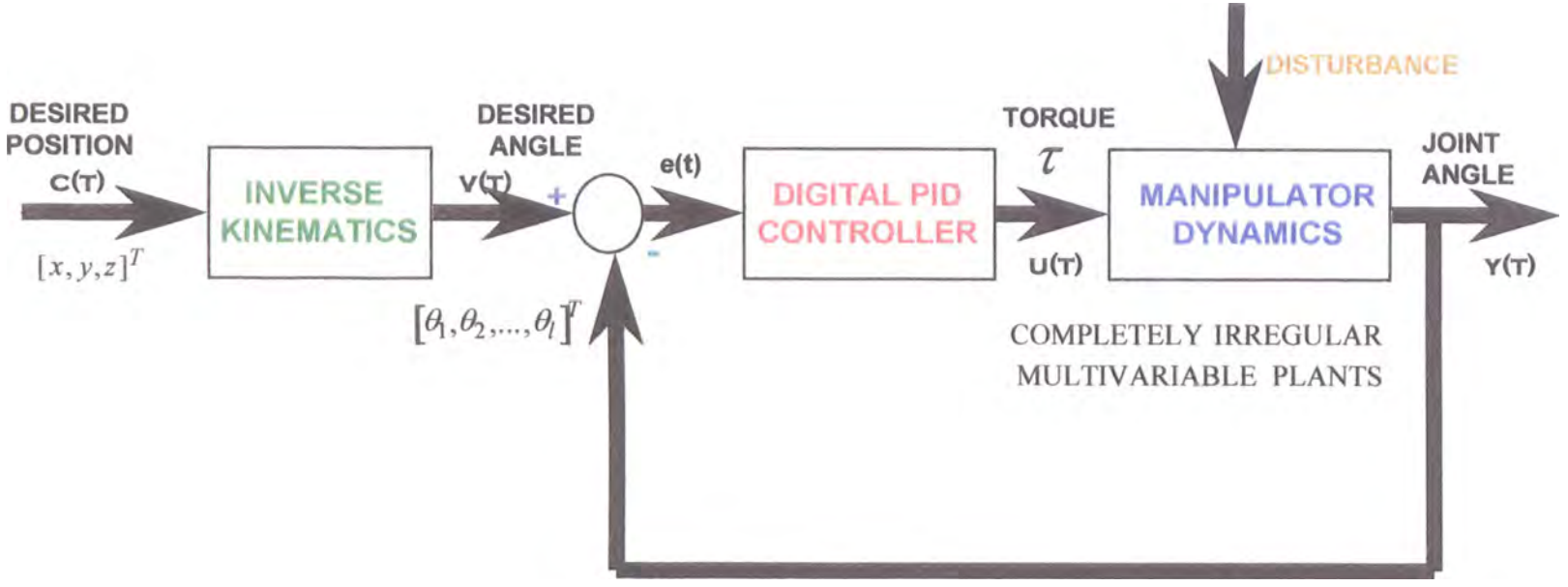

Figure 3.2: Block diagram of trajectory-tracking system for completely irregular linear multivariable plants

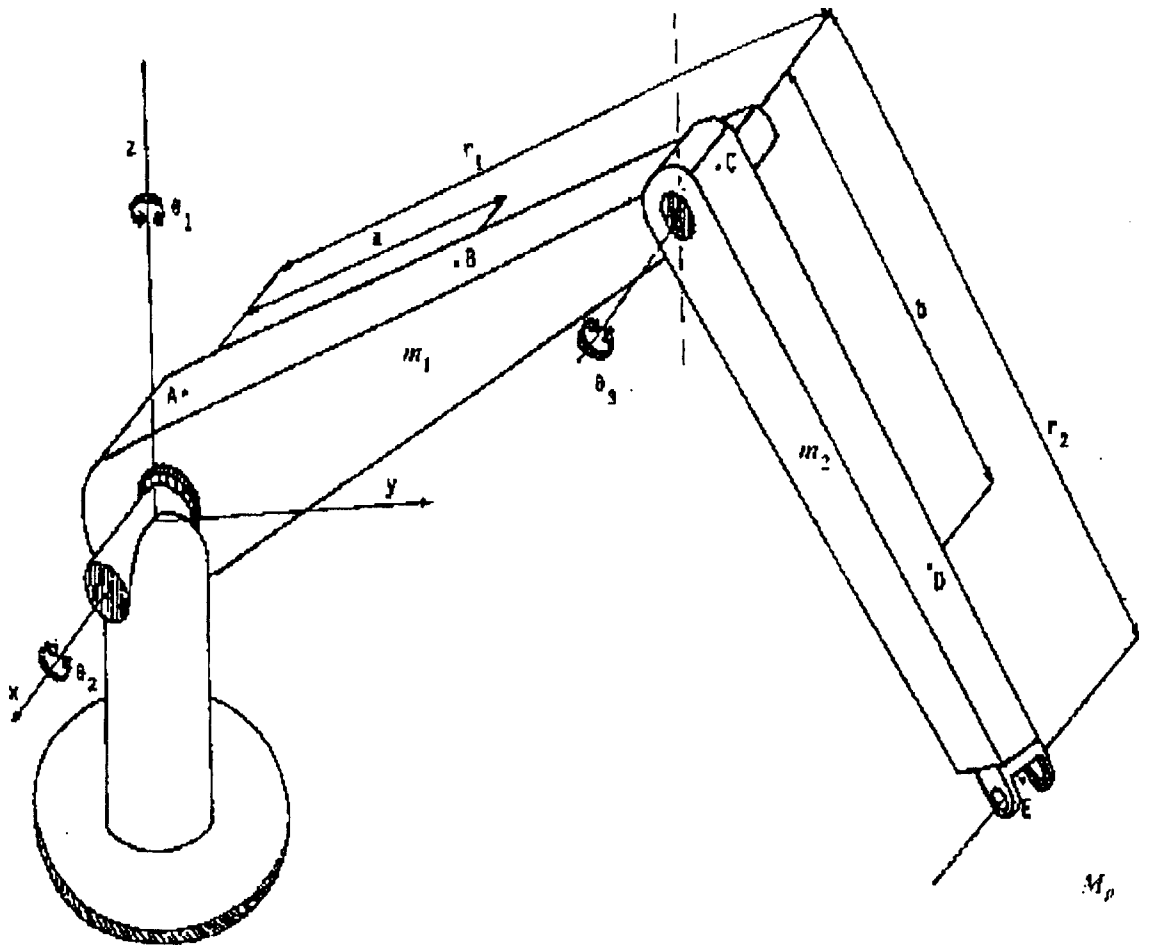

Figure 3.3: Three-link direct-drive robotic manipulator schematic representation and dimensions.

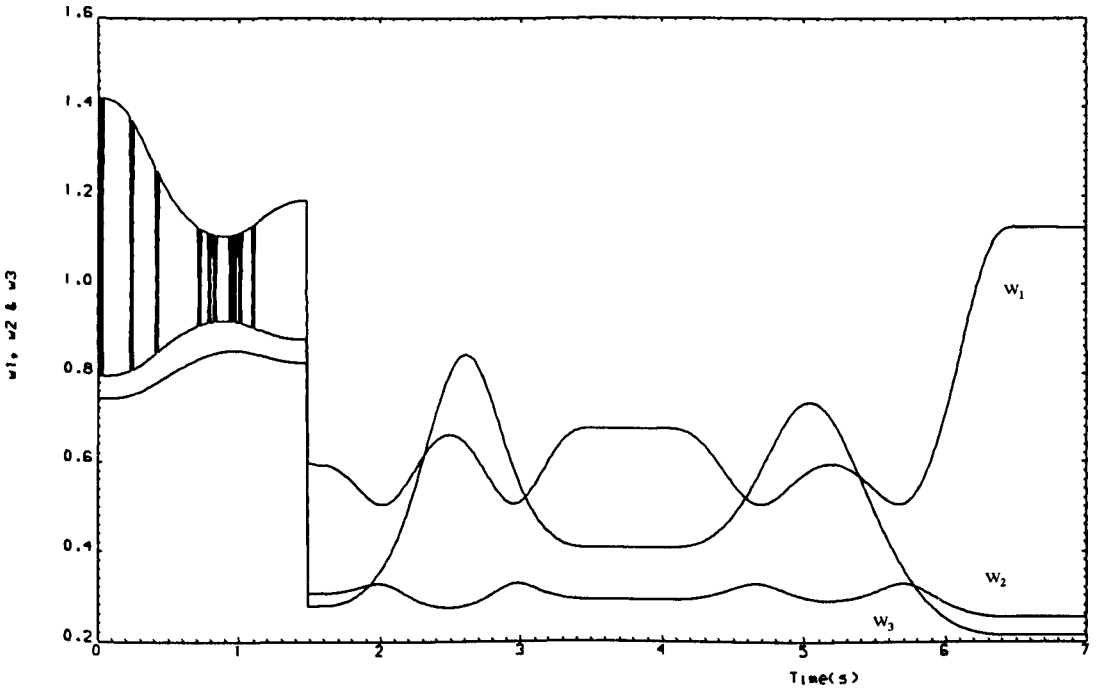

Figure 3.4: Time-domain behaviour of the spectrum of the perturbation matrix $M(x_1)^{-1}\, M(x_{1d})$ using singular perturbation designed digital PID controller.

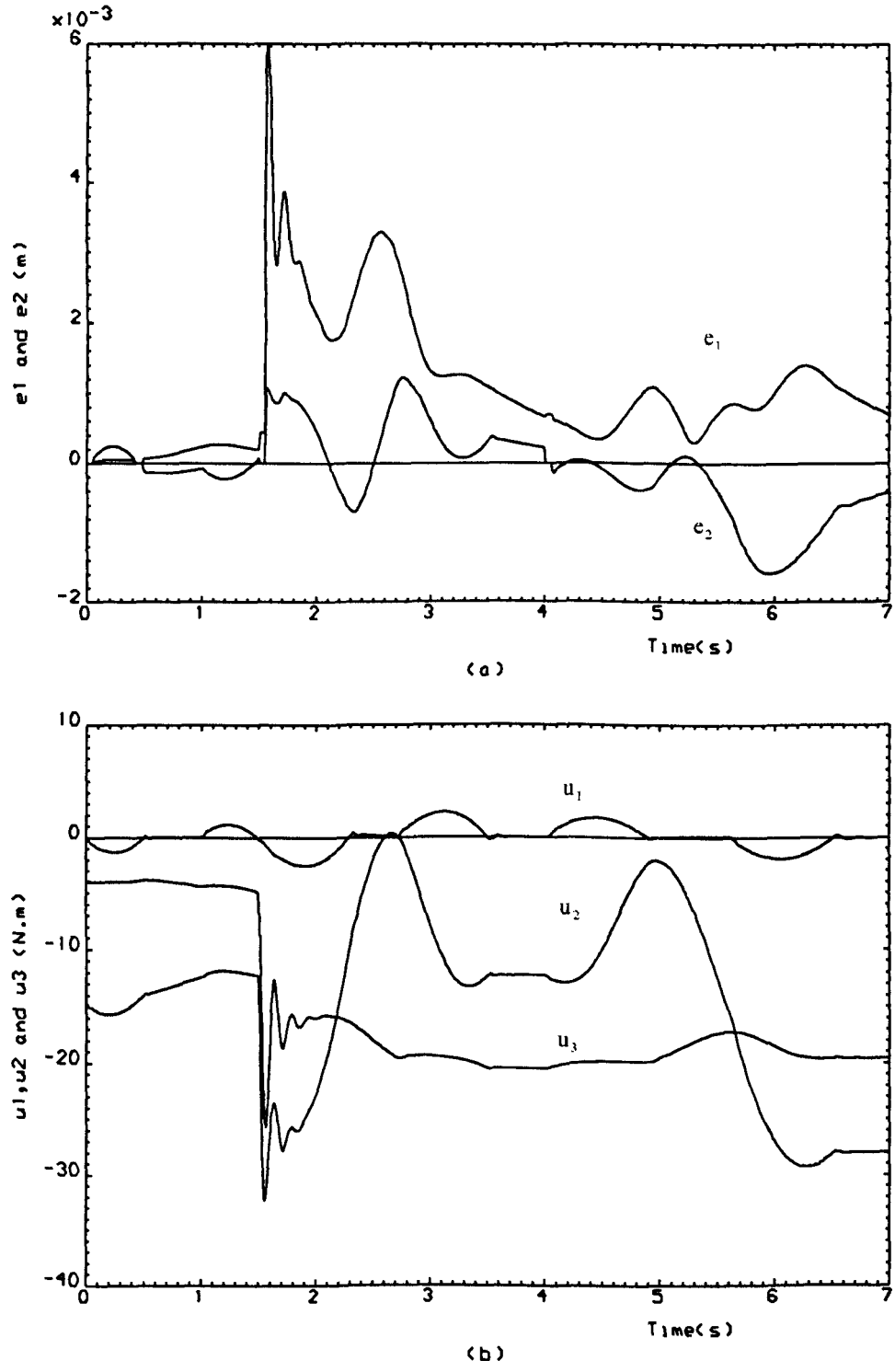

Figure 3.5: Time-domain behaviour of errors and torques for singular perturbation design of digital PID controller.

PART III

CONSTRAINED DESIGN OF DIGITAL TRAJECTORY-TRACKING CONTROLLERS USING EVOLUTIONARY ALGORITHMS

Chapter 4

DESIGN OF DIGITAL CONTROLLERS FOR ROBOTIC MANIPULATORS USING GENETIC ALGORITHMS

4.1 INTRODUCTION

By using singular perturbation techniques, *Porter*, *Manganas*, and *Manganas* (1988) developed a methodology in which only the open-loop step-response matrices of plants are needed to design the appropriate fast-sampling error-actuated digital PID controllers. In the face of plant-parameter variations, *Porter* and *Abidin* (1990) subsequently established the robustness characteristics of such non-adaptive and non-evolutionary controllers in the case of completely irregular multivariable plants as summarised in Chapter 3. It was thus shown by *Porter* and *Abidin* (1990a) that the plant-parameter variations tolerable by fast-sampling error actuated digital PID controller can be expressed very simply in terms of the step-response matrices of the nominal and actual plants.

In practice, these theoretical results are valid only asymptotically as $f \rightarrow \infty$ (i.e., as the sampling frequency of the digital PID controller considered in Chapter 3 becomes infinite). It has so far proved impossible to obtain theoretical non-asymptotic robustness results for digital PID controllers with finite sampling frequencies. It is therefore desirable to develop an effective tuning procedure for such non-asymptotic conditions. In this chapter, the use of genetic algorithms

(Appendix A.1) is accordingly proposed for the design of digital trajectory-tracking controllers for robotic manipulators that was treated by singular perturbation methods in Chapter 3. This genetic design process is effected by incorporating the controller design equation (3.14) within the genetic algorithms so as to optimise the various tuning parameters. In this way, the best trajectory-tracking behaviour is achieved in accordance with a chosen evaluation criterion for the specified typical sequence of trajectory-tracking tasks presented in Chapter 3 (which includes a sudden change of payload).

The optimal quadruple $\{\alpha , \delta , \sigma , \rho\}$ of design parameters for fast-sampling digital PID controllers thus obtained allows a direct comparison to be made in the non-asymptotic case between the performance of those controllers with genetically tuned parameters and those with parameters tuned by *Porter* and *Abidin* (1990). In addition, it is shown that the genetic design procedure does not violate the robustness characteristics which need to be considered in the case of robotic manipulators due to the time-varying characteristics of such systems as well as to the discrepancy introduced by the use of a nominal plant for design purposes. Indeed, the genetic design procedure makes an implicit use of the robustness theorem and corollary presented in Chapter 3 by ensuring that the genetically designed controllers remain stable during the specified sequence of tracking tasks.

This genetic design approach is a natural extension to robotic control problems of the previously obtained non-robotic results of *Porter* and *Jones* (1992), *Porter*,

Mohamed and *Jones* (1993), and *Porter* and *Hicks* (1994).

4.2 GENETIC DESIGN PROCEDURE

The closed-loop digital trajectory-tracking system under investigation incorporates the following two principal components as shown in Figure 4.1

(i) an inverse kinetic transformation block which generates a desired set-point command vector in the joint space, v(t), in response to a set-point command vector in the task space, c(t);

(ii) a multivariable PID controller that generates an appropriate control input vector, u(t), in response to the error between the set-point command vector, v(t), and the plant output vector, y(t).

It is intended that such fast-sampling digital PID controller produces high-accuracy trajectory-tracking behaviour by causing the output vector , y(t), to track the set-point command, v(t).

Indeed, the digital trajectory-tracking systems under consideration, as shown in Figure 4.1, consist of linear multivariable plants together with digital PID controllers. Such plants are assumed to exhibit minimum-phase characteristics

and are therefore amenable to fast-sampling error-actuated control [*Bradshaw* and *Porter* (1980)]. These linear multivariable plants are governed on the continuous-time set $T = [0,\infty)$ by state and output equations of the respective forms [*Manganas* (1985)]

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

and

$$y(t) = Cx(t), \tag{4.2}$$

where

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} \in \Re^n . \tag{4.3}$$

In these equations, $x(t) \in \Re^n$ is the plant state vector, $u(t) \in \Re^l$ is the plant input vector, $y(t) \in \Re^l$ is the plant output vector, $A \in \Re^{nxn}$ is the plant matrix, $B \in \Re^{nxl}$ is the input matrix, $C \in \Re^{lxn}$ is the output matrix, $x_1(t) \in \Re^{n-l}$,and $x_2(t) \in \Re^l$. Furthermore, in view of the assumed complete irregularity of the considered plants, the first Markov parameter, CB, of the continuous-time open-loop plant is null [*Abidin* (1991)], ie.,

$$CB = 0 \quad , \tag{4.4}$$

whilst the second Markov parameter, CAB, is of full rank, ie.,

$$rank\ CAB = l \quad . \tag{4.5}$$

The digital PID controllers under consideration are governed on the discrete-time set $T_T = \{0,T, 2T,...\}$ by control-law equations of the form [*Porter et al* (1985)]

$$u(kT) = K_1 r(kT) + K_2 z(kT) \quad . \tag{4.6}$$

In equation (4.6), $K_1 \in \Re^{l \times l}$ and $K_2 \in \Re^{l \times l}$ are the proportional and integral controller matrices, whilst the vectors $r(kT) \in \Re^l$ and $z(kT) \in \Re^l$ are generated in accordance with the difference equations [*Porter* (1987)]

$$s\{(kT + 1)T\} = -\alpha s(kT) + e(kT), \tag{4.7}$$

$$r(kT) = -\frac{2}{T}(1 + \alpha)Ds(kT) + \left(I_l + \frac{2}{T}D\right)e(kT), \tag{4.8}$$

and

$$z\{(k + 1)T\} = z(kT) + Tr(kT). \tag{4.9}$$

Moreover, in equations (4.7), (4.8) and (4.9), $\alpha \in (-1, +1]$, $s(kT) \in \Re^l$, $e(kT) =$

$v(kT) - y(kT) \in \Re^l$ is the error vector, $v(kT)$ is the set-point command vector, and the derivative matrix $D \in \Re^{lxl}$ is such that

$$rank\, D = l\,. \tag{4.10}$$

The design methodology proposed by *Porter* and *Abidin* (1990) and presented in Chapter 3 indicates that set-point tracking behaviour can be achieved asymptotically if the following design equations for the fast-sampling digital PID controllers are used:

$$K_1 = T\tilde{H}^{-1}(T)\Sigma(TI_l + 2D)^{-1} \in \Re^{lxl} \tag{4.11}$$

and

$$K_2 = \rho T\tilde{H}^{-1}(T)\Sigma(TI_l + 2D)^{-1} \in \Re^{lxl} \tag{4.12}$$

In these equations,

$$\Sigma = \sigma I_l\,(\sigma \in \Re^+) \tag{4.13}$$

is the positive diagonal tuning matrix,

$$D = \delta I_l\,(\delta \in \Re^+) \tag{4.14}$$

is the positive diagonal derivative matrix, and ρ is the positive tuning parameter integral-to-proportional ratio.

In addition, in equations (4.11) and (4.12),

$$\tilde{H}(T) = \int_0^T \tilde{C}\, e^{\tilde{A}\, t}\, \tilde{B}\, dt \qquad (4.15)$$

is the step-response matrix of the nominal open-loop plant with state-space triple $(\tilde{A},\tilde{B},\tilde{C})$ which is used for design purposes in obtaining the controller for the actual open-loop plant with state-space triple (A,B,C) governed by equations (3.1) and (3.2).

However, it is important to remember that despite the elegance of the established robustness theorems of *Porter* and *Abidin* (1990) presented in Chapter 3, these results are restricted to the asymptotic case of fast-sampling error-actuated digital PID controllers for which $f \rightarrow \infty$. Furthermore, it transpires that these asymptotic results involve only the pair (α,σ) of design parameters rather than the complete quadruple $\{\alpha,\sigma,\rho,\delta\}$ of design parameters involved in the controller design equations (4.11) and (4.12).

In fact, under asymptotic conditions, consideration is given only to the reduced controller parameter set (α,σ), which affect most importantly the 'fast' modes of the system. The remaining controller parameters, which essentially determine the asymptotically uncontrollable and unobservable 'slow' modes of the closed-

loop system, are often assigned fixed values in the asymptotic design process. Indeed, the use of fast-sampling error-actuated digital PID trajectory-tracking controllers guarantees the removal of 'slow' modes from the plant outputs, which in turn ensures that the discrete-time tracking systems exhibit set-point tracking characteristics that are both fast and non-interacting.

However, under non-asymptotic conditions for finite sampling frequencies, 'slow' modes tend to be present in the outputs. It is therefore necessary to investigate the effects of both 'fast' and 'slow' modes on trajectory-tracking behaviour by using the complete quadruple $\{\alpha,\sigma,\rho,\delta\}$ for such finite sampling frequencies.

In view of equations (3.46), there is a clear indication that the existence of a disparity between the nominal plant $(\tilde{A},\tilde{B},\tilde{C})$ and the actual plant (A,B,C) will affect the values of the elements of Z_4 (see equation 3.83) (the 'fast' modes) and will therefore tend to alter the stability characteristics of the closed-loop system. Indeed, taking into account the non-linear and time-varying nature of robotic manipulators, it is essential to ensure that the relevant controllers are robust. Such a process of robustification (ie, of ensuring that excellent trajectory-tracking performance is maintained over a set of operating conditions) is obviously crucial to the successful operation of such tracking systems.

It is evident that the absence of a theoretical solution to the non-asymptotic robustness problem highlights the need to consider the following general

robustness problem for the non-asymptotic case [*Porter* and *Allaoui* (1995a)]:

In the case of finite sampling frequencies, determine the set of actual plants with state-space triple (A,B,C) *tolerable by digital PID controllers designed for the nominal plant with state-space triple* $(\tilde{A},\tilde{B},\tilde{C})$ *and characterised by the quadruple* $\{\alpha,\sigma,\rho,\delta\}$ *of controller design parameters.*

This genetic robustification approach consists in effectively using genetic algorithms to solve the following version of the robustness problem [*Porter* and *Allaoui* (1995a)]:

In the case of finite sampling frequencies, determine the quadruple $\{\alpha,\sigma,\rho,\delta\}$ *of the controller design parameters such that optimal trajectory-tracking behaviour is obtained when a given robotic manipulator is controlled so as to track a given trajectory.*

It is clear that the solution to this non-asymptotic robustness problem will provide an optimal quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller design parameters which is dependent upon the given manipulator, the given task, and the measure of trajectory-tracking accuracy used in the optimisation procedure.

There are clearly many different measures of optimality. However, it is convenient initially to regard a minimum integral over the task time, τ , of the Euclidean norm of trajectory-tracking error vector in Cartesian space as the

ultimate design requirement. Thus, genetic algorithms can be readily used to select the appropriate set of controller parameters such that

$$\Gamma = \int_0^{\tau} \| e(t) \| \, dt \tag{4.18}$$

is minimised, where τ is the duration of the tracking task, $e(t) \in \Re^l$ is the trajectory-tracking error vector in Cartesian space, and $\|.\|$ denotes the Euclidean norm.

However, in order to use genetic algorithms to solve this non-asymptotic robustness problem , it is necessary to encode the quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller design parameters in accordance with a system of concatenated, multi-parameter, mapped fixed point coding as proposed by *Goldberg* (1989). Thus, each quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller parameters is represented by a string of binary digits to form an individual member of a particular population. Then, following any choice of randomly generated initial population of such strings, successive generations of strings can be rapidly obtained using the basic genetic operations of selection, crossover, and mutation (Appendix A.5). In addition to the basic genetic operation mentioned above, a certain degree of elitism which can affect both the fittest and worst individuals of a generation is introduced. Indeed, the first step of this elitist scheme consists of insuring the survival of the fittest individual or chromosome amongst the entire set of two successive generations of parents and children. Whereas, the second step safeguards the retention of the best or fittest of the worst individuals of two

successive generations as the worst individual of the current generation. As a result, the successive generations of error-actuated digital PID controllers produced by the genetic algorithms tend to exhibit progressively improving trajectory-tracking behaviour with respect to the selected performance measure for any given robotic manipulator.

4.3 ILLUSTRATIVE EXAMPLE

This general procedure for the genetic design of fast-sampling digital PID controllers for trajectory-tracking systems, which are required to exhibit robust performance in the face of the time-varying plant parameters inherent in robotic manipulators, can be conveniently illustrated by reference to the three-degree-of-freedom robotic manipulator previously investigated by *Petropoulakis* (1986). This manipulator was also used by *Porter* and *Abidin* (1990) to illustrate the asymptotic robustness results presented in Chapter 3 for fast-sampling error-actuated PID controllers for completely irregular linear multivariable plants. This robotic manipulator is governed on $T = [0, +\infty]$ by state and output equations of the respective forms [*Petropoulakis* (1986)]

$$M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = u \quad , \tag{4.19a}$$

and

$$y = f(\theta) \quad . \tag{4.19b}$$

The numerical values of the inertial and kinematics parameters of the typical three-degree-of-freedom robotic manipulator under consideration are those given by *Petropoulakis* (1986). The robotic manipulator is, for design purposes, considered to be in the neighbourhood of the arbitrarily selected operating point corresponding to the end-effector position (0,0.45,0)m which corresponds to the joint space coordinates ($\pi/2$, 0.27π, -0.27π) proposed by *Porter* and *Abidin* (1990).

The genetically designed controller is used while the end-effector of the robotic manipulator is caused to track the straight-line trajectories defined in Chapter 3. In the same manner as in the illustrative example of Chapter 3, the robotic manipulator grasps an additional payload of 5 Kg after the initial transition **I** $\rightarrow$ **II** (see section 3.5) in order to test the robustness of the controller.

The structure of the computational implementation of genetic algorithms in the case of the trajectory-tracking system is shown in Figure 4.2. In this way, the genetic design of the controller for this selected trajectory- tracking task can be readily effected so as to minimise the cost function

$$\Gamma = \int_0^{\tau} \|e(t)\| \, dt \qquad (4.20)$$

by determining the value of the associated optimal quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller parameters.

In this case, the controller parameter set $\{\alpha,\sigma,\rho,\delta\}$ is encoded in accordance with a system of concatenated, multi-parameter, fixed-point coding. Furthermore, in the genetic algorithm, a population size N=30, a crossover probability p_c=0.6, and a set of mutation probabilities {p_m= 0.008, p_m = 0.3, p_m = 0.98} are used over 50 generations.

The choice of different values for the mutation probability, p_m, is motivated by the very nature and function of mutation. Thus, when using genetic algorithms, the population slowly converges from the initial population, where the individuals are quite dissimilar, towards a state in which the individuals are quite similar. In this last state, further improvement may only be possible by using a favourable mutation probability, p_m. In other words, in genetic algorithms mutation is usually treated as a background operator ensuring that the population consists of a diverse pool of individuals that can evolve mainly by crossover. In the past, common settings of the fixed mutation probability, p_m, were obtained in experimental investigations conducted by *De Jong* (1975) who proposed a value p_m = 0.001, by *Grefenstette* (1986) who proposed in turn a value of p_m = 0.01, and by *Schaffer et al* (1989) who suggested values within an interval $p_m \in [0.005\ 0.01]$.

These values for the mutation rate can be taken as guides; but other experts in the field such as *Bäck* (1993) and *Mühlenbein* (1992) proposed different values. Thus, it was proposed that $p_m = 1/l$ (where l is the length of the binary

string representing the encoded controller parameters) is always a good starting point because it will not perform worse than any smaller setting of the mutation probability. However, in a later study of heuristics for setting the mutation probability for genetic algorithms, *Bäck* (1997) stated that it is impossible to draw a general conclusion from the different experimental investigations proposed. This is because the optimal mutation probability is dependent on both population size and objective function, and this dependency affects the convergence of the genetic algorithm. *Bäck* (1997) concluded by acknowledging that no useful analytical results are known for the dependence of the optimal mutation probability on parameters such as population size or objective function.

Indeed, it is commonly accepted in the genetic algorithms community that, when using a genetic algorithm strategy, success in any application area can only be determined by experimentation and tends to be problem dependent [*Winter et al* (1995)].

In the present computational implementation of genetic algorithms, the evolutionary process involved in the genetic design of the trajectory-tracking digital PID controller for the robotic manipulator searches for an optimal quadruple $\{\alpha,\sigma,\rho,\delta\}$ of tuning parameters. Three different sampling periods, T = 0.01s, T = 0.02s, and T = 0.04s, were selected so that the genetic algorithm is tested over a range of non-asymptotic condition. The optimal quadruple $\{\alpha,\sigma,\rho,\delta\}$ is selected by this evolutionary process so as to minimise the

performance measure, Γ. In all three cases, the best performance was attained using the mutation probability p_m = 0.008, for which the corresponding genetically designed controller will be referred to as the prime design in the remaining sections of this chapter. The results of the genetic design procedure for the mutation probabilities p_m= 0.008, p_m = 0.3, and p_m = 0.98 under the non-asymptotic condition T = 0.01s are shown in Figures 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10, 4.11 and 4.12 over 50 generations. In Figures 4.3, 4.6 and 4.9, the associated best-of-generation and generation-average values of the cost function, Γ, for the mutation probabilities p_m = 0.008, p_m = 0.3, and p_m = 0.98, respectively, are plotted against generation number. It is clearly shown in Tables 4.1, 4.2 and 4.3 that the optimal quadruple of controller parameters obtained for the best design depicted in Figure 4.12 is $\{\alpha,\sigma,\rho,\delta\}$ = {0.174, 0.831,16.751,0.0126}, for which the corresponding value of the cost function Γ = 1.47×10^{-3} is evidently the smallest amongst all different proposed genetically designed controllers. It is also clear from Figures 4.5, 4.8 and 4.11 that the eigenvalues of the perturbation matrix $(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1}$ corresponding to the design under the non-asymptotic condition T = 0.01s always lie on the interval $\left(0, (1+\alpha)/\sigma\right)$ permitted by the robustness corollary [*Porter* and *Abidin* (1990b)] presented in Chapter 3. This fact indicates that the controller system remains locally stable during the sequence of tracking tasks for the selected sampling periods T = 0.01s. This prediction is again confirmed by the plots of the trajectory-tracking errors and joint torques corresponding to the prime design shown in Figures 4.4, 4.7 and 4.10.

Sampling Period $T = 0.01$	Controller Parameters α	σ	ρ	δ	Cost Function Γ
Asymptotic design	0.100	0.700	1.000	0.010	8.64×10^{-3}
Genetic design (p_m= 0.008)	0.174	0.831	16.751	0.0126	1.47×10^{-3}
Genetic design (p_m = 0.30)	0.104	0.842	20.509	0.013	1.62×10^{-3}
Genetic design (p_m = 0.98)	0.161	0.796	18.258	0.018	1.66×10^{-3}

Table 4.1 Comparative results for genetic and asymptotic designs (T = 0.01s).

Sampling Period $T = 0.02$	Controller Parameters α	σ	ρ	δ	Cost Function Γ
Genetic design (p_m= 0.008)	0.118	0.849	16.256	0.0690	9.67×10^{-3}
Genetic design (p_m = 0.30)	0.076	0.891	15.957	0.065	10.53×10^{-3}
Genetic design (p_m = 0.98)	0.114	0.867	18.173	0.082	10.28×10^{-3}

Table 4.2 Comparative results for genetic designs (T = 0.02s).

Sampling Period $T = 0.04$	Controller Parameters				Cost Function
	α	σ	ρ	δ	Γ
Genetic design (p_m= 0.008)	0.062	0.893	9.222	0.135	60.91 x 10^{-3}
Genetic design (p_m = 0.30)	0.086	0.898	7.893	0.111	63.67 x 10^{-3}
Genetic design (p_m = 0.98)	0.078	0.890	9.448	0.130	63.69 x 10^{-3}

Table 4.3 Comparative results for genetic designs (T = 0.04s).

In this genetic design of fast-sampling robust PID controllers for robotic manipulators, a significant improvement in performance was achieved for the proposed genetically designed controllers (under the non-asymptotic condition T = 0.01s) in comparison to the performance obtained under the same non-asymptotic condition T = 0.01s for the controller designed by *Abidin*(1990b) using the singular perturbation methods presented in Chapter 3. Indeed, the designs resulted in good trajectory-tracking accuracy characterised by a small peaks error of 4mm (see Figures 4.4, 4.7 and 4.10) during the transient period following the introduction of a sudden 5 Kg load. This compares very favourably with the peak error of 6mm (see Figure 3.5) obtained for the controller designed by singular perturbation methods. In addition, the overall trajectory-tracking accuracy performance measured by Γ = 1.47 x 10^{-3} , Γ = 1.62 x 10^{-3} and Γ = 1.66 x 10^{-3} are clearly superior to their singular perturbation counterpart

presented in Chapter 3 which produced a value of $\Gamma = 8.64 \times 10^{-3}$. This superiority holds for all the proposed genetically designed controllers which achieved values of the objective function $\Gamma \in [1.47 \times 10^{-3}, 1.66 \times 10^{-3}]$. The comparative results obtained in solving this design problem genetically and by using the singular perturbation methods are summarised in Table 4.1. This table also displays the values of the optimal quadruples, $\{\alpha,\sigma,\rho,\delta\}$, of controller parameters and the corresponding values of the cost function.

So far, the discussion has been confined to the performance of the genetically designed controllers compared with their counterpart designed by singular perturbation methods. It is therefore relevant to extend this comparison to the different genetic designs. Indeed, from Table 4.1 it transpires that the design under the non-asymptotic condition $T = 0.01$s is to be credited with the best performance by achieving the smallest cost function value of all proposed designs. However, it is clear from Table 4.1, 4.2 and 4.3 that the genetic procedure can be effective under different non-asymptotic condition corresponding to sampling period $T= 0.01$s, $T= 0.02$s and $T= 0.04$s. As expected, an increase of the sampling period is accompanied by a degradation of the tracking performance as depicted in Figures 4.14, 4.16, 4.18, 4.20, 4.22, and 4.24.

It is also clear from Figures 4.3(a), 4.6(a), 4.9(a), 4.13(a), 4.15(a), 4.17(a), 4.19(a), 4.21(a) and 4.23(a) that the convergence towards the optimal cost function value is fastest in the design case corresponding to $p_m = 0.008$. It is

also evident that, the higher the mutation probability p_m, the slower the convergence. In fact, large mutation probabilities have an adverse effect on the speed of convergence, due to their tendency to disrupt some relevant information contained in the population of individuals during the evolutionary process. Indeed, it is clear from Figures 4.3(b), 4.6(b), 4.9(b), 4.13(b), 4.15(b), 4.17(b), 4.19(b), 4.21(b) and 4.23(b), which represent the average cost function of the population of individuals for the successive generations, that an increase in the mutation probability, p_m, is a source of disruption to the population of individuals.

It is thus evident that, when using genetic algorithms, an optimal strategy must maintain a balance between the exploitation of the best individuals found so far and the continued exploration for potentially better individuals which can be achieved using an adequate mutation probability.

It is also noticeable that the larger is the sampling period T the bigger is the degradation of the performance, which confirms the fact that design procedure of *Porter* and *Abidin* (1990a) presented in Chapter 3 and used throughout this chapter derives from the fast-sampling methodologies of *Porter et al* (1985).

4.4 CONCLUSION

It has been shown in this chapter that genetic algorithms can be conveniently

used to tune the parameter set $\{\alpha,\sigma,\rho,\delta\}$ for fast-sampling digital PID controllers. This genetic design procedure has been illustrated by the design of a trajectory-tracking system for the three-degree-of-freedom robotic manipulator for which a PID digital controller was previously designed using the singular perturbation technique presented in Chapter 3. These genetic designs are characterised by an evident improvement in the trajectory-tracking performance when compared with that obtained from the singular perturbation design (Chapter 3 under identical non-asymptotic condition $T = 0.01$s).

In the present genetic design procedure, genetic algorithms have been used to 'robustify' the fast-sampling digital PID controllers embodied in digital trajectory-tracking system in the case of non-linear time-varying and completely irregular plants for which no non-asymptotic robustness theorem currently exists. This process of robustification has been effected by using genetic algorithms to determine the optimal set of controller tuning parameters for trajectory-tracking tasks (which include sudden changes of payload) by making implicit use of the robustness corollary [*Porter* and *Abidin* (1990b)] presented in Chapter 3.

The attractive features of the genetic design procedure include simplicity of performance specification, tuning flexibility, and the implicit use the robustness characteristics established by *Porter* and *Abidin* (1990). The simplicity of the performance specification feature permits the selection of performance measures which embody practical engineering constraints such as amplitude, or rate limits on the inputs, or outputs of complex devices such as robotic

manipulators. The use of such performance measures is investigated in Chapter 6.

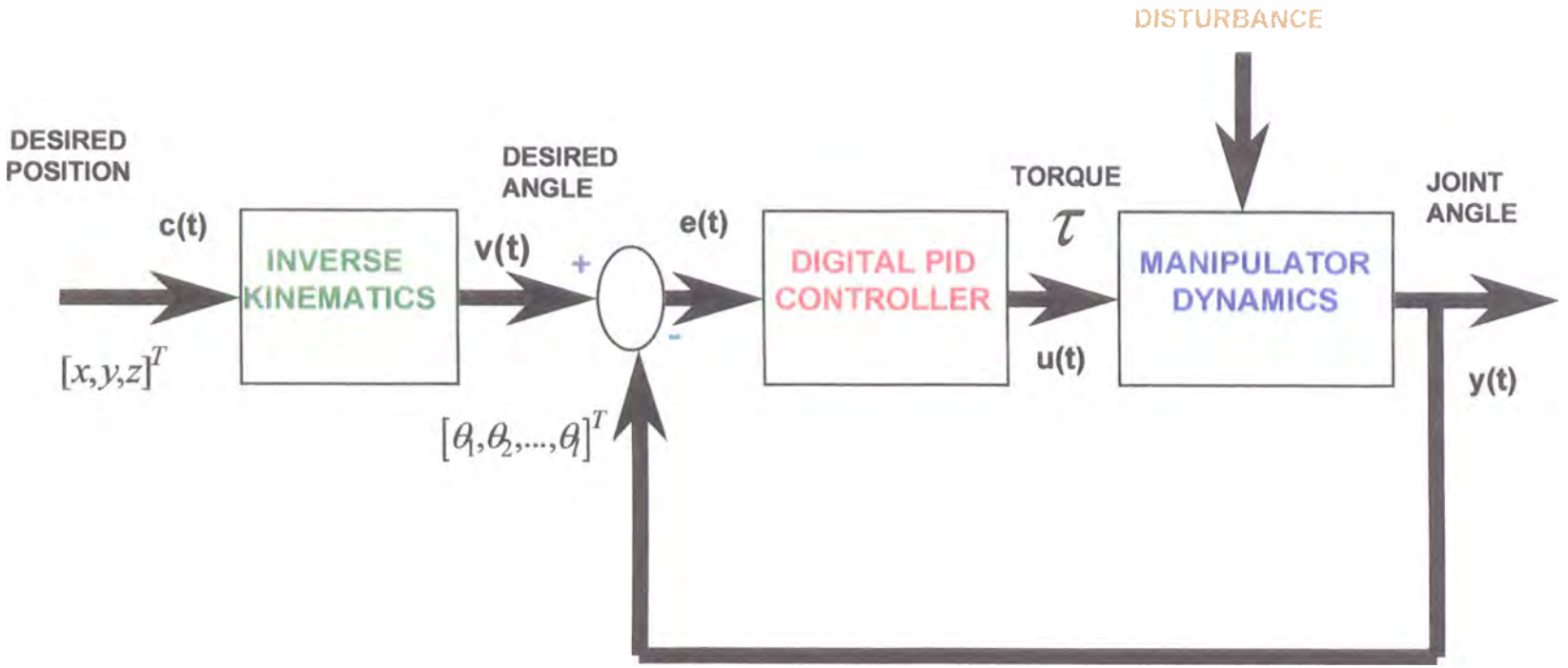

Figure 4.1: Block diagram representation of the closed-loop trajectory-tracking system.

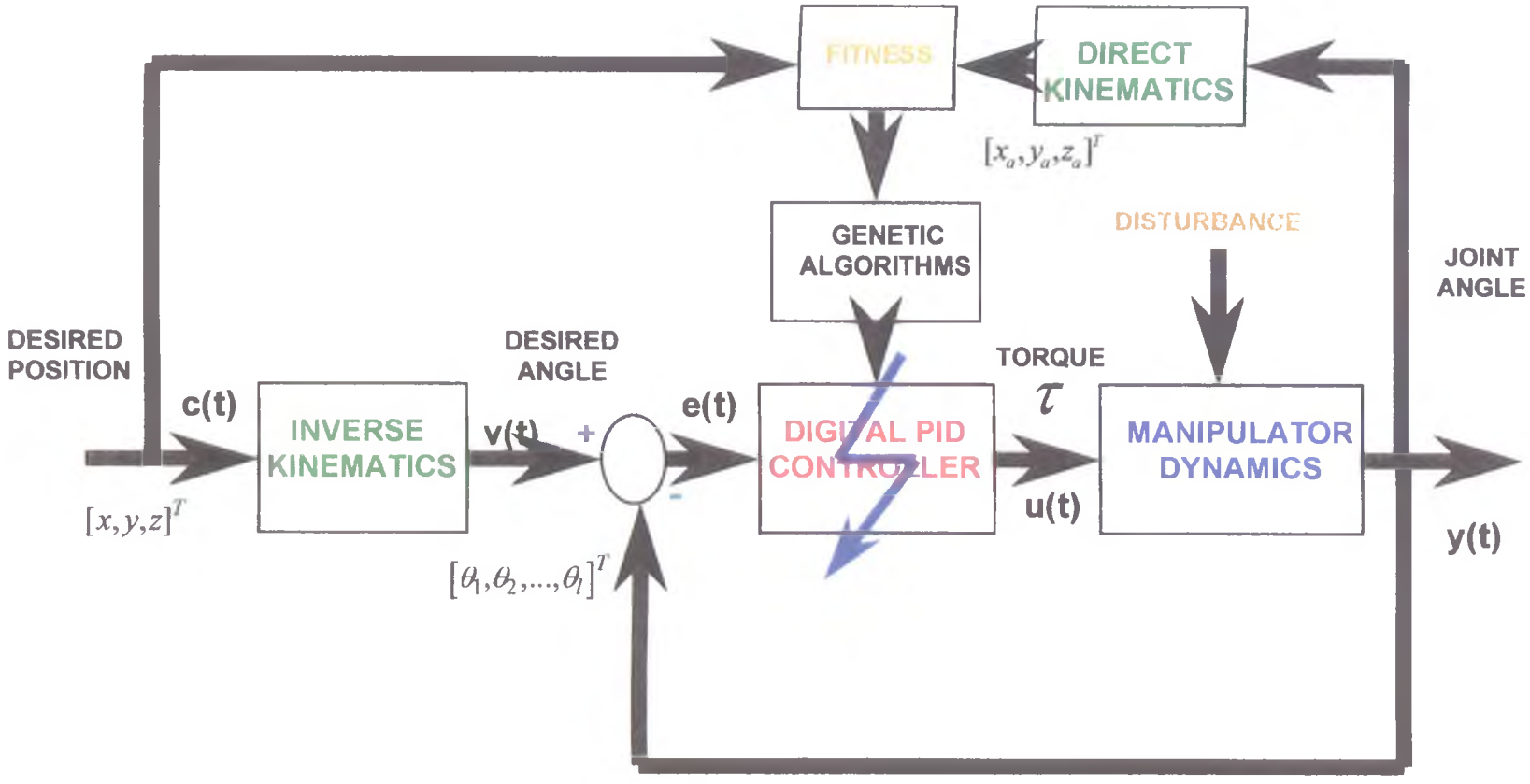

Figure 4.2: computational implementation of trajectory-tracking system and its associated genetic algorithms mechanism

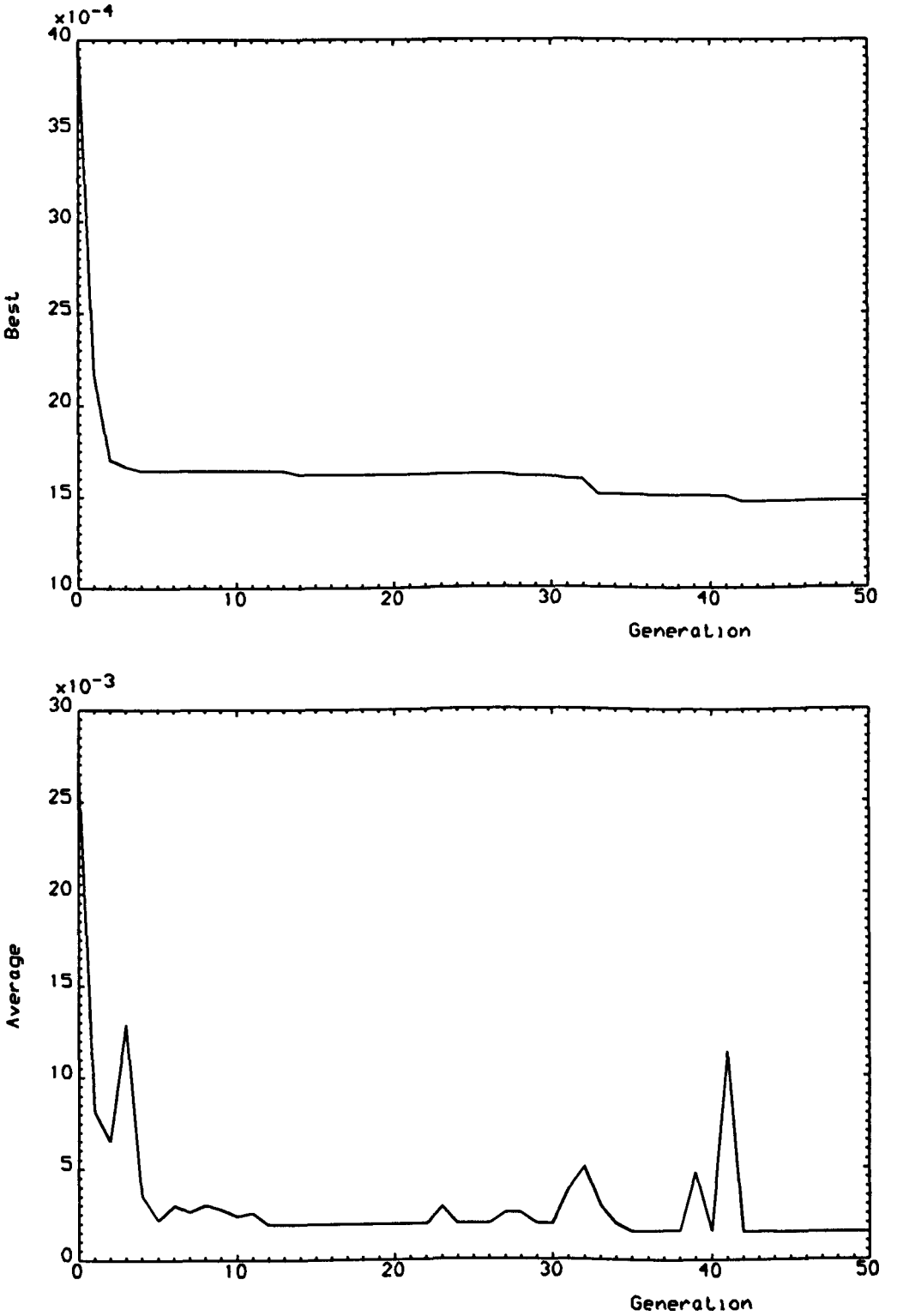

Figure 4.3: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.008, T = 0.01).

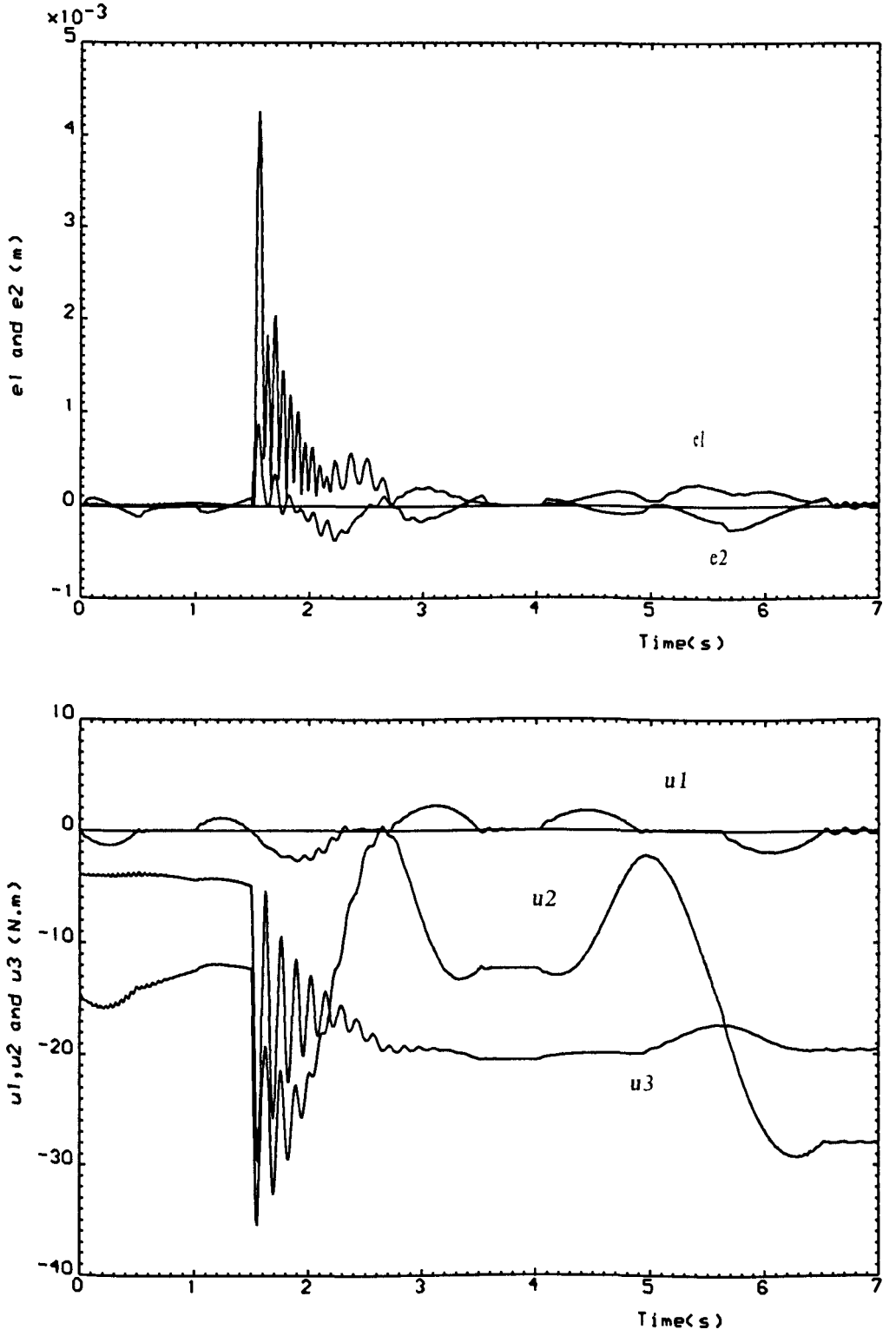

Figure 4.4: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.008,T=0.01).

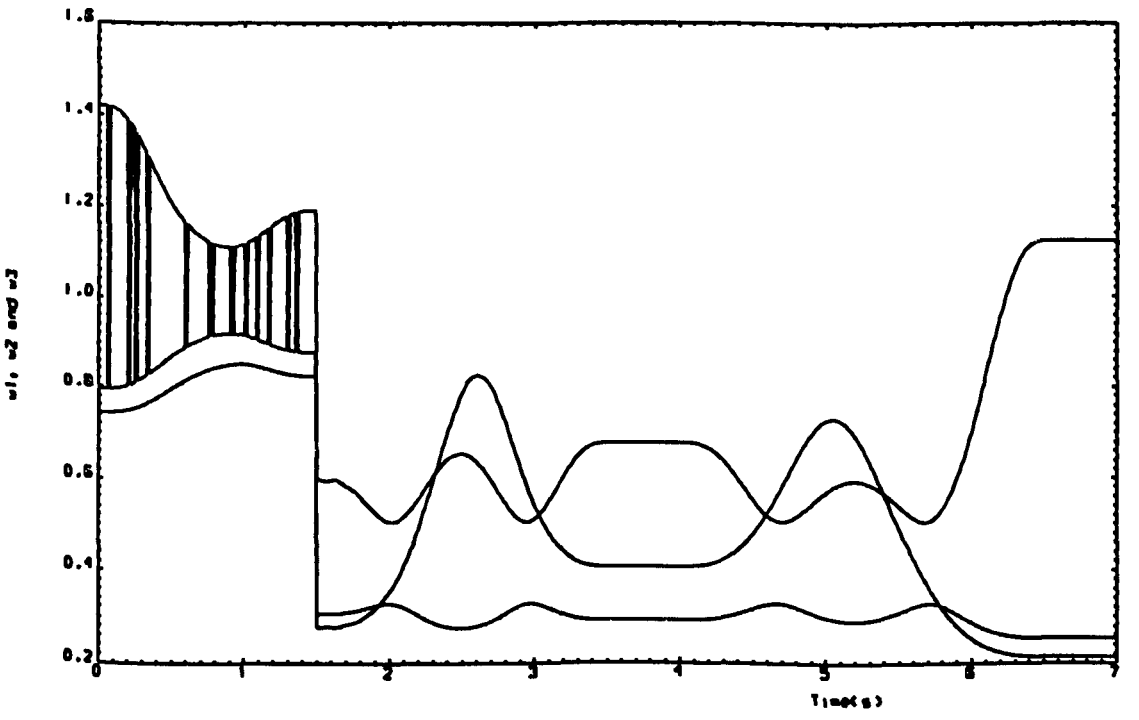

Figure 4.5: Time-domain behaviour of eigenvalues of plant perturbation for genetically designed digital PID Controller (p_m = 0.008, T = 0.01).

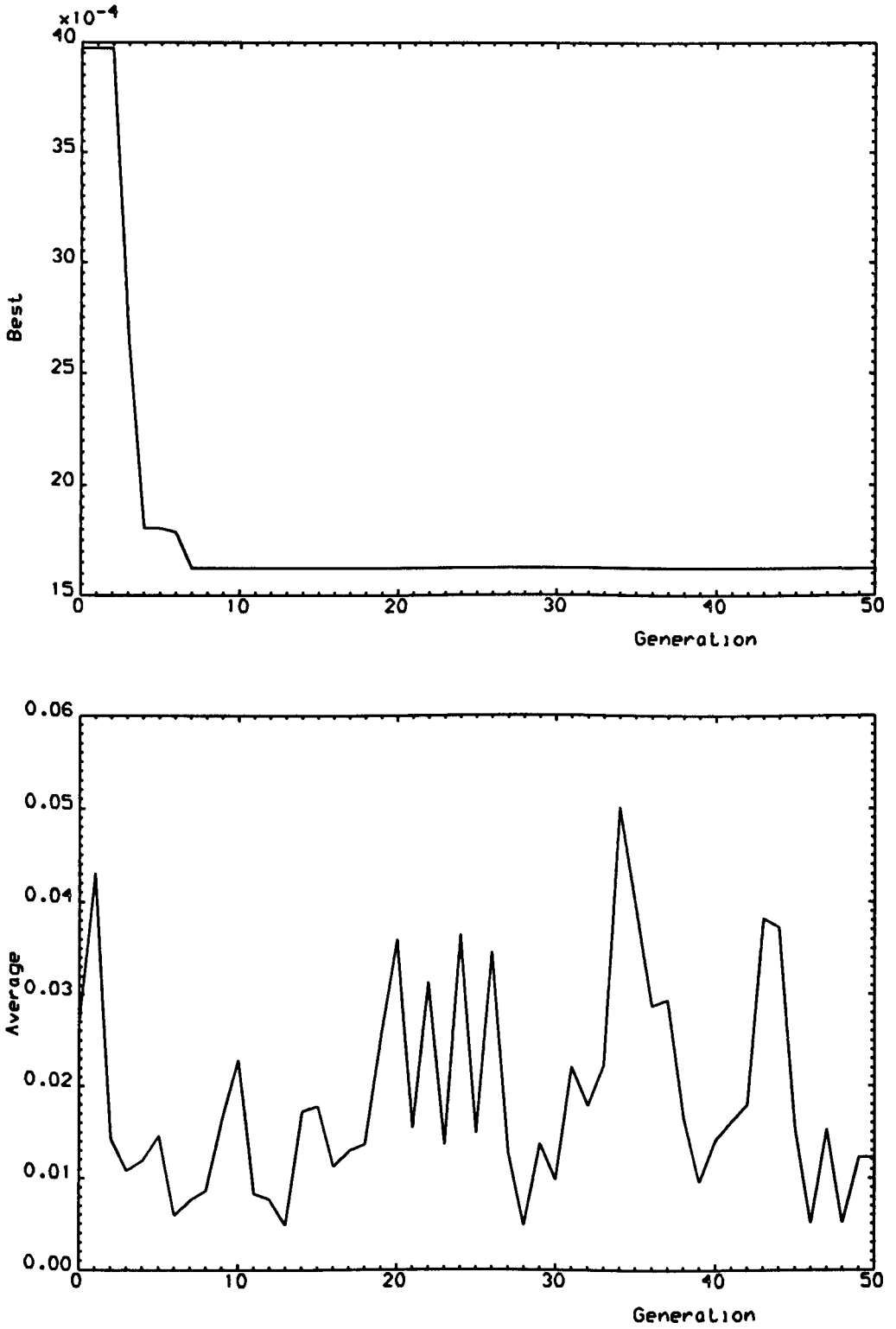

Figure 4.6: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.3, T = 0.01).

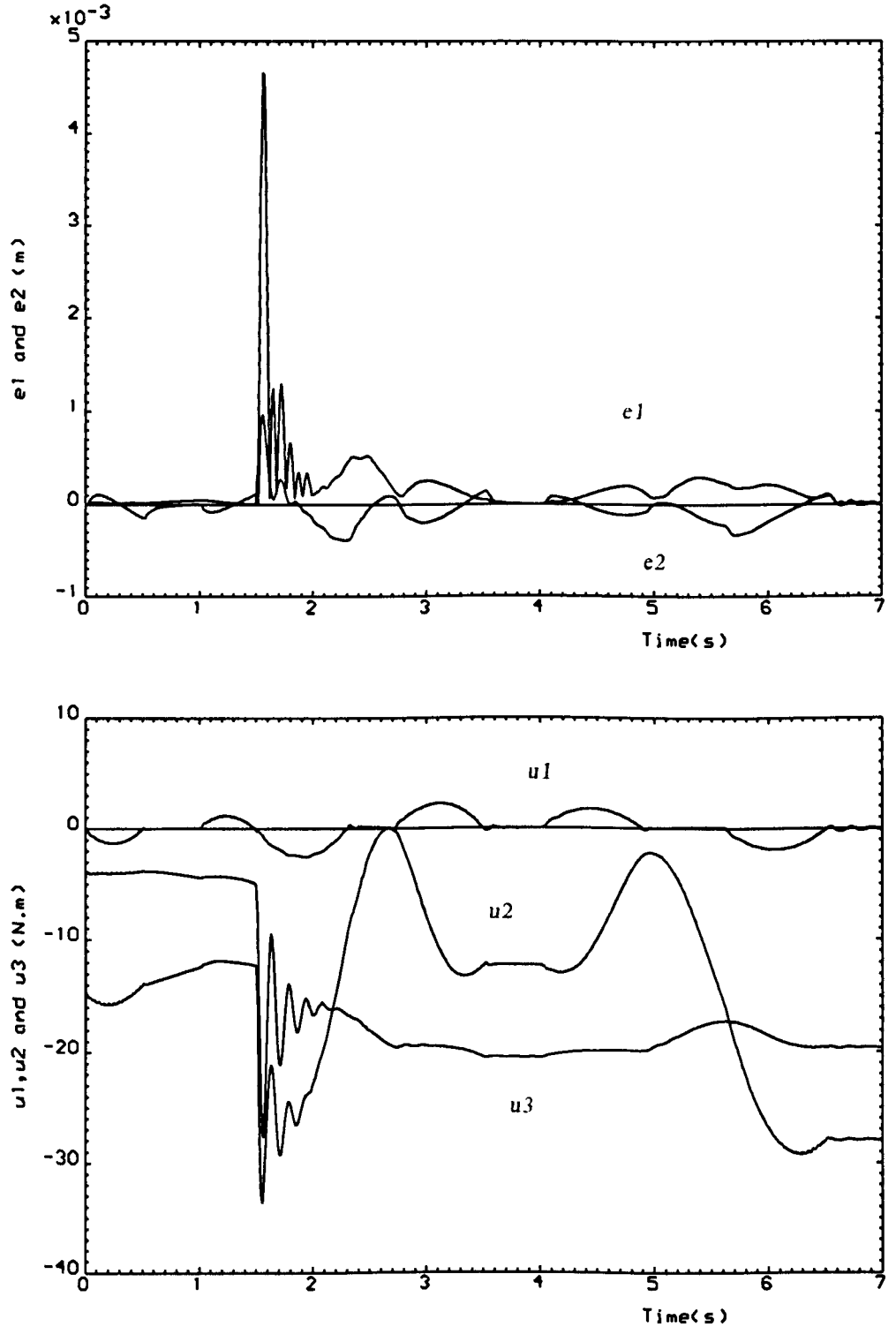

Figure 4.7: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.3,T=0.01).

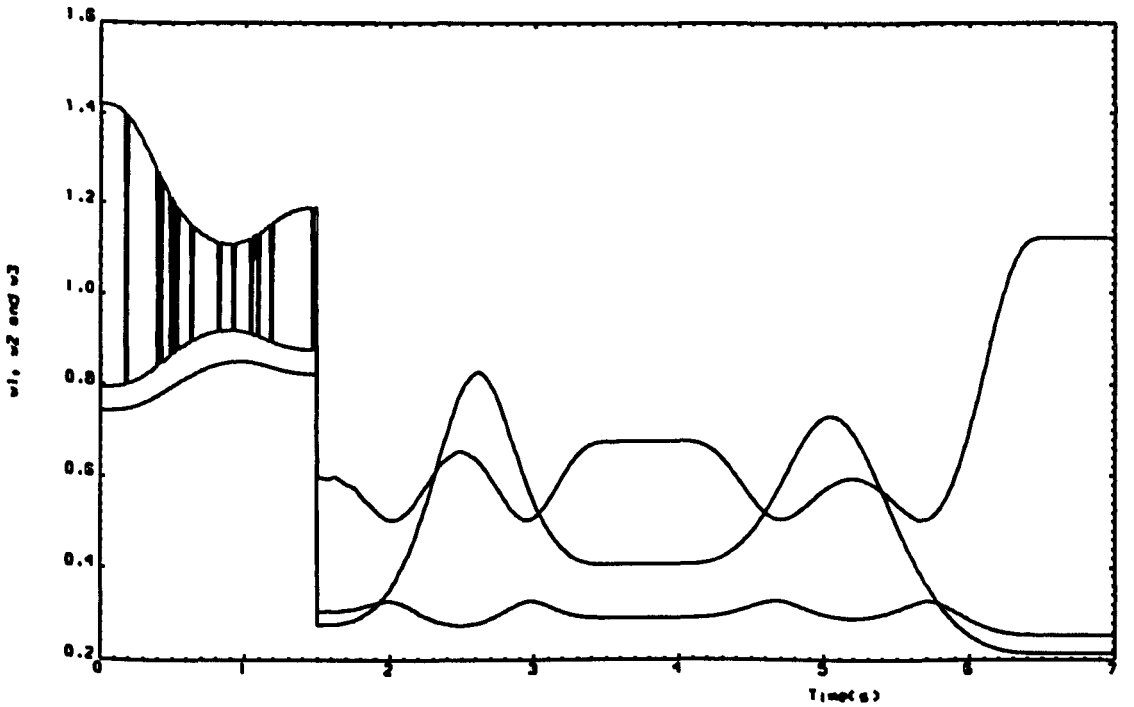

Figure 4.8: Time-domain behaviour of eigenvalues of plant perturbation for genetically designed digital PID Controller (p_m = 0.3, T = 0.01).

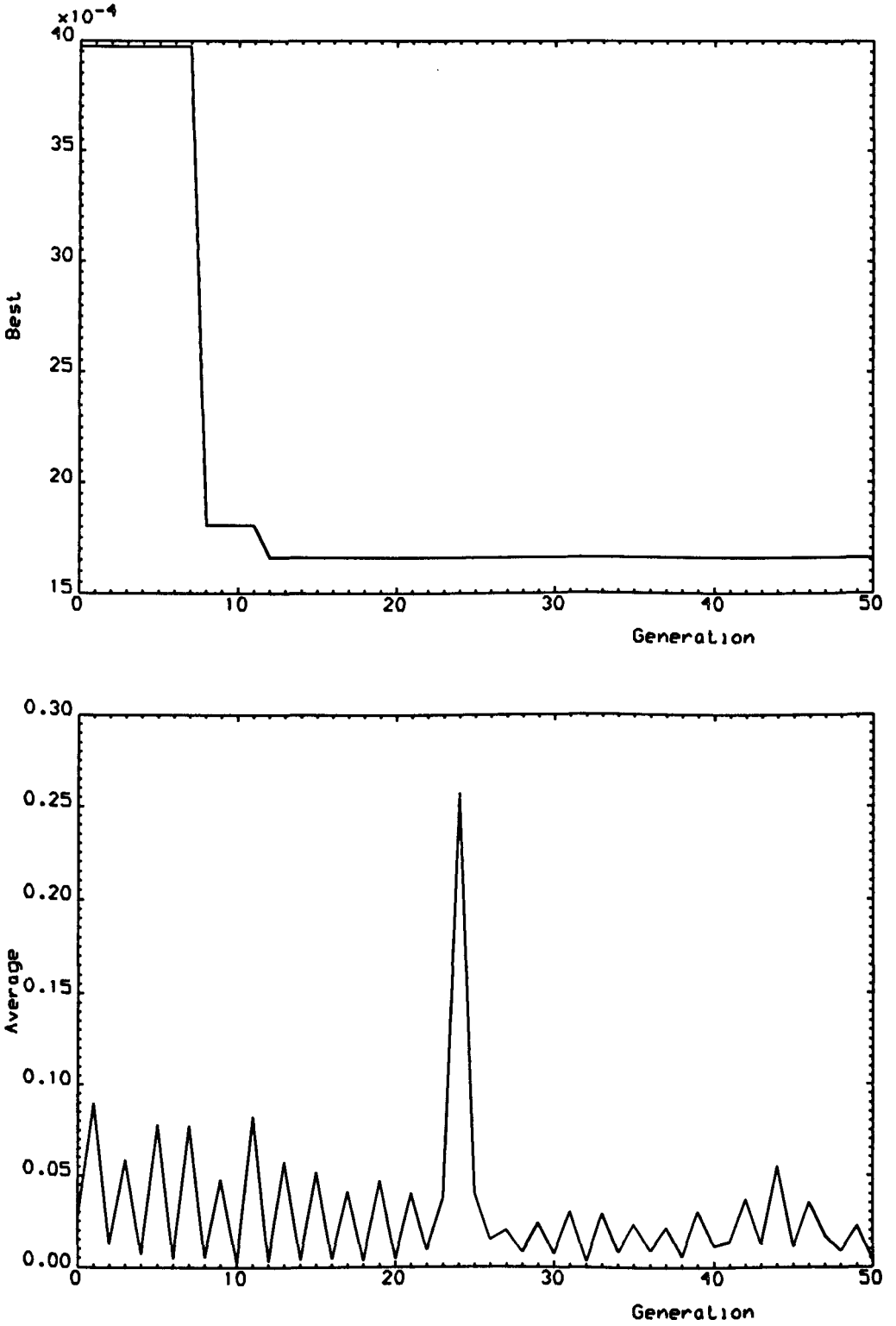

Figure 4.9: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.98, T = 0.01).

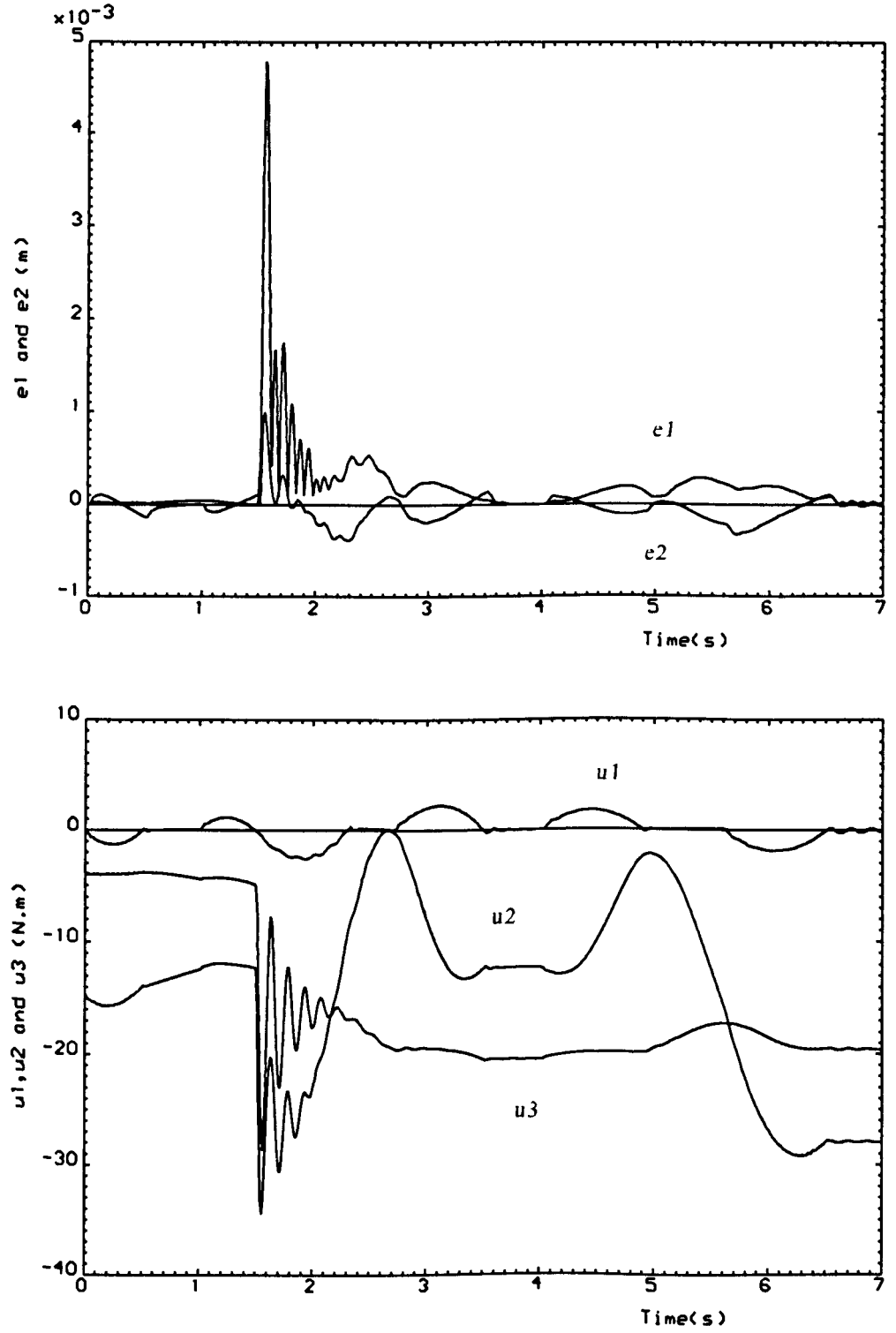

Figure 4.10: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.98,T=0.01).

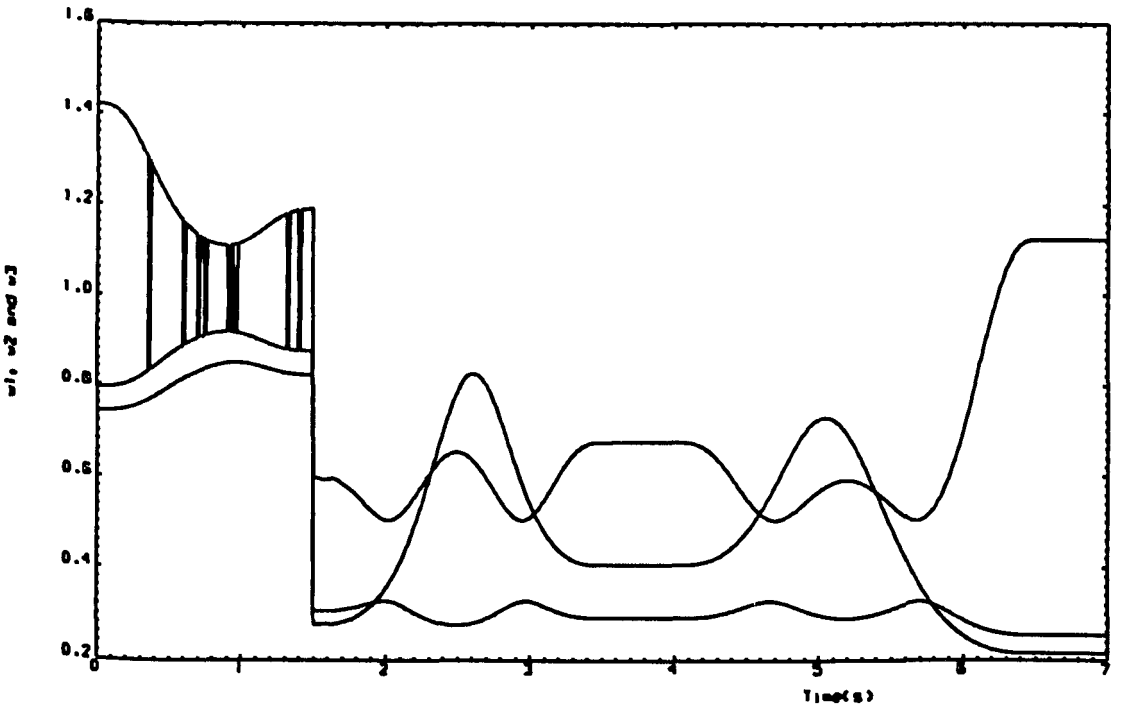

Figure 4.11: Time-domain behaviour of eigenvalues of plant perturbation for genetically designed digital PID Controller (p_m = 0.98, T = 0.01).

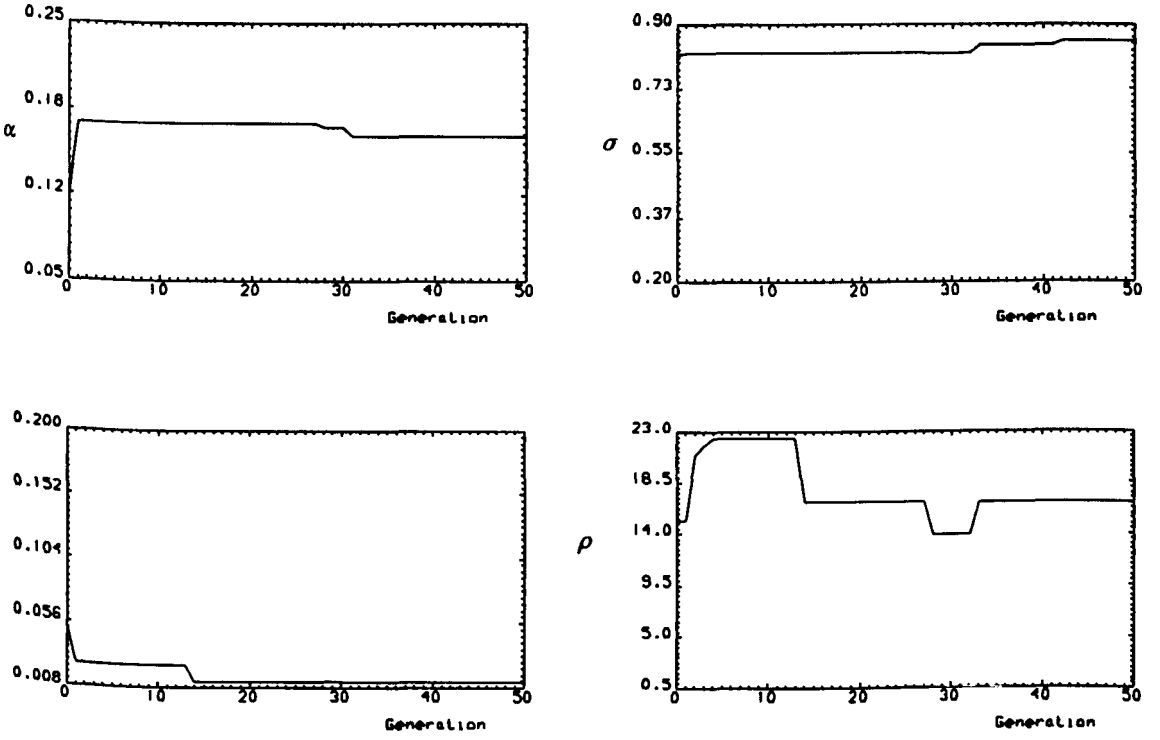

Figure 4.12: Best-of-generation of the genetically tuned set of controllers Parameters $\{\alpha,\delta,\sigma,\rho\}$ (p_m = 0.008, T = 0.01).

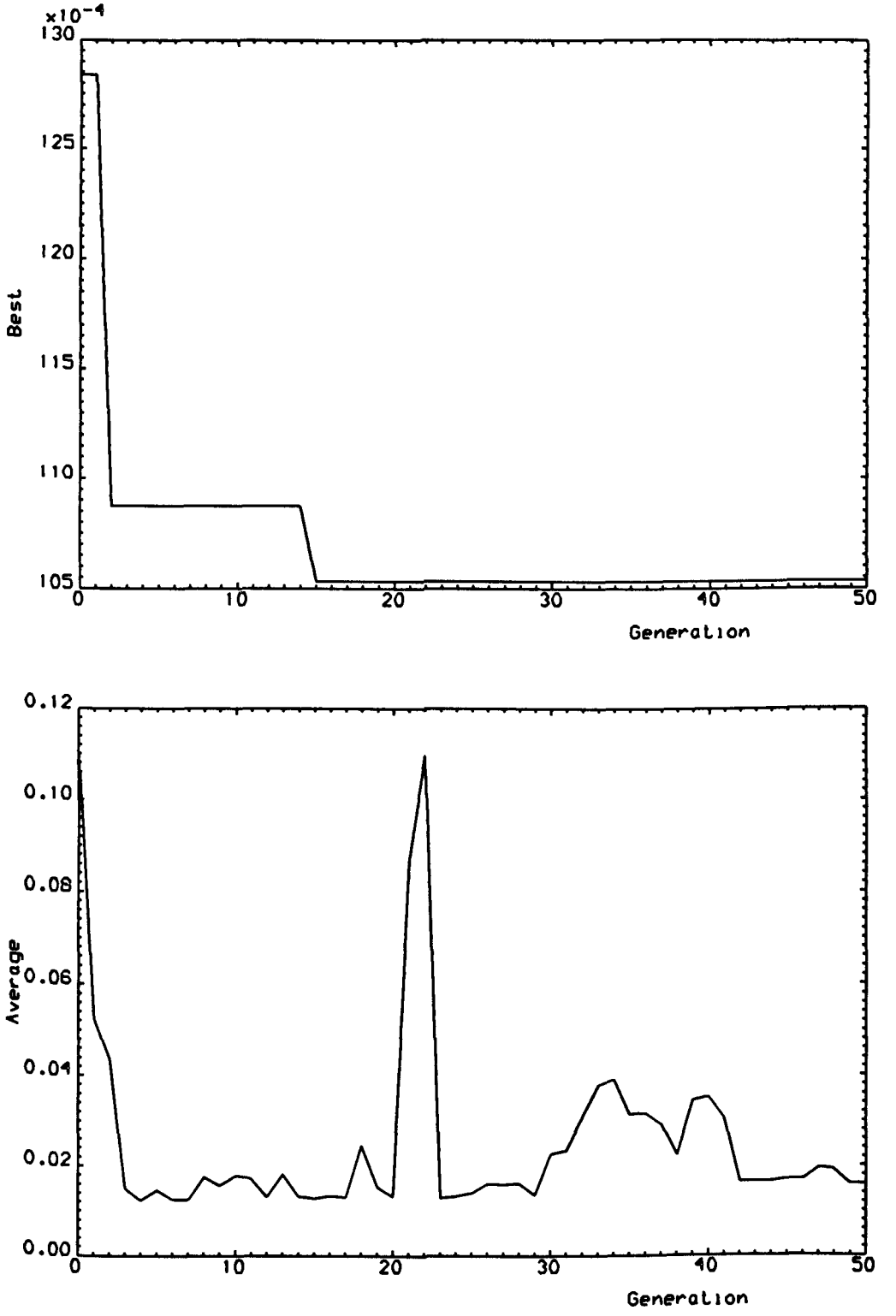

Figure 4.13: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.008, T = 0.02).

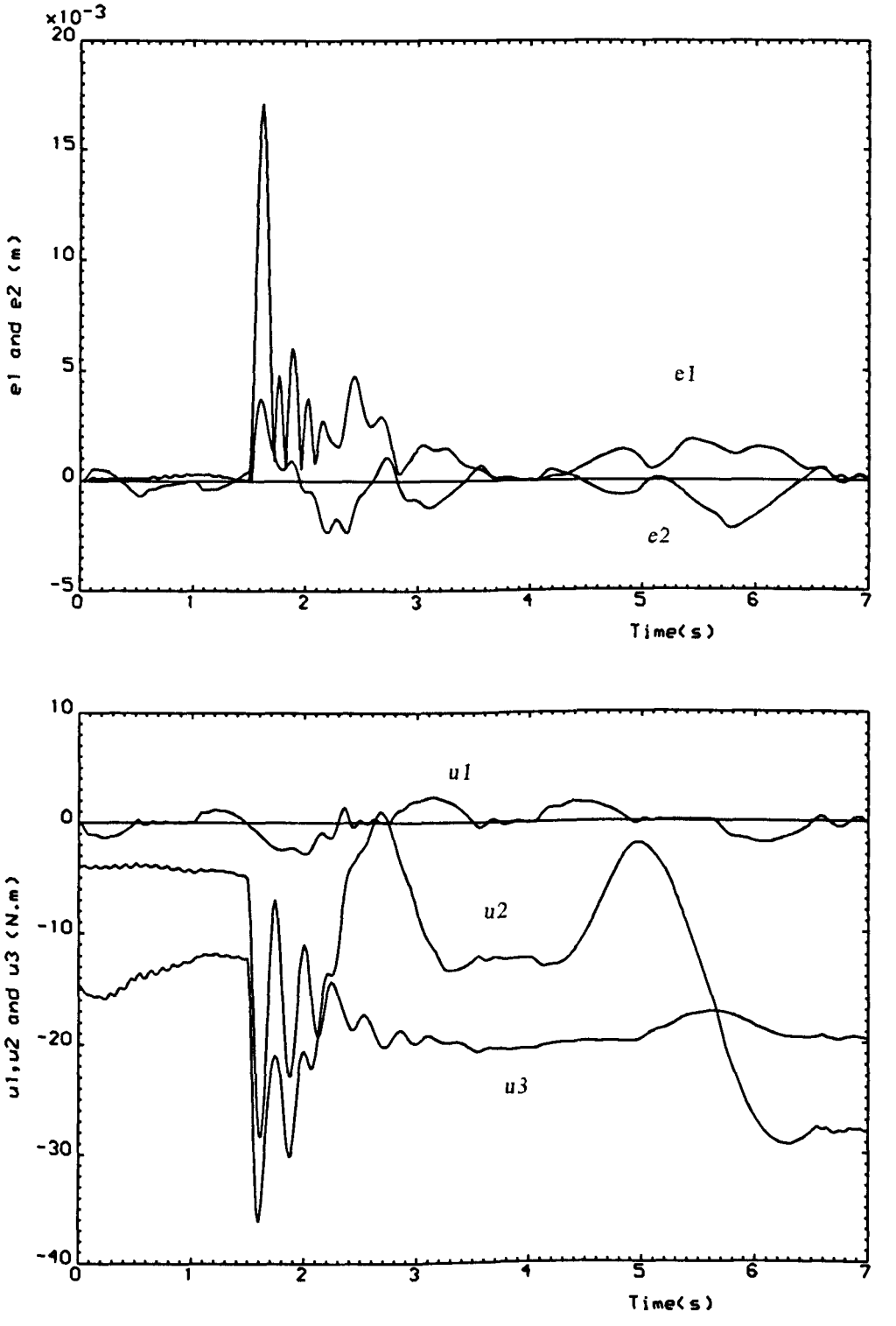

Figure 4.14: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.008,T=0.02).

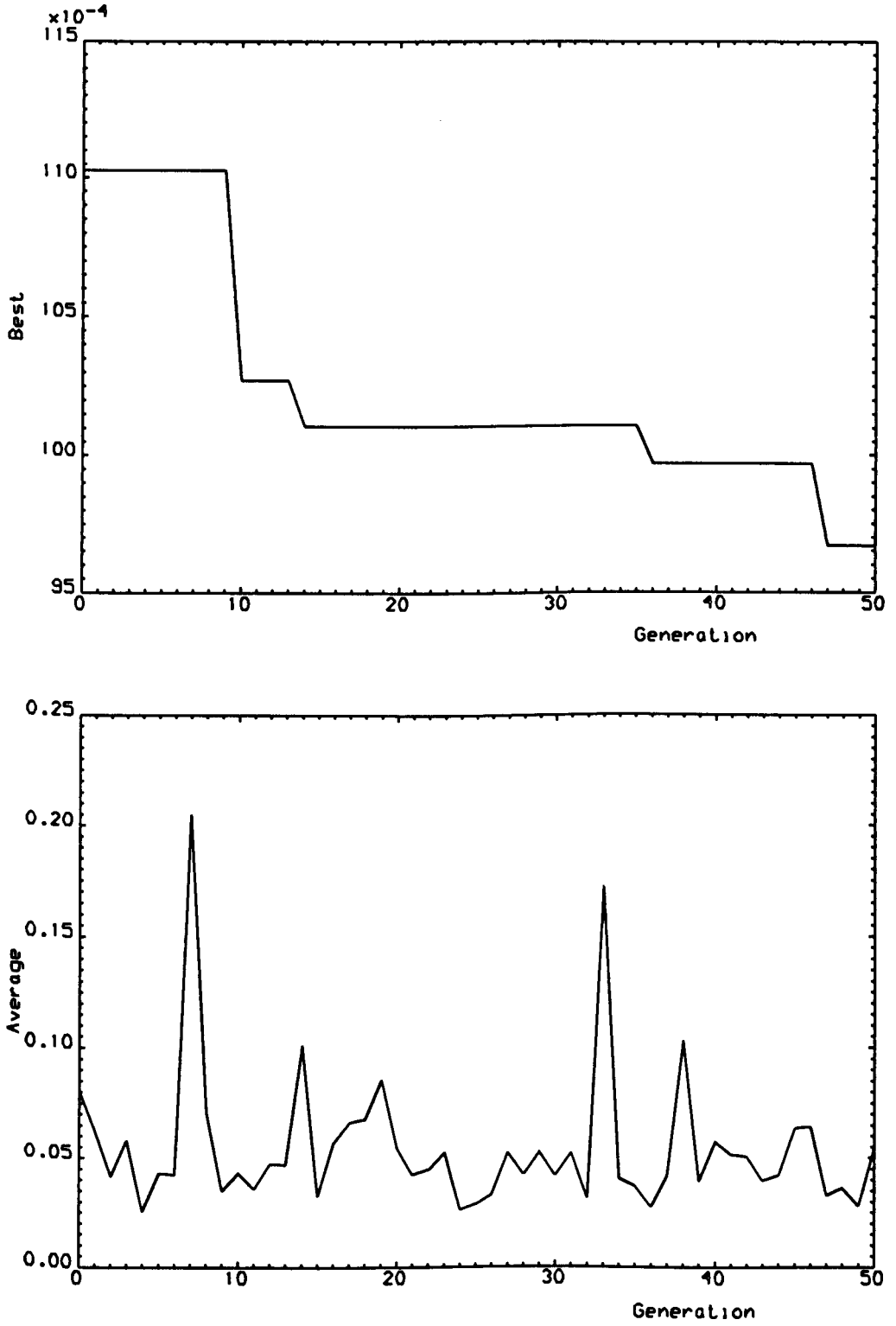

Figure 4.15: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.3, T = 0.02).

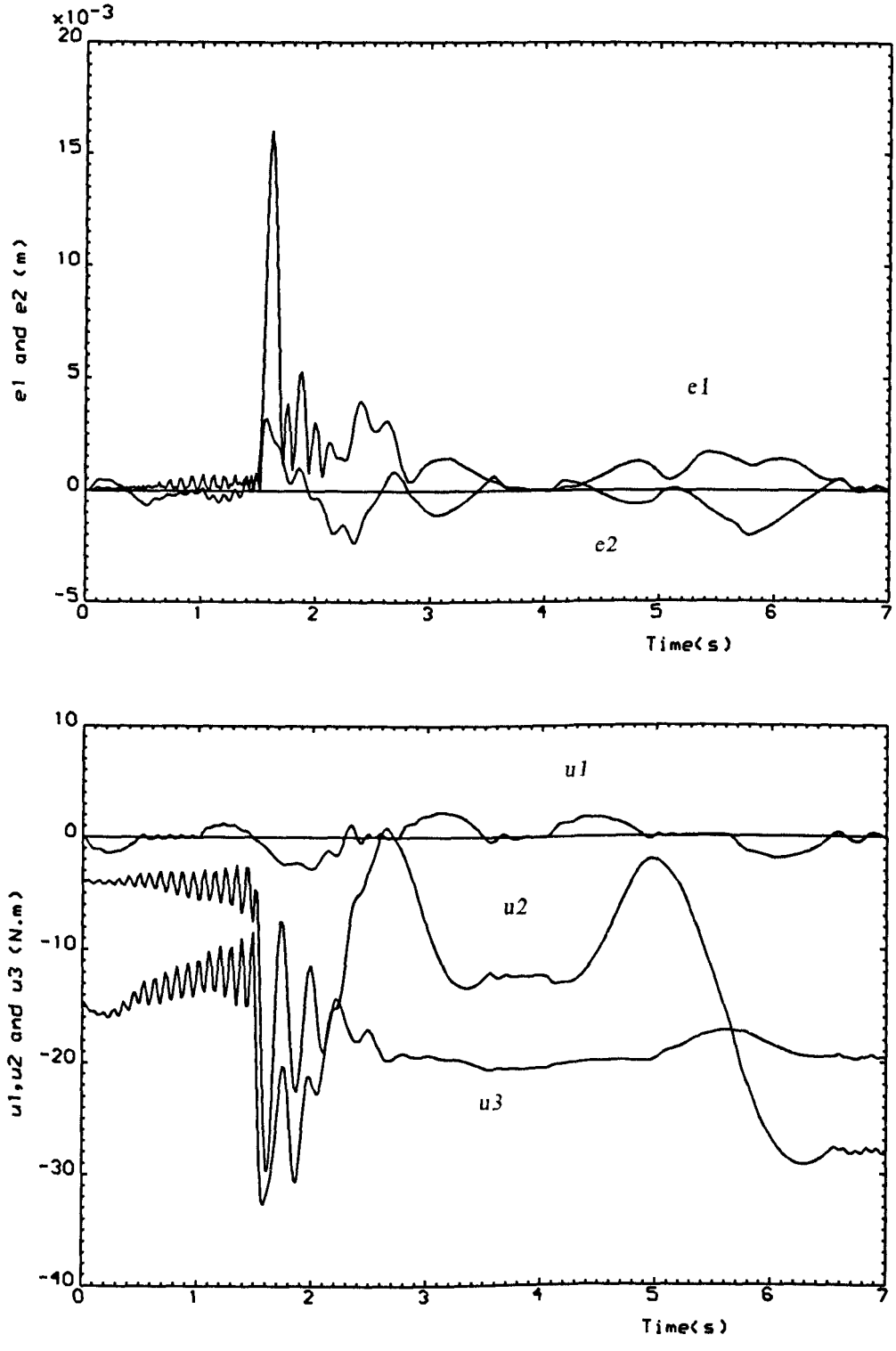

Figure 4.16: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.3,T=0.02).

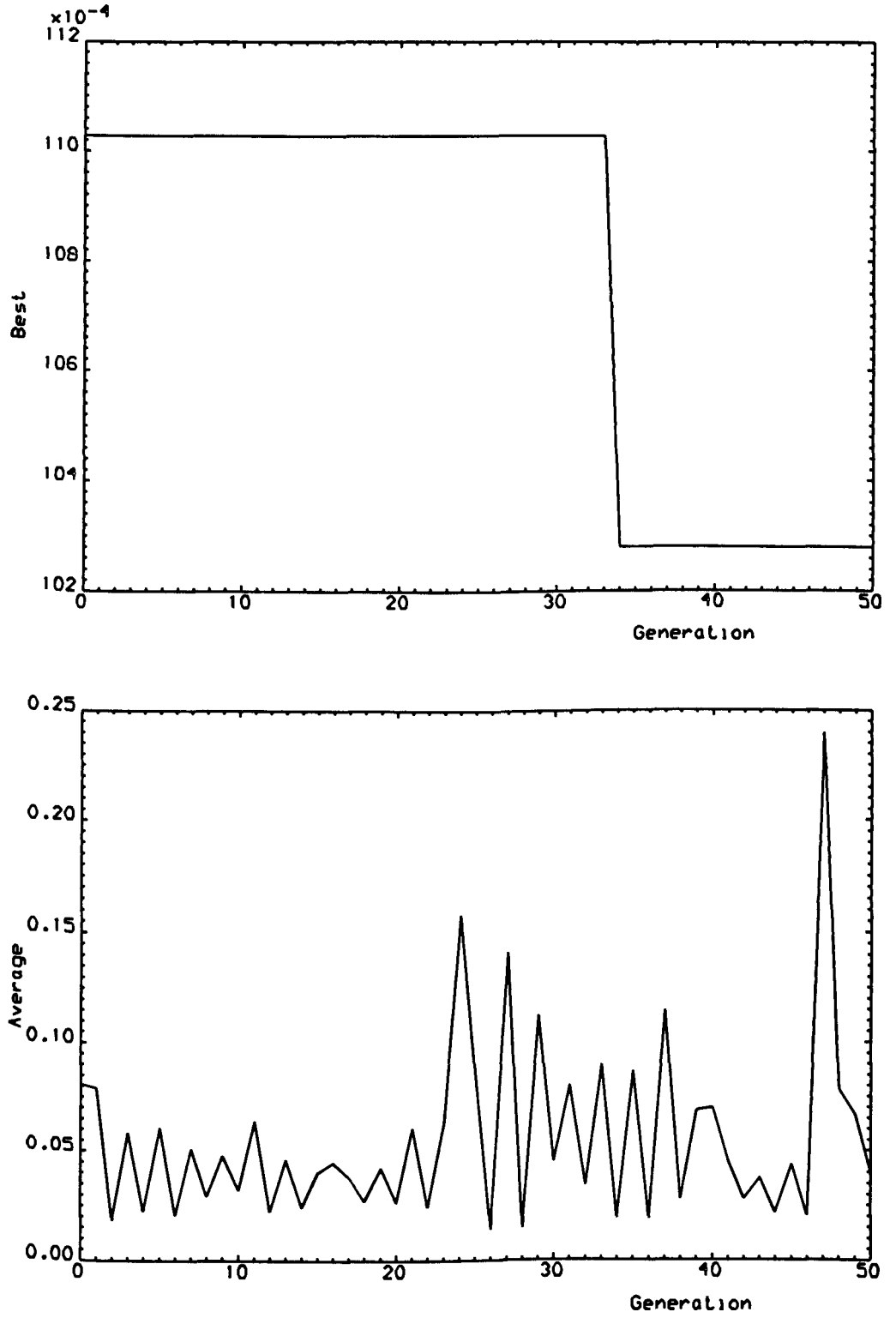

Figure 4.17: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.98, T = 0.02).

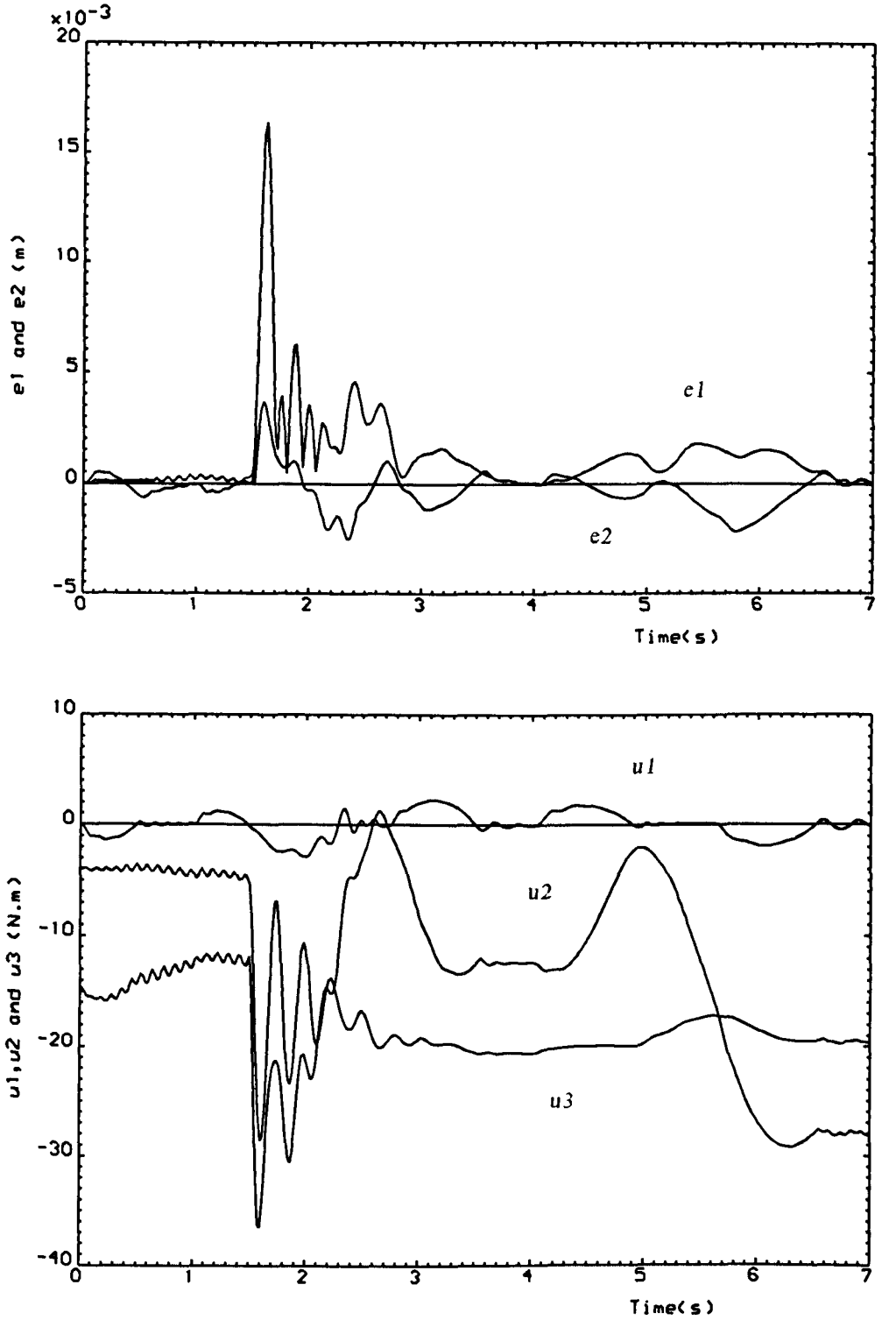

Figure 4.18: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.98,T=0.02).

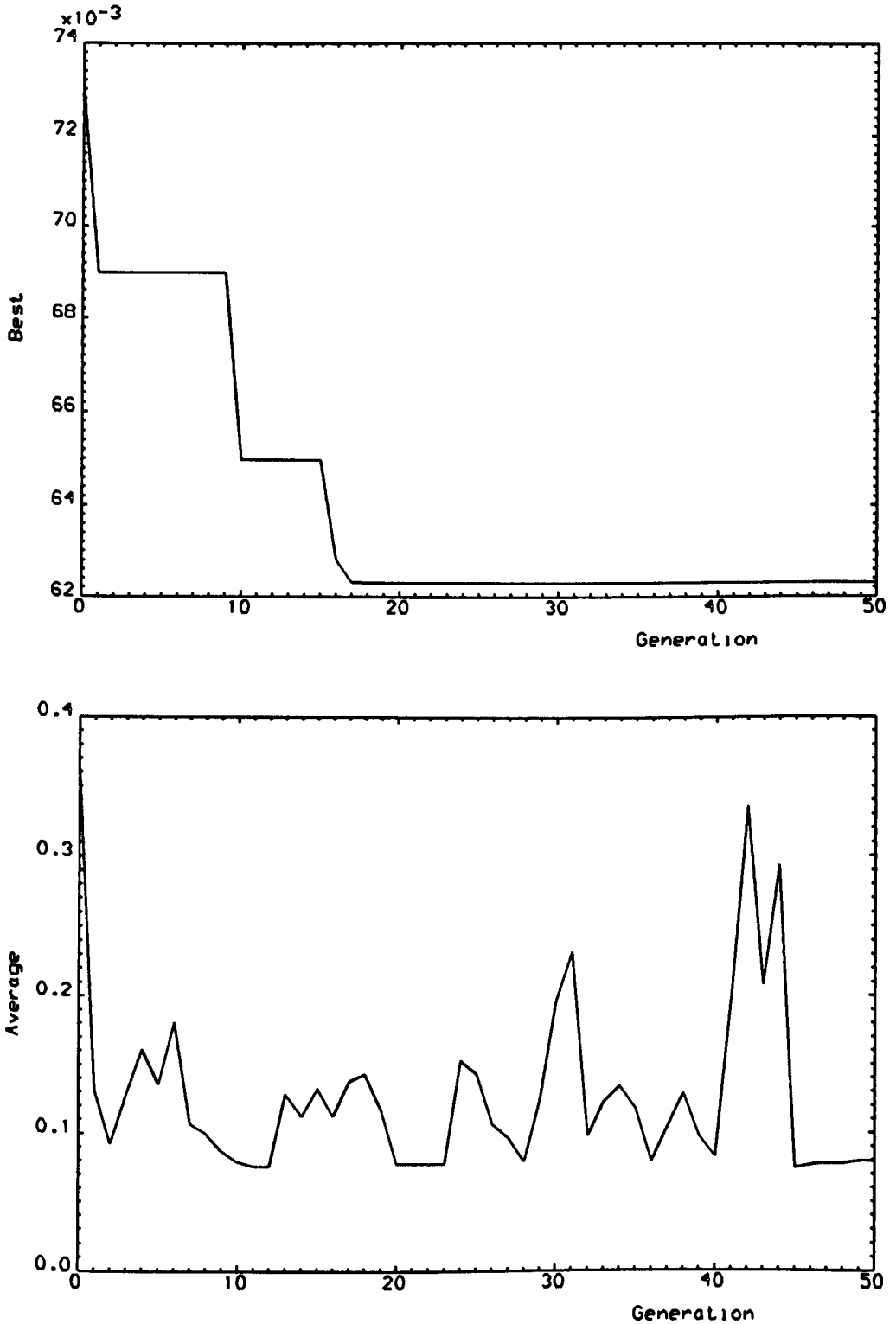

Figure 4.19: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.008, T = 0.04).

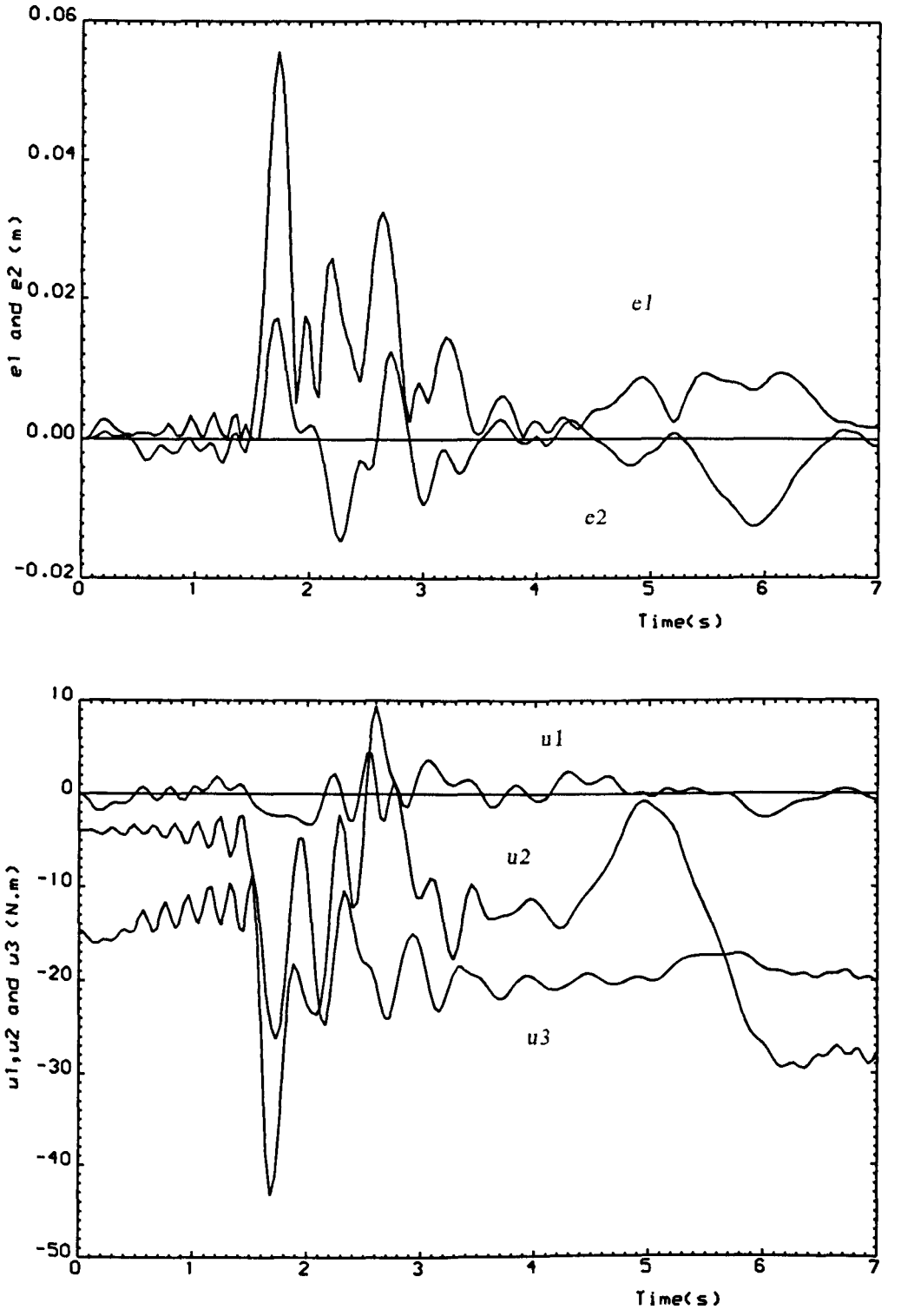

Figure 4.20: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.008,T=0.04).

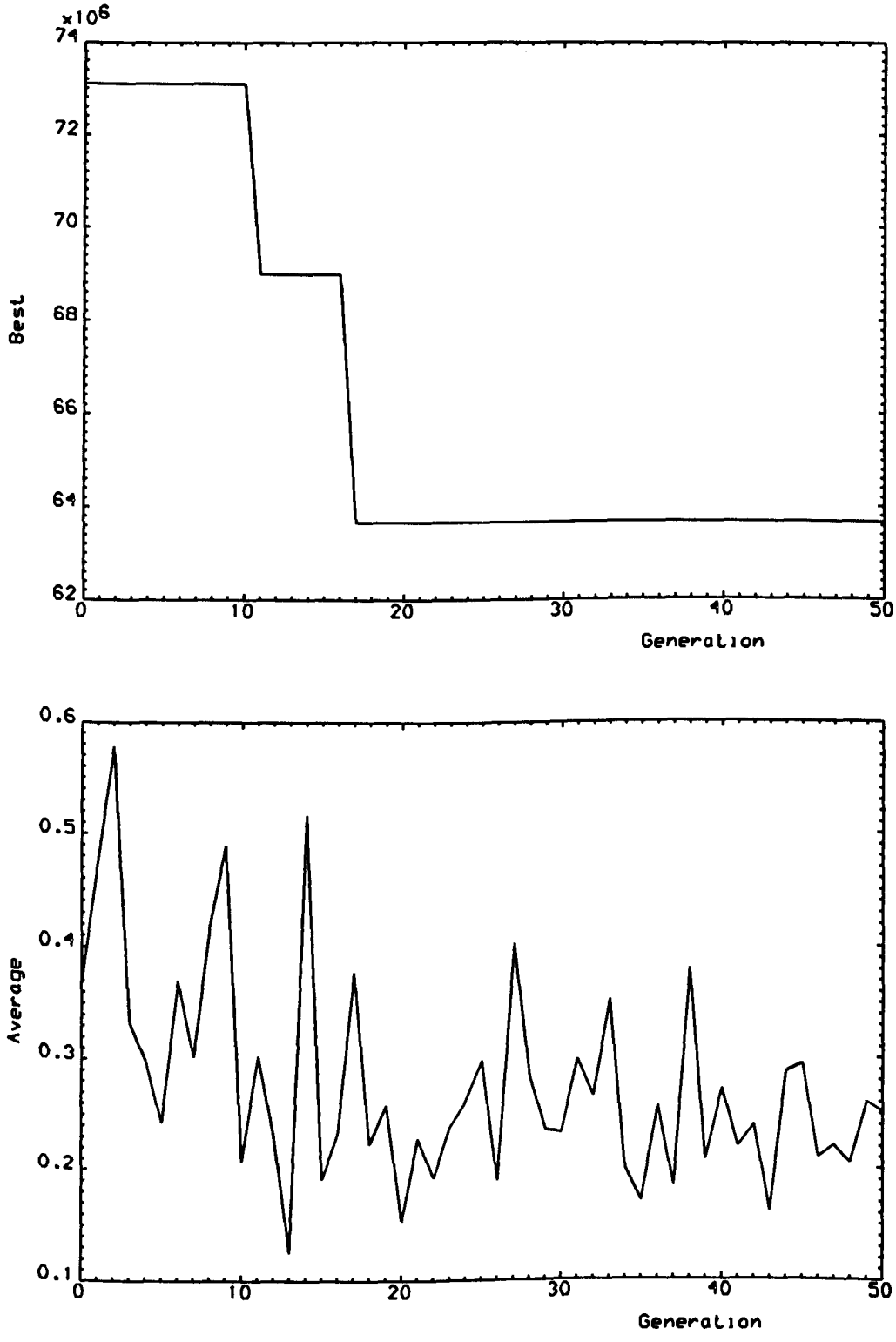

Figure 4.21: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.3, T = 0.04).

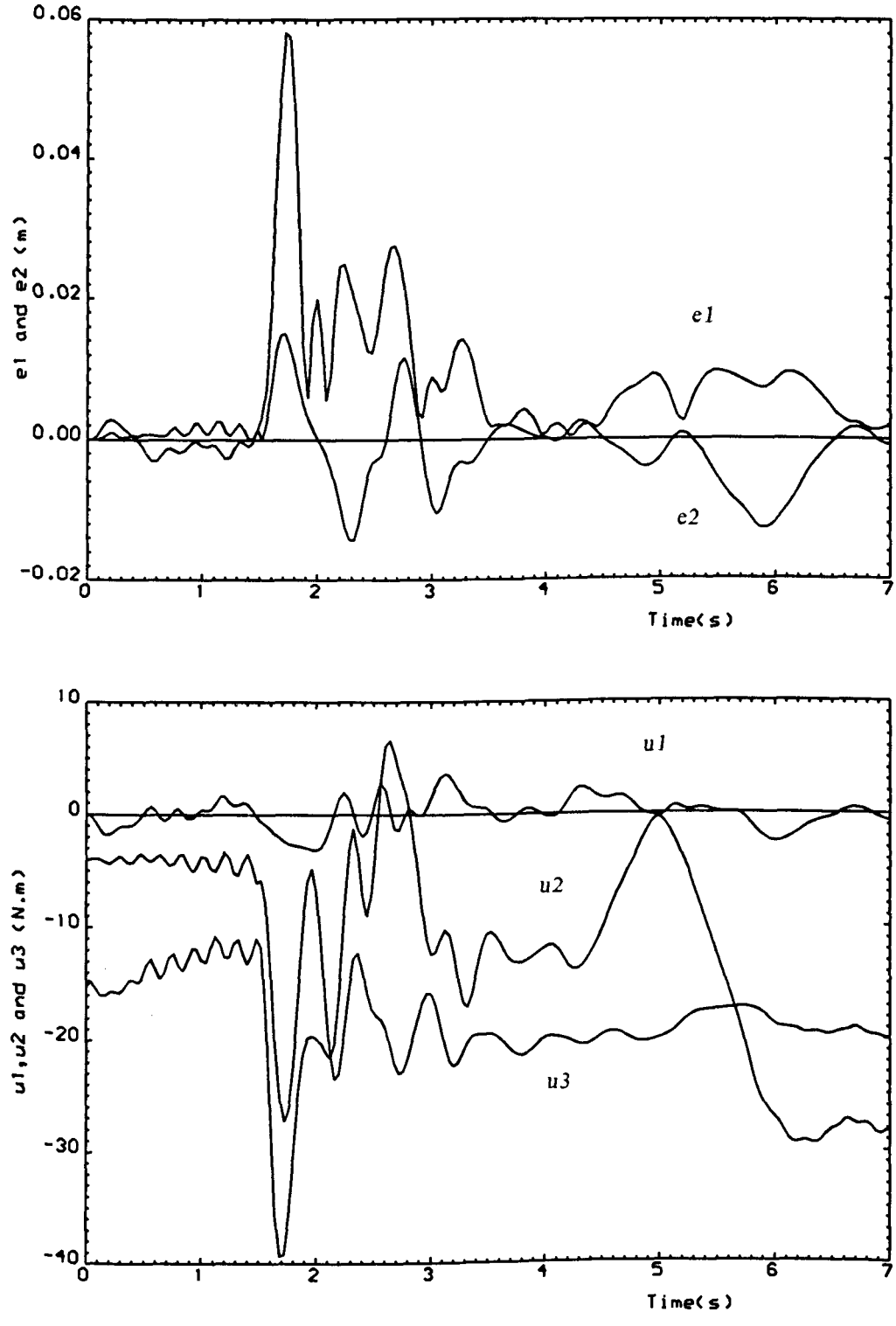

Figure 4.22: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.3,T=0.04).

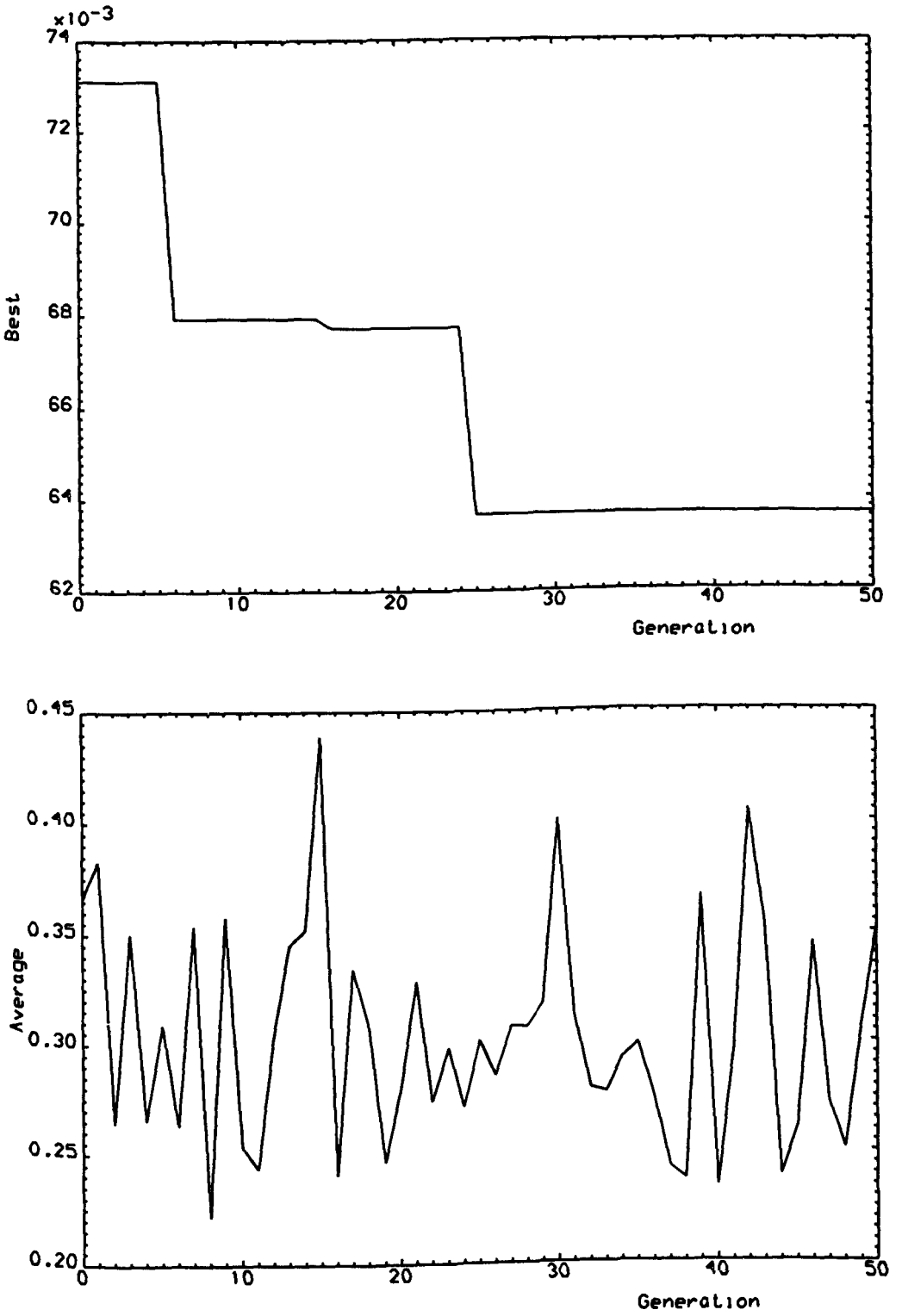

Figure 4.23: Best-of-generation and average-of-generation of cost function for genetically designed controller (p_m = 0.98, T = 0.04).

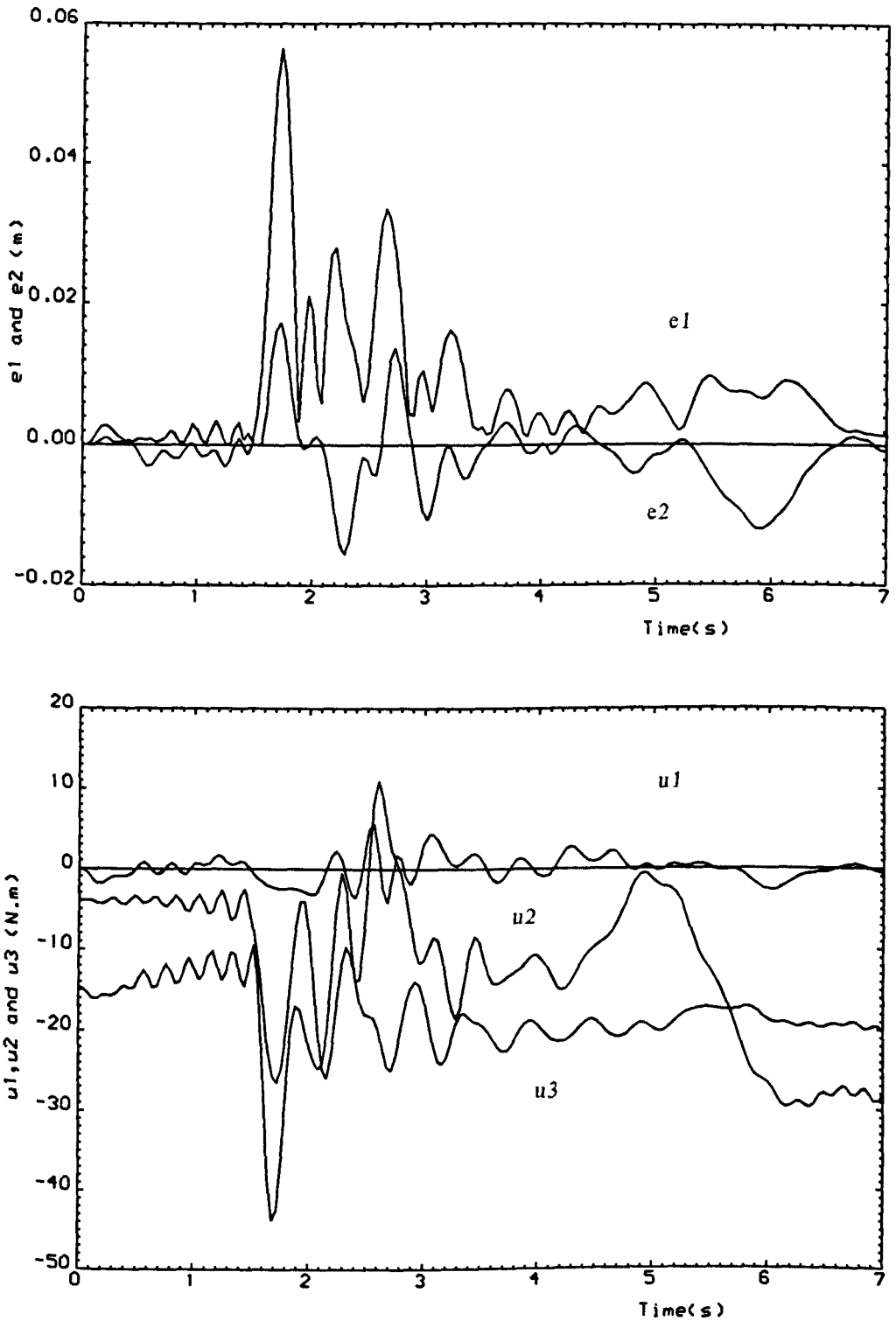

Figure 4.24: Time-domain behaviour of errors and torques for the genetically designed digital PID controller (p_m=0.98,T=0.04).

Chapter 5

DESIGN OF DIGITAL CONTROLLERS FOR ROBOTIC MANIPULATORS USING EVOLUTION STRATEGIES

5.1 INTRODUCTION

In Chapter 4, genetic algorithms were used for the design of robust digital trajectory-tracking PID controllers for robotic manipulators: such design was previously effected by the singular perturbation methods presented in Chapter 3. It was thus shown in Chapter 4 that genetic algorithms can be used to develop an effective tuning procedure for such controllers in the non-asymptotic case for which the sampling period is non-zero.

However, in the case of very complex robotic manipulators performing wide ranges of trajectory-tracking tasks, there is a practical need to accelerate the evolutionary processes involved in the genetic robustification methodologies. This need prompts the search for algorithms that are more powerful than genetic algorithms. In fact, the speed of convergence is an important measure of the power of a given evolutionary approach. This is because sufficiently rapid convergence ensures that the algorithm will at least produce a local optimum within reasonable time, which is a critical factor for any practical application. Indeed, *Back, Rudolph* and *Schwefel* (1993) have shown that evolution strategies are able to achieve a rate of

convergence which is claimed by *Voigt, Muhlenbein* and *Cvetkovic'* (1995) to be the best convergence rate achievable for evolutionary algorithms. Therefore, it is proposed in this chapter to use evolution strategies without recombination as an alternative to genetic algorithms for the design of digital trajectory-tracking PID controllers for robotic manipulators. Such evolution strategies are simpler than genetic algorithms (since crossover in the form of recombination is eliminated) and are therefore frequently less computationally burdensome than genetic algorithms. It is therefore relevant to explore the relative merits of these alternative evolutionary optimisation techniques in the context of digital control system design.

In fact, two different variants of evolution strategies are proposed. The first is the standard non-adaptive approach which consists of using fixed mutation probabilities during the evolutionary process. In contrast with this, the second approach is characterised by the presence of a self-adaptative mutation mechanism. Moreover, it is proposed in this chapter to use the $(\mu + \lambda)$ evolution strategies introduced by *Schewefel* (1977) (see Appendix A.2) which generate λ offspring from μ parents and then select the μ best individuals from the entire set of $\mu + \lambda$ individuals (parents and offspring). In the case of both the non-adaptive and adaptive evolution strategies, the evolutionary design processes are effected by incorporating the controller design equation (3.14) within the evolutionary strategies so as to optimise the various tuning parameters. In this way, the best trajectory-tracking behaviour is achieved in accordance with a chosen evaluation performance for the specified sequence of typical trajectory-tracking tasks

presented in Chapter 3 and 4 (which includes a sudden change of payload).

The optimal quadruple $\{\alpha, \delta, \sigma, \rho\}$ of design parameters for digital PID controllers thus obtained allows a direct comparison to be made in the non-asymptotic case between the performance of those controllers with the evolutionary parameters tuned by evolution strategies and those with parameters tuned by the singular perturbation methodologies of *Porter* and *Abidin* (1990a). In addition, it is shown that the evolution strategies --as in the case of genetic algorithms-- respect the robustness characteristics which need to be considered in the case of robotic manipulators. Indeed, the evolutionary design procedure makes implicit use of the robustness theorem and corollary presented in Chapter 3 by ensuring that the evolutionarily designed controllers remain stable during the specified sequence of tracking tasks.

This design approach using evolution strategies is a natural extension to robotic control problems of the previously obtained non-robotic results of *Porter* and *Zadeh* (1997) and *Porter* and *Merzougui* (1997).

5.2 EVOLUTIONARY DESIGN PROCEDURE

5.2.1 EVOLUTION STRATEGIES

In this evolutionary design process, $(\mu + \lambda)$-evolution strategies are used to

design robust digital-tracking PID controllers for complex non-linear and time-varying parameters robotic manipulators for non-asymptotic conditions in which realistic and practical finite sampling periods are considered. It is evident that, as in the genetic approach, it is intended that the solution of this problem provide an optimal quadruple, $\{\alpha,\sigma,\rho,\delta\}$, of controller design parameters which is dependent upon the given manipulator, the given task, and the measure of trajectory-tracking performance.

In order to apply $(\mu + \lambda)$-evolution strategies to solve this non-asymptotic design problem, it is necessary only to encode the quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller design parameters in accordance with a system of concatenated, multi-parameter, mapped fixed-point coding. In order to make an effective comparison between the performance of the different proposed evolutionary algorithms (see Chapter 7), the binary representation of the quadruple $\{\alpha,\sigma,\rho,\delta\}$ used in the genetic approach of Chapter 4 is retained.

Notice that, for the sake of consistency and easy comparison, it is also convenient to regard the minimum integral of the Euclidean norm of the trajectory-tracking error vector in Cartesian space as the ultimate design requirement. Thus, $(\mu + \lambda)$-evolution strategies are used to select the appropriate quadruple, $\{\alpha,\sigma,\rho,\delta\}$, of controller parameters in the design equations (4.7), (4.8), (4.11) and (4.12) such that

$$\Gamma = \int_0^{\tau} \|e(t)\| \, dt \tag{5.7}$$

is minimised, where τ is the duration of the tracking task, $e(t) \in \Re^l$ is the trajectory-tracking error vector in Cartesian space, and $\|.\|$ denotes the Euclidean norm.

In order to ensure objectivity in the comparison of the performance of different evolutionary algorithms (see Chapter 7), the evolution strategies are started by randomly generating an initial population of strings as in the case of genetic algorithms. Successive generations of strings are then obtained using the evolutionary operations of selection and mutation (Appendix A.5). In this case, unlike that of genetic algorithms where the elitist approach affected only the best and worst of each generation, the greater degree of elitism has the potential to affect the entire generation. Indeed, in the case of $(\mu + \lambda)$-evolution strategies, elitism consists in selecting the μ best individuals from the entire set of $\mu + \lambda$ individuals (parents and offspring).

5.2.2 NON-ADAPTIVE $(\mu + \lambda)$-EVOLUTION STRATEGIES

In the absence of an automated self-adaptive mutation mechanism, it is required in the case of non-adaptive $(\mu + \lambda)$-evolution strategies to proceed with the selection of fixed mutation probabilities by means of a trial-and-error

process. However, such a selection process provides an insight into the impact of fixed mutation probabilities on the behaviour of non-adaptive (μ + λ)-evolution strategies. Indeed, the choice of different values for the mutation probability, p_m, in the non-adaptive (μ + λ)-evolution strategy is motivated by the very nature and function of mutation. This mutation parameter has a significant effect on the overall performance of the recombination-free evolution strategies proposed in this chapter. Indeed, specialists in the field of evolution strategies such as *Back* (1997) emphasise that —due to the predominance of the mutation operation in recombination-free evolution strategies— varying the probability of mutation has a far more dramatic effect on the performance than in the case of genetic algorithms. In the latter algorithms, recombination is considered to be the most important search operator as advocated by *Holland* (1975),and *Goldberg* (1989). Indeed, those authors consider that recombination and selection are the minimum requirement for evolution to occur. It is also well-documented that those who emphasise the importance of recombination use mutation only as a dedicated background operator of small importance (*Holland* (1975)), which ensures only that the population consists of a diverse pool of individuals.

5.2.3 ADAPTIVE (μ + λ)-EVOLUTION STRATEGIES

The major quality of the (μ + λ) adaptive strategy is seen in its ability to incorporate a self-adaptive mutation mechanism in the evolution strategy, thus

avoiding the time-consuming task of manual tuning. Indeed, efficiency can be gained by changing the value of the mutation parameter during the operation of the algorithm by taking feedback from the current state of the search. It is initially convenient to describe this adaptive (μ + λ)-evolution strategy for the particular case in which μ = λ, i.e., when the number of parents equals the number of children. Then, the strategy proceeds with the selection of the fittest μ individuals which constitute the new generation of parents amongst the entire set of 2 x μ (parents + children) individuals. In order to understand the self-adaptive mutation mechanism, it can be initially described it in its simplest form for which μ = λ= 1 (*Porter* (1997)).

Thus, as proposed by *Porter* (1997), let χ_k be the n-dimensional binary vector representing the parental chromosome in the k-th generation (k = 0, 1, 2,...,v) of an adaptive (1 + 1)-evolution strategy. In addition, let $\beta_k \in [1,n]$ be the positive integer representing the associated current mutation rate. Moreover, let β_k also determine the interval, [- β_k, + β_k], of a uniform probability distribution function, p(β_k). Then, in order to generate χ_{k+1} and β_{k+1}, consider that mutation occurs such that

$$\chi_k \rightarrow \chi'_k \tag{5.8}$$

and

$$\beta_k \rightarrow \beta'_k \tag{5.9}$$

In (5.8), χ'_k is obtained by adding 1 to each of β_k randomly selected bits of χ_k; in (5.9), β'_k is obtained by firstly selecting an integer randomly from the interval $[-\beta_k, +\beta_k]$ as the value of $\Delta\beta_k$ and by then recalculating

$$\beta'_k = \beta_k + \Delta\beta_k \quad . \quad (5.10)$$

Next, consider the fitnesses

$$\Phi_k = \Phi(\chi_k) \quad (5.11)$$

and

$$\Phi'_k = \Phi(\chi'_k) \quad (5.12)$$

of the chromosomes χ_k and χ'_k. Then, if there is no increase in fitness so that

$$\Phi'_k \leq \Phi(\chi'_k) \quad , \quad (5.13)$$

choose

$$\chi_{k+1} = \chi_k \quad , \quad (5.14)$$

and

$$\beta_{k+1} = \beta'_k \quad . \quad (5.15)$$

Or else, if there is an increase in fitness so that

$$\Phi'_k > \Phi_k \quad , \quad (5.16)$$

choose

$$\chi_{k+1} = \chi'_k \quad (5.17)$$

and

$$\beta_{k+1} = \beta_k \quad . \quad (5.18)$$

In other words, adaptive evolution strategies are caused to evolve as follows: if the current mutation parameter does not lead to an increase in fitness, then it is changed to its mutated value but the current chromosome is retained; or else, if the current mutation parameter does lead to an increase in fitness, it is retained but the current chromosome is changed to its mutated value. Such an adaptive process can be readily extended to embrace the general case of

entire populations of chromosomes for which $\mu > 1$ and $\lambda \geq 1$.

In summary, the ($\mu + \lambda$)-evolution strategies use recombination-free evolution. This is characterised by a fixed mutation rate parameter strategy in the non-adaptive variant, and by a self-adaptive mutation rate parameters strategy in the adaptive version. In addition, a pronounced degree of elitism is present in these strategies.

5.3 ILLUSTRATIVE EXAMPLE

5.3.1 SYSTEM DESCRIPTION

The use of (μ + λ)-evolution strategies to design fast-sampling digital PID controllers for trajectory-tracking systems, which are required to exhibit robust performance in the face of the time-varying plant parameters inherent in robotic manipulators, can be conveniently illustrated by reference to the three-degree-of-freedom robotic manipulator previously investigated by *Petropoulakis* (1986).

This manipulator was used by *Porter* and *Abidin* (1990b) to illustrate the asymptotic robustness results presented in Chapter 3 for fast-sampling error-actuated PID controllers for completely irregular linear multivariable plants. The same robotic manipulator was also used to assess the performance of the genetically designed fast-sampling error-actuated PID controllers presented in

Chapter 4. The need to make a valid comparison of the performance of the different proposed evolutionary algorithms motivates the choice of this same illustrative environment (robotic manipulator, task and performance measure). The robotic manipulator in question is governed on T = [0, +∞] by state and output equations of the respective forms [*Petropoulakis* (1986)]

$$M(\theta)\ddot{\theta}+c(\theta,\dot{\theta})+g(\theta)=u \quad , \tag{5.19a}$$

and

$$y=f(\theta) \quad . \tag{5.19b}$$

The numerical values of the inertial and kinematic parameters of the typical three-degree-of-freedom robotic manipulator under consideration are those given by *Petropoulakis* (1986) and presented in Chapter 3. Notice that the highly non-linear and coupled dynamics of this type of robotic manipulator make it a difficult candidate for accurate trajectory control and, therefore, a suitable vehicle for designing error-actuated digital PID controllers using evolution strategies.

The evolutionarily designed controllers are used while the end-effector of the robotic manipulator is caused to track the straight-line trajectories defined in Chapter 3. In the same manner as in the illustrative example of Chapter 3, the robotic manipulator grasps an additional payload of 5 Kg after the initial transition **I** → **II** (see section 3.5) in order to test the robustness of such

evolutionarily designed controllers. Indeed, the sudden change in the dynamics of the manipulator resulting from the introduction of the payload requires that such controllers embody the robustness characteristics needed in the face of such severe conditions. In fact, the acquired robustness characteristic ensure that the evolutionarily designed trajectory-tracking system remains stable for the duration of the entire task.

The structure of the computational implementation of (μ + λ)-evolution strategies in the case of the trajectory-tracking system is shown in Figure 5.1. In this way, the evolutionary design of the controller for this selected trajectory-tracking task can be readily effected by minimising the cost function

$$\Gamma = \int_0^{\tau} \|e(t)\| \; dt \quad , \tag{5.20}$$

thus determining the associated optimal quadruple $\{\alpha, \sigma, \rho, \delta\}$ of controller parameters.

5.3.2 NON-ADAPTIVE EVOLUTION STRATEGY

In the present case, a non-adaptive (30 + 30)-evolution strategy was used with a 16 bit representation for each of the four controller parameters and evolution occurred over 50 generations.

In order to study the effect of the mutation probability, p_m, on the performance of the non-adaptive (30 + 30)-evolution strategy, the three mutation probabilities, $p_m = 0.02$, $p_m = 0.001$ and $p_m = 0.9$ were selected. The entire set of results thus obtained is presented in Tables 5.1, 5.2 and 5.3 from which it emerges that the controllers designed using the mutation probability $p_m = 0.02$ outperform those designed using the other two mutation probabilities. It is also clear from these tables that this outperforming trend associated with the mutation probability $p_m = 0.02$ holds for all three selected sampling times T = 0.01s, T = 0.02s, and T = 0.04s. Indeed, the resulting optimal values of the cost function shown in Table 5.1 indicate that the evolution strategy based on the mutation probability $p_m = 0.02$ leads to significant reductions of the optimal values of the cost function. These reductions range between 12% and 55%, if the optimal values associated with the mutation probability $p_m = 0.02$ are taken as a bench-mark for comparison with those optimal values obtained using the mutation probability $p_m = 0.001$, presented in Table 5.2. Similarly, these bench-mark optimal values of Table 5.1 corresponding to the mutation probability $p_m = 0.02$ reveal a reduction in the range of 3% to 17% compared to those optimal cost function values associated with the evolution strategy based on the mutation probability $p_m = 0.9$ shown in Table 5.3.

In the time-domain, the superiority of the performance of the design with $p_m = 0.02$ translates into a smaller value of the peak error associated with the sudden introduction of the payload (see Figures 5.2.a, 5.2.b and 5.2.c). In addition, this design provides smaller magnitudes of the error throughout the

entire duration of the task (see Figs 5.3.a, 5.3.b, 5.3.c, 5.4.a, 5.4.b and 5.4.c).

	Cost Function	Controller Parameters			
Sampling period T	Γ	α	σ	ρ	δ
0.01	0.00131	0.1246	0.8999	22.9080	0.0194
0.02	0.00904	0.0727	0.8981	12.9096	0.0383
0.04	0.06103	0.0510	0.8993	7.7487	0.0997

Table **5.1** : Evolutionarily designed (non-adaptive ES, p_m = **0.02**) controllers for different sampling times

	Cost Function	Controller Parameters			
Sampling period T	Γ	α	σ	ρ	δ
0.01	0.00287	0.2053	0.6124	8.4140	0.0102
0.02	0.01103	0.1182	0.8285	13.2132	0.0541
0.04	0.06953	0.0532	0.8375	9.5388	0.1569

Table **5.2** : Evolutionarily designed (non-adaptive ES, p_m = **0.001**) controllers for different sampling times

	Cost Function	Controllers Parameters			
Sampling period T	Γ	α	σ	ρ	δ
0.01	0.00153	0.1566	0.8573	15.8305	0.0163
0.02	0.01088	0.0877	0.8688	9.8507	0.0431
0.04	0.06330	0.0816	0.8963	5.1098	0.0599

Table **5.3** : Evolutionarily designed (non-adaptive ES, p_m = **0.9**) controllers for different sampling times

It is clear from the results of Table 5.1, 5.2 and 5.3 that all three proposed non-adaptive evolution strategies result in an evident improvement in the trajectory-tracking performance when compared with that obtained from the singular perturbation design. Indeed, the corresponding value of the cost function Γ = 8.64×10^{-3} in the latter case is 3 to 6 times larger than that obtained with non-adaptive evolution strategies. In addition, it is important to notice that the non-adaptive evolution strategy corresponding to the mutation probability p_m = 0.02 shows an improvement in the optimal values of the cost function varying between 3% and 15% compared with their best genetically designed counterpart presented in Table 4.2.

The rate of evolution of non-adaptive evolution strategies is discussed later in this chapter, and compared with that of the adaptive evolution strategies.

5.3.2 ADAPTIVE EVOLUTION STRATEGY

In a similar way , the adaptive (30 + 30)-evolution strategy retains the 16 bit representation of each the four controller parameters over 50 generations. However, in the adaptive case the user is not required to specify mutation probabilities *a priori*, since these are automatically generated by the self-adaptive mutation mechanism incorporated in the evolution strategy.

This incorporation of self-adaptive mutation generates the results presented in Table 5.4. These results indicate that the adaptive evolution strategy produces the best performance so far. Indeed, in every run and for all three proposed sampling times T = 0.01s, T = 0.02s and T = 0.04s, these adaptively generated designs outperform their non-adaptively generated counterparts.

	Cost function	**Controller parameters**			
Sampling period T	Γ	α	σ	ρ	δ
0.01	0.00129	0.1341	0.8998	22.9808	0.0187
0.02	0.00894	0.0868	0.8981	11.3849	0.0335
0.04	0.06091	0.0501.	0.8999	8.5902	0.1161

Table **5.4** : Evolutionarily designed (**adaptive** ES) controllers for different sampling times

5.3.3 NON-ADAPTIVE VERSUS ADAPTIVE EVOLUTION STRATEGIES

It is possible to argue that the improvement in the optimal values of the cost function, Γ, resulting from the incorporation of self-adaptive mutation is quite insignificant when compared to their best non-adaptive counterparts. Indeed, Table 5.5 confirms this claim. In the same way, the time-domain behaviour of the errors and torques for the adaptive designs depicted in Figures 5.5.a, 5.5.b and 5.5.c are practically identical to their best non-adaptive counterparts depicted in Figures 5.2.a, 5.2.b and 5.2.c. However, this claim is moderated by the fact that the adaptive evolution strategies did not require any tedious and time-consuming selection of optimal mutation parameters in order to attain this level of performance.

In the case of the non-adaptive evolution strategies, the associated best-of-generation and average-generation values of the cost function, Γ, for the mutation probabilities $p_m = 0.02$, $p_m = 0.001$, and $p_m = 0.9$ are shown in Figures 5.6.a, 5.6.b, 5.6.c, 5.7.a, 5.7.b, 5.7.c, 5.8.a, 5.8.b and 5.8.c, over 50 generations. Similarly, in the case of the adaptive evolution strategies, the associated best-of-generation and average-generation values of the cost function, Γ, are depicted in Figures 5.9.a, 5.9.b, 5.9.c. However, the initial study of these plots does not provide conclusive evidence regarding the rate of evolution of the proposed evolution strategies; But a deeper study produces the comprehensive comparative Table 5.5, from which it is possible to derive

conclusions regarding the speeds of the different proposed evolution strategies. Since the best performances were attained using adaptive evolution strategies, the corresponding optimal values of the performance measure, Γ, are selected as benchmark values. Then, two characteristics are evaluated to assess the performance of the proposed evolution strategies. Thus, it is firstly considered that the number of generations taken for the best cost function of a particular evolution strategy to attain 10% of its final optimal value is an indication of the speed of convergence of the evolution strategy.

In addition, as an indication of the quality of this convergence, the final optimal value obtained by each evolution strategy is measured as a percentage relative to the optimal values attained by the bench-mark adaptive evolution strategy. It is clear from Table 5.5 that the best performances are those associated with the adaptive evolution strategies, and that the worst are those associated with the non-adaptive strategy with $p_m = 0.001$.

It is also clear from the different values of the controller parameters shown in Table 5.1, 5.2, 5.3, and 5.4 that, as predicted by the theory of fast-sampling digital PID controllers proposed by *Porter* and *Abidin* (1990b), the optimal values of ρ reduce as the sampling time increases. It is also clear from Figure 5.10.a, 5.10.b and 5.10.c that the eigenvalues of the perturbation matrix $(CAB)(\tilde{C}\tilde{A}\tilde{B})^{-1}$ corresponding to the adaptively designed digital PID controllers always lie on the interval $(0, (1+\alpha)/\sigma)$ permitted by the robustness corollary

[*Porter* and *Abidin* (1990b)] presented in Chapter 3. This fact indicates that these adaptively designed controllers maintain local stability during the sequence of tracking tasks for the selected sampling periods T = 0.01s, T = 0.02s, and T = 0.04s.

	Number of generations **(% difference from bench-mark)**		
Type of Evolution Strategy	T = 0.01	T = 0.02	T = 0.04
Adaptive ES	13 **(0%)**	4 **(0%)**	6 **(0%)**
Non-adaptive ES with p_m = 0.001	32 **(122.5%)**	29 **(23.4%)**	NA **(14.2%)**
Non-adaptive ES with p_m = 0.02	20 **(1.6%)**	8 **(1.1%)**	7 **(0.2%)**
Non-adaptive ES with p_m = 0.9	1 **(18.6%)**	NA **(21.7%)**	22 **(3.9%)**

Table **5.5**: Number of generations required for the algorithm to reach within the 10% of the final optimal value and percentage of error from the bench-mark adaptive optimal values

This design procedure for error-actuated digital PID controllers using $(\mu + \lambda)$-evolution strategies represents an improvement on the genetic procedure presented in Chapter 4. This improvement resides in the fact that the proposed recombination free $(\mu + \lambda)$-evolution strategies are much simpler to implement and yet they perform as well as, or better than, genetic algorithms.

5.4 CONCLUSION

It has been shown in this chapter that evolution strategies in both their non-adaptive and adaptive variants can be conveniently used to tune the parameters of fast-sampling digital PID controllers. These evolutionary procedures have been illustrated by the design of a trajectory-tracking system for the three-degree-of-freedom robotic manipulator for which a digital PID controller was previously designed using both the singular perturbation technique presented in Chapter 3 and the genetic technique presented in Chapter 4. These $(\mu + \lambda)$-evolution strategies represent a simplification and a refinement of genetic algorithms, and produce an evident improvement in the trajectory-tracking performance of the digital PID controllers.

In Chapter 6, these strategies are used in conjunction with an extended class of measures of trajectory-tracking performance.

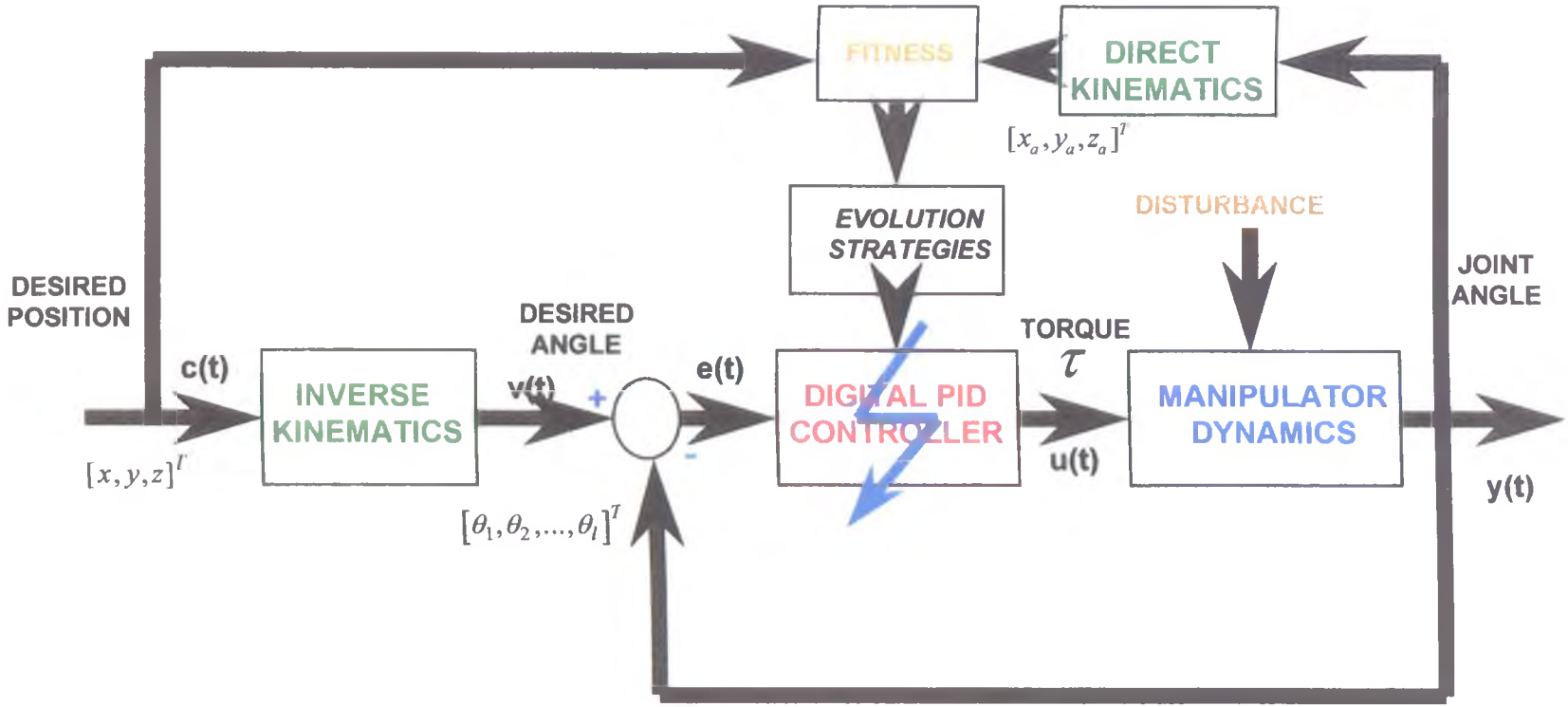

Figure 5.1: Computational implementation of trajectory-tracking system and its associated evolution strategies tuning mechanism

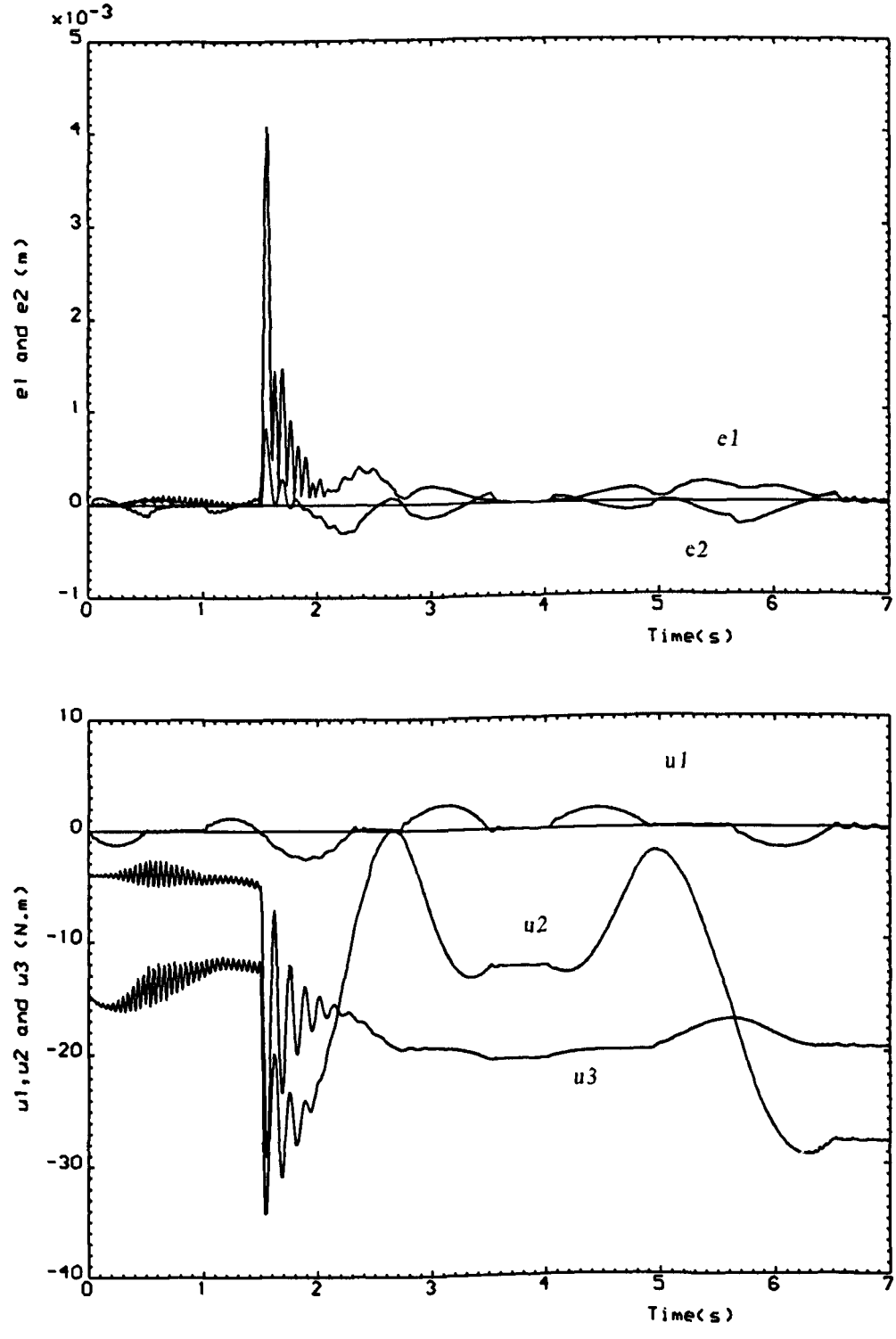

Figure 5.2.a: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.02,T=0.01).

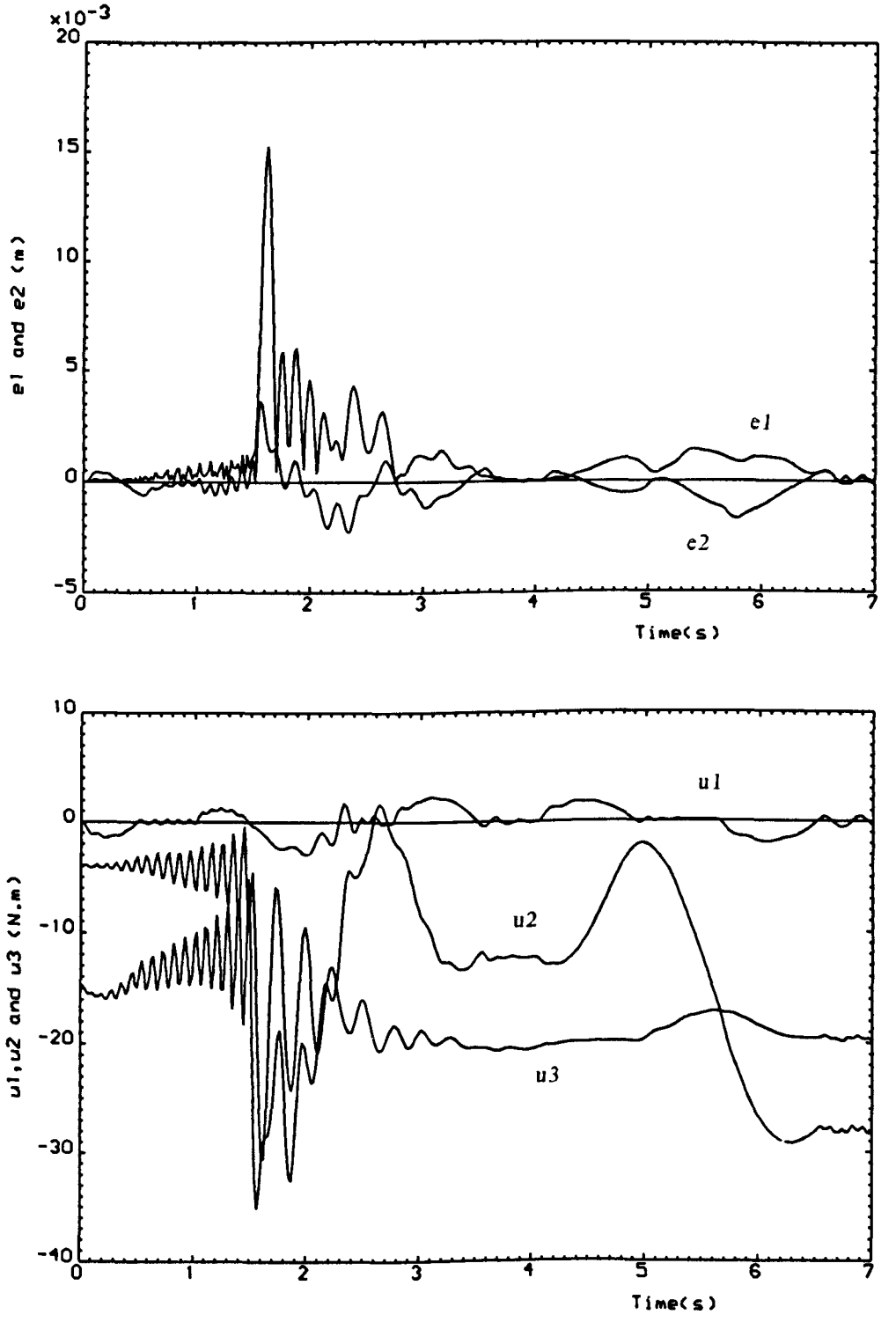

Figure 5.2.b: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.02T=0.02).

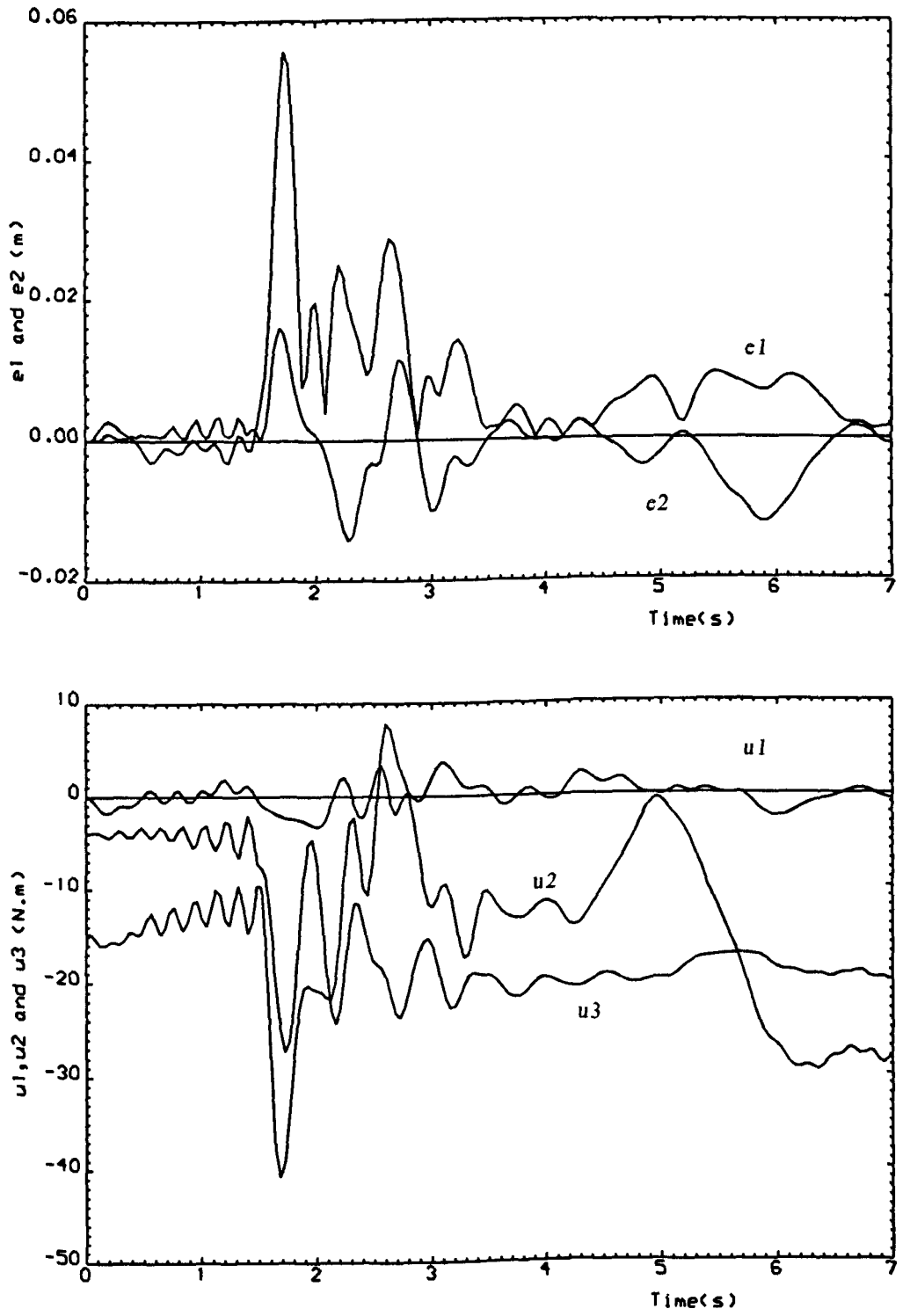

Figure 5.2.c: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.02,T=0.04).

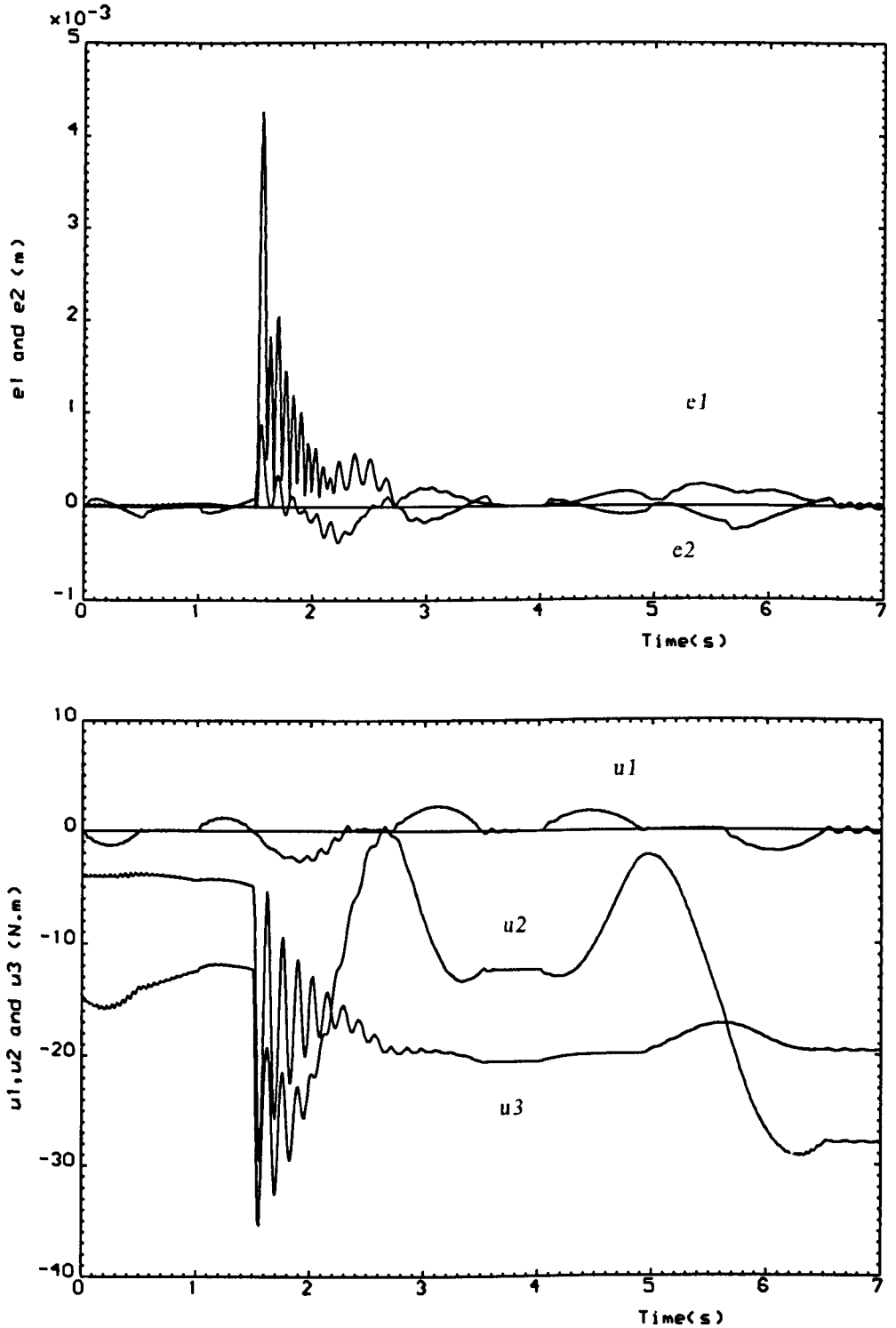

Figure 5.3.a: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.001,T=0.01).

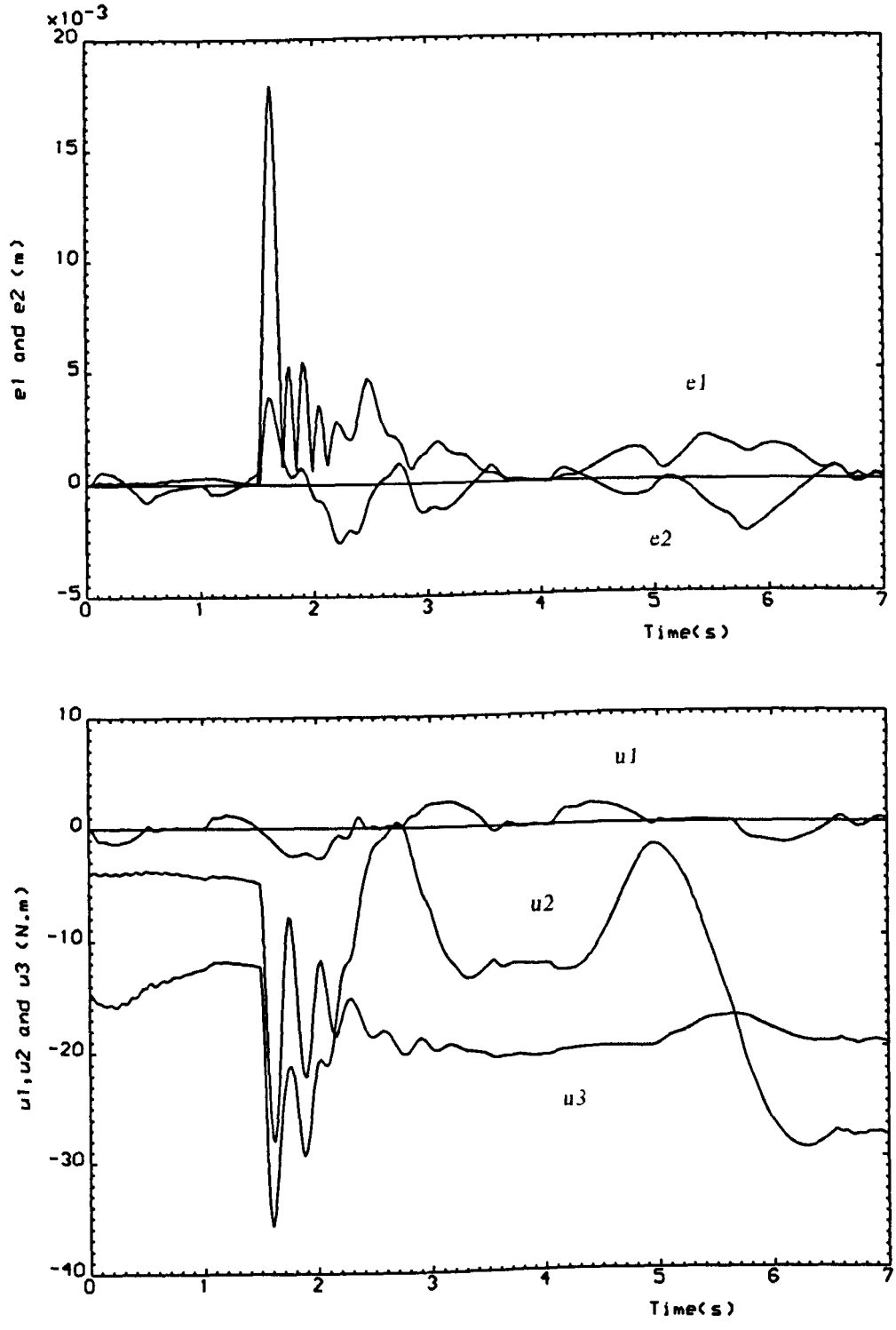

Figure 5.3.b: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.001,T=0.02).

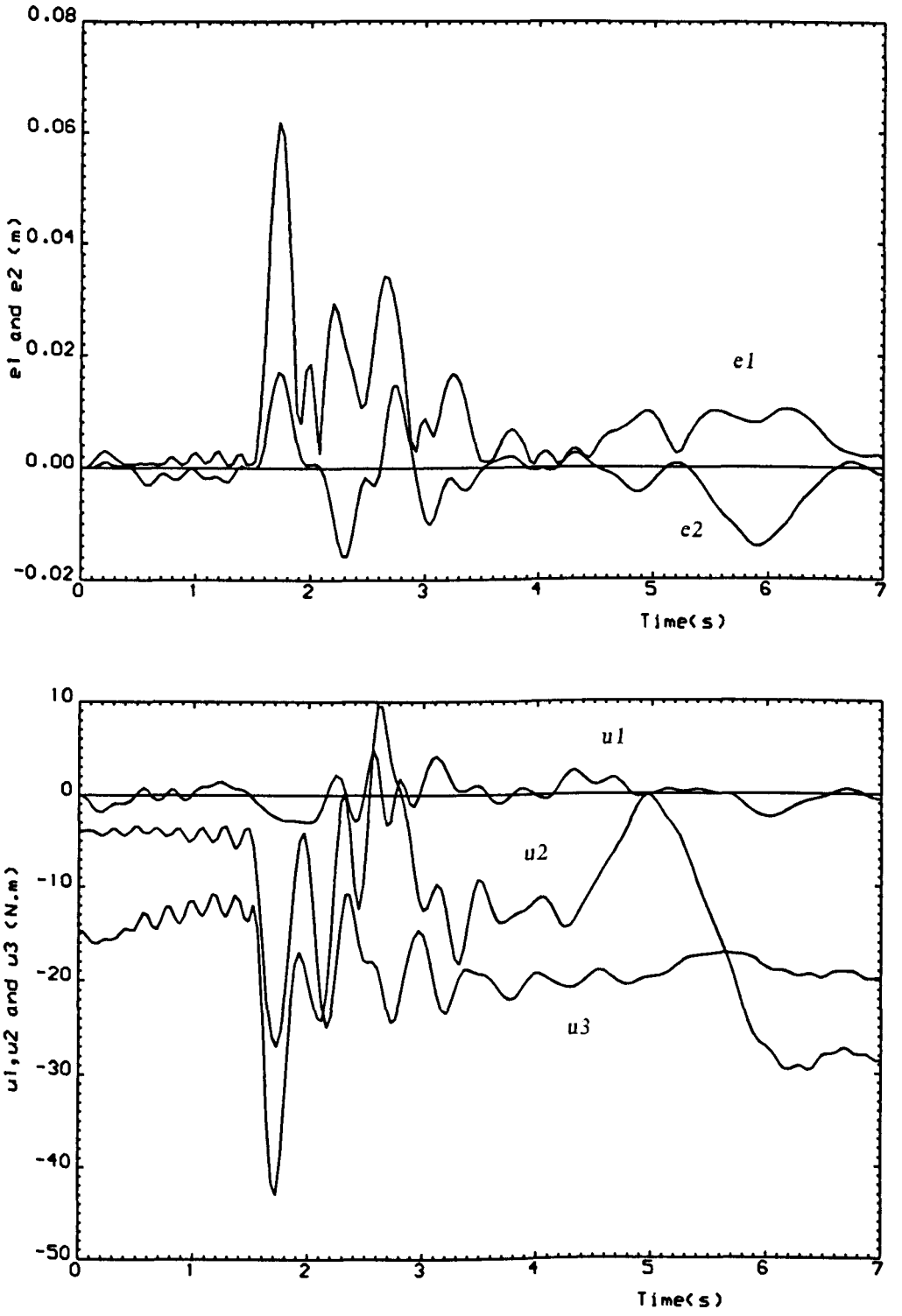

Figure 5.3.c: Time-domain behaviour of errors & torques for non-adaptive ES design of digital PID controller (P_m=0.001,T=0.04).

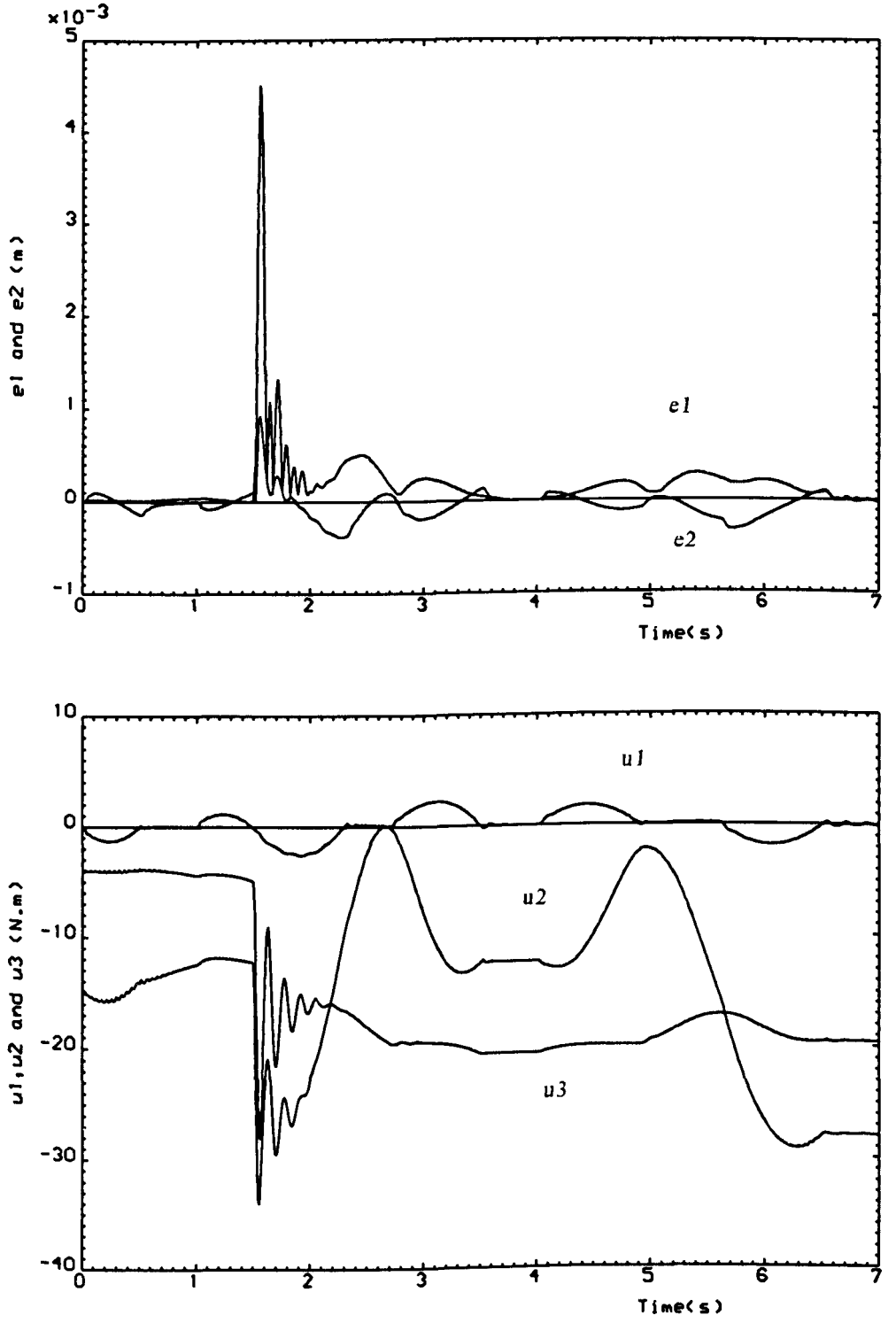

Figure 5.4.a: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.9,T=0.01).

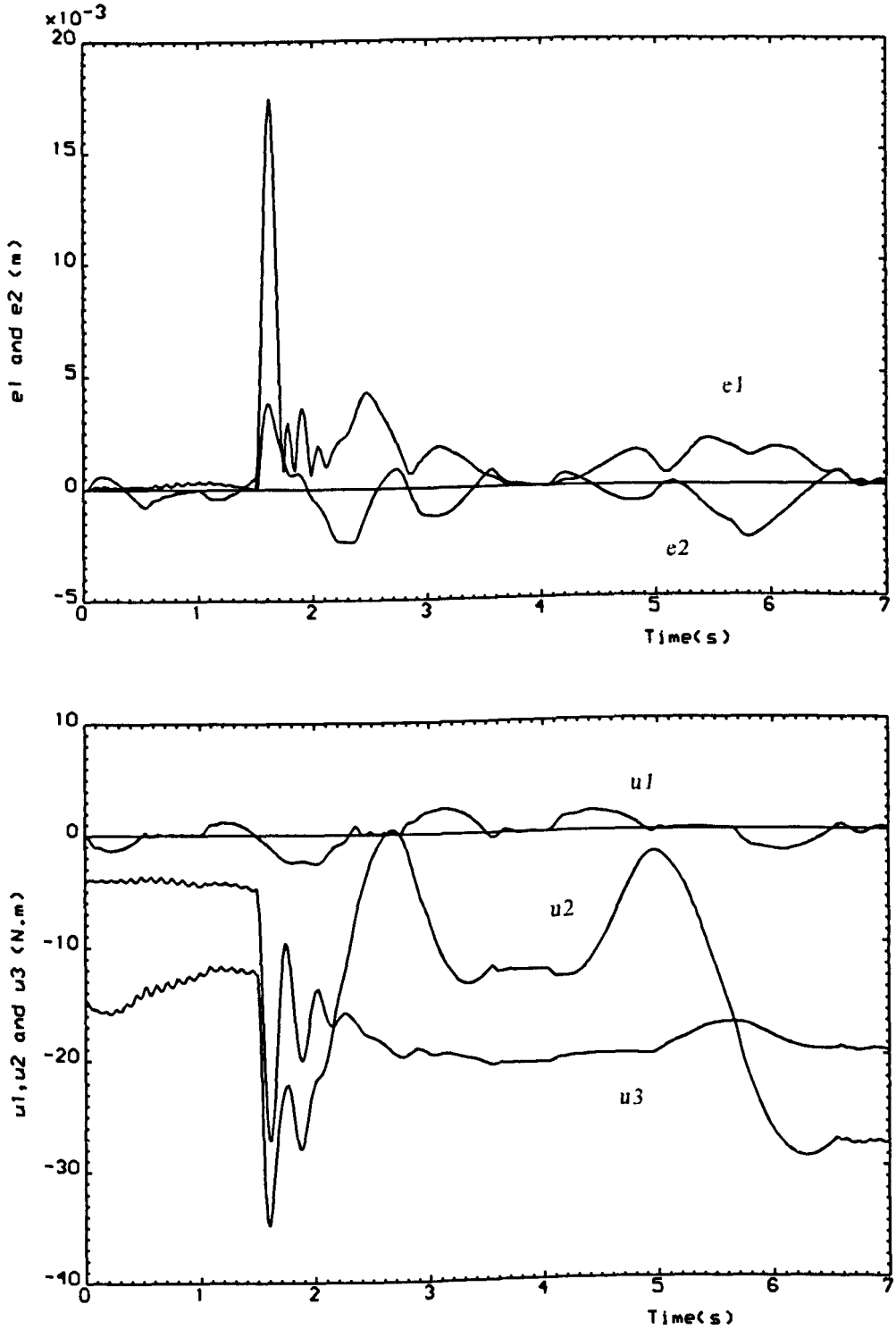

Figure 5.4.b: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.9,T=0.02).

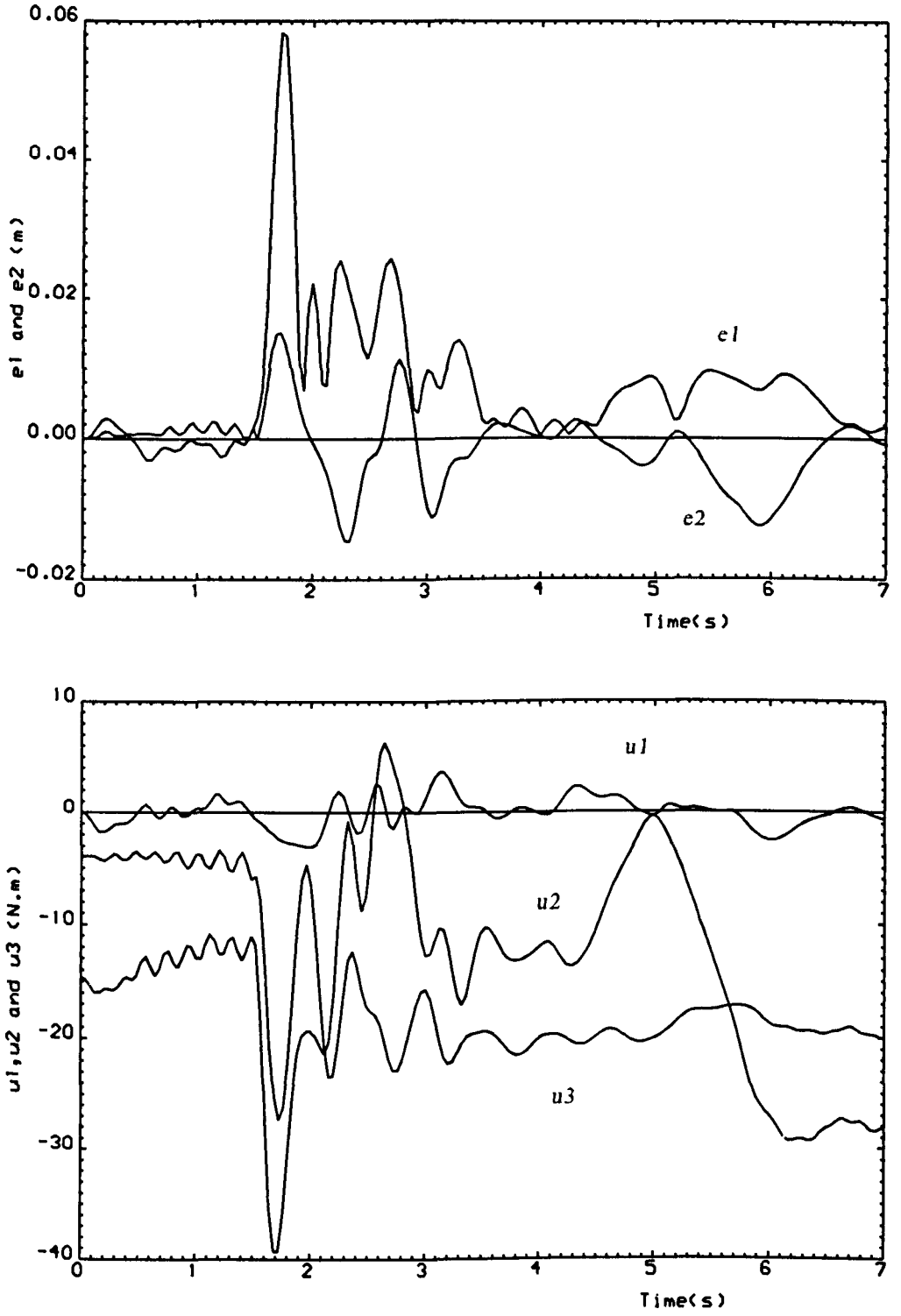

Figure 5.4.c: Time-domain behaviour of errors & torques for non-adaptive ES design of a digital PID controller (P_m=0.9,T=0.04).

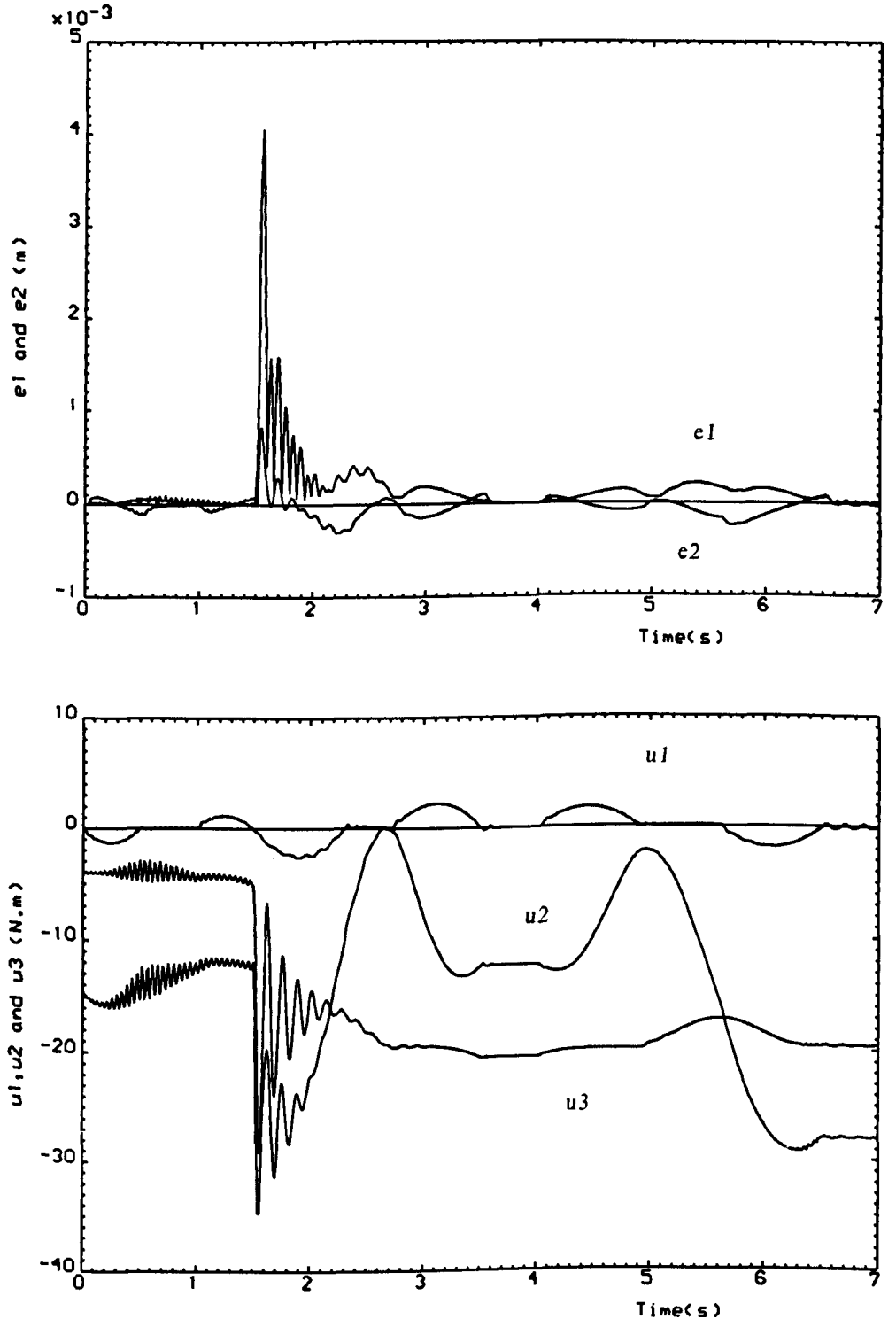

Figure 5.5.a: Time-domain behaviour of errors & torques for adaptive ES design of a digital PID controller (T=0.01).

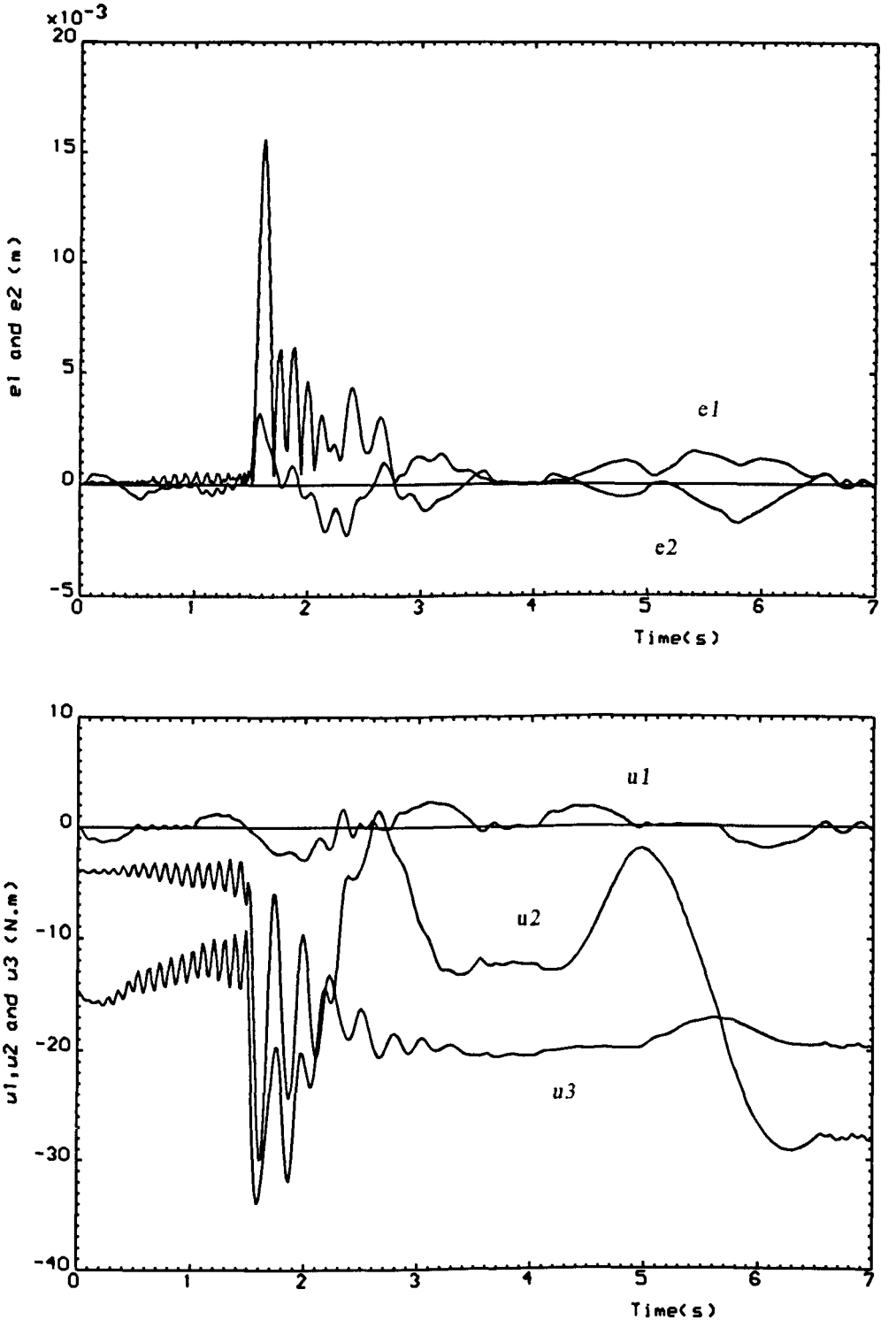

Figure 5.5.b: Time-domain behaviour of errors & torques for adaptive ES design of a digital PID controller (T=0.02).

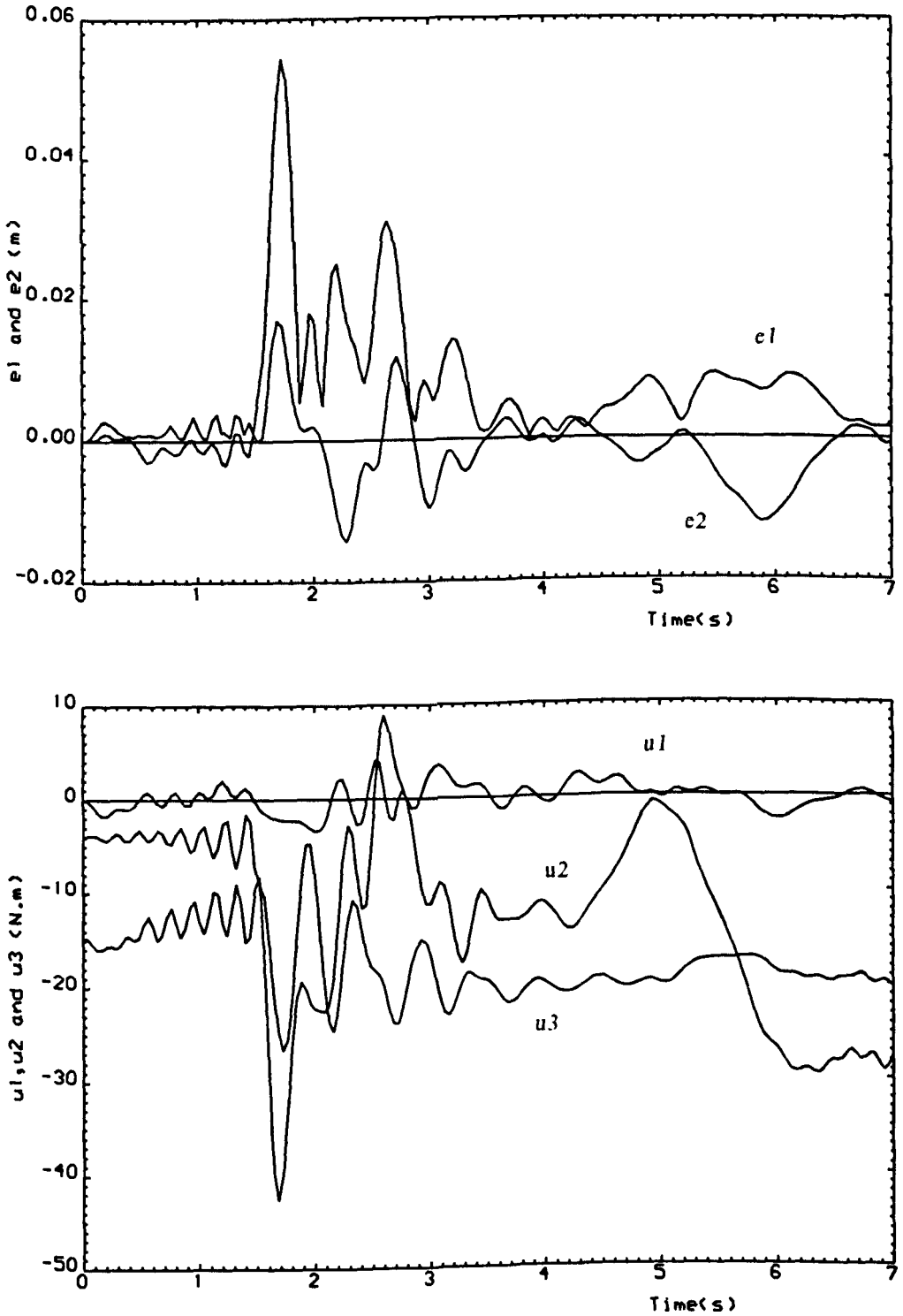

Figure 5.5.c: Time-domain behaviour of errors & torques for adaptive ES design of a digital PID controller (T=0.04).

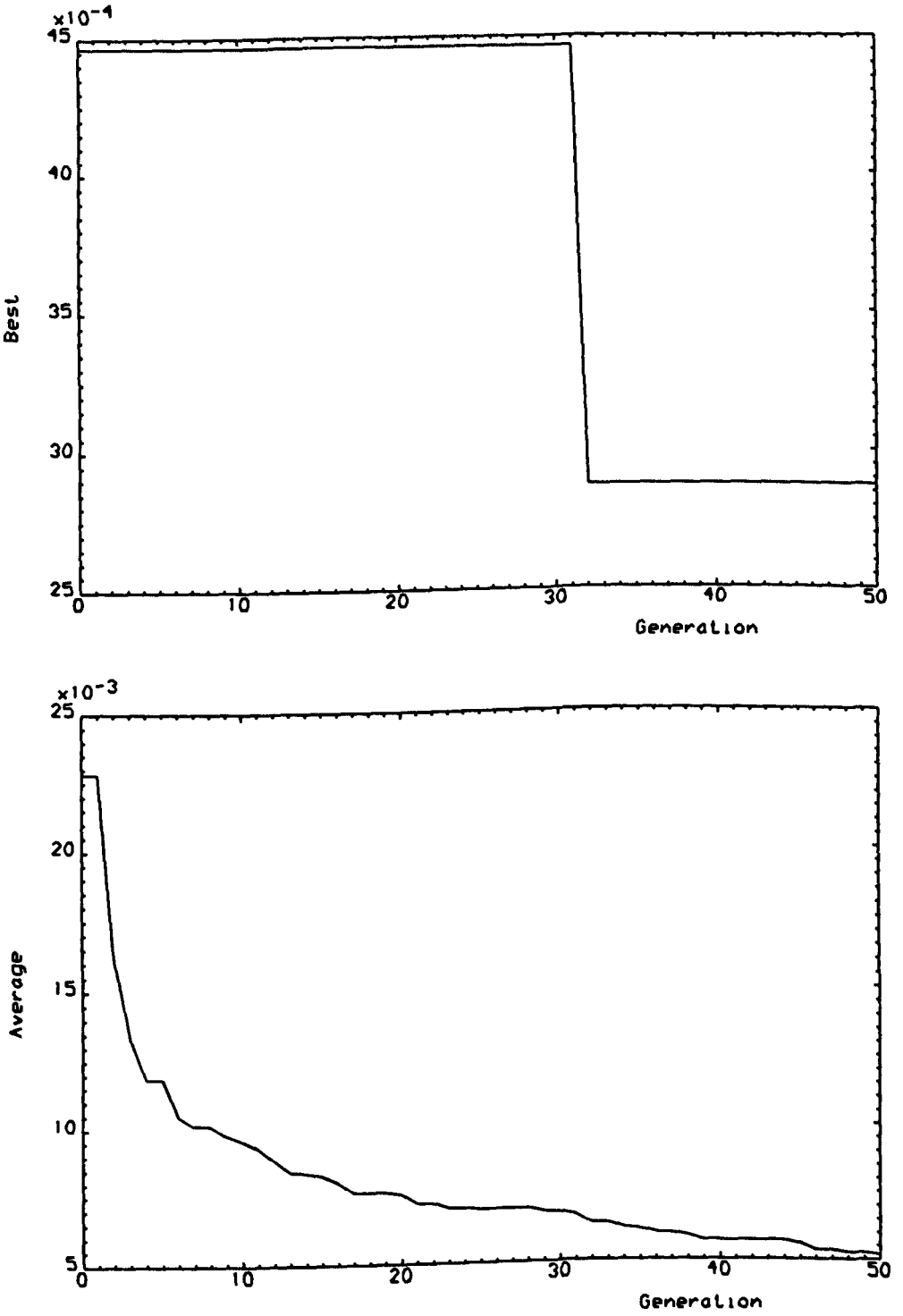

Figure 5.6.a: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.001, T = 0.01).

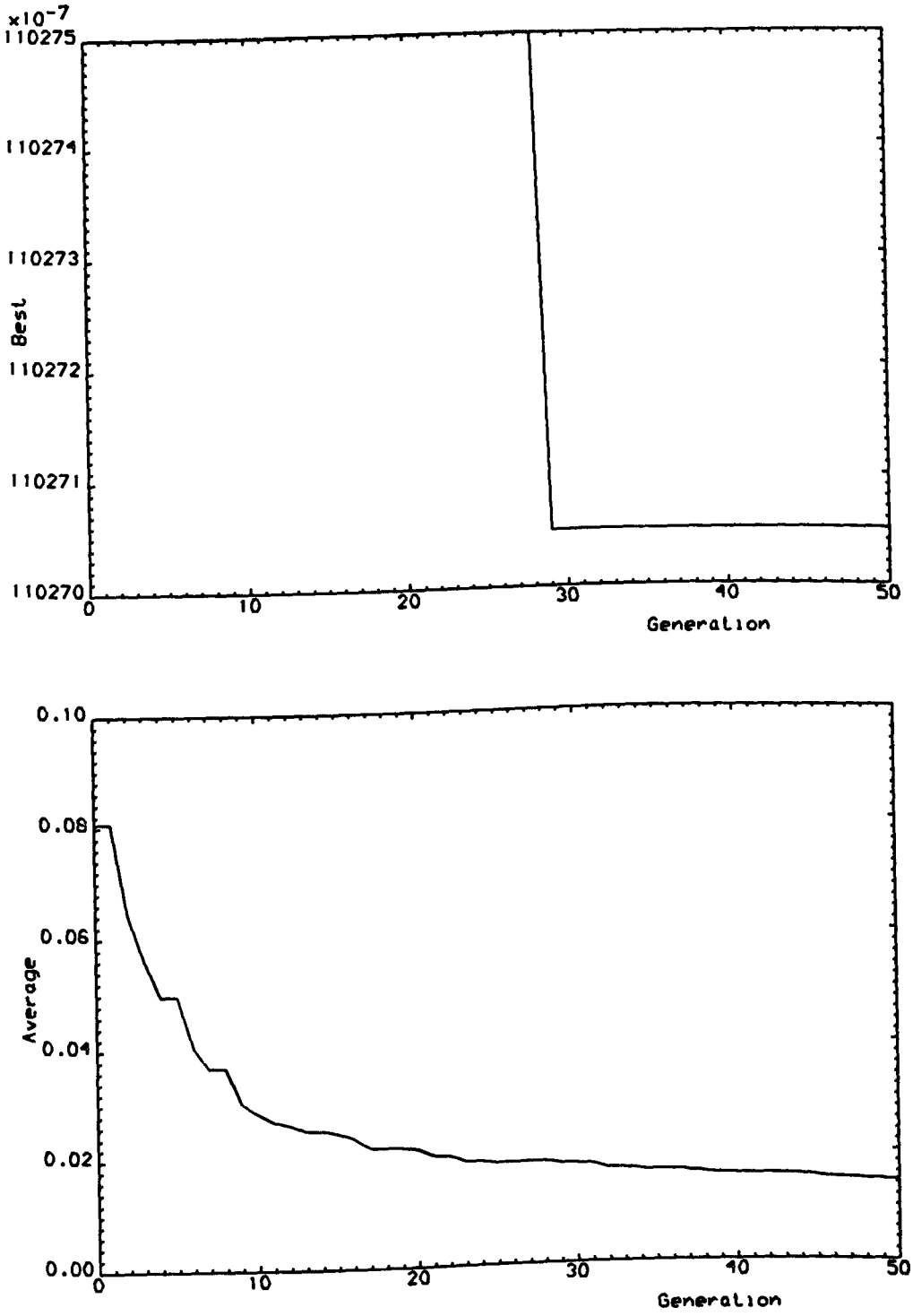

Figure 5.6.b: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.001, T = 0.02).

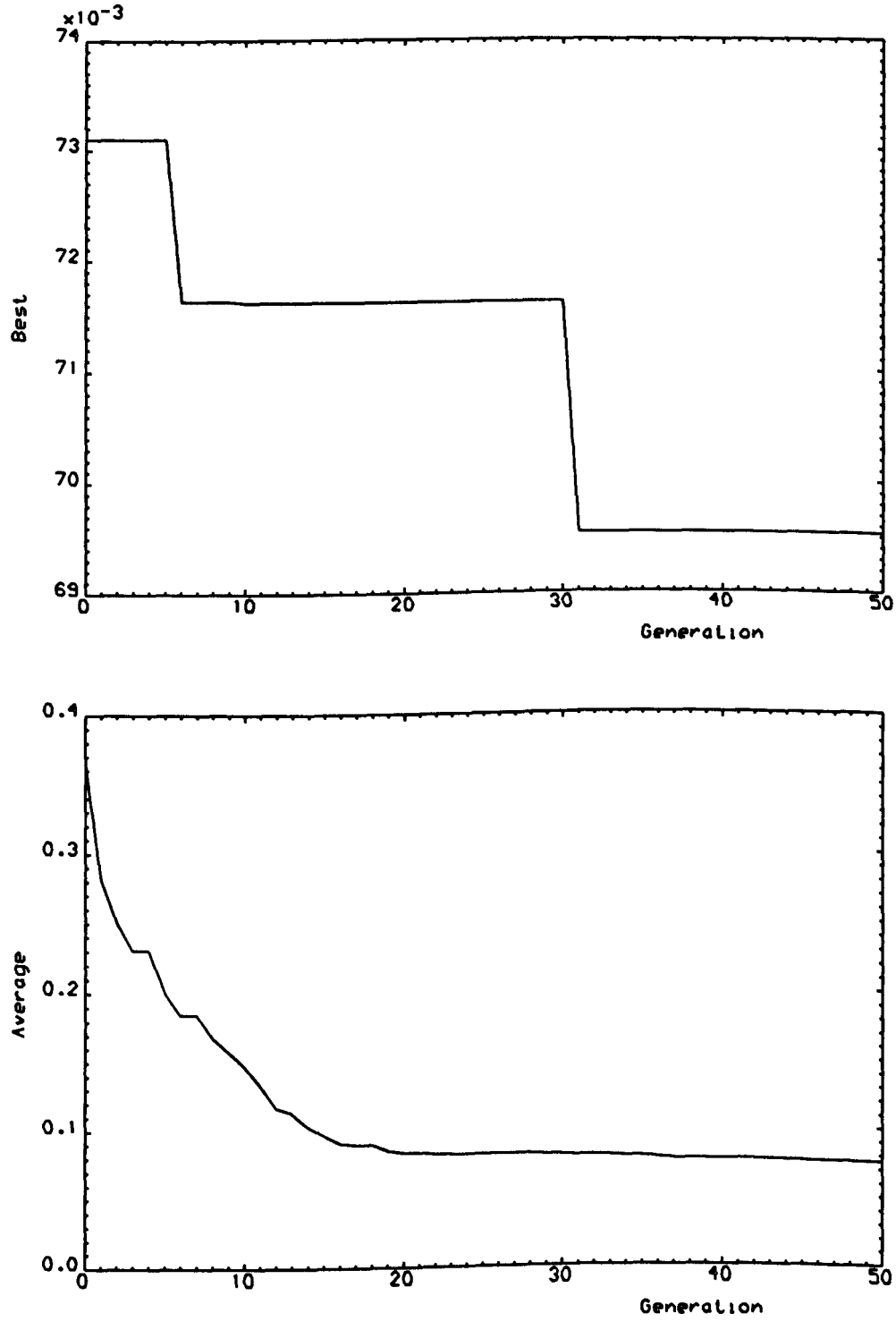

Figure 5.6.c: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.001, T = 0.04).

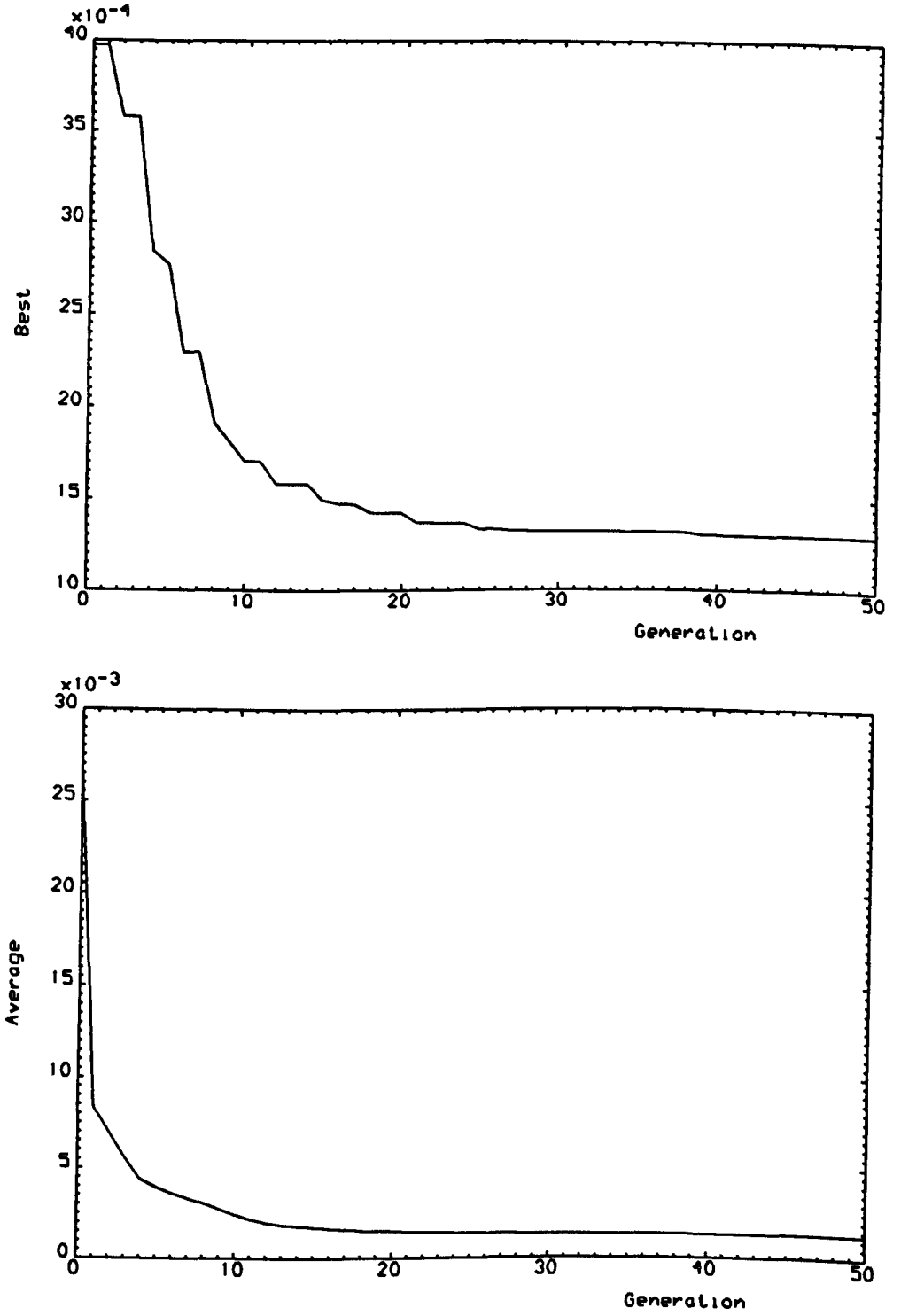

Figure 5.7.a: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.02, T = 0.01).

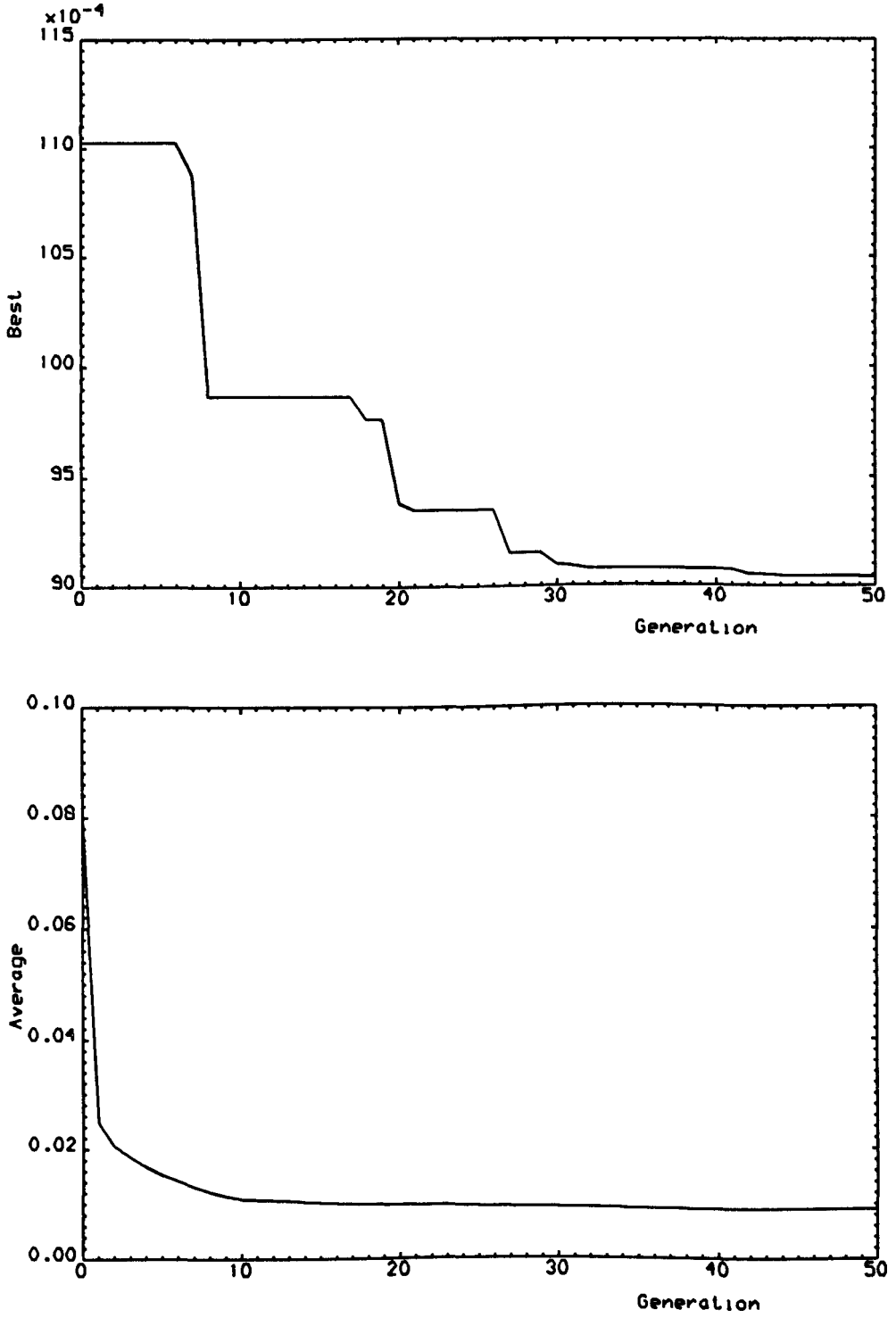

Figure 5.7.b: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.02, T = 0.02).

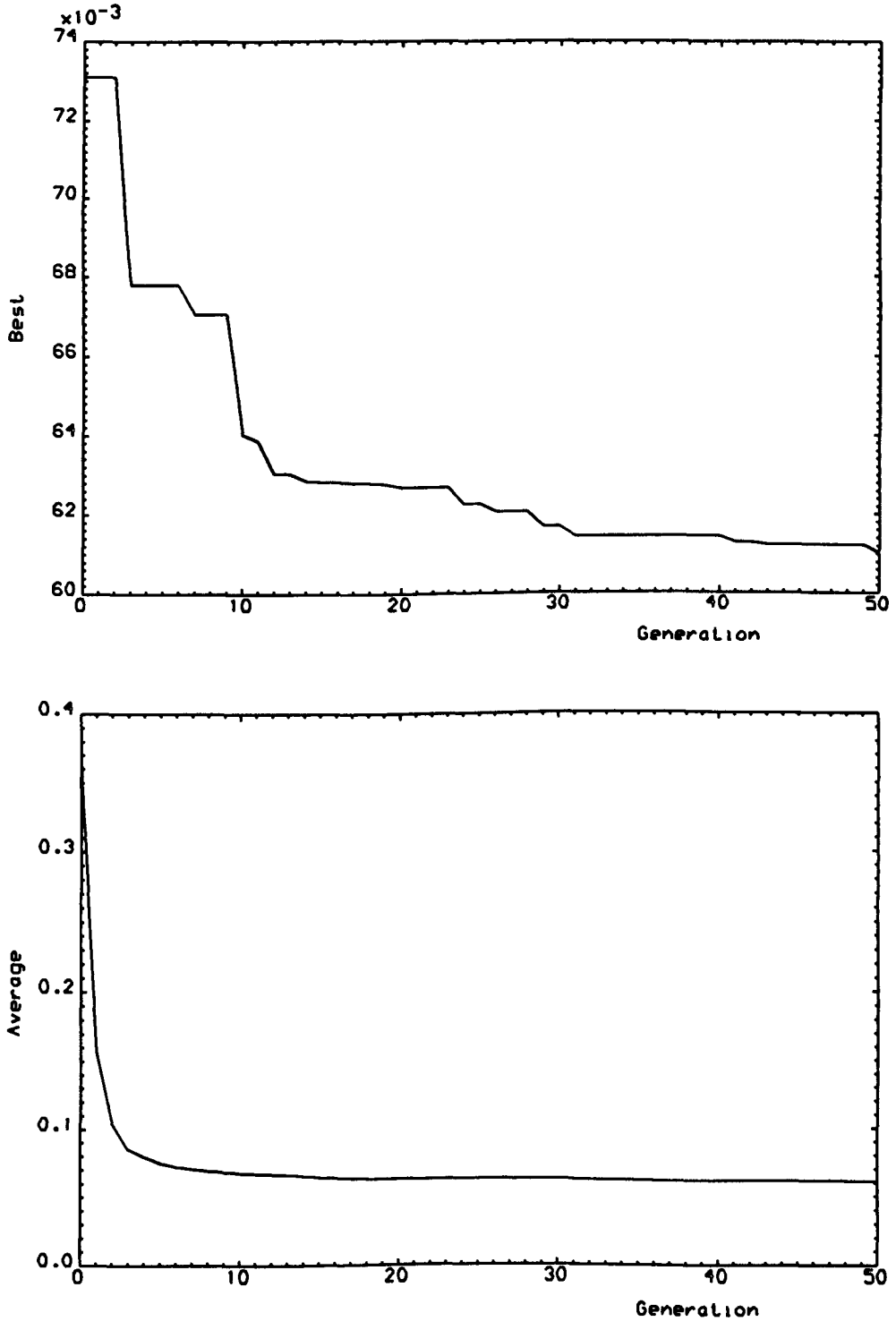

Figure 5.7.c: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.02, T = 0.04).

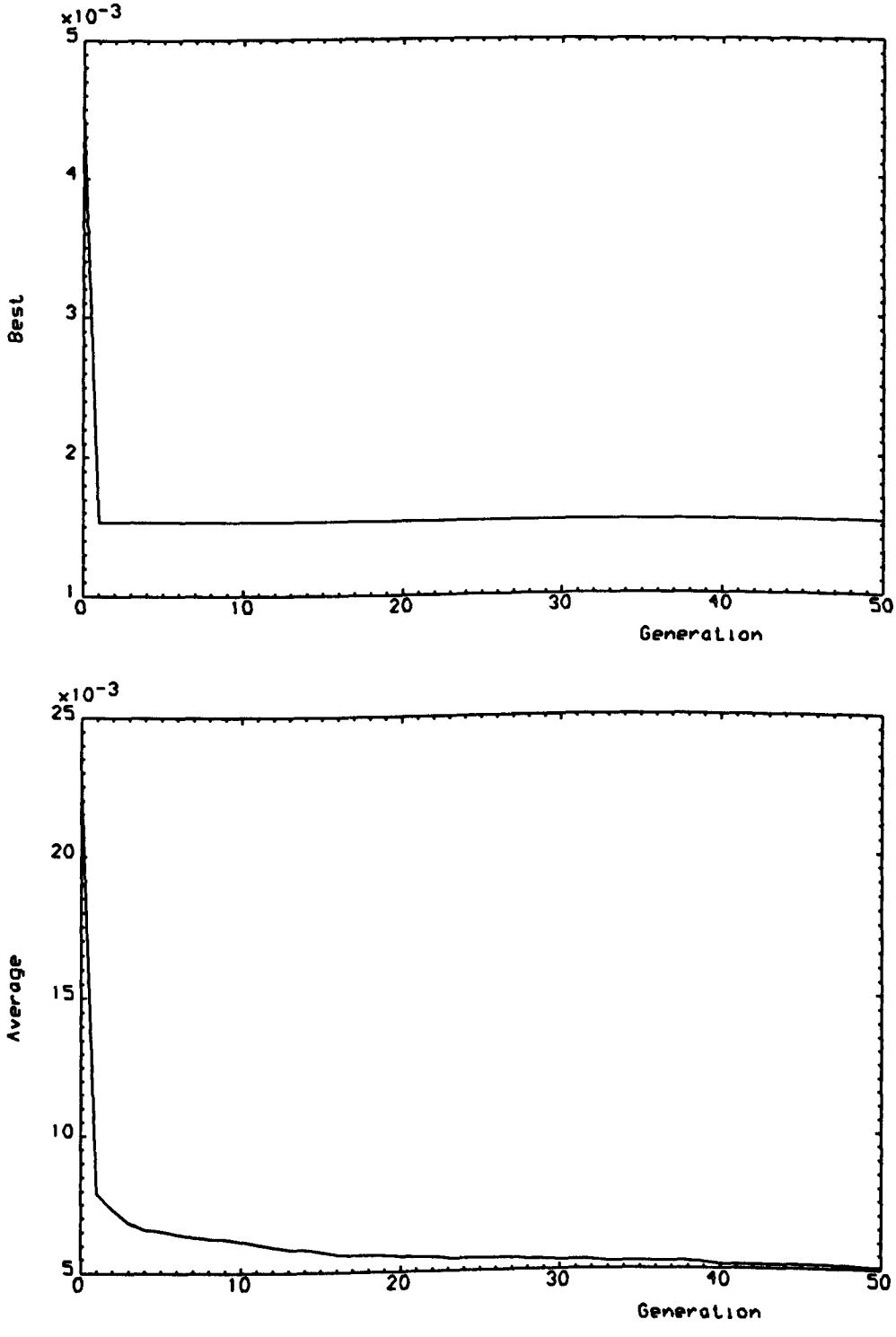

Figure 5.8.a: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.9, T = 0.01).

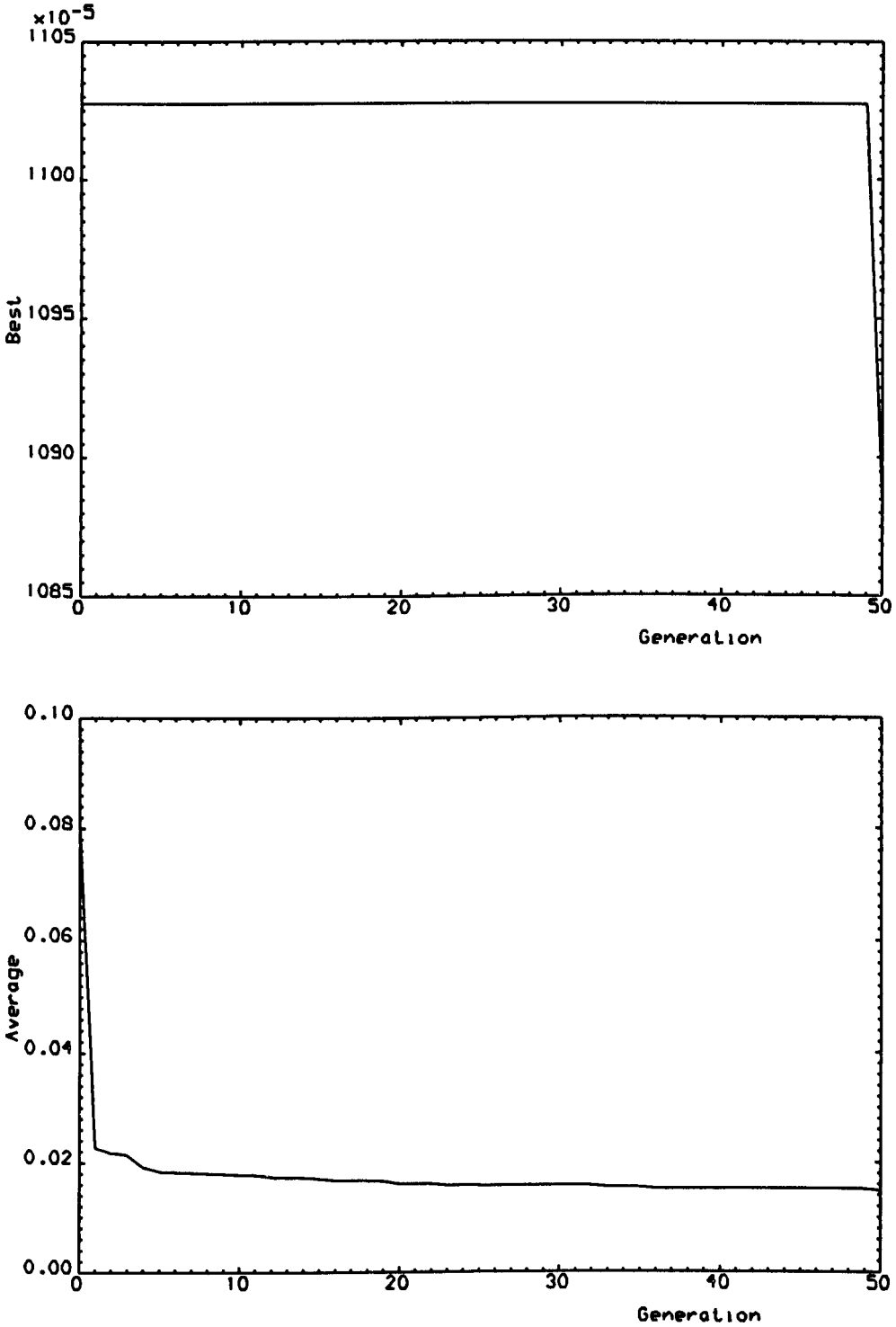

Figure 5.8.b: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.9, T = 0.02).

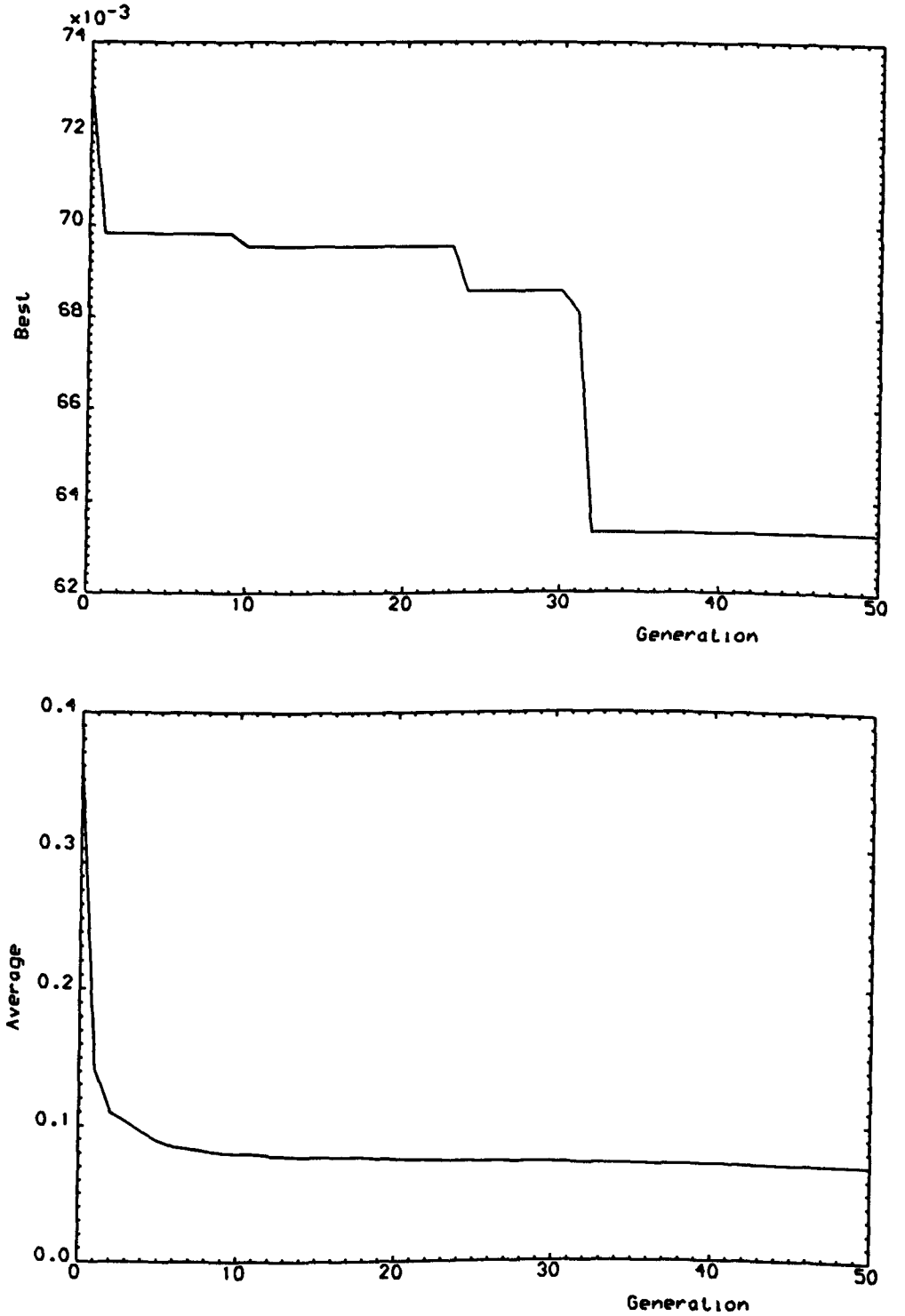

Figure 5.8.c: Best-of-generation and average-of-generation for evolutionarily designed controller (p_m = 0.9, T = 0.04).

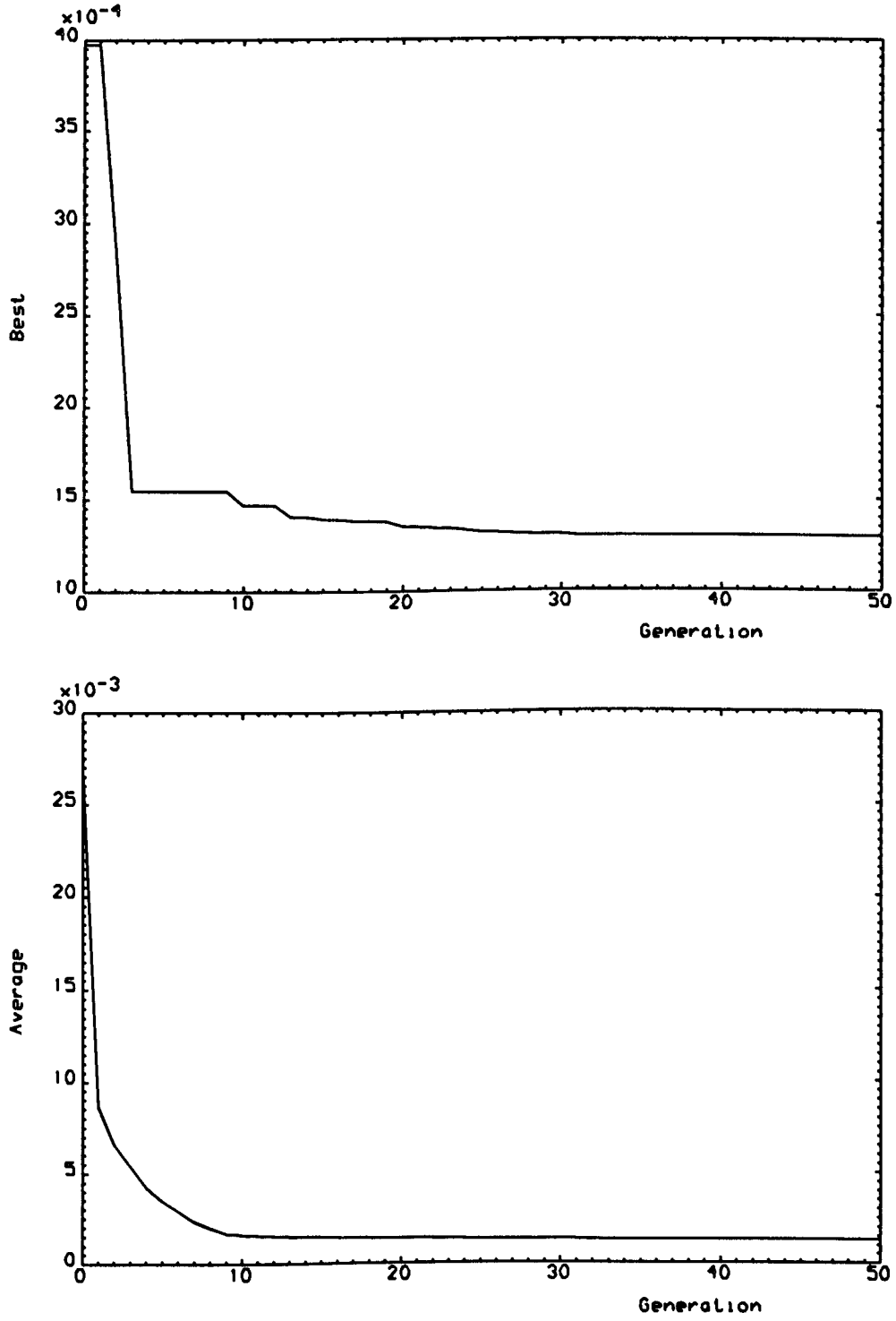

Figure 5.9.a: Best-of-generation and average-of-generation for evolutionarily designed controller (adaptive, T = 0.01).

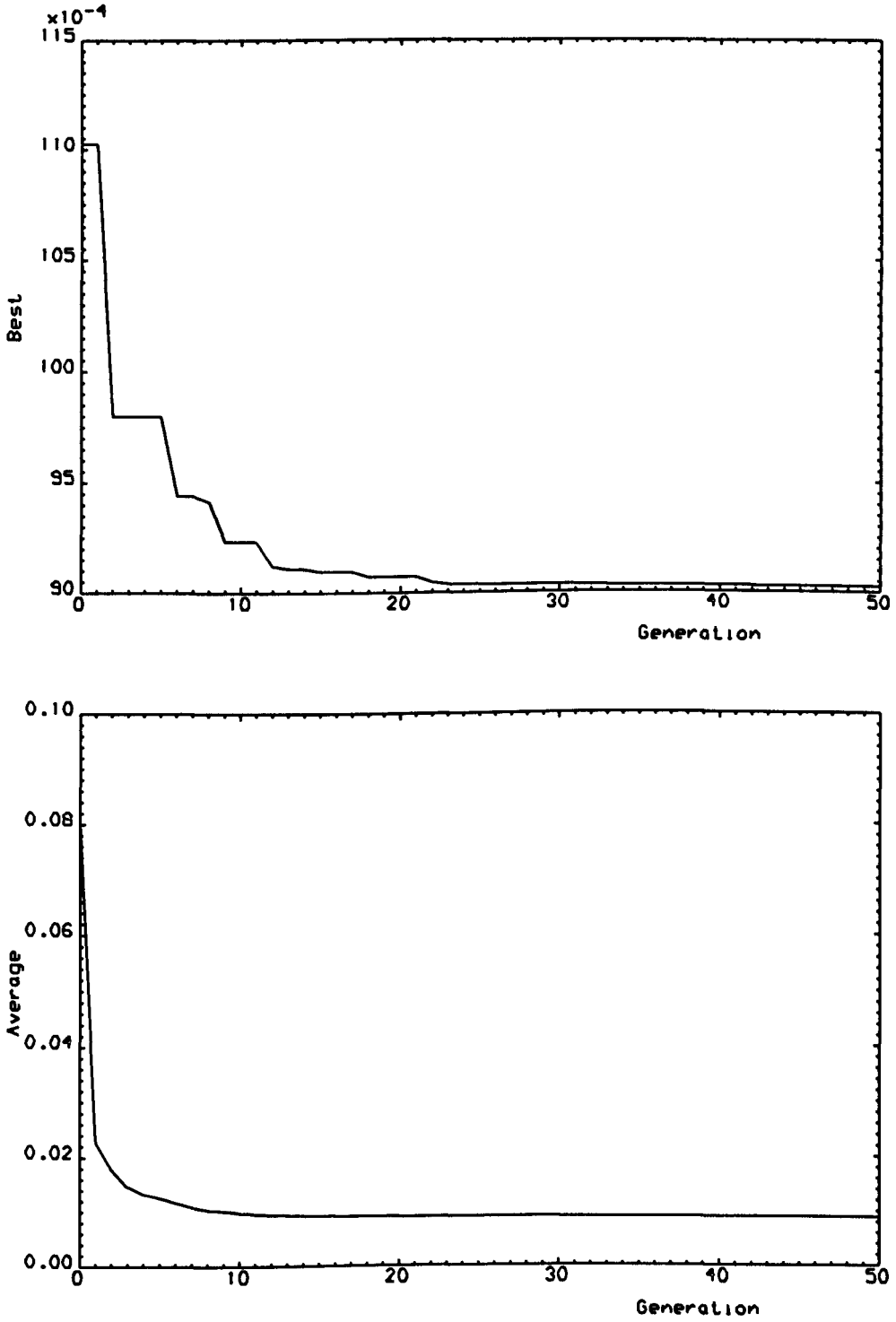

Figure 5.9.b: Best-of-generation and average-of-generation for evolutionarily designed controller (adaptive, T = 0.02).

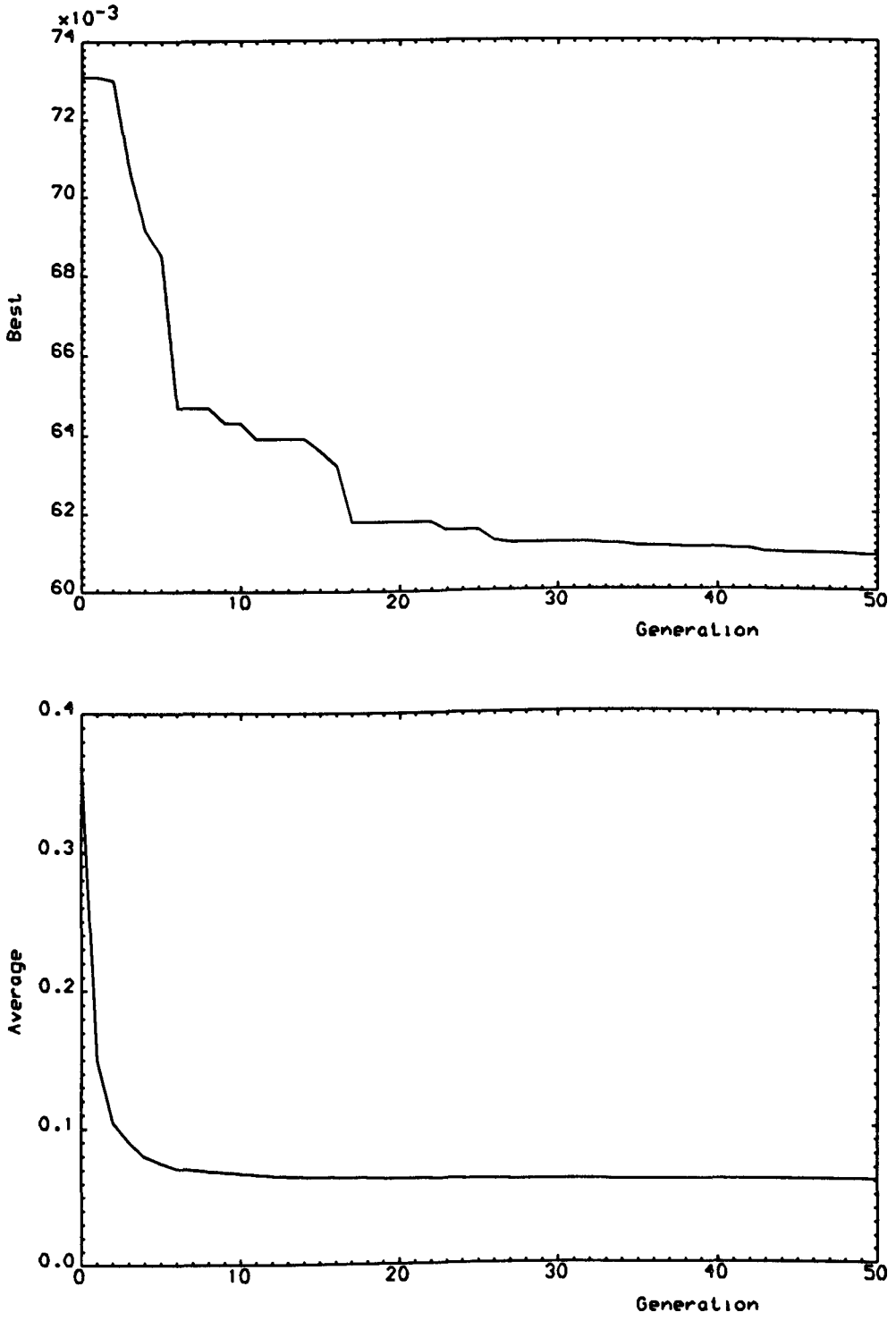

Figure 5.9.c: Best-of-generation and average-of-generation for evolutionarily designed controller (adaptive, T = 0.04).

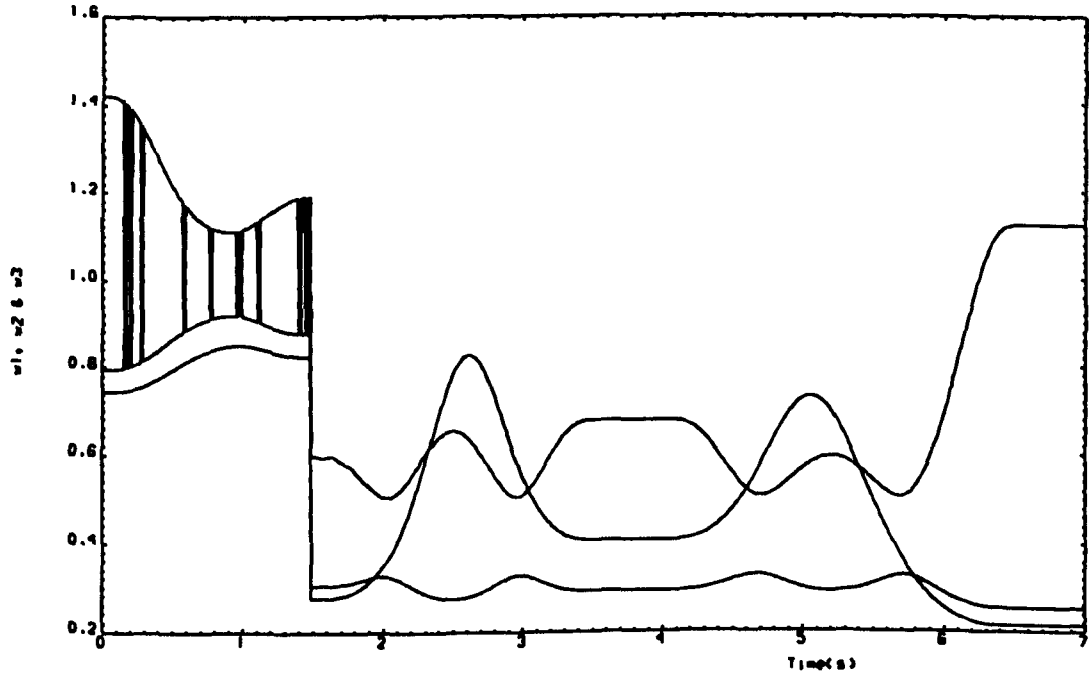

Figure 5.10.a: Time-domain behaviour of eigenvalues of plant perturbation for evolutionarily designed digital PID controller (ES Adaptive T = 0.01).

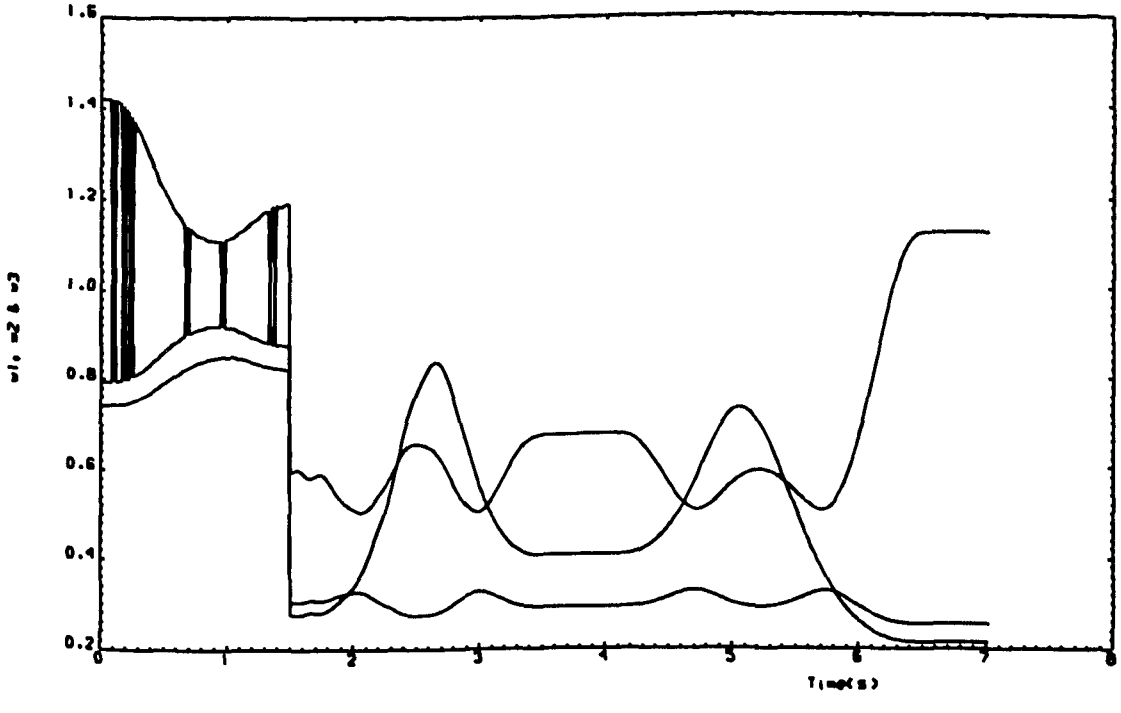

Figure 5.10.b: Time-domain behaviour of eigenvalues of plant perturbation for evolutionarily desinged digital PID controller (ES Adaptive T = 0.02).

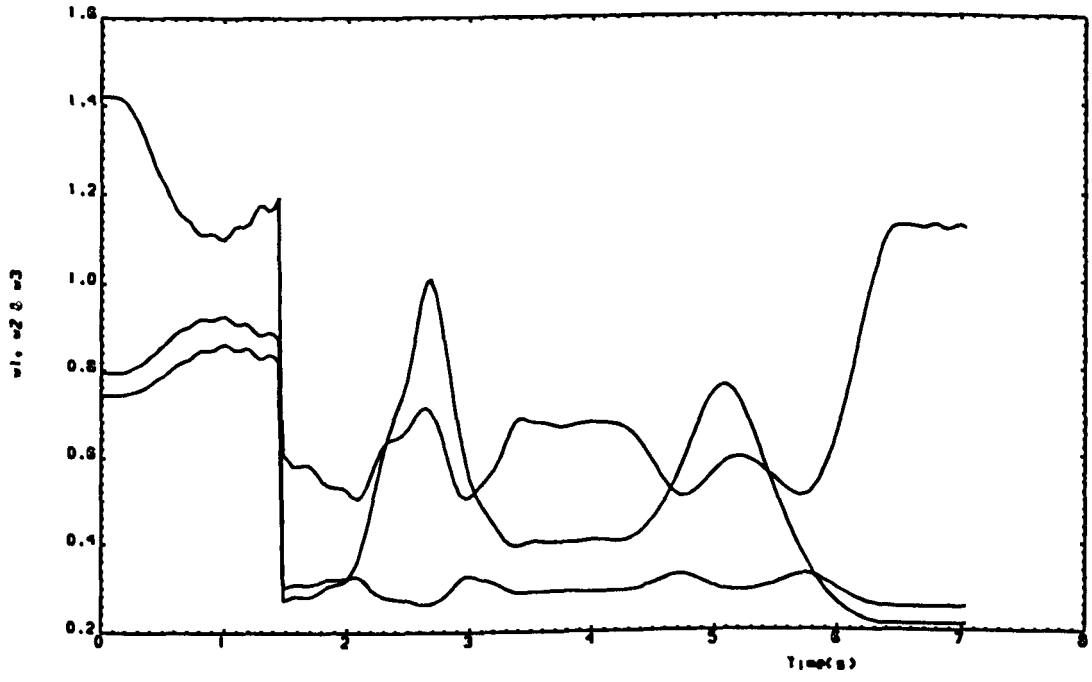

Figure 5.10.c: Time-domain behaviour of eigenvalues of plant perturbation for evolutionarily designed digital PID controller (ES Adaptive T = 0.04).

Chapter 6

PERFORMANCE MEASURES IN THE DESIGN OF DIGITAL CONTROLLERS FOR ROBOTIC MANIPULATORS USING EVOLUTIONARY ALGORITHMS

6.1 INTRODUCTION

In Chapters 4 and 5, genetic algorithms and evolution strategies were respectively used for the design of robust digital trajectory-tracking controllers for robotic manipulators. It is an established fact that complex modern control systems -such as those for robotic manipulators- require the use of sophisticated performance criteria in order to assess the suitability of alternative designs. In the particular evolutionary design procedures presented in Chapters 4 and 5, the minimisation of the integral error norm for any trajectory-tracking task was regarded as the ultimate design requirement.

However, as pointed out by *Schultz* and *Rideout* (1961) in their survey of control system performance measures, the application of various alternative performance measures is of great interest in the evaluation of control system designs. Early control experts were mainly concerned with the stability of linear systems and therefore concentrated their work on establishing criteria that enabled the linear system designer to answer the question, "Is the system stable?". The complexity of modern control systems, for which various kinds of stability can be assessed at least in the non-linear sense, compelled control system experts to conclude that

stability alone, although a necessary requirement of a good system, does not guarantee a suitable or usable system design in the practical sense.

Inevitably, the principal goal for a designer is to try to optimise a particular dominant performance criterion. However, the designer may also have other performance characteristics in mind that are deemed to be equally important when a range of practical considerations need to be met. In this way, it is important to note that alternative performance measures can be readily and effectively used in the evolutionary design of digital multivariable PID controllers.

In this chapter, the simplicity of specifying such performance measures is accordingly exploited to allow the selection of a set of three performance measures which embody practical engineering considerations such as trajectory-tracking performance, and amplitude or rate limits on the inputs or outputs of complex devices such as robotic manipulators. Indeed, a detailed comparison is made of the performance of evolutionarily designed trajectory-tracking controllers for robotic manipulators when the following three performance measures are used:

(i) the integral error norm;

(ii) the weighted sum of the integral error norm and the integral error velocity norm;

(iii) the weighted sum of the integral error norm and the integral control effort velocity norm.

These general results are illustrated by the evolutionary design of robustified digital trajectory-tracking PID controllers for the three-degree-of-freedom robotic manipulator previously considered in Chapters 3, 4, and 5.

6.2 EVOLUTIONARY DESIGN PROCEDURE

In Chapters 4 and 5, it was indicated that there are many different measures of optimality. However, it was convenient initially to regard the minimum integral over the task time, τ, of the Euclidean norm of trajectory-tracking error vector in Cartesian space as the ultimate design requirement. Thus, genetic algorithms and evolution strategies were used to select the appropriate set of controller parameters such that

$$\Gamma_1 = \int_0^{\tau} \left\| e(t) \right\| \, dt \tag{6.7}$$

is minimised, where τ is the duration of the tracking task, $e(t) \in \mathfrak{R}^l$ is the trajectory-tracking error vector in Cartesian space, and $\|.\|$ denotes the Euclidean norm.

It is evident that the actual process of selecting which performance measure to use is a problem-related matter. It is here that a certain amount of personal preference is found. Thus, the choice should be governed by relevance, in that the particular performance measure which is selected must be a convenient one

to use, as well as one which yields practical results. In the present case, the initial choice of Γ_1 (equation 6.7) as the performance measure was most relevant to high-accuracy trajectory-tracking behaviour. This is because this measure encompasses the notion of error norm whose minimisation evidently is the most desirable requirement when designing high-accuracy trajectory-tracking PID controllers for robotic manipulators. In practice, it is advisable to use performance measures that are useful in additionally preventing undesirable behaviour of the inputs and outputs. These alternative performance indices can therefore embody certain practical constraints such as amplitude or rate limits on the inputs and outputs associated with complex non-linear plants such as robotic manipulators.

Indeed, in order to introduce the rate limits on the outputs, it is convenient to consider the performance index

$$\Gamma_2 = \int_0^\tau \left\| \Delta e(t) \right\| \, dt \quad , \tag{6.8}$$

where τ is the duration of the tracking task, $e(t) \in \Re^l$ is the trajectory-tracking error vector in Cartesian space, $\Delta e(t) \in \Re^l$ is the change in the error vector $e(t) \in \Re^l$ over a sampling period, and $\|.\|$ denotes the Euclidean norm. This performance index, Γ_2, evidently takes into consideration the rate limits to be imposed on the output of the system since the joint space error is defined as

$$e(t) = v(t) - y(t) \quad , \tag{6.9}$$

where $v(t) \in \mathfrak{R}^l$ is the joint-space reference vector, and $y(t) \in \mathfrak{R}^l$ is the joint output vector of the robotic manipulator end-effector in joint space.

In contradistinction to considering the rate limits on the output, a performance index considering rate limits on the control inputs can be formulated in the form

$$\Gamma_3 = \int_0^\tau \left\| \Delta u(t) \right\| dt \tag{6.10}$$

where τ is the duration of the tracking task, $\Delta u(t) \in \mathfrak{R}^j$ is the change in the control vector $u(t) \in \mathfrak{R}^j$ over a sampling period, and $\|.\|$ denotes the Euclidean norm.

It is also possible to combine these three performance indices. Thus, for example, the cost function

$$\Gamma = \lambda_1 \Gamma_1 + \lambda_2 \Gamma_3 + \lambda_3 \Gamma_3 \tag{6.11}$$

can be used, where λ_1, λ_2 and λ_3 are non-negative weighting parameters.

In exactly the same manner as in Chapters 4 and 5, evolutionary algorithms can be used to design digital trajectory-tracking PID controllers for robotic

manipulators for any choice of performance measure. Indeed, this design problem is immediately amenable to the non-asymptotic approach presented in Chapters 4 and 5. Therefore, it is natural to adopt a similar procedure to that used in Chapters 4 and 5 , which consists of using both genetic algorithms and $(\lambda + \mu)$ -evolution strategies to solve the non-asymptotic design problem..

It is clear that the solution of this problem will provide an optimal quadruple $\{\alpha, \sigma, \rho, \delta\}$ of controller design parameters which is dependent upon the given manipulator, the given task, and the measure of trajectory-tracking performance used in the optimisation procedure. It is therefore evident from equation (6.11) that the nature of optimality is to be determined by the choice of the parameters λ_1, λ_2, and λ_3.

In the same way as that used in Chapters 4 and 5 to solve the non-asymptotic robustness problem using genetic algorithms and evolution strategies, it is necessary only to encode the quadruple $\{\alpha, \sigma, \rho, \delta\}$ of controller design parameters in accordance with a system of concatenated, multi parameter, mapped, fixed-point coding. Thus, each quadruple $\{\alpha, \sigma, \rho, \delta\}$ of controller parameters is represented by a string of binary digits. Then, following any random choice of the initial generation of such strings, successive generations of strings are obtained using the genetic operations of selection and mutation in the case of $(\lambda + \mu)$-evolution strategies, and of selection, crossover and mutation in the case of genetic algorithms. These operations ensure that the successive generations of error-actuated digital PID controller thus produced by the genetic algorithms

and (λ + μ)-evolution strategies tend to exhibit improving trajectory-tracking performance (in respect of the different measure of the quality of such performance specified by the selection of appropriate weighting parameters λ_1, λ_2, and λ_3 for any given manipulator).

6.3 ILLUSTRATIVE EXAMPLE

6.3.1 SYSTEM DESCRIPTION

The evolutionary design procedure can be conveniently illustrated by designing trajectory-tracking digital PID controllers when variants of the performance measures Γ in equation (6.11) are used in the case of the three-degree-of-freedom robotic manipulator previously investigated in Chapters 4 and 5. This robotic manipulator is governed on T = [0, +∞] by state and output equations of the respective forms [*Petropoulakis* (1986)]

$$M(\theta)\ddot{\theta} + c(\theta, \dot{\theta}) + g(\theta) = u \quad , \qquad (6.12a)$$

and

$$y = f(\theta) \quad . \qquad (6.12b)$$

The numerical values of the inertial and kinematic parameters of the typical three-degree-of-freedom robotic manipulator under consideration are those

given by *Petropoulakis* (1986). The robotic manipulator is, for design purposes, considered to be in the neighbourhood of the arbitrarily selected operating point corresponding to the end-effector position (0,0.45,0)m which corresponds to the joint space coordinates ($\pi/2$, 0.27π, -0.27π) proposed by *Porter* and *Abidin* (1990b).

The evolutionarily designed controller is used while the end-effector of the robotic manipulator is caused to track the straight-line trajectories defined in Chapters 3, 4, and 5. In the same manner as in the illustrative example of Chapters 3, 4, and 5, the robotic manipulator grasps an additional payload of 5 Kg after the initial transition **I** $\rightarrow$ **II** (see section 3.5) in order to test the robustness of the controller..

The structure of the computational implementation of the evolutionary algorithms in the case of the trajectory-tracking system is the same as that shown in Figures 4.2 and 5.1. In this way, the evolutionary design of the controller for this selected trajectory- tracking task can be readily effected so as to minimise the cost function

$$\Gamma = \lambda_1\Gamma_1 + \lambda_2\Gamma_3 + \lambda_3\Gamma_3 \quad , \qquad (6.13)$$

where λ_1, λ_2 and λ_3 are non-negative weighting parameters, and Γ_1, Γ_2 and Γ_3 are as defined in equations (6.7), (6.8) and (6.10). Thus, substituting equations (6.7), (6.8) and (6.9) into equation (6.13), it follows that

$$\Gamma = \lambda_1 \int_0^\tau \|e(t)\| \, dt + \lambda_2 \int_0^\tau \|\Delta e(t)\| \, dt + \lambda_3 \int_0^\tau \|\Delta u(t)\| \, dt . \qquad (6.14)$$

The evolutionary design of controllers for the given trajectory task can be readily undertaken by minimising any particular measure of performance specified by the designer. In particular, it is instructive to use the cost function in equation (6.14) in the following three particular cases:

(i) the integral error norm for which $\Gamma = \Gamma_1$ (λ_1=1, λ_2= 0, λ_3= 0);

(ii) the sum of the integral error norm and the weighted integral error velocity norm for which $\Gamma = \Gamma_1 + \lambda_2 \Gamma_2$ (λ_1=1, λ_3= 0);

(iii) the sum of the integral error norm and the weighted integral control velocity norm for which $\Gamma = \Gamma_1 + \lambda_3 \Gamma_3$ (λ_1=1, λ_2= 0).

6.3.2 GENETIC ALGORITHMS

The genetic design of PID controllers considered in Chapter 4 was that corresponding to the first of these cases. However, the results of performing the genetic robustification procedure over 50 generations in more general cases are shown in Tables 6.1 and 6.2 for a population size N = 30, a crossover probability p_c= 0.6, and a mutation probability p_m= 0.008. Thus, the results obtained by genetically minimising Γ with λ_1 = 1 and λ_3= 0, while λ_2 is left to

vary within the selected interval [0, 5 x 10^4], are shown in Table 6.1; whilst the corresponding results obtained by genetically minimising Γ with λ_1= 1 and λ_2= 0, while λ_3 is allowed to vary within the interval [0, 1] are shown in Table 6.2.

λ_2(λ_1=1, λ_3=0)	Γ	Γ_1	Γ_2	Γ_3	α	σ	ρ	δ
0	1.471 10^{-3}	1.47 10^{-3}	1.33 10^{-5}	3.29	0.1743	0.8312	16.7511	0.0126
10^3	1.571 10^{-3}	1.50 10^{-3}	1.06 10^{-5}	4.05	0.1057	0.8786	18.2956	0.0199
2 x 10^3	1.700 10^{-3}	1.54 10^{-3}	1.01 10^{-5}	3.00	0.0862	0.8568	18.6693	0.0213
3 x 10^3	1.729 10^{-3}	1.59 10^{-3}	7.63 10^{-6}	4.06	0.1045	0.8832	19.9679	0.0255
6 x 10^3	1.863 10^{-3}	1.71 10^{-3}	4.18 10^{-6}	4.3	0.1092	0.8810	18.2146	0.0237
5 x 10^4	2.161 10^{-3}	1.80 10^{-3}	2.49 10^{-6}	2.6	0.0951	0.8870	11.4397	0.0178

Table **6.1**: Genetic designs of PID controllers (λ_1 = 1, $\lambda_2 \in$ [0, 5 x 10^4], λ_3 = 0).

λ_3 (λ_1=1, λ_2=0)	Γ	Γ_1	Γ_2	Γ_3	α	σ	ρ	δ
0	1.471 10^{-3}	1.47 10^{-3}	1.33 10^{-5}	3.29	0.1743	0.8312	16.7511	0.0126
10^{-3}	3.673 10^{-3}	1.76 10^{-3}	1.09 10^{-7}	1.88	0.1485	0.8319	10.7541	0.0124
5 x 10^{-3}	9.985 10^{-3}	2.28 10^{-3}	6.21 10^{-6}	1.54	0.1493	0.8517	15.9273	0.0304
10^{-2}	17.67 10^{-2}	2.62 10^{-3}	5.28 10^{-6}	1.51	0.1343	0.8528	13.3393	0.0295
10^{-1}	138.9 10^{-3}	7.78 10^{-3}	3.76 10^{-5}	1.31	0.1248	0.7321	10.4856	0.0707
1	1305 10^{-3}	17.53 10^{-3}	1.68 10^{-4}	1.29	0.1142	0.6997	6.0821	0.1016

Table **6.2** : Genetic designs of PID controllers (λ_1 = 1, λ_2 = 0, $\lambda_3 \in$ [0, 1]).

By varying λ_2, a trade-off between the individual performance indices Γ_1 and Γ_2 is introduced. It is evident from Table 6.1 that, as λ_2 increases, the value of Γ_2 decreases while the value of Γ_1 increases: this implies an increase of the importance of Γ_2 in the overall performance measure Γ, with the result that Γ_1 becomes less significant in the process. The existence of such a compromise is demonstrated by the associated increasing 'smoothing' of the trajectory-tracking

performance of the manipulator present in the time-domain behaviour of the error vector, *e(t)*, shown in Figures 6.2(a), 6.3(a), 6.4(a), 6.5(a) and 6.6(a). This is particularly visible during the transient period following the sudden change of pay-load, in that the greater the value of λ_2 the smoother is the transition.

It is clear that the introduction of the performance measure, Γ_2, has the expected effect on the rate of variation of the error vector, *e(t)*. Indeed, a closer look at the different time-domain responses allows a comparison of the time-domain behaviour of the error vector, *e(t)*, depicted in Figures 6.1(a) —corresponding to the design in which only the minimisation of the performance index, Γ_1, is taken into account— with the behaviour of the error vector, *e(t)*, obtained for the successive values of λ_2 when the performance index,Γ_2 ,is considered. It is clear in Figure 6.2(a) that the transient period following the introduction of the pay-load is smoother than in Figure 6.1(a), whereas the transient peak and the amplitude of the error vector, *e(t)*, over the duration τ of the entire task are slightly bigger. This pattern is confirmed and accentuated in the successive time-domain behaviours depicted in Figures 6.3(a), 6.4(a), 6.5(a) and 6.6(a). However, this pattern of behaviour is mostly evident in Figure 6.6(a) which corresponds to the design with the highest value of $\lambda_2=5x10^4$, where the transient oscillatory behaviour vanishes. In other words, the bigger is the penalisation of the performance index Γ_2 (bigger value of λ_2) in the overall performance index, Γ, the smaller is the numerical value of Γ_2 (see

Table 6.1). In contrast to the decrease in the numerical value of the performance index Γ_2, there corresponds an increase in the numerical value of the performance index, Γ_1, caused by the increasing amplitude of the error vector, $e(t)$, in the process.

Similarly, it is evident from Table 6.2 that, as λ_3 increases, the value of Γ_3 decreases while the value of Γ_1 increases. In this case, a compromise between the performance indices Γ_1 and Γ_3 is introduced as a result of varying λ_3. The joint torques generated by the digital PID controllers consequently exhibit an increasingly 'smooth' time-domain response associated with an increase in the value of λ_3, as shown in Figures 6.7(b), 6.8(b), 6.9(b), 6.10(b) and 6.11(b).

It is clear from these time-domain responses that the overall performance measure, Γ, which includes the performance indices Γ_3 and Γ_1, ensures 'smooth' behaviour for both the error vector, $e(t)$, and joint torque vector, $u(t)$. Inevitably, the only limitation resulting from this particular combination of these performance indices is the rapidly deteriorating trajectory-tracking behaviour resulting from the selection of large values for λ_3 (see Figures 6.10(a) and 6.11(a)). Indeed, it is clear from Table 6.2 that the use of the performance measure $\Gamma = \Gamma_1 + \lambda_3\Gamma_3$ (λ_1=1, λ_2= 0), which consists of the sum of the integral error norm and the weighted integral control velocity norm, has an increasing impact on the trajectory-tracking behaviour of the robotic manipulator as the value of the weighting parameter λ_3 increases. Initially, this pattern is illustrated

by the net increase in the value of the transient peak which is 2 to 3 times bigger in Figures 6.10(a) and 6.11(a) than the peak value present in Figure 6.1(a). The same trend characterises the amplitude of the error vector, *e(t)*, throughout the task period τ (see Figures 6.10(a) and 6.11(a)). This deteriorating trajectory-tracking behaviour is confirmed by the correspondingly large values of the performance index, Γ_1. In fact, in order to ensure the existence of an appropriate trade-off between the combined performance indices, care must be taken to avoid using inadequate weighting parameters, λ_3. Indeed, failing to take such a precaution can produce an unbalanced overall performance measure, Γ, thus causing the deterioration of the trajectory-tracking performance due to the loss of significance of the performance measure, Γ_1, in the overall performance measure, Γ.

It is also clear from Table 6.1 and 6.2 that, for 10 out of the 11 proposed designs, the value of the performance measure, Γ_1, is smaller than that obtained by singular perturbation methods (see Table 4.1). This attests to the ability of genetic algorithms to minimise this performance measure. Indeed, following the study of the time-domain behaviour and the evaluation of the individual performance indices of all 11 proposed designs, it is clear that the design which corresponds to the combination of the weighting parameters $\lambda_1 = 1$, $\lambda_2 = 0$, and $\lambda_3 = 5 \times 10^{-3}$ is the most appropriate one. Indeed, in this particular case, the time-domain responses depicted in Figures 6.8(a) and 6.8(b) exhibit a qualitative performance which appears to outperform the remaining proposed designs. This particular design is characterised by the 'smoothness' of the time-

domain behaviour of the joint torques (see Figure 6.8(b)) (which is important for practical implementation) and by the acceptable value of its performance index $\Gamma_1 = 2.28 \times 10^{-3}$ (which ensures good trajectory-tracking behaviour).

In view of the importance of the trajectory-tracking error performance measure, Γ_1, (see equation (6.18)) in evaluating the quality of the trajectory-tracking behaviour, it is clear that the optimum setting of the controller design parameters must ultimately be achieved without appreciably deteriorating the performance measure, Γ_1. This translates conveniently into making appropriate choices of the weighting parameters associated with the individual performance indices without adversely affecting the trajectory-tracking behaviour best assessed by the corresponding value of Γ_1.

6.3.3 EVOLUTION STRATEGIES

In the present case, a (30 + 30)-evolution strategy in both its non-adaptive and adaptive variants was used while retaining the 16 bit representation for each of the four controller parameters as well as the 50 generations-long evolution process of Chapter 5. In addition, the non-adaptive (30 + 30)-evolution strategy retains the best mutation strategy of Chapter 5 based on the mutation probability, $p_m = 0.02$.

It is clear from Tables 6.3 and 6.4 that the first design in each table corresponds to the first case of Table 5.1, for which $\Gamma = \Gamma_1$ when (λ_1=1, λ_2= 0, and λ_3= 0).

This also applies to the first cases of the adaptive (30 + 30)-evolution strategy depicted in Tables 6.5 and 6.6 which correspond to their counterparts of Table 5.4. Thus, the results obtained by evolutionarily minimising Γ with $\lambda_1 = 1$ and $\lambda_3 = 0$, while λ_2 is left to vary within the selected interval $[0, 5 \times 10^4]$, are shown for the non-adaptive (30 + 30)-evolution strategy in Table 6.3; whilst the corresponding results obtained by evolutionarily minimising Γ with $\lambda_1 = 1$ and $\lambda_2 = 0$, while λ_3 is allowed to vary within the interval [0, 1] are shown in Table 6.4. Similarly, the results generated by the adaptive (30 + 30)-evolution strategy corresponding to the same variation of the weighting parameters λ_1,λ_2, and λ_3 are respectively shown in Tables 6.5 and 6.6

It is clear from Tables 6.3 and 6.5 that, as in the case of genetic algorithms, the introduction of the performance measures, Γ_2, plays an increasingly significant role in the overall performance measure, Γ_1, as its associated weighting parameter λ_2 increases. This implies that, as the value of Γ_2 decreases due to its increasing importance in the overall performance measure, the corresponding value of Γ_1 increases. Hence, the trade-off between the individual performance measures is introduced as the weighting parameter, λ_2, varies. As a result, the time- domain behaviour of the error vector, *e(t)*, shown in Figures 6.12(a), 6.13(a), 6.14(a), 6.15(a), 6.16(a), 6.17(a), 6.23(a), 6.24(a), 6.25(a), 6.26(a), 6.27(a) and 6.28(a) exhibits to a different degree smoother behaviour during the transient period following the sudden introduction of the payload. Indeed, the smoothness of the transition phase intensifies with the

increase in the value of the weighting parameter, λ_2. It is also clear from these time-domain plots that the trade-off introduced as a result of varying the value of λ_2 translates not only in the reduction of the rate of variation of the error vector, *e(t)*, but also in the increase of the transient peak and of the amplitude of the error vector, *e(t)*, over the duration, τ, of the entire task. Thus, as formulated in (ii), the overall performance measure, Γ, takes account of the constraint on the rate of variation of the error vector, *e(t)*, which causes the (30 + 30)-evolution strategy to generate appropriate set of controllers parameter. These latter parameters are obviously dependent on the value of the weighting factor, λ_2, used in the overall measure performance, Γ. Similarly, the result obtained in Tables 6.4 and 6.6 show that as λ_3 is permitted to vary within the interval [0, 1] while imposing $\lambda_1 = 1$ and $\lambda_2 = 0$, a trade-off between the individual performance measures Γ_1 and Γ_3 is initiated. Hence, an increase in the value of λ_3 is marked by an increase in the value of the performance index, Γ_1, whereas the same increase in the value of λ_3 causes the value of the performance index, Γ_3, to decrease. It is clear from equation (6.14) that, since the joint torques (control effort) generated by the digital PID controllers are incorporated in the formulation of the overall performance measure, Γ, varying λ_3 will automatically affect the shape of the joint torque vector, *u(t)*, by reducing the rate of variation of *u(t)*. Indeed, the corresponding time-domain behaviour shown in Figures 6.18, 6.19, 6.20, 6.21 and 6.22 in the non-adaptive (30 + 30)-evolution strategy case, and in Figures 6.29, 6.30, 6.31, 6.32 and 6.33 in the adaptive (30 + 30)-evolution strategy case, are characterised by the 'smooth' behaviour of both the

joint torque vector ,*u(t)*, and error vector, *e(t)*. However, a deterioration in the trajectory-tracking performance is detected for larger values of λ_3 (see the values of the performance index Γ_1 in Tables 6.4 and 6.6) which requires the designer to take certain precautions when selecting values for the weighting parameter, λ_3. Indeed, it is strongly recommended that the designer avoid excessive values of λ_3 which will ultimately create an unbalance between the individual performance indices ,Γ_1 and Γ_3, resulting in the deterioration of the tracking performance.

Finally, it is important to note from Tables 6.3, 6.4, 6.5 and 6.6 the quantitative superiority of the controller performance offered by the adaptive (30 + 30)-evolution strategy over its non-adaptive counterpart regardless of the selected overall performance measure, Γ. Most importantly, the evolution strategies in both variants show a better ability than genetic algorithms in successfully discriminating between the different optimal quadruples, $\{\alpha,\sigma,\rho,\delta\}$, which frequently do not differ very much from each other, yet still produce rather different time-domain behaviours.

λ_2(λ_1=1, λ_3=0)	Γ	Γ_1	Γ_2	Γ_3	α	σ	ρ	δ
0	$1.301\ 10^{-3}$	$1.30\ 10^{-3}$	$1.34\ 10^{-5}$	3.48	0.1264	0.8999	22.9080	0.0194
10^3	$1.455\ 10^{-3}$	$1.37\ 10^{-3}$	$9.04\ 10^{-6}$	3.03	0.1554	0.8920	15.1580	0.0117
$2\ 10^3$	$1.531\ 10^{-3}$	$1.39\ 10^{-3}$	$1.07\ 10^{-5}$	4.51	0.1205	0.8981	21.0292	0.0165
$3\ 10^3$	$1.577\ 10^{-3}$	$1.42\ 10^{-3}$	$6.64\ 10^{-6}$	4.04	0.1309	0.8986	14.6073	0.0122
$6\ 10^3$	$1.739\ 10^{-3}$	$1.51\ 10^{-3}$	$4.44\ 10^{-6}$	3.55	0.1405	0.8945	11.2577	0.0105
$5\ 10^4$	$2.139\ 10^{-3}$	$1.82\ 10^{-3}$	$6.77\ 10^{-6}$	3.07	0.0511	0.8542	20.2217	0.0299

Table **6.3**: Non-adaptive evolution strategies designs (λ_1=1, $\lambda_2 \in$ [0, 1], λ_3=0, p_m = 0.02).

λ_3 (λ_1=1, λ_2=0)	Γ	Γ_1	Γ_2	Γ_3	α	σ	ρ	δ
0	$1.301\ 10^{-3}$	$1.30\ 10^{-3}$	$1.34\ 10^{-5}$	3.48	0.1264	0.8999	22.9080	0.0194
10^{-3}	$3.521\ 10^{-3}$	$1.52\ 10^{-3}$	$9.17\ 10^{-6}$	1.80	0.2024	0.9000	20.7422	0.0254
$5\ 10^{-3}$	$9.984\ 10^{-3}$	$2.26\ 10^{-3}$	$5.91\ 10^{-6}$	1.54	0.1495	0.8480	15.7088	0.0295
10^{-2}	$17.50\ 10^{-2}$	$2.79\ 10^{-3}$	$4.38\ 10^{-6}$	1.47	0.1494	0.8328	13.7022	0.0319
10^{-1}	$138.7\ 10^{-3}$	$7.69\ 10^{-3}$	$4.00\ 10^{-5}$	1.31	0.0808	0.7244	9.0506	0.0585
1	$1303\ 10^{-3}$	$15.11\ 10^{-3}$	$2.02\ 10^{-4}$	1.29	0.0564	0.6812	7.5592	0.1041

Table **6.4**: Non-adaptive evolution strategies designs (λ_1=1, λ_2= 0, $\lambda_3 \in [\ 0, 1\ ]$, p_m = 0.02).

λ_2(λ_1=1, λ_3=0)	Γ	Γ_1	Γ_2	Γ_3	α	σ	ρ	δ
0	$1.290\ 10^{-3}$	$1.29\ 10^{-3}$	$1.33\ 10^{-5}$	3.29	0.1341	0.8998	22.9808	0.0187
10^{3}	$1.414\ 10^{-3}$	$1.31\ 10^{-3}$	$1.06\ 10^{-5}$	4.05	0.1270	0.9000	20.3631	0.0159
$2\ 10^{3}$	$1.563\ 10^{-3}$	$1.32\ 10^{-3}$	$1.01\ 10^{-5}$	3.00	0.1490	0.8959	19.0104	0.0145
$3\ 10^{3}$	$1.571\ 10^{-3}$	$1.34\ 10^{-3}$	$7.63\ 10^{-6}$	4.06	0.1313	0.8999	15.9988	0.0129
$6\ 10^{3}$	$1.731\ 10^{-3}$	$1.48\ 10^{-3}$	$4.18\ 10^{-6}$	4.3	0.1306	0.8992	11.4939	0.0110
$5\ 10^{4}$	$1.922\ 10^{-3}$	$1.80\ 10^{-3}$	$2.49\ 10^{-6}$	2.6	0.1006	0.8996	14.7570	0.0221

Table **6.5**: Adaptive evolution strategies designs (λ_1 = 1, $\lambda_2 \in [\ 0, 1\ ]$, λ_3 = 0).

λ_3(λ_1=1, λ_2=0)	Γ	Γ_1	Γ_2	Γ_3	α	σ	ρ	δ
0	$1.290\ 10^{-3}$	$1.29\ 10^{-3}$	$1.33\ 10^{-5}$	3.29	0.1341	0.8998	22.9808	0.0187
10^{-3}	$3.409\ 10^{-3}$	$1.57\ 10^{-3}$	$8.08\ 10^{-6}$	1.84	0.1497	0.8762	19.8165	0.0239
$5\ 10^{-3}$	$9.948\ 10^{-3}$	$2.22\ 10^{-3}$	$6.21\ 10^{-6}$	1.54	0.1687	0.8562	16.3611	0.0305
10^{-2}	$17.44\ 10^{-2}$	$2.47\ 10^{-3}$	$5.28\ 10^{-6}$	1.50	0.2433	0.8996	15.0344	0.0334
10^{-1}	$138.7\ 10^{-3}$	$7.47\ 10^{-3}$	$3.76\ 10^{-5}$	1.31	0.0685	0.7241	9.1001	0.0569
1	$1302\ 10^{-3}$	$14.43\ 10^{-3}$	$1.68\ 10^{-4}$	1.28	0.0599	0.6858	6.1473	0.0787

Table **6.6**: Adaptive evolution strategies designs (λ_1 = 1, λ_2 = 0, $\lambda_3 \in [\ 0, 1\ ]$).

6.4 CONCLUSION

In this chapter, evolutionary algorithms have been used to design digital multivariable PID controllers for robotic manipulators for typical trajectory-tracking tasks when various different performance measures are used. It has thus been shown that, by using an appropriate performance measure (which embodies practical engineering constraints), the set of controller design parameters can be readily found that determines the optimal time-domain behaviour of robotic manipulators for such tasks. This use of the power of evolutionary algorithms has been illustrated by the design of digital trajectory-tracking PID controllers for the three-degree-of-freedom robotic manipulator previously investigated in Chapters 4 and 5.

The evolutionary design approach used in this chapter and in Chapters 4 and 5 constitutes a quick and effective way of achieving good overall trajectory-tracking performance of robotic manipulators. However, it has been shown that the controllers designed using genetic algorithms are outperformed by those designed using evolution strategies.

It is clear in this chapter that only the performance of the controllers designed using all three different evolutionary algorithms has been evaluated. In Chapter 7, the performance of all three evolutionary algorithms in terms of ease of design and convergence characteristics will be therefore evaluated.

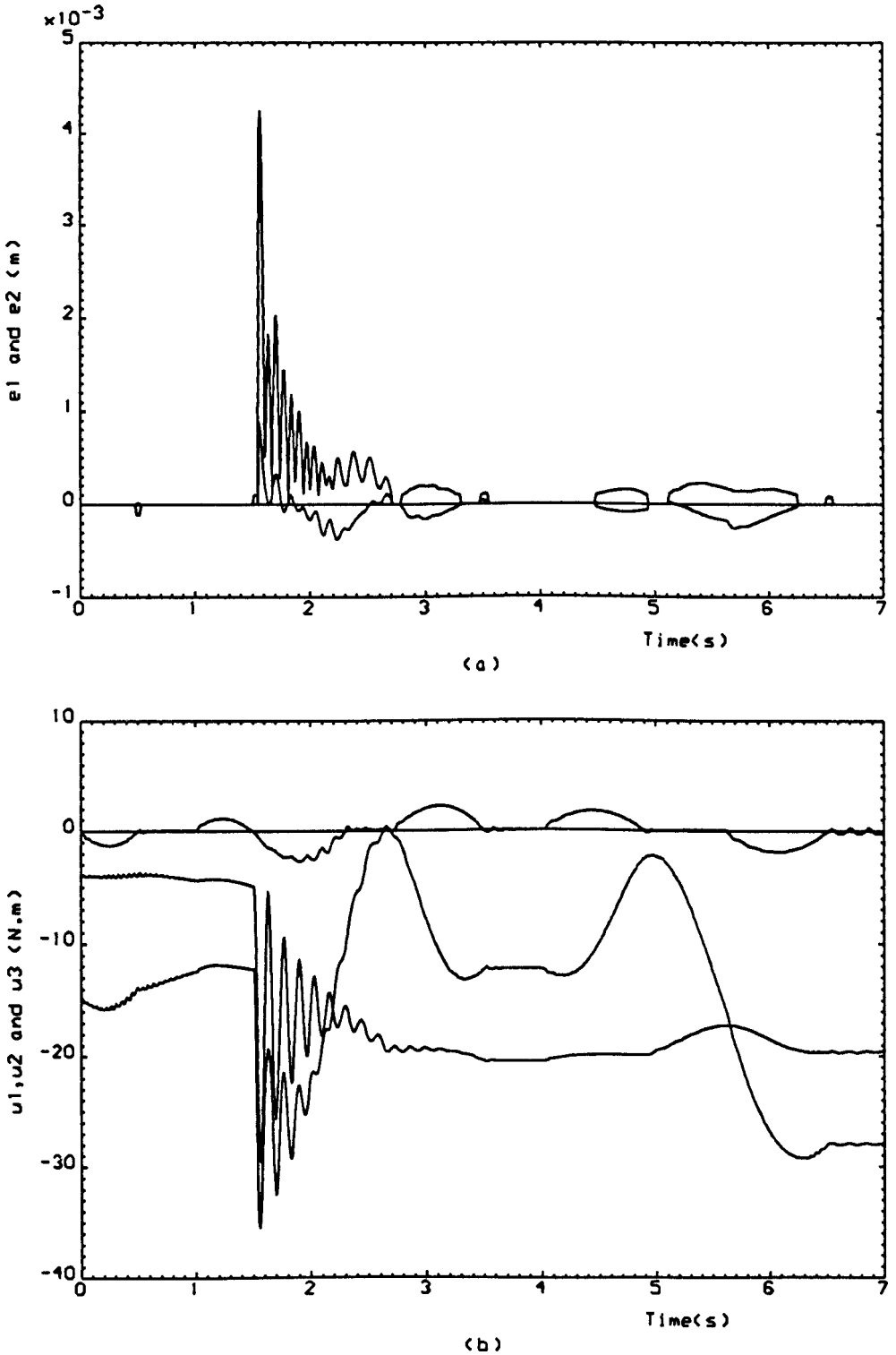

Figure 6.1: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1;\lambda_2 = 0;\lambda_3 = 0$).

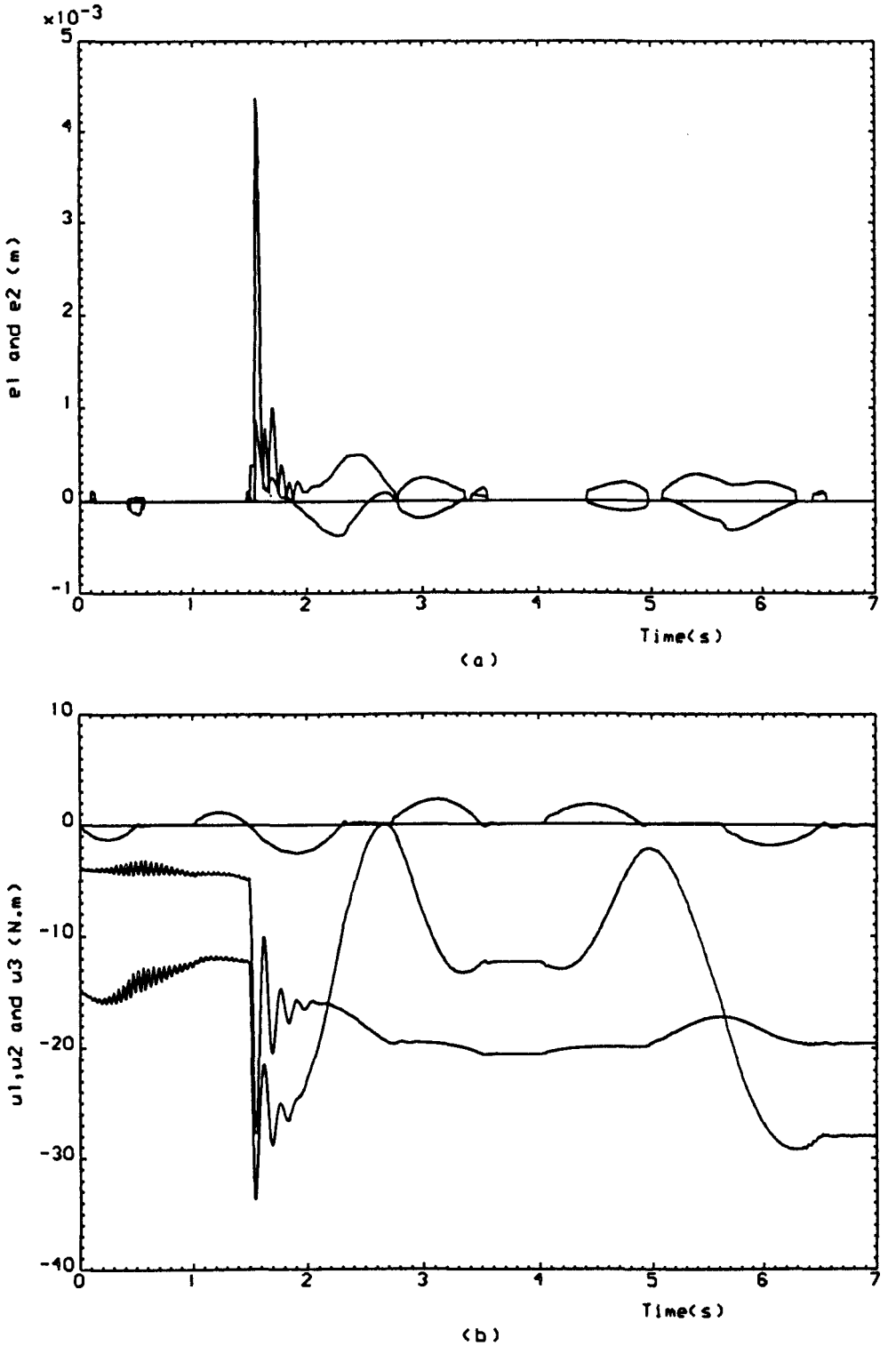

Figure 6.2: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1;\lambda_2 = 10^3;\lambda_3 = 0$).

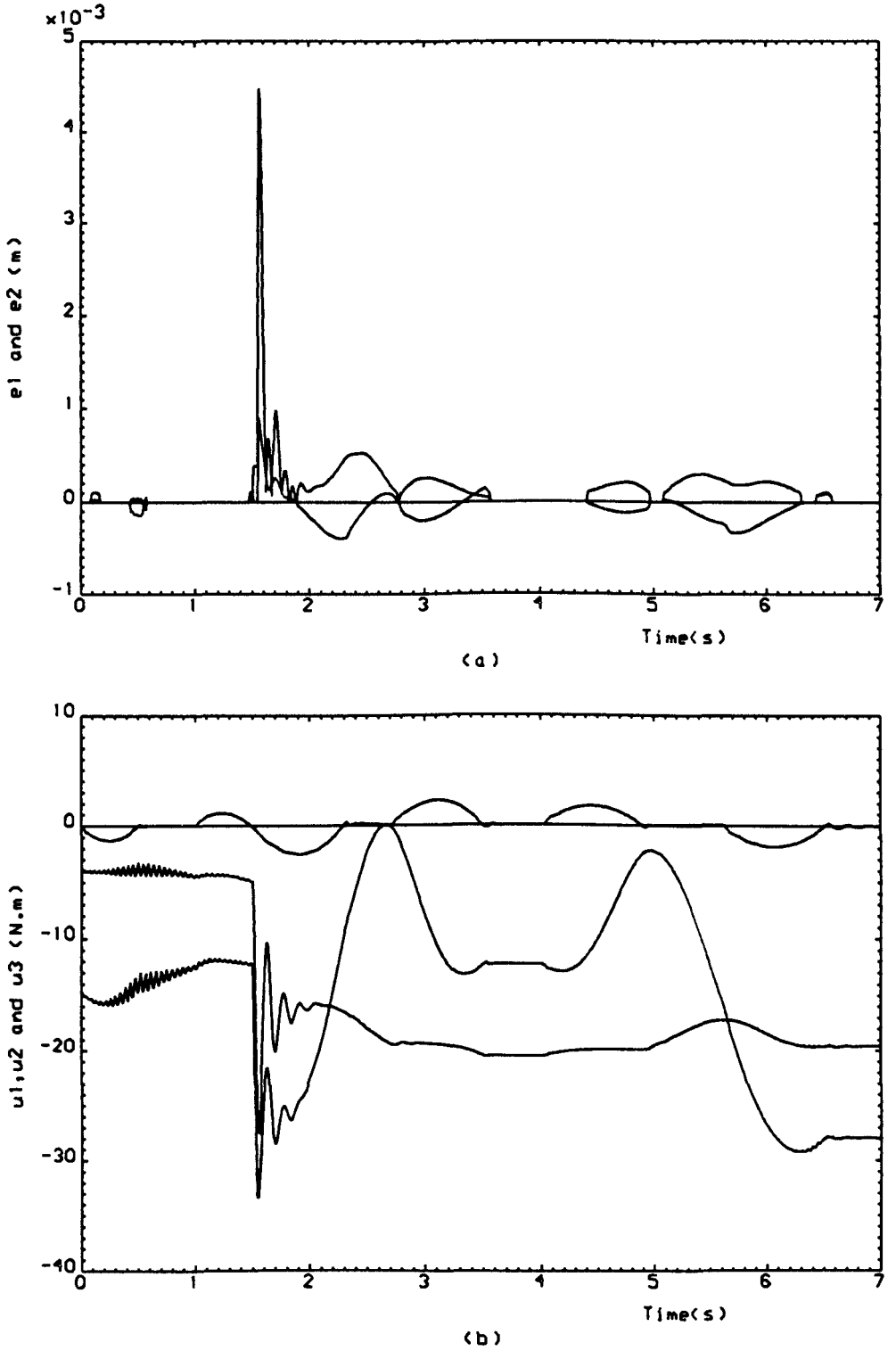

Figure 6.3: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1;\lambda_2 = 2 \times 10^3;\lambda_3 = 0$).

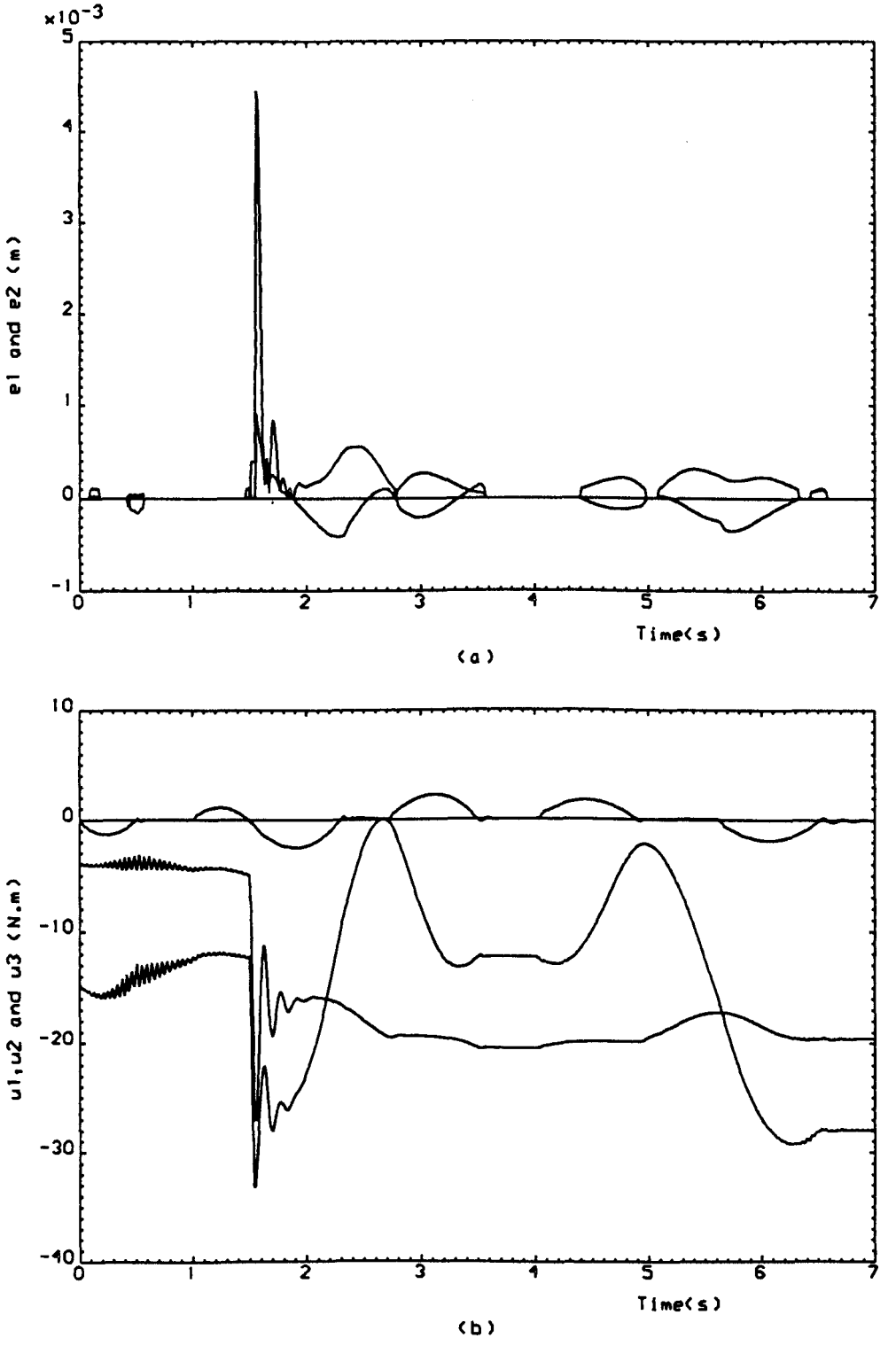

Figure 6.4: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 3 \times 10^3; \lambda_3 = 0$).

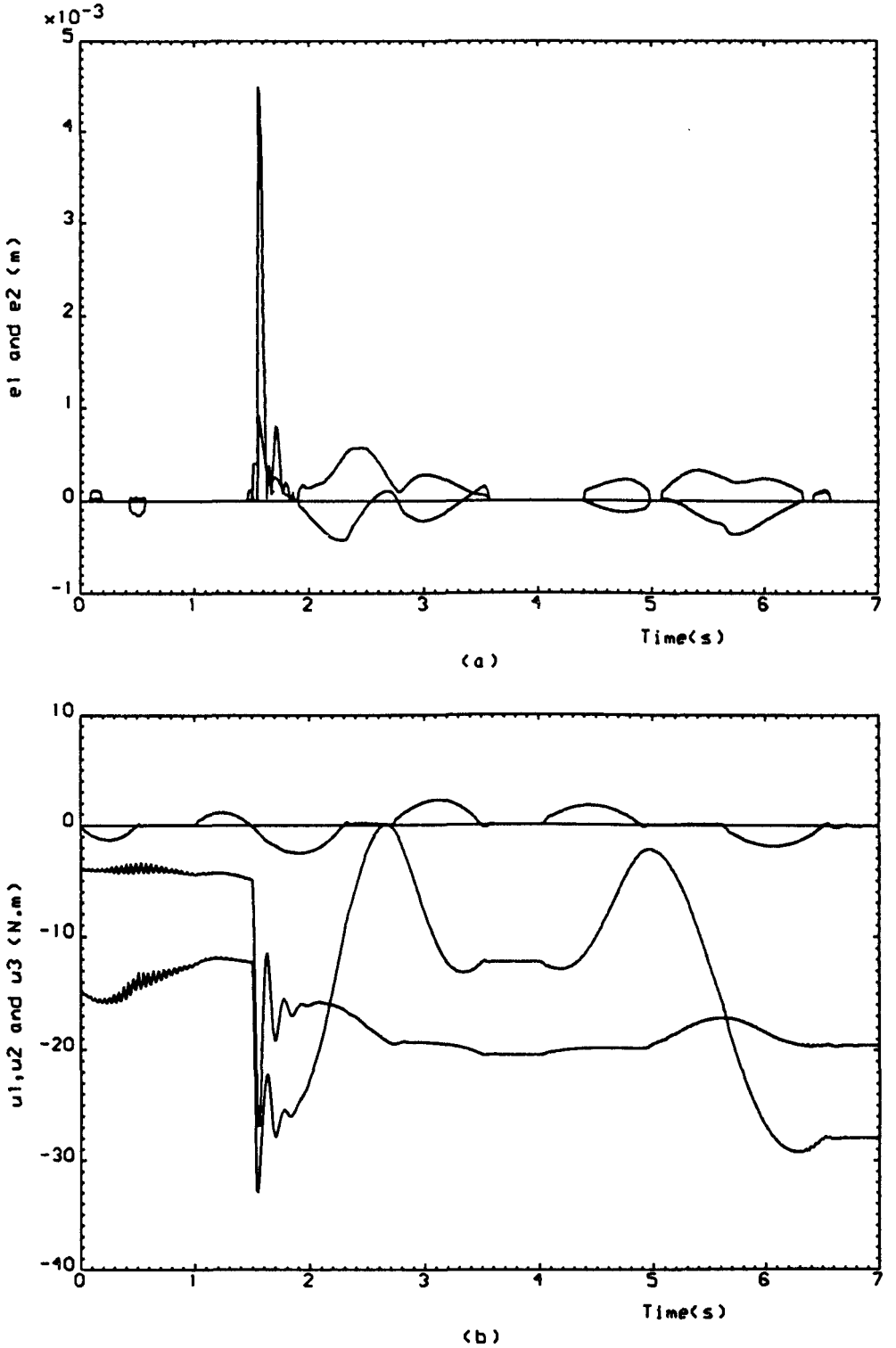

Figure 6.5: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 6 \times 10^3; \lambda_3 = 0$).

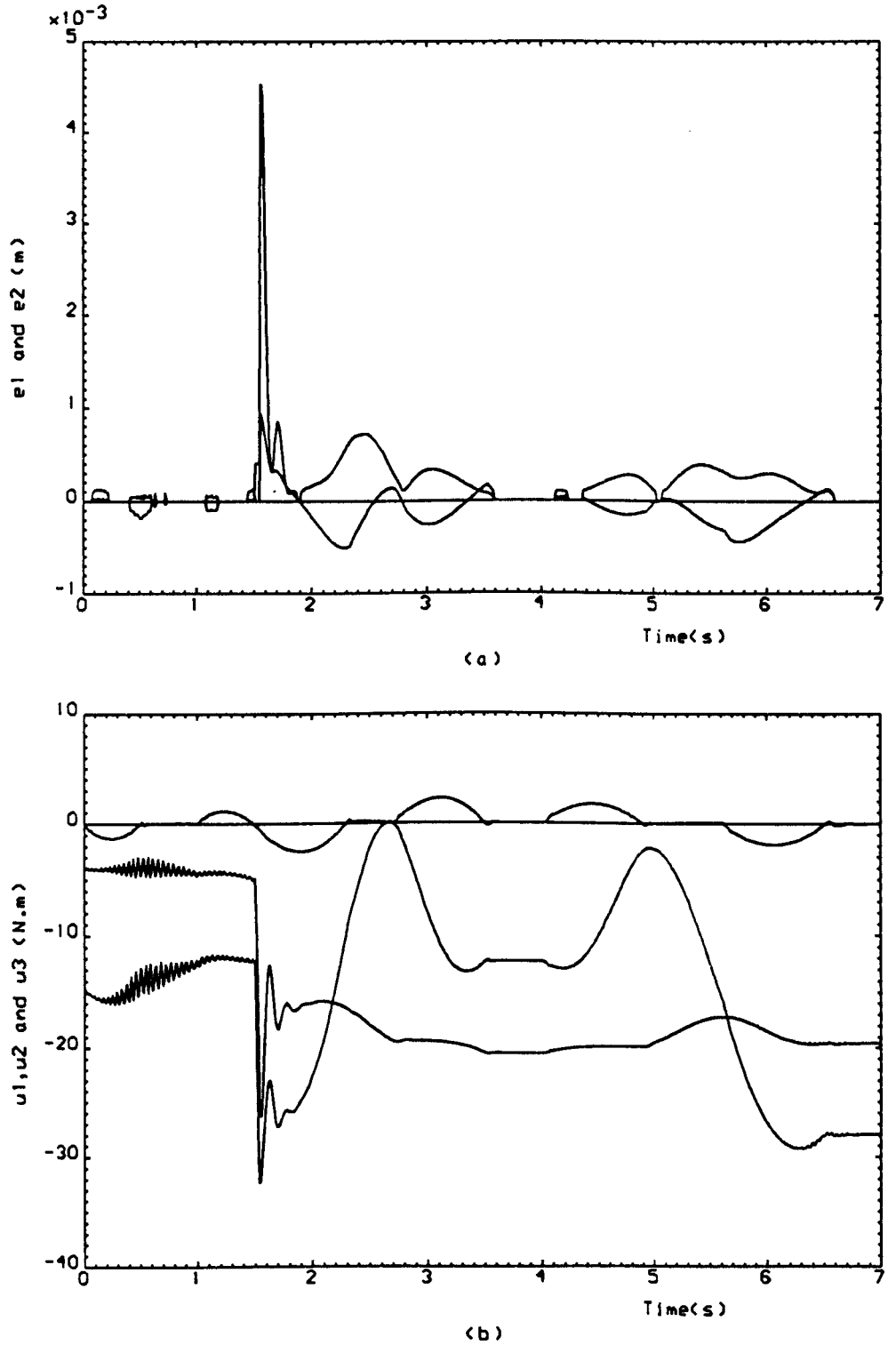

Figure 6.6: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 5 \times 10^4; \lambda_3 = 0$).

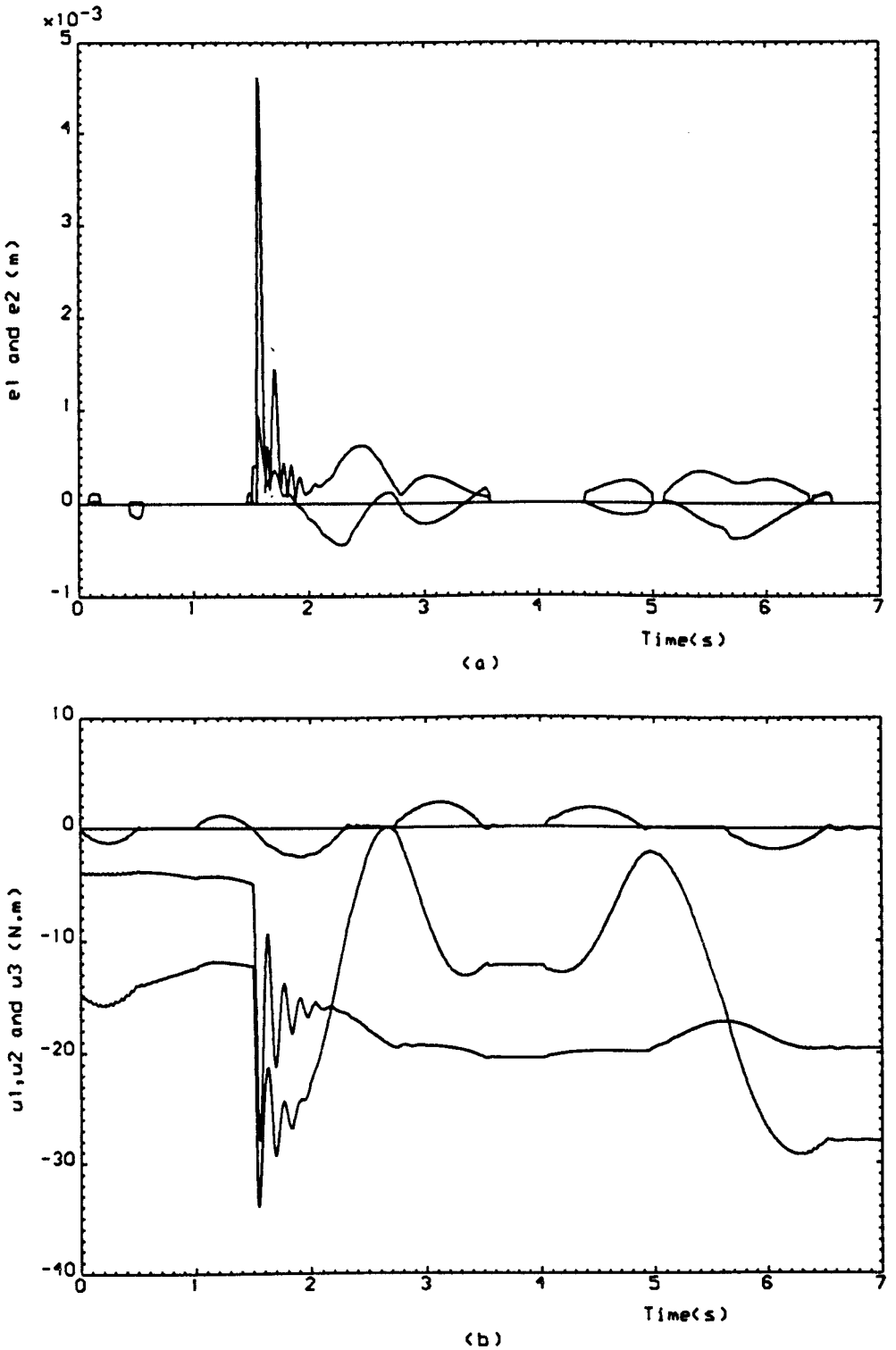

Figure 6.7: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 0; \lambda_3 = 10^{-3}$).

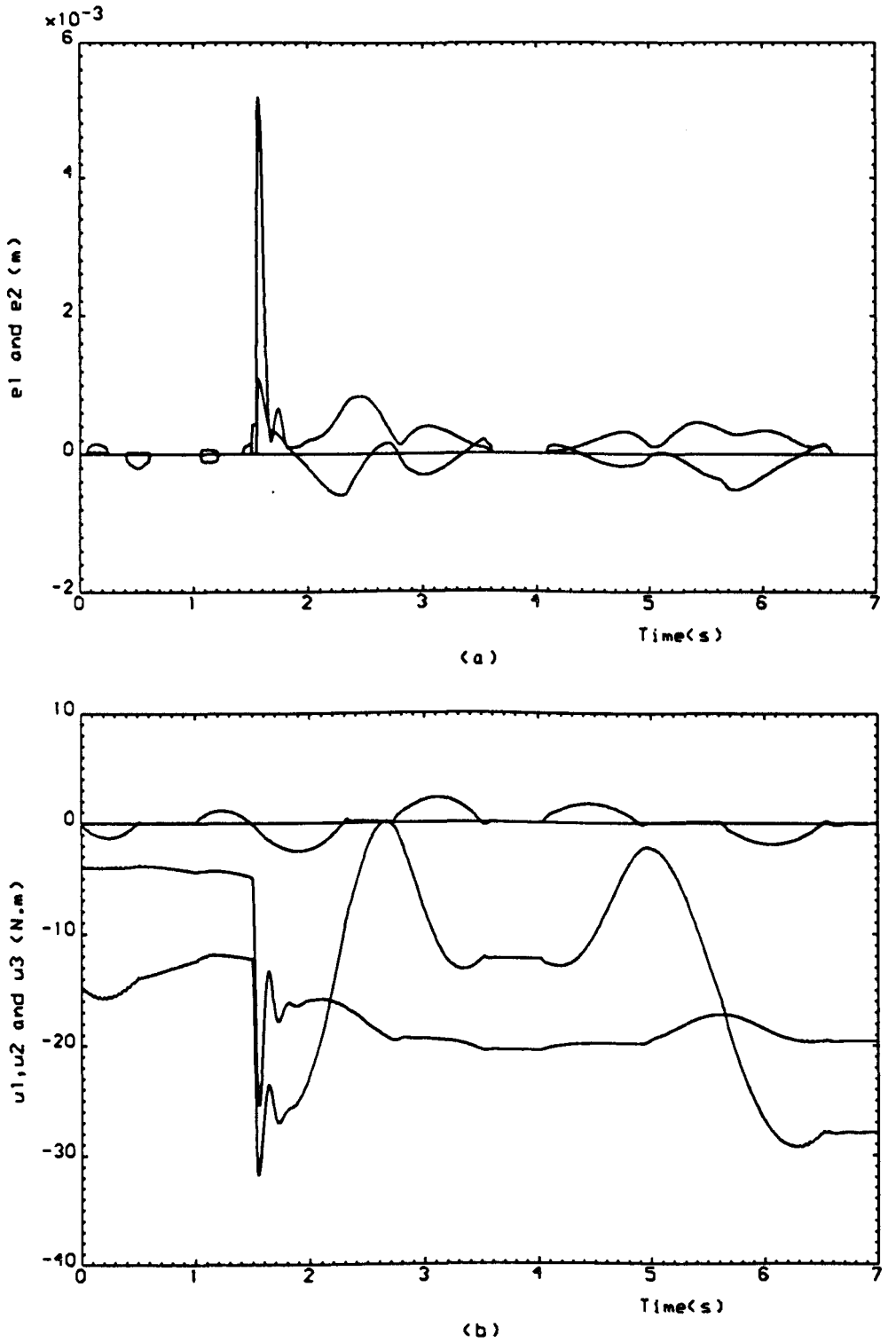

Figure 6.8: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 0; \lambda_3 = 5 \times 10^{-3}$).

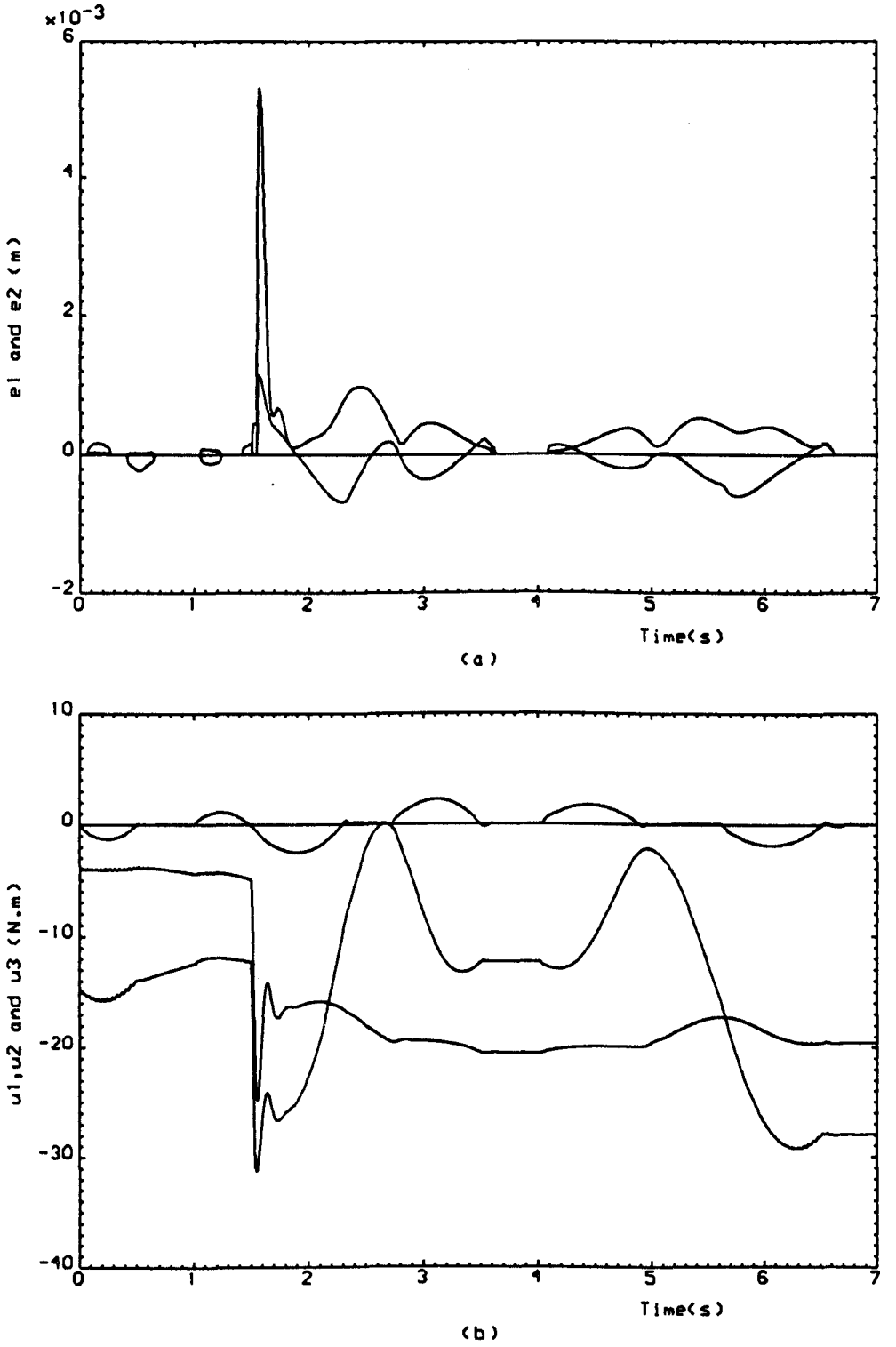

Figure 6.9: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 0; \lambda_3 = 10^{-2}$).

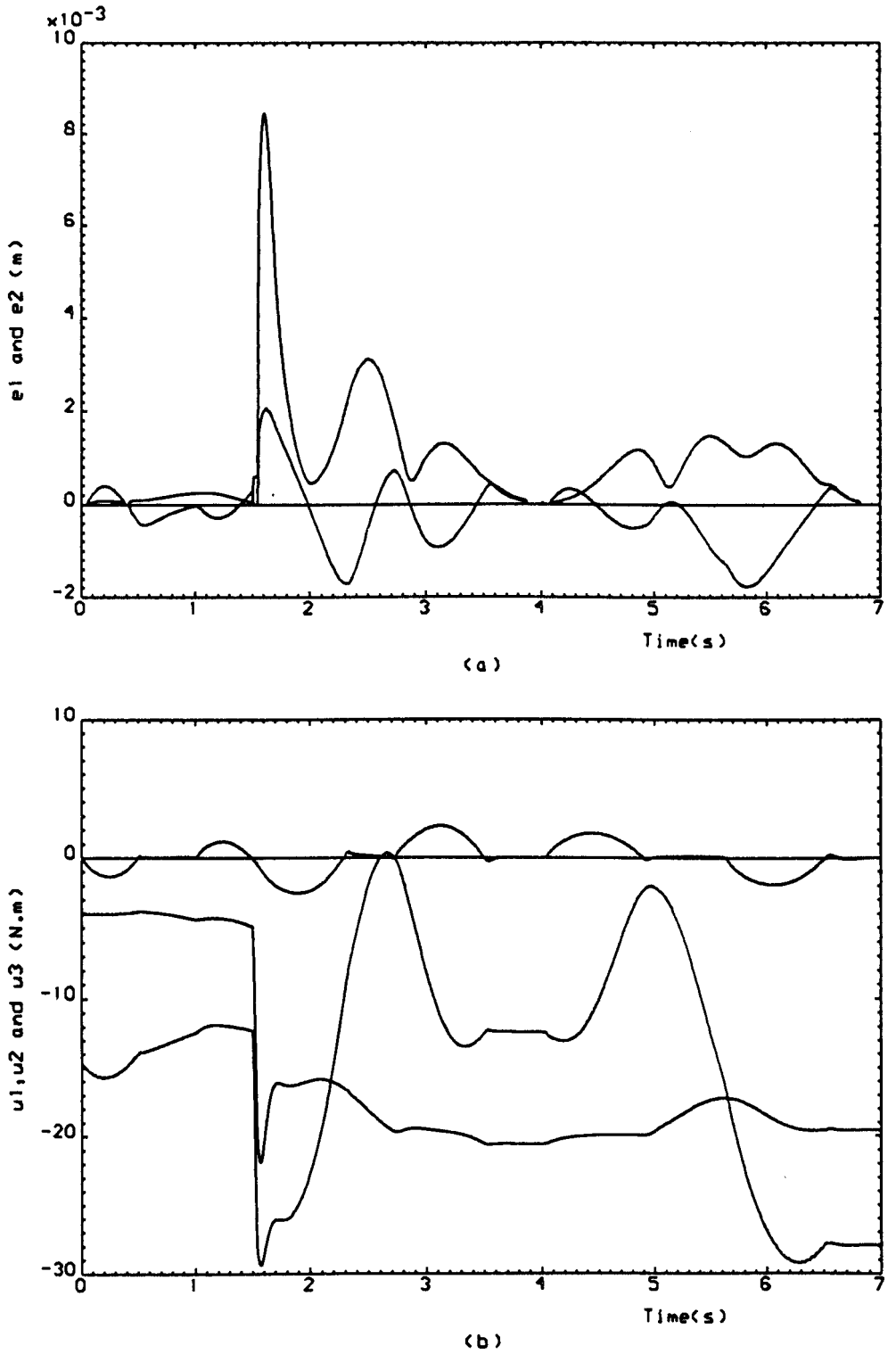

Figure 6.10: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 0; \lambda_3 = 10^{-1}$).

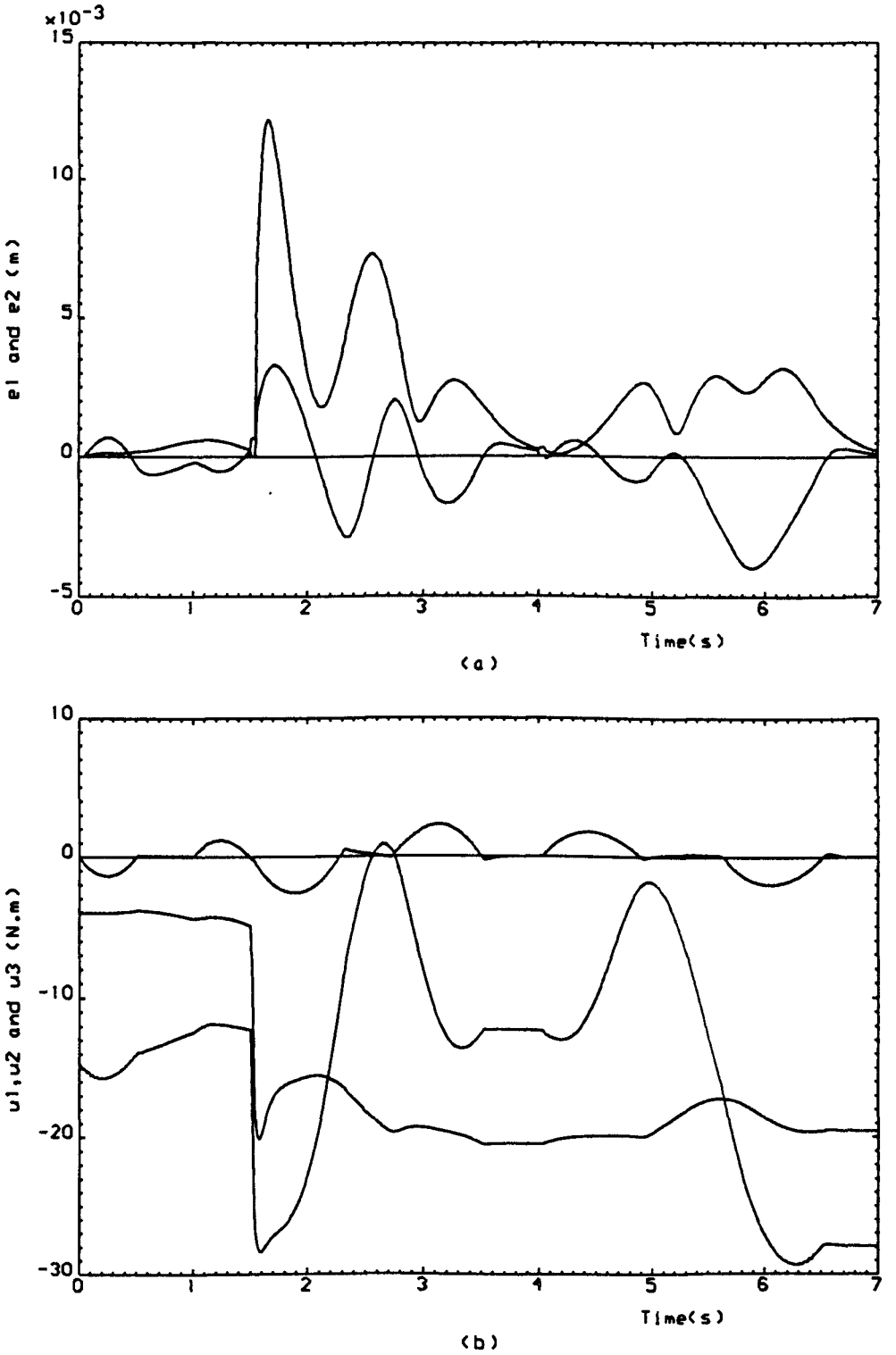

Figure 6.11: Time-domain behaviour of errors and torques for the genetic design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 0; \lambda_3 = 1$).

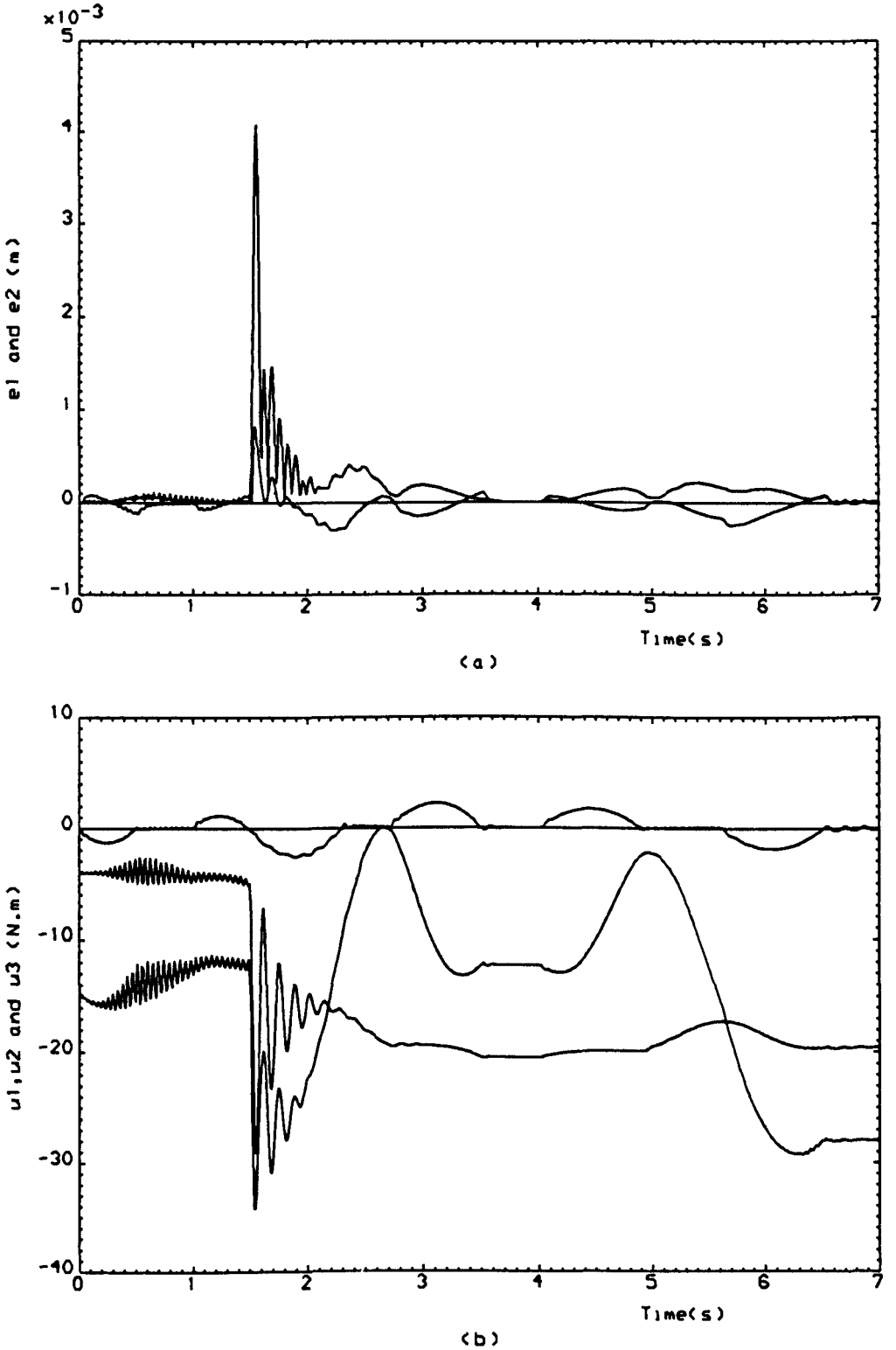

Figure 6.12: Time-domain behaviour of errors and torques for the non- adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 0;λ_3 = 0).

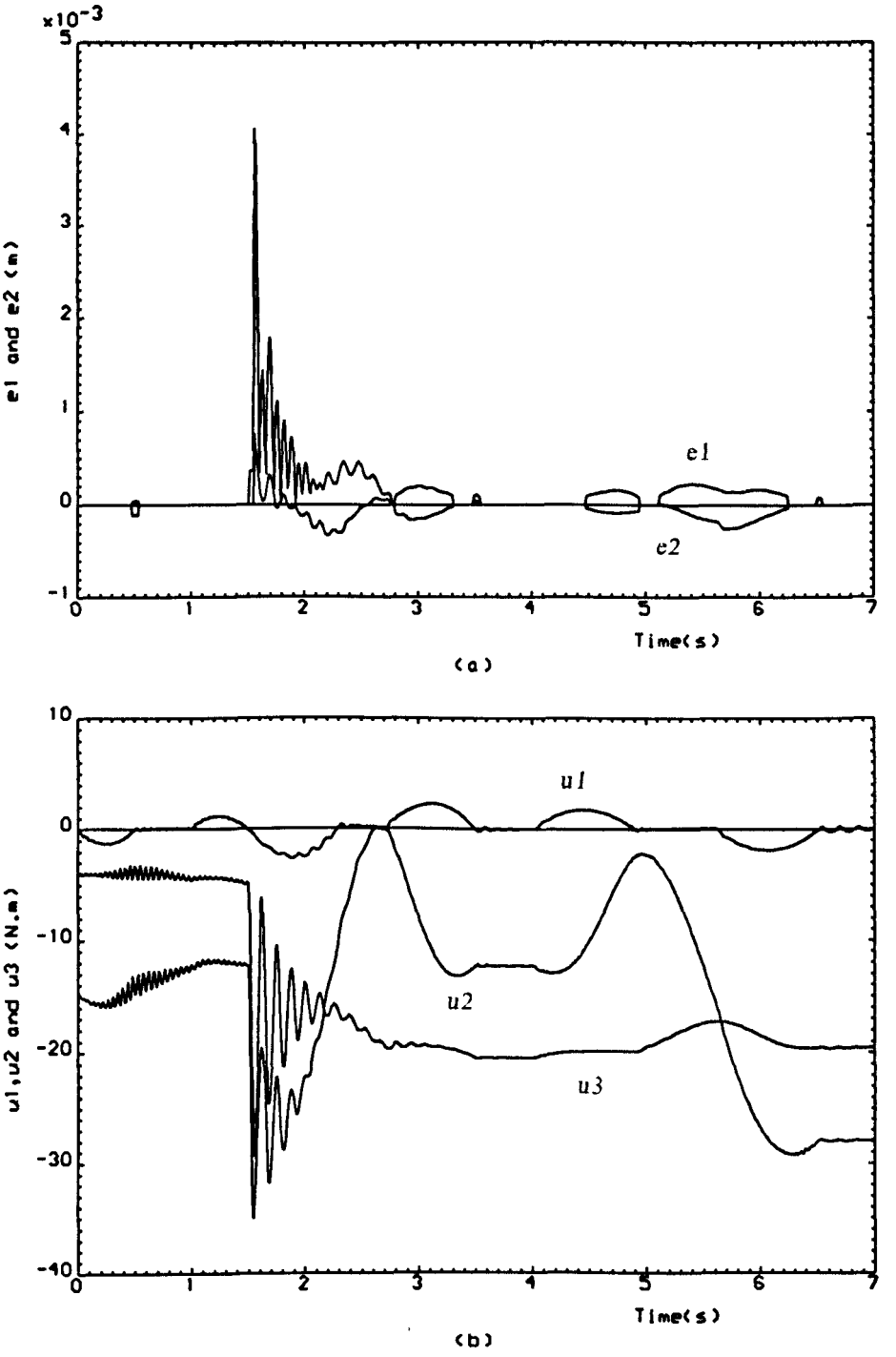

Figure 6.13: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (pm = 0.02; $\lambda_1 = 1$;$\lambda_2 = 10^3$;$\lambda_3 = 0$).

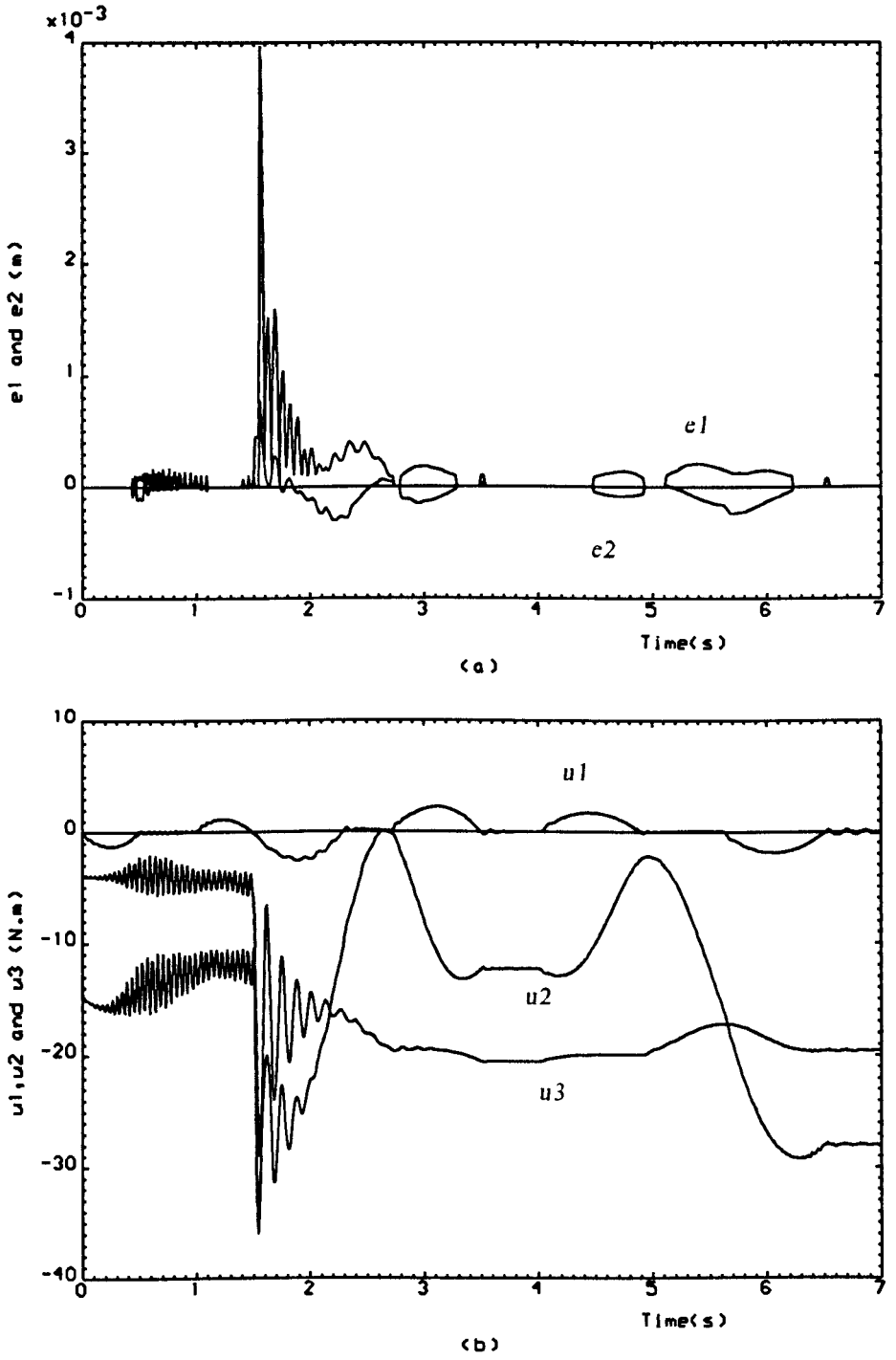

Figure 6.14: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 2 x 10^3;λ_3 = 0).

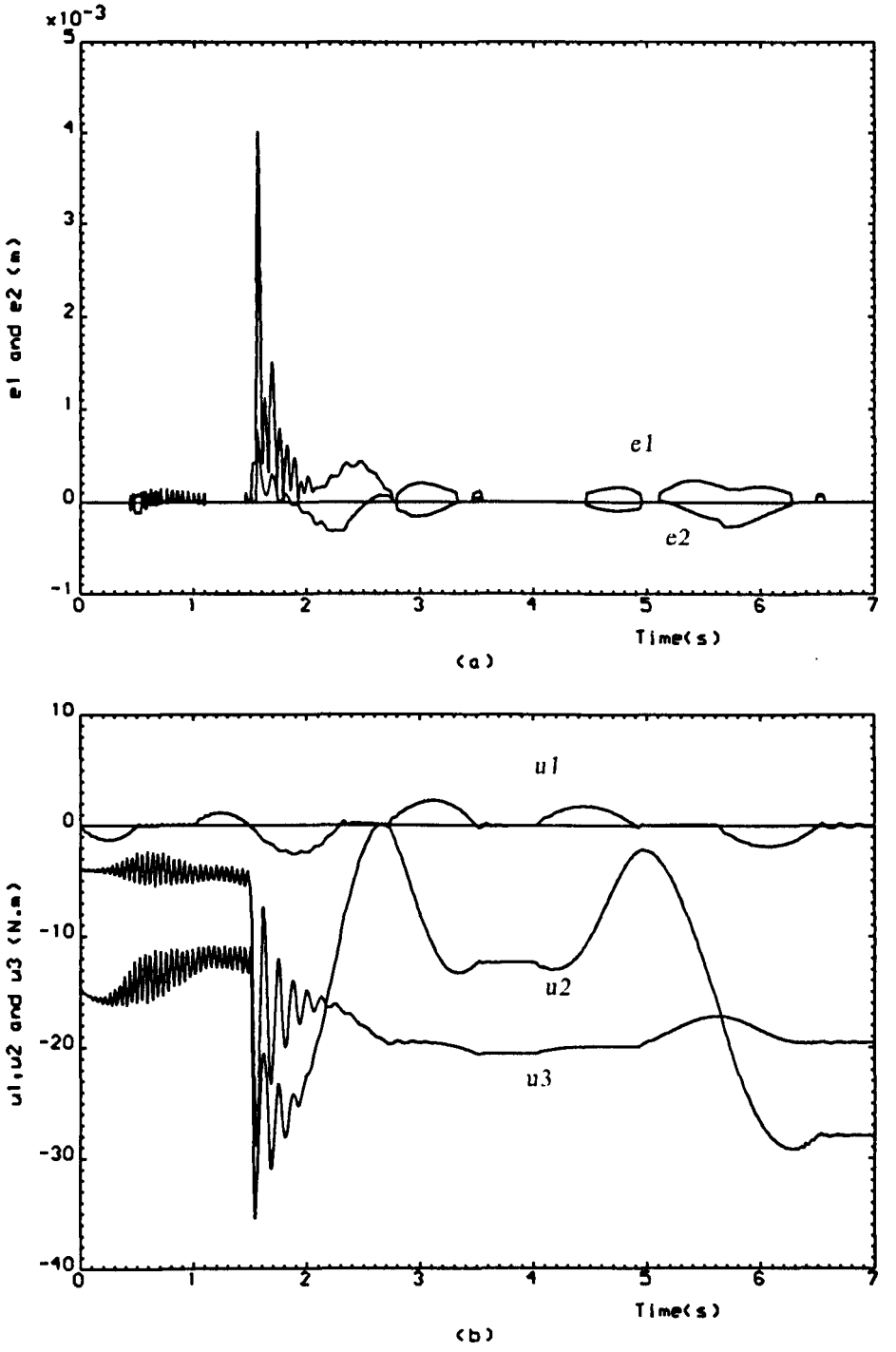

Figure 6.15: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 3 x 10^3;λ_3 = 0).

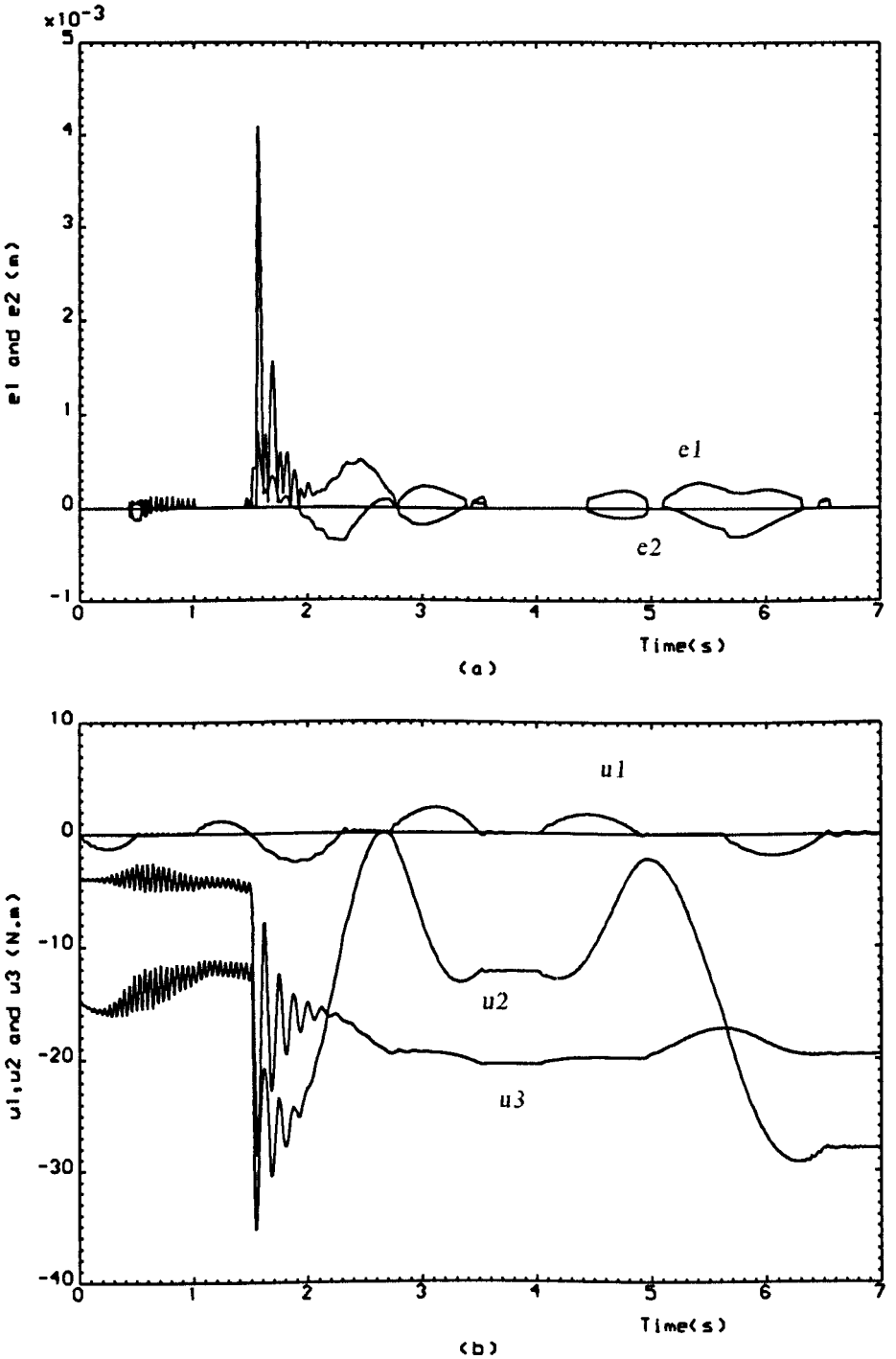

Figure 6.16: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 6 x 10^3;λ_3 = 0).

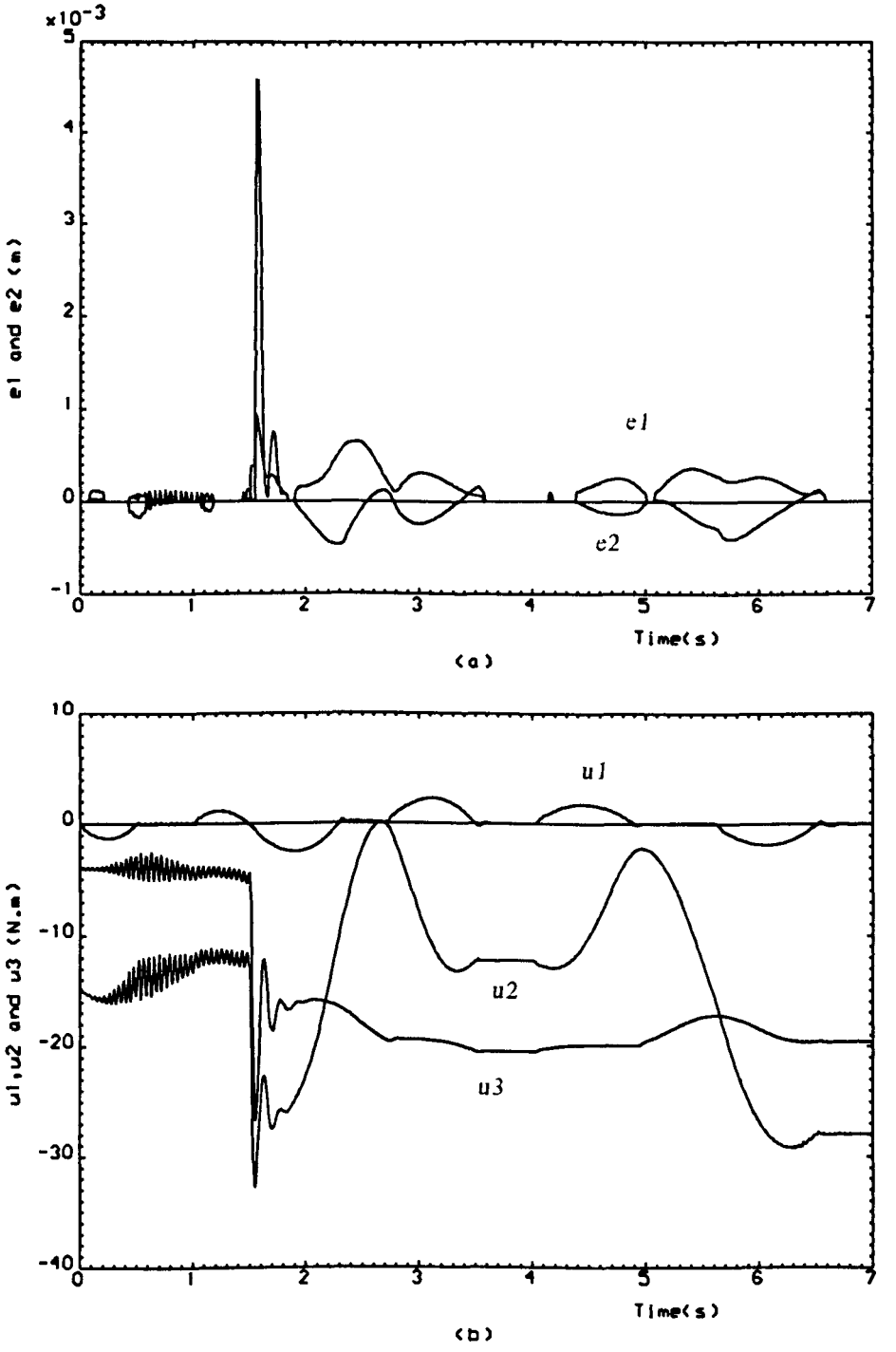

Figure 6.17: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 5 x 10^4;λ_3 = 0).

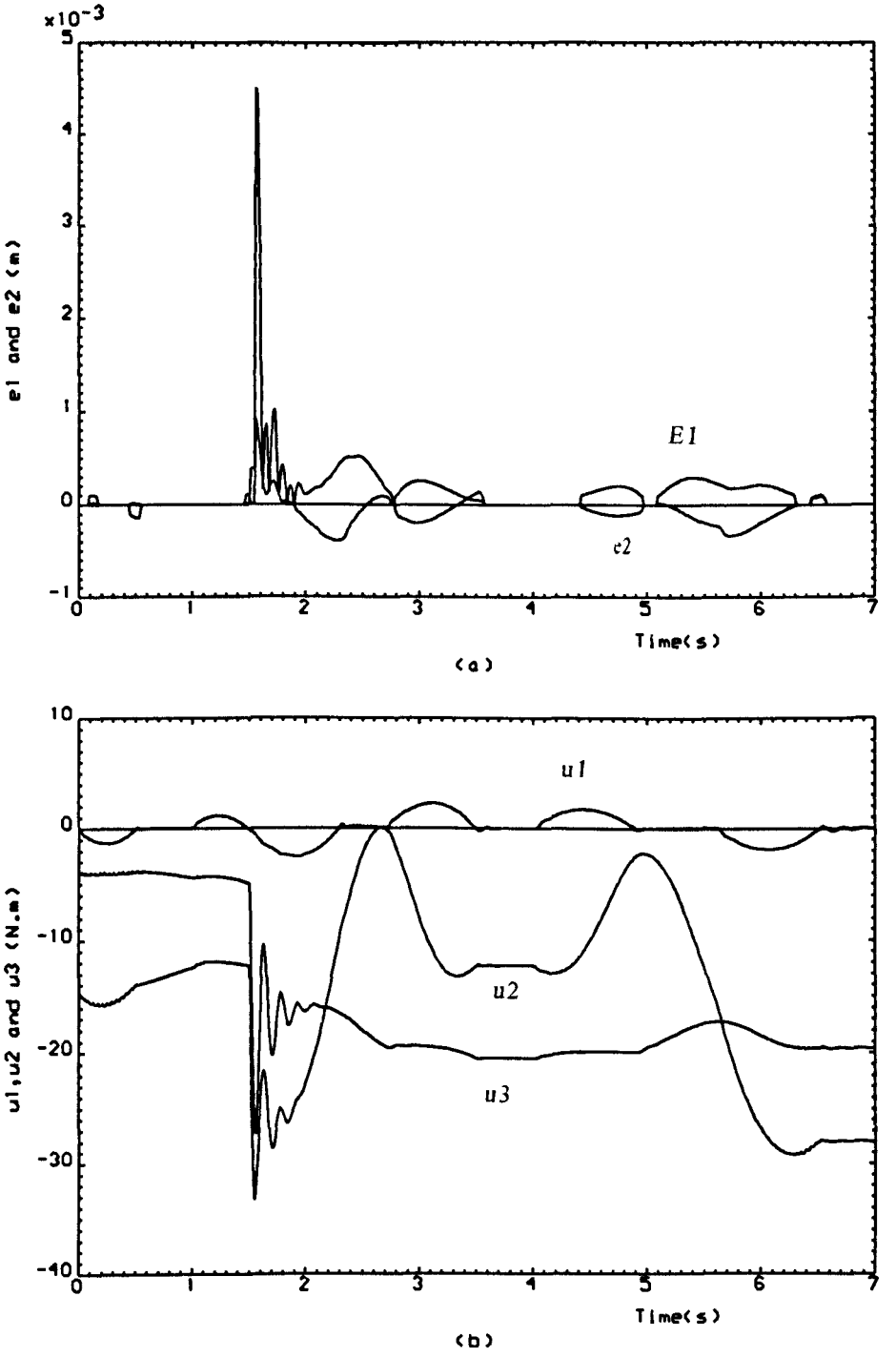

Figure 6.18: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 0;λ_3 = 10^{-3}).

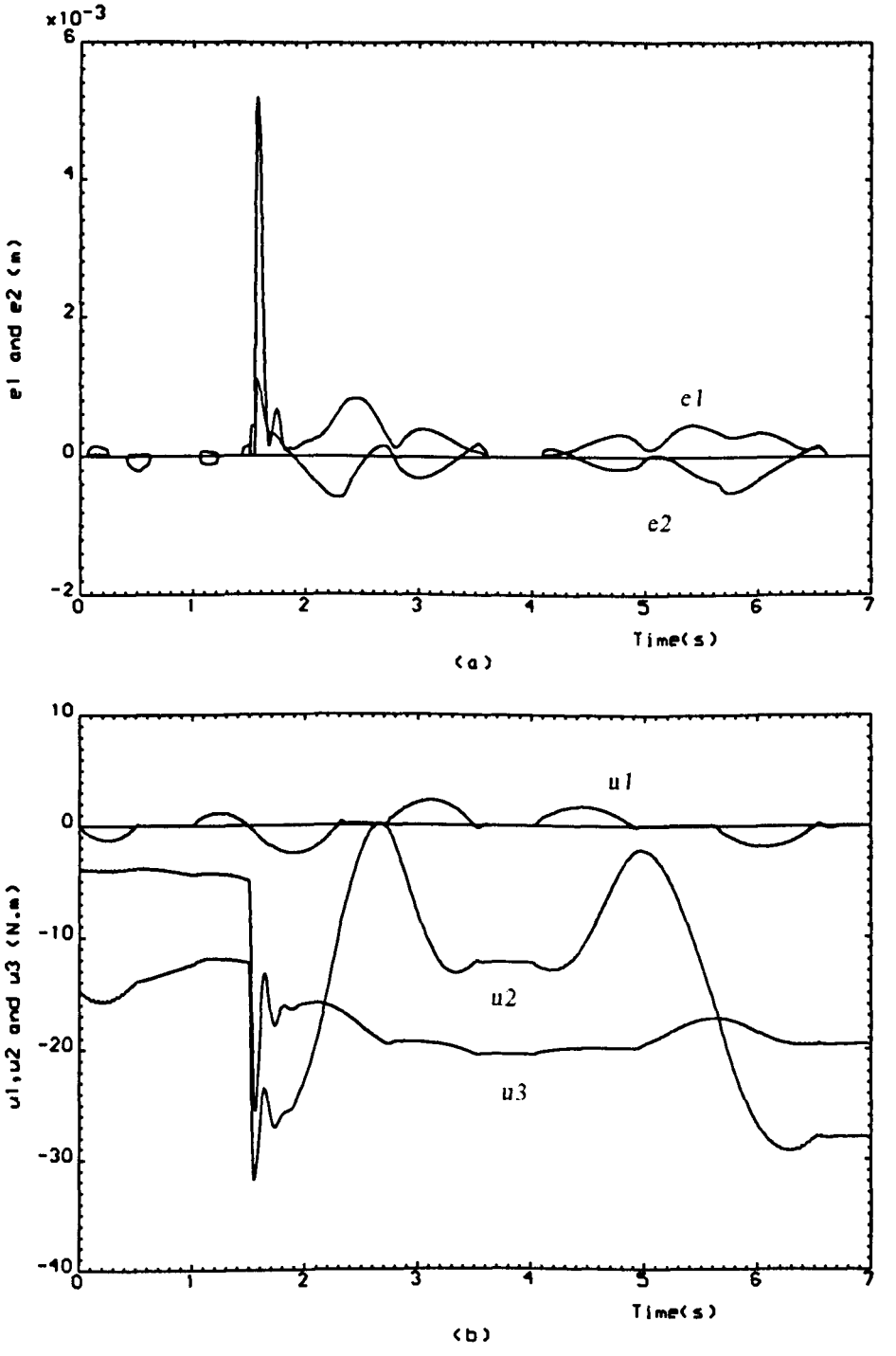

Figure 6.19: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 0;λ_3 = 5 x 10^{-3}).

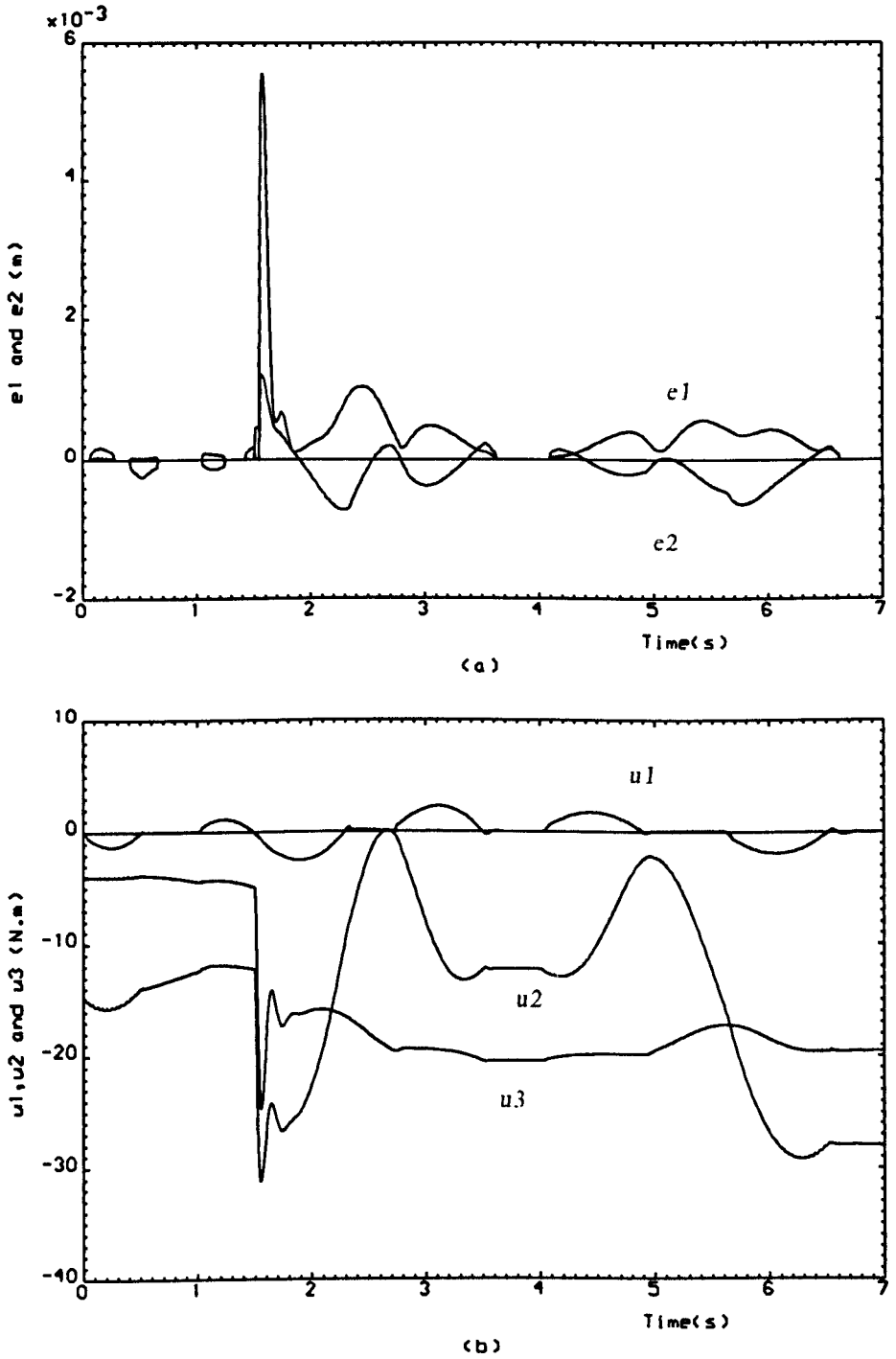

Figure 6.20: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 0;λ_3 = 10^{-2}).

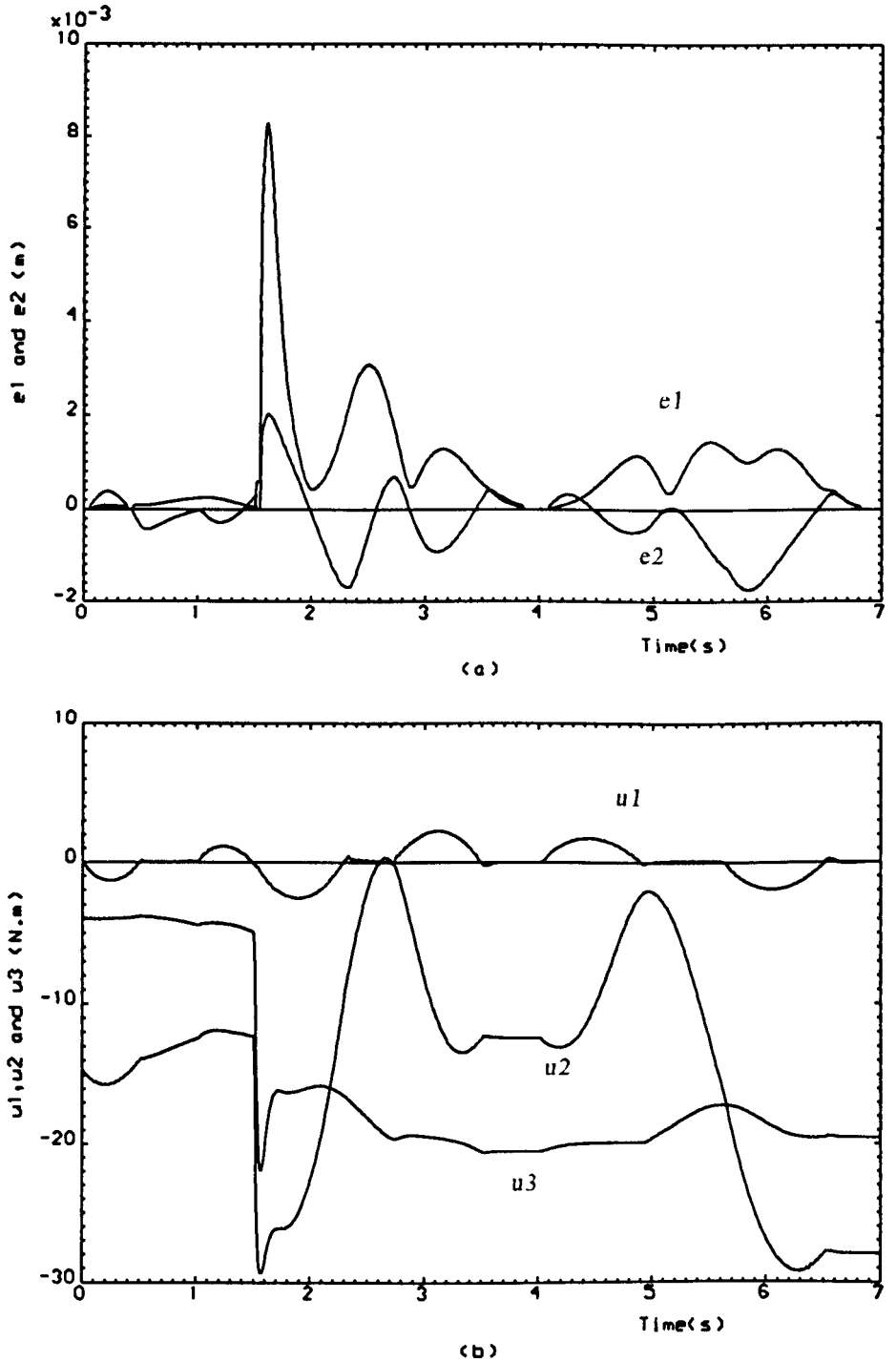

Figure 6.21: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 0;λ_3 = 10^{-1}).

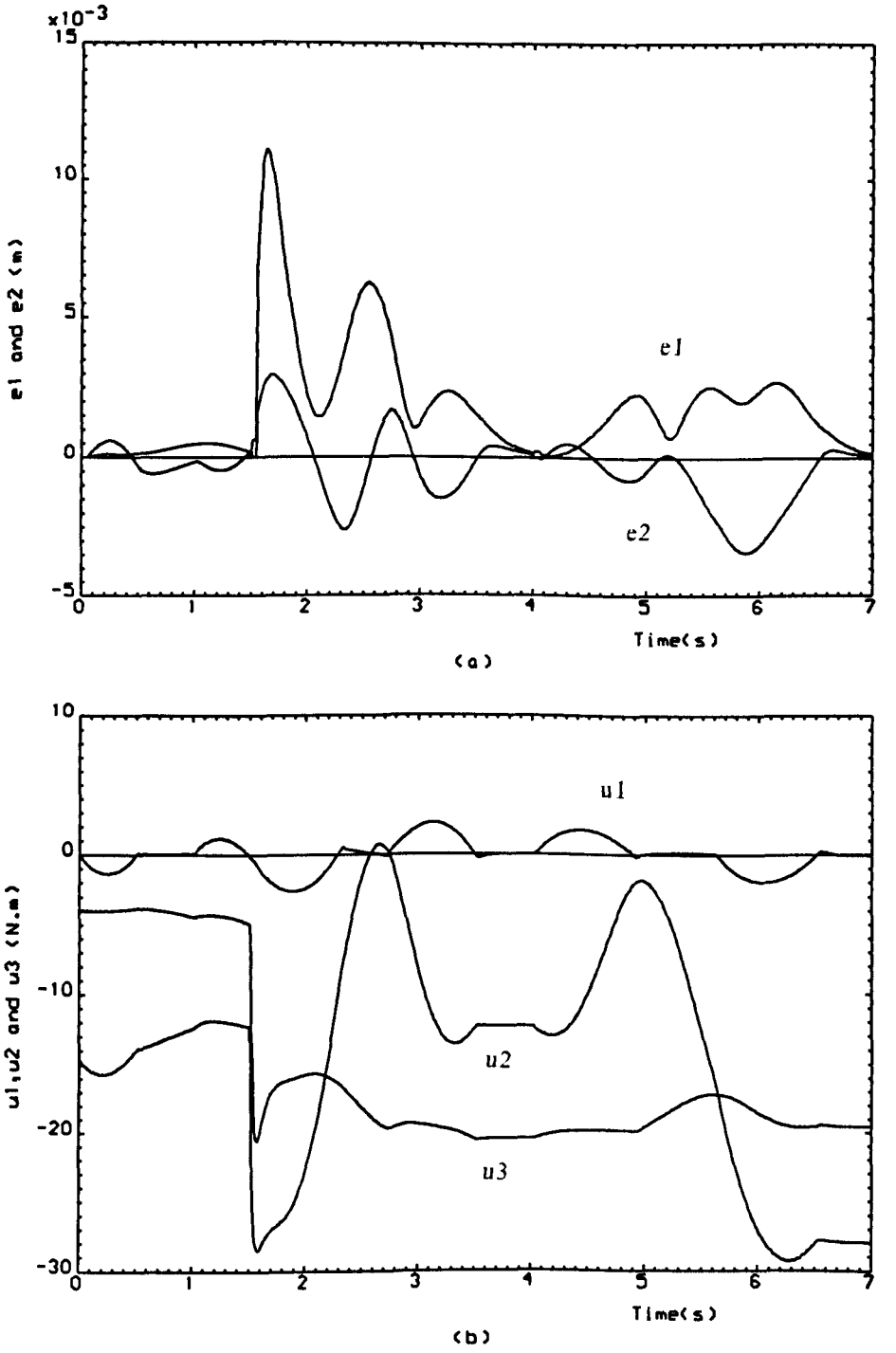

Figure 6.22: Time-domain behaviour of errors and torques for the non-adaptive Evolutionary Strategies design of a digital PID controller with (p_m = 0.02; λ_1 = 1;λ_2 = 0;λ_3 = 1).

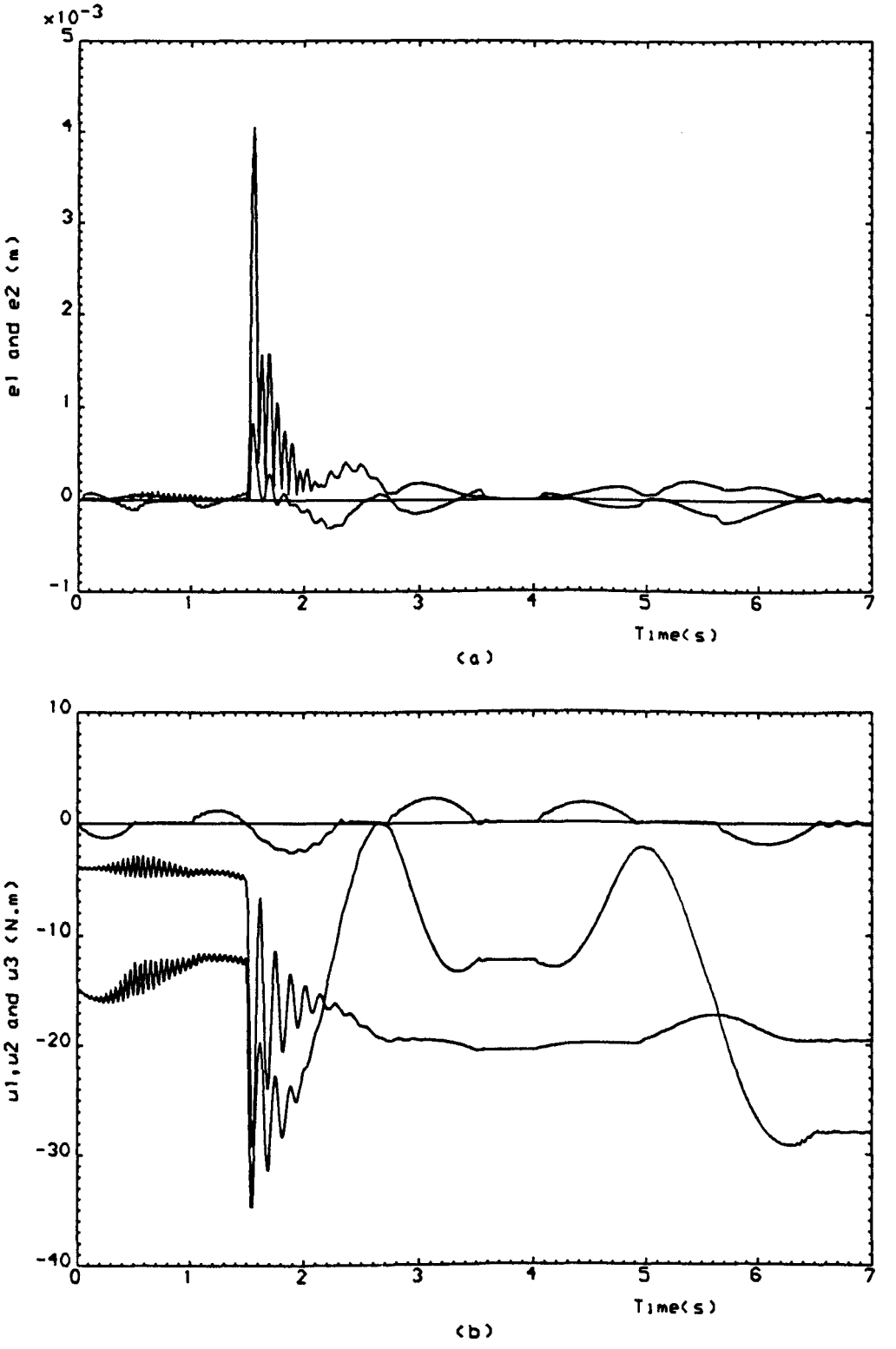

Figure 6.23: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1$; $\lambda_2 = 0$; $\lambda_3 = 0$).

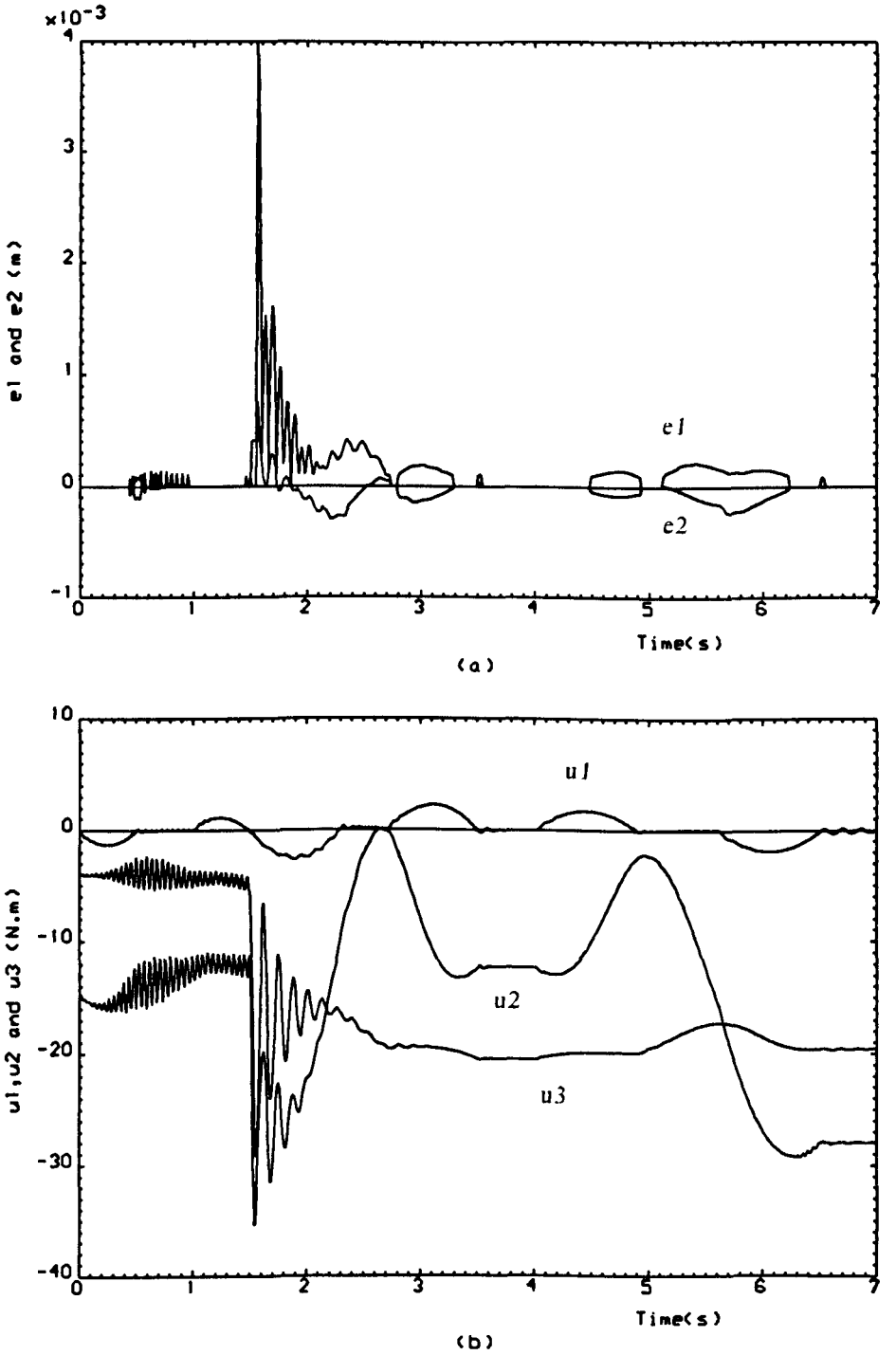

Figure 6.24: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1;\lambda_2 = 10^3;\lambda_3 = 0$).

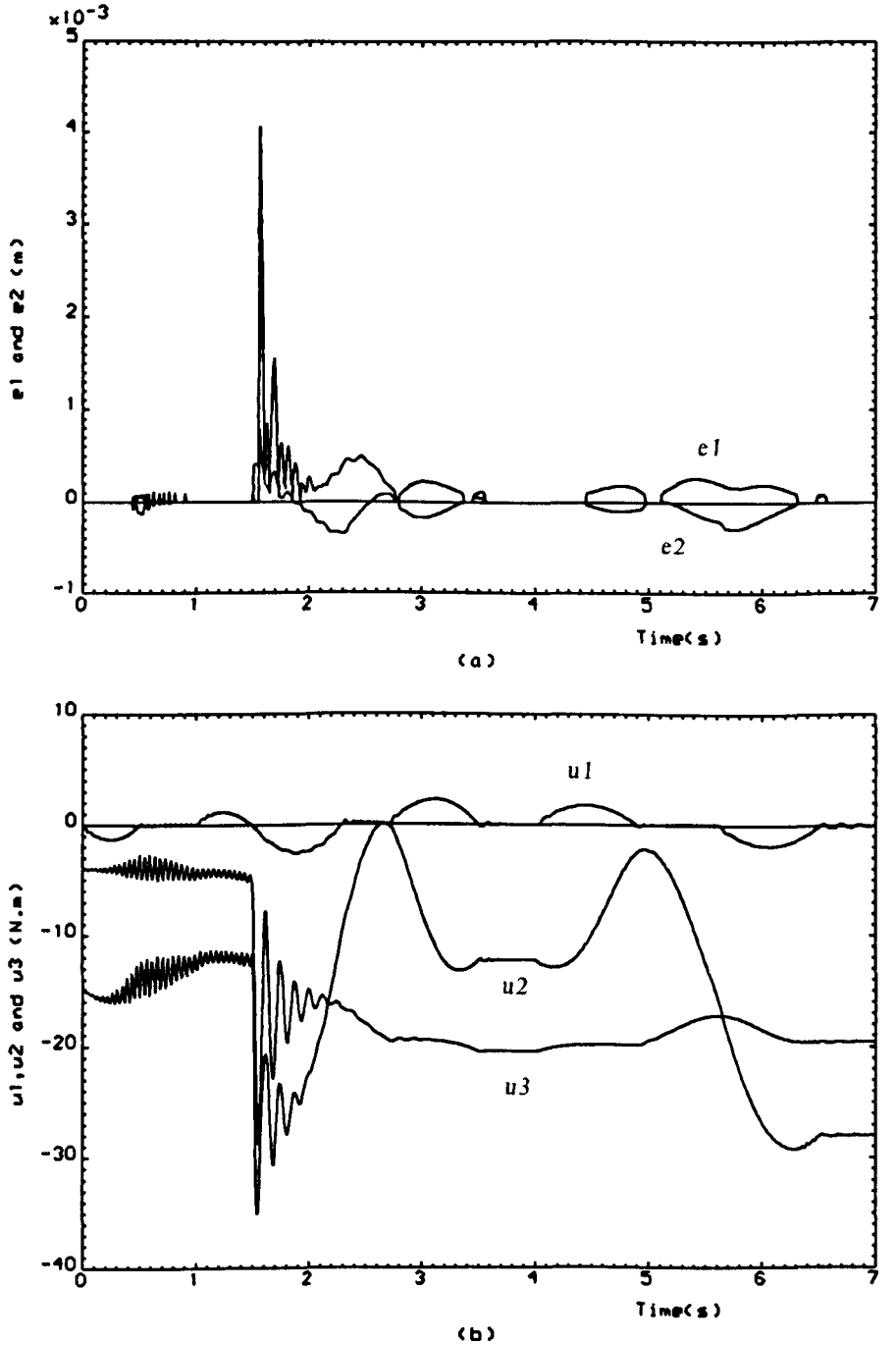

Figure 6.25: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 2 \times 10^3; \lambda_3 = 0$).

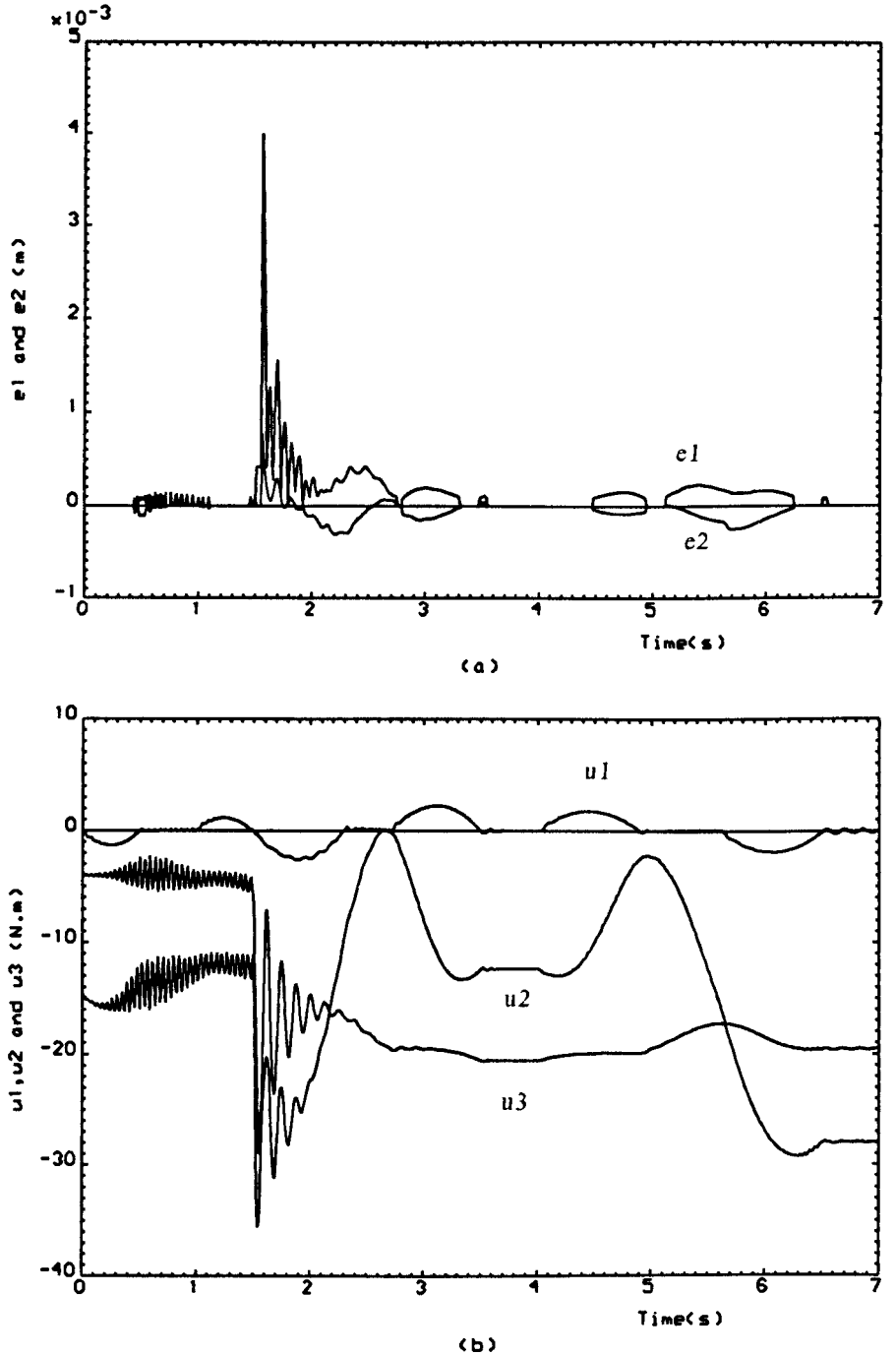

Figure 6.26: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of adigital PID controller with ($\lambda_1 = 1; \lambda_2 = 3 \times 10^3; \lambda_3 = 0$).

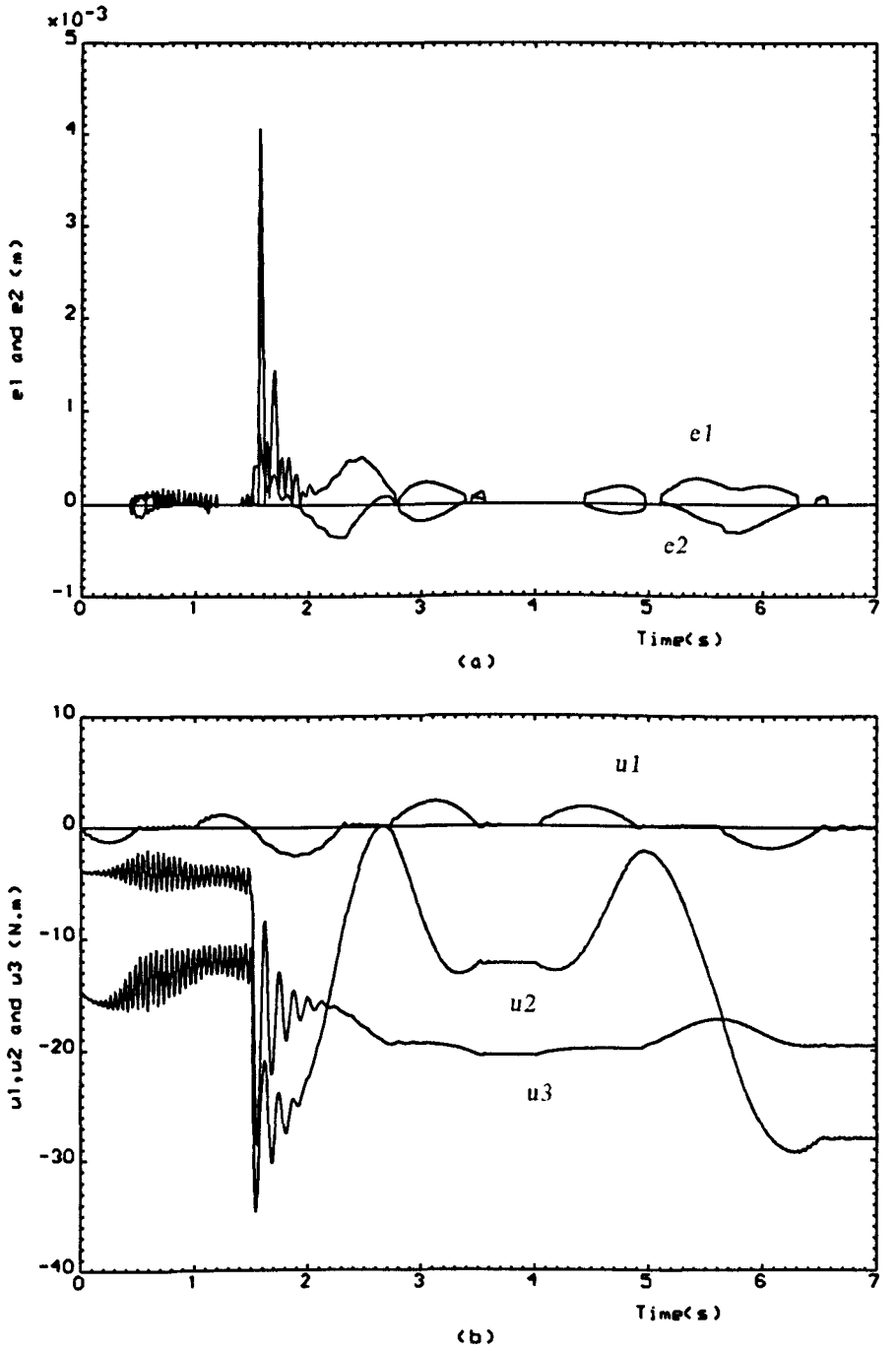

Figure 6.27: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 6 \times 10^3; \lambda_3 = 0$).

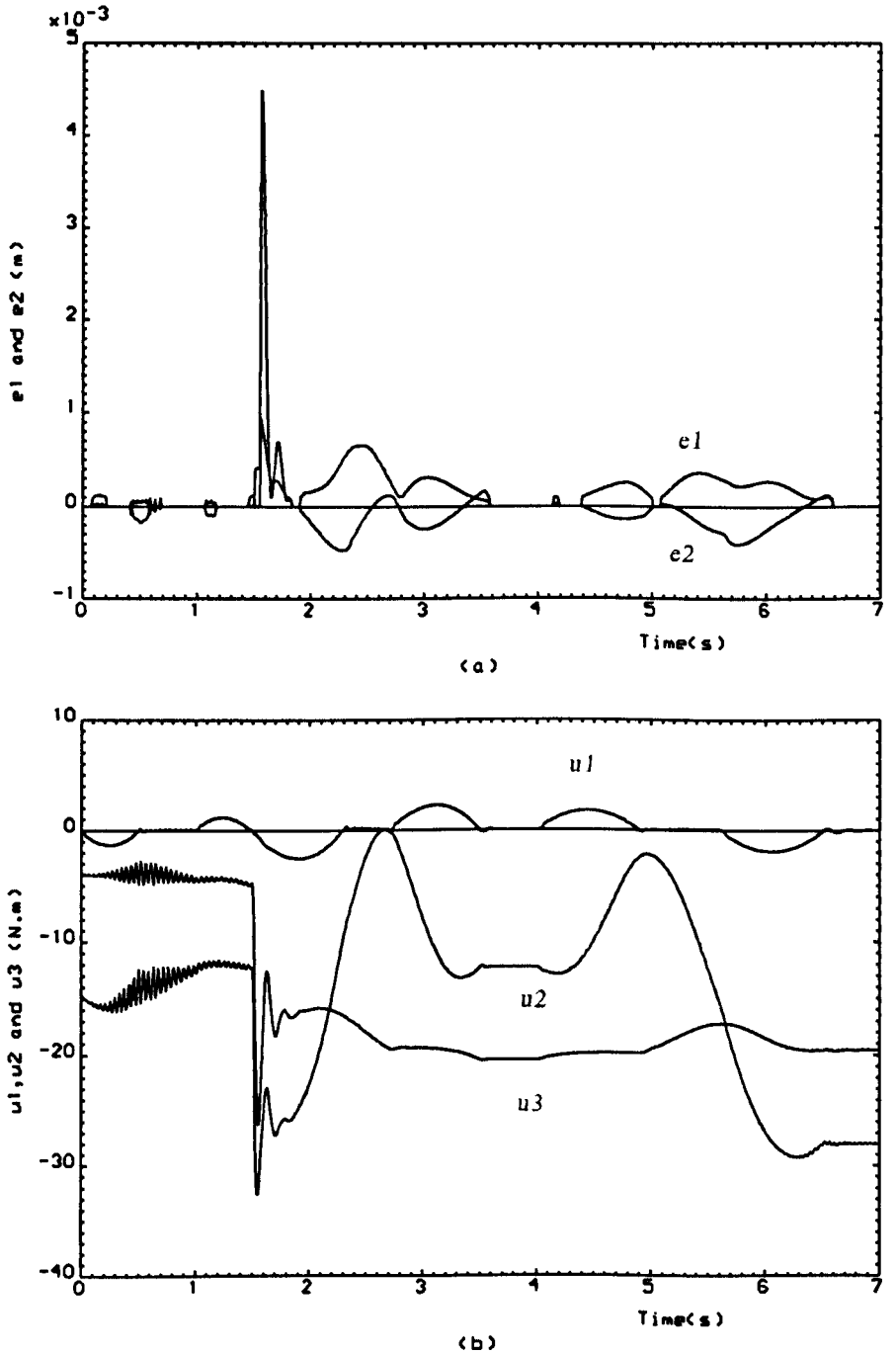

Figure 6.28: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 5 \times 10^4; \lambda_3 = 0$).

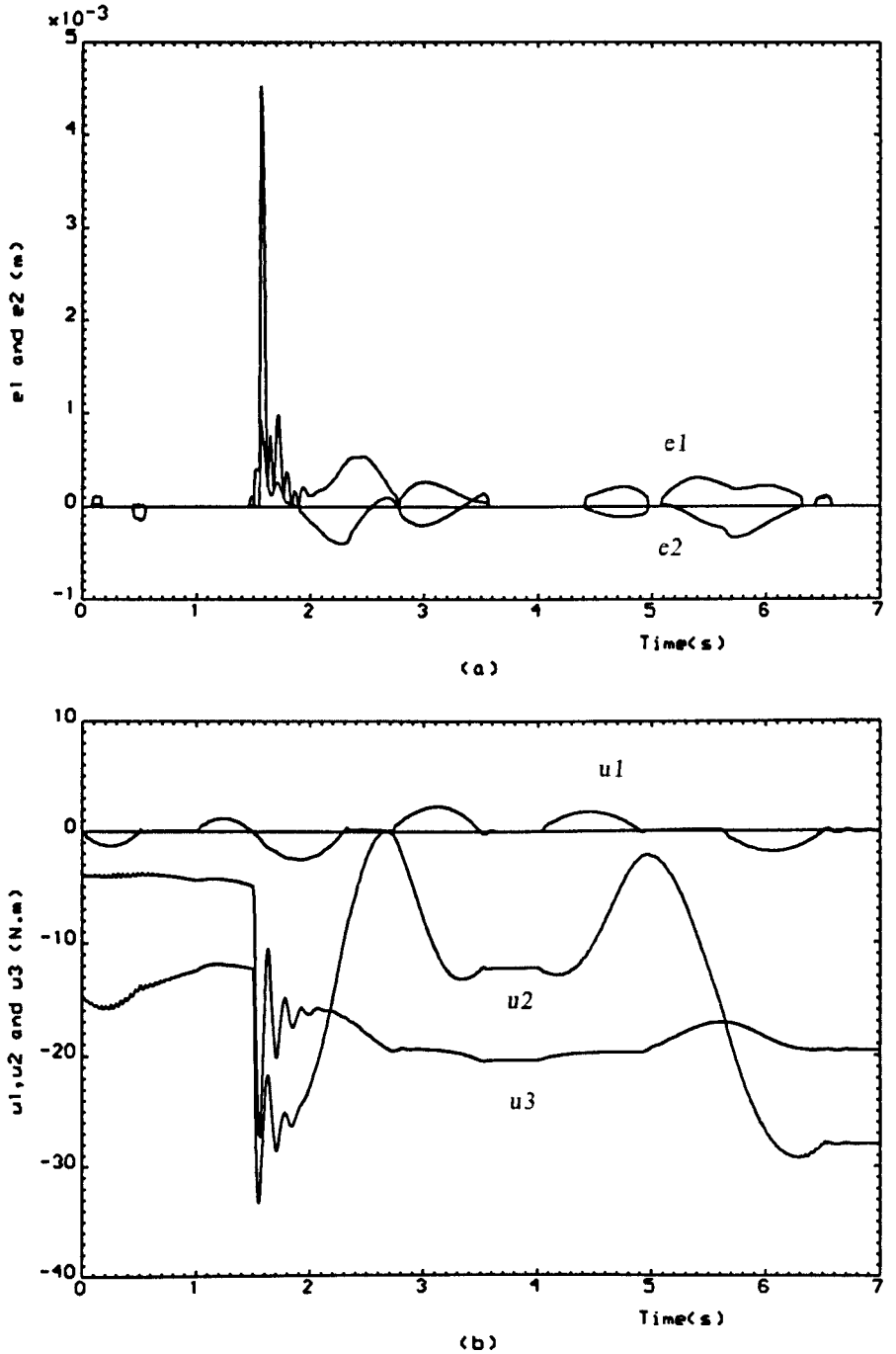

Figure 6.29: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1;\lambda_2 = 0;\lambda_3 = 10^{-3}$).

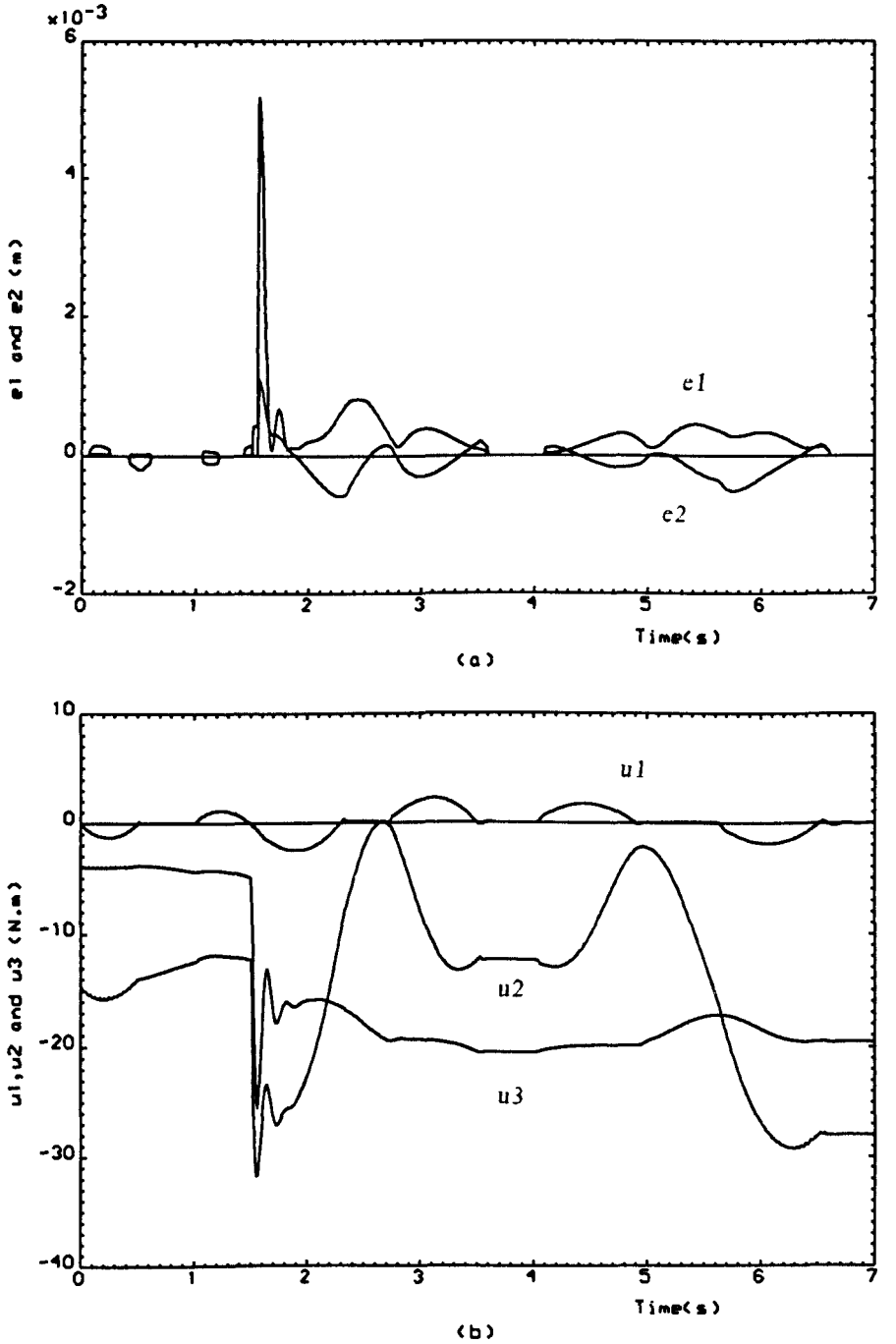

Figure 6.30: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 0; \lambda_3 = 5 \times 10^{-3}$).

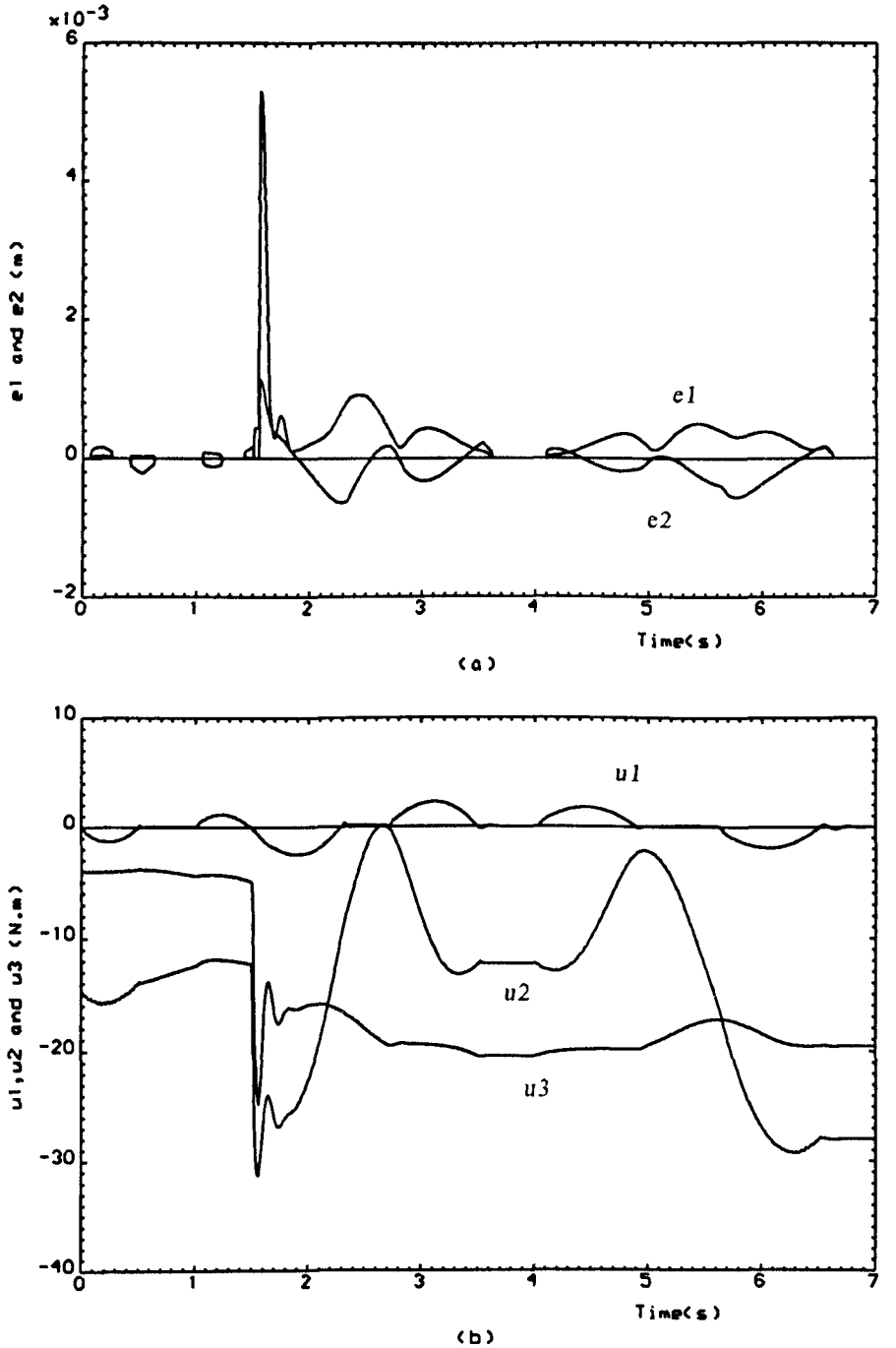

Figure 6.31: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1;\lambda_2 = 0;\lambda_3 = 10^{-2}$).

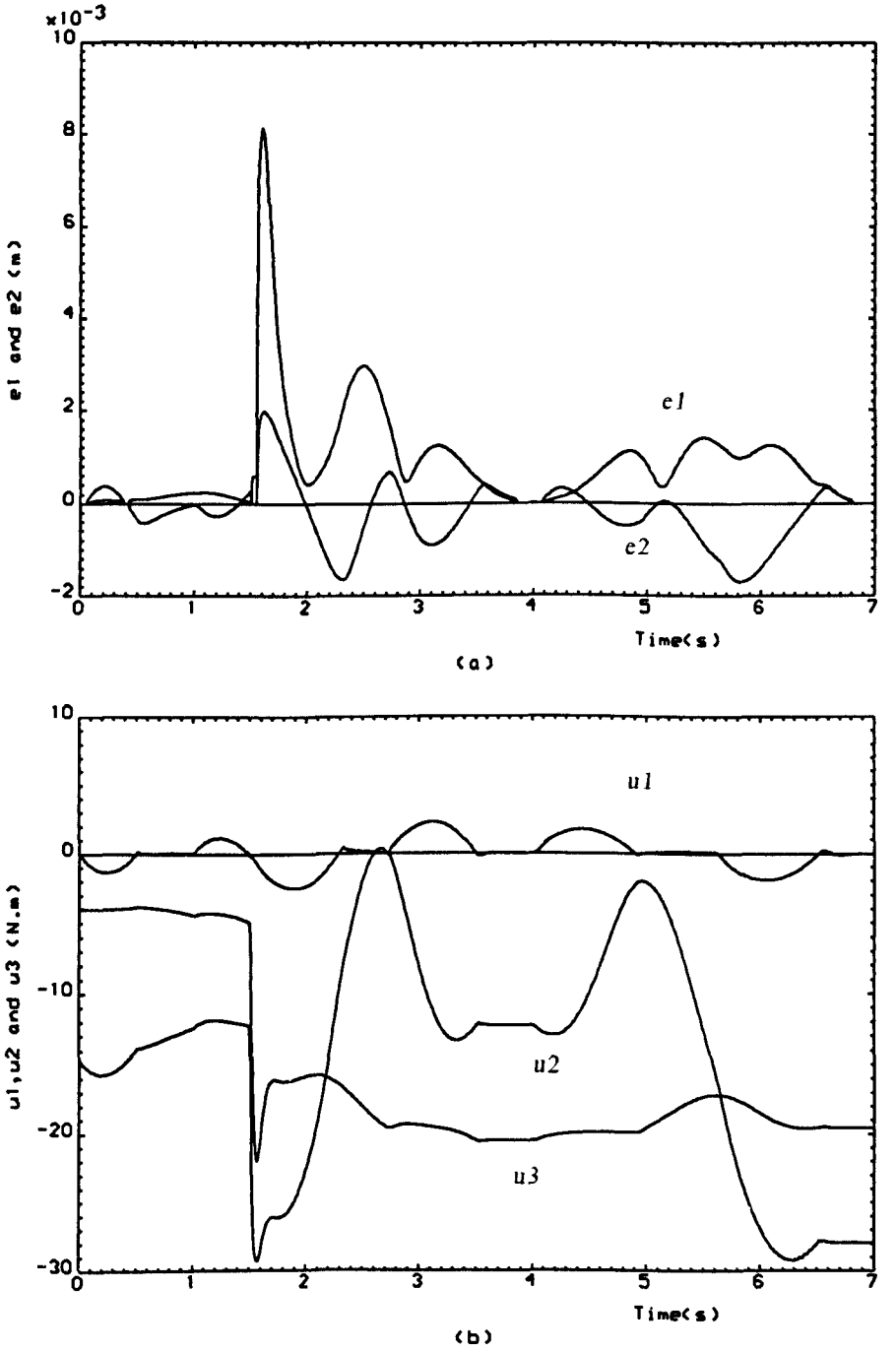

Figure 6.32: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1; \lambda_2 = 0; \lambda_3 = 10^{-1}$).

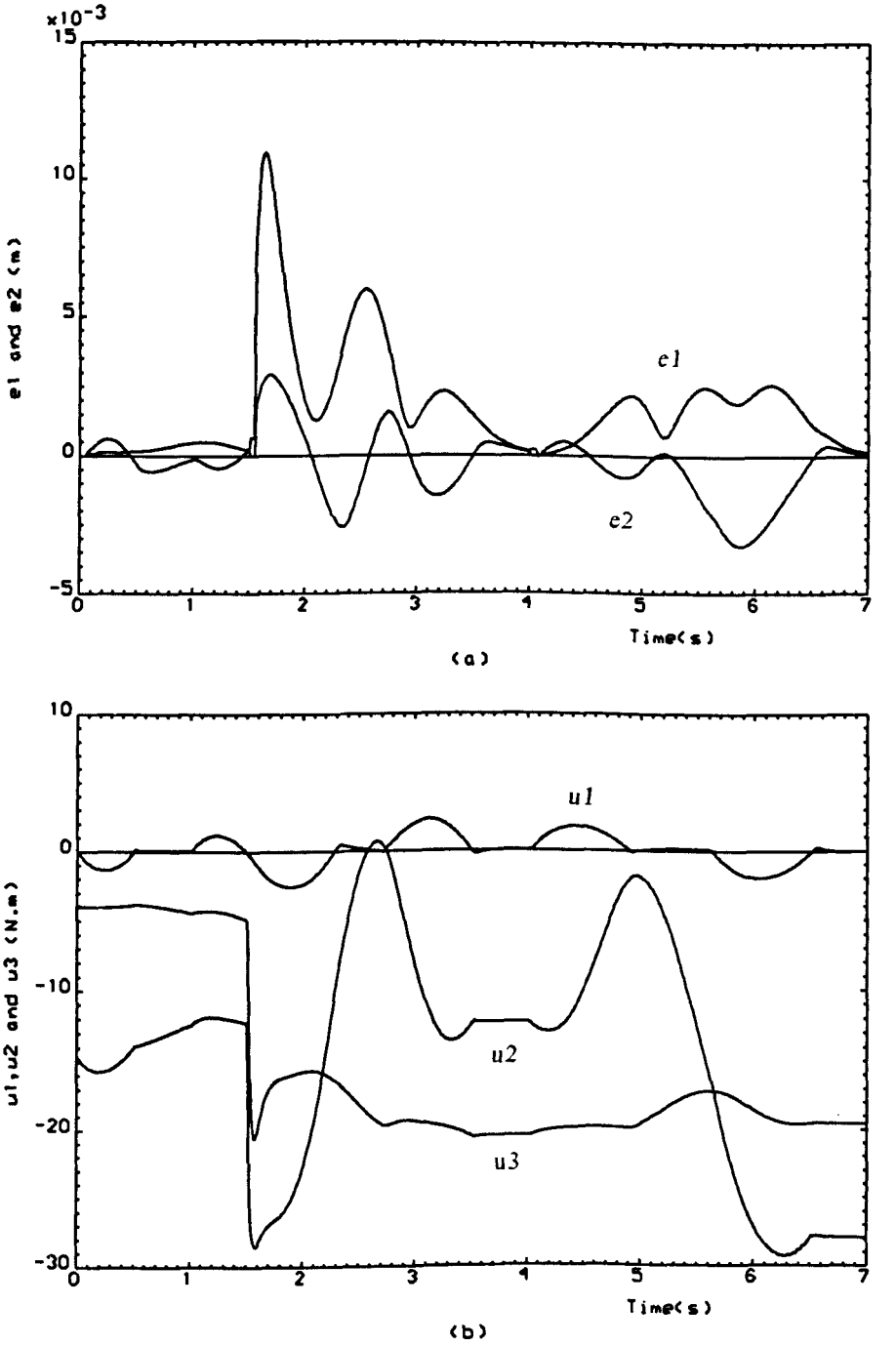

Figure 6.33: Time-domain behaviour of errors and torques for the adaptive Evolutionary Strategies design of a digital PID controller with ($\lambda_1 = 1;\lambda_2 = 0;\lambda_3 = 1$).

Chapter 7

EVALUATION OF EVOLUTIONARY DESIGNS OF DIGITAL CONTROLLERS FOR ROBOTIC MANIPULATORS

7.1 INTRODUCTION

In Chapters 4, 5 and 6, genetic algorithms, non-adaptive evolution strategies, and adaptive strategies were used to determine the optimal sets of parameter values for digital multivariable PID controllers for robotic manipulators. In this chapter, the performance of the evolutionary designed controllers is compared in the case of a typical three-degree-of-freedom robotic manipulator performing trajectory-tracking tasks with associated payload variations. This comparison considers both the quality of the results and the convenience associated with the use of these various evolutionary algorithms, as discussed by *Porter* and *Allaoui* (1998).

7.2 COMPARISON OF GENETIC ALGORITHMS AND EVOLUTION STRATEGIES IN CONTROLLERS DESIGNS

The design of digital PID controllers for robotic manipulator using evolutionary algorithms is an interesting test problem by which to asses the performance of all three evolutionary algorithms as used in Chapters 4 and 5. The analysis of the results of the different runs carried out in Chapters 4 and 5 is summarised in

Table 7.1. It is relevant to mention the fact that, for the sake of objectivity of comparison, the runs of the all three evolutionary algorithms were conducted while retaining the same bit representation, population size, initial population, limits of the search space, and number of generations. Indeed, in all cases each quadruple $\{\alpha, \sigma, \rho, \delta\}$ of PID controller parameters was represented by a binary string containing 56 bits; a population of 30 binary strings was caused to evolve; and evolution occurred for 50 generations. The results presented in Table 7.1 were obtained as an average over three runs for each individual case.

In this attempt to evaluate the performance of the designed PID controllers using all three evolutionary algorithms. it is necessary to point out that both genetic algorithms and non-adaptive (30 + 30)-evolution strategy required extensive experimentation to select respectively an appropriate crossover probability, p_c, and mutation probability, p_m, for genetic algorithms, and an appropriate mutation probability, p_m, for the non-adaptive evolution strategy. Thus, the presence of different mutation probabilities in Tables 7.1 and 7.2 for the genetic algorithm, and for the non-adaptive (30 + 30)-evolution strategy, is a clear indication of the importance of this parameter and the sensitivity of both these evolutionary algorithms to its variation.

In Table 7.1, the numbers of generation taken by the various evolutionary algorithms to reach within 10% of its final value are shown in a regular font. In addition, the associated percentage value representing the difference from the bench-mark which corresponds to the best of all trajectory-tracking performances —obtained using the adaptive (30 + 30)-evolution strategy— are

shown between brackets in Table 7.1. The quantitative study of the results depicted in Table 7.1 indicates the superiority of the trajectory-tracking performance of those digital PID controllers tuned using the adaptive (30 + 30)-evolution strategy over the other two alternative algorithms. Thus, for all three sampling periods T = 0.01s, T = 0.02s, and T = 0.04s, the superiority of the trajectory-tracking performance is confirmed for those designs obtained using adaptive (30 + 30)-evolution strategies. This superiority of the adaptive design is reflected in the time-domain plots as it can be seen from Figures 5.5.a, 5.5.b, and 5.5.c. However, this trajectory-tracking superiority is less evident compared to the best of the non-adaptive (30 + 30)-evolution strategy. However, the time-consuming trial-and-error process of choosing the mutation probability, p_m, implies that the non-adaptive evolution strategy is not as convenient as its adaptive counterpart.

Similarly, the study of the best-of-generation values of the cost function, Γ, for all three evolutionary algorithms presented in Chapters 4 and 5 confirms the interpretation of Table 7.1 previously stated in Chapter 5. Indeed, it was clearly stated that the number of generations taken for the best cost function of a particular evolutionary algorithm to attain 10% of its final optimal value is an indication of the rate of evolution of the evolutionary algorithm towards optimality. In addition, the difference between this final value and the benchmark optimal value of the cost function obtained using the evolution strategy is an indication of the quality of this evolution. The results of Table 7.1 indicate that —in all cases except one— optimisation of the PID controller is completed by the adaptive evolution strategy in fewest generations. This

adaptive evolution strategy is thus shown not only to be the best but also the most convenient algorithm to use, since it requires no *a priori* determination of crossover and mutation probabilities.

	Number of generations **(% difference from bench-mark)**		
Evolution algorithms	$T = 0.01$	$T = 0.02$	$T = 0.04$
Adaptive ES	13 **(0%)**	4 **(0%)**	6 **(0%)**
Non-adaptive ES ($p_m = 0.001$)	32 **(122.5%)**	29 **(23.4%)**	NA **(14.2%)**
Non-adaptive ES ($p_m = 0.02$)	20 **(1.6%)**	8 **(1.1%)**	7 **(0.2%)**
Non-adaptive ES ($p_m = 0.9$)	1 **(18.6%)**	NA **(21.7%)**	22 **(3.9%)**
Genetic Algorithm ($p_m = 0.008$)	32 **(13.9%)**	2 **(17.8%)**	10 **(2.2%)**
Genetic Algorithm ($p_m = 0.3$)	7 **(25.6%)**	10 **(8.0%)**	10 **(4.5%)**
Genetic Algorithm ($p_m = 0.98$)	8 **(28.7%)**	NA **(15.0%)**	6 **(4.5%)**

Table **7.1**: Number of generations required for the algorithm to reach within 10% of its final value and percentage difference from the bench-mark adaptive optimal values.

In Table 7.2, the average running-time of each individual case is given, together with the confirmation, or not, of the necessity to proceed with the pre-selection of a required crossover probability p_c and/or mutation probability p_m. Indeed, the pre-selection of the crossover and/or mutation probabilities introduces a hidden

time cost accumulated while proceeding with this trial-and-error selection process. However, the adaptive (30 + 30)-evolution strategy does not require the designer to indulge in such a time-consuming process, in view of the automated mutation mechanism incorporated into the evolutionary algorithm.

	Running-Time (**Is crossover/mutation pre-tuning required?**)		
Evolution algorithms	T = 0.01	T = 0.02	T = 0.04
Adaptive ES	1.9 Hrs **(No)**	2.5 Hrs **(No)**	3.1 Hrs **(No)**
Non-adaptive ES (p_m = 0.001)	3.2 Hrs **(Yes)**	3.5 Hrs **(Yes)**	3.9 Hrs **(Yes)**
Non-adaptive ES (p_m = 0.02)	2.8 Hrs **(Yes)**	3.1 Hrs **(Yes)**	3.6 Hrs **(Yes)**
Non-adaptive ES (p_m = 0.9)	3.8 Hrs **(Yes)**	4.0 Hrs **(Yes)**	4.3 Hrs **(Yes)**
Genetic Algorithm (p_m = 0.008)	3.2 Hrs **(Yes)**	3.9 Hrs **(Yes)**	5.3 Hrs **(Yes)**
Genetic Algorithm (p_m = 0.3)	4.2 Hrs **(Yes)**	4.7 Hrs **(Yes)**	5.6 Hrs **(Yes)**
Genetic Algorithm (p_m = 0.98)	5.0 Hrs **(Yes)**	5.3 Hrs **(Yes)**	5.5 Hrs **(Yes)**

Table **7.2**: Running-time for completion of the evolutionary algorithms and pre-tuning of crossover and/or mutation probabilities.

It emerges from the combined analysis of Tables 7.1 and 7.2 that, by eliminating chromosomal recombination in the form of crossover (present in genetic algorithms) and by incorporating time-varying mutation probabilities as

an integral part of the strategy (as opposed to the fixed mutation probability present in the non-adaptive evolution strategies), a simplified and convenient form of adaptive evolutionary algorithm can produce the best results in the most effective way. Thus, the worst performance is attributed to the genetically tuned controllers, followed by the non-adaptive evolution strategies as second best to the adaptive evolution strategies.

7.3 CONCLUSION

In this chapter, the performance of genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies has been compared in the case of a typical robotic manipulator performing trajectory-tracking tasks. It has been shown, by comparing the performance of the various evolutionary algorithms used in Chapters 4 and 5 for the design optimal of digital trajectory-tracking PID controllers, that the adaptive evolution strategy produces the best results and also that it is the most convenient algorithm to use.

PART IV

UNCONSTRAINED DESIGN OF DIGITAL TRAJECTORY-TRACKING CONTROLLERS USING EVOLUTIONARY ALGORITHMS

Chapter 8

UNCONSTRAINED DESIGN OF DIGITAL CONTROLLERS FOR ROBOTIC MANIPULATORS USING EVOLUTIONARY ALGORITHMS

8.1 INTRODUCTION

In Chapters 4, 5 and 6, genetic algorithms and evolution strategies were used to design robust digital trajectory-tracking controllers for robotic manipulators. Indeed, in all the designs presented in Chapters 4, 5 and 6, the PID controller design equations obtained from singular perturbation theory were incorporated into the evolutionary algorithms in order to obtain the optimal quadruple $\{\alpha, \sigma, \rho, \delta\}$ of controller design parameters. This clearly constrains the resulting design to lie in a sub-space of the entire space of design parameters.

It is accordingly proposed in this chapter to address the problem of designing *unconstrained* fast-sampling digital PID controllers. Indeed, removing the constraints imposed on the structure of the controller results in the incorporation of all the elements of the controller matrices into the evolutionary algorithms rather than just the quadruple $\{\alpha, \sigma, \rho, \delta\}$ of controller design parameters as is the case in the *constrained* design. Hence, this *unconstrained* configuration introduces a degree of complexity characterised by the high dimensionality of its search space. This is considered to provide an appropriate test of the performance of all three proposed evolutionary algorithms, namely, genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies.

The present *unconstrained* controller design methodology is illustrated for the three-degree-of-freedom robotic manipulator previously considered in Chapters 3, 4, 5 and 6, so that comparison can then be made with the evolutionarily tuned *constrained* designs presented in Chapters 4 and 5.

8.2 UNCONSTRAINED EVOLUTIONARY DESIGN PROCEDURE

The trajectory-tracking systems under consideration are controlled by digital PID controllers governed by control-law equation of the form

$$u(kT) = K_1 r(kT) + K_2 z(kT) \quad . \tag{8.1}$$

In equation (8.1), $K_1 \in \Re^{l \times l}$ and $K_2 \in \Re^{l \times l}$ are the proportional and integral controller matrices, whilst the vectors $r(kT) \in \Re^l$ and $z(kT) \in \Re^l$ are generated in accordance with the difference equations [*Porter* (1987)]

$$s\{(kT + 1)T\} = -\alpha s(kT) + e(kT) , \tag{8.2}$$

$$r(kT) = -\frac{2}{T}(1 + \alpha)Ds(kT) + \left(I_l + \frac{2}{T}D\right) e(kT), \tag{8.3}$$

and

$$z\{(k + 1)T\} = z(kT) + Tr(kT) . \tag{8.4}$$

Moreover, in equations (8.2), (8.3) and (8.4), $\alpha \in (-1, +1]$, $s(kT) \in \Re^l$, $e(kT) = v(kT) - y(kT) \in \Re^l$ is the error vector, $v(kT)$ is the set-point command vector, and the derivative matrix $D \in \Re^{l \times l}$.

In Chapters 4, 5 and 6, it was shown that the design of *constrained* digital PID controllers is amenable to the evolutionary tuning of the quadruple $\{\alpha, \sigma, \rho, \delta\}$ of controller parameters. This *constrained* configuration of the controller (see equations (4.11) and (4.12)) introduces constraints on the possible values that the controller matrices K_1, K_2, and D can take, thus restricting the search space for the evolutionary algorithms. However, adopting the *unconstrained* configuration of the digital PID controller permits the tuning of all individual controller parameters. This evidently requires the incorporation of the entire controller matrices K_1, K_2, and D, together with the controller parameter α, into the evolutionary algorithms.

In order to maintain objectivity in the comparison of the performance of the *unconstrained* evolutionary designed controllers with the performance of their *constrained* counterparts, the trajectory-tracking performance measure used in Chapters 4 and 5 is retained. This implies that minimisation of the performance measure

$$\Gamma = \int_0^{\tau} \|e(t)\| \; dt \tag{8.5}$$

is regarded as the ultimate design requirement.

8.3 ILLUSTRATIVE EXAMPLE

8.3.1 SYSTEM DESCRIPTION

The procedure for the evolutionary design of *unconstrained* digital PID controllers can be conveniently illustrated by considering the three-degree-of-

freedom robotic manipulator for which digital PID controllers were previously designed (using a *constrained* structure as opposed to an *unconstrained* structure) in Chapters 4, 5, and 6.

The robotic manipulator in question is governed on T = [0, +∞] by state and output equations of the respective forms [*Petropoulakis* (1986)]

$$M(\theta)\ddot{\theta}+c(\theta,\dot{\theta})+g(\theta)=u \quad , \tag{8.6a}$$

and

$$y=f(\theta) \quad . \tag{8.6b}$$

The numerical values of the inertial and kinematic parameters of the typical three-degree-of-freedom robotic manipulator under consideration are those given by *Petropoulakis* (1986) and presented in Chapter 3. The evolutionarily designed controllers are used while the end-effector of the robotic manipulator is caused to track the straight-line trajectories defined in Chapter 3.

The complexity of the design task associated with the *unconstrained* case results from the enlargement of the hyper-space of search to accommodate the twenty eight (28) different controller parameters to be tuned using, in turn, genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies. Hence, this increased dimensionality increases the difficulty of the design task compared with the *constrained* case, in which only the quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller parameters was tuned. Indeed, the increased number of

parameters of the digital PID controller arises from the *unconstrained* structure of the controller parameter set $\{K_1, K_2, D, \alpha\}$, where

$$K_1 = \begin{bmatrix} K_{11}, & K_{12}, & K_{13} \\ K_{14}, & K_{15}, & K_{16} \\ K_{17}, & K_{18}, & K_{19} \end{bmatrix}, \tag{8.7}$$

$$K_2 = \begin{bmatrix} K_{21}, & K_{22}, & K_{23} \\ K_{24}, & K_{25}, & K_{26} \\ K_{27}, & K_{28}, & K_{29} \end{bmatrix}, \tag{8.8}$$

and

$$D = \begin{bmatrix} D_1, & D_2, & D_3 \\ D_4, & D_5, & D_6 \\ D_7, & D_8, & D_9 \end{bmatrix}. \tag{8.9}$$

It is clear that, during this design process, each element of each of the different controller matrices will be individually tuned so as to minimise the performance measure of equation (8.5).

In the same way as was used in Chapters 4, 5, and 6 in the case of evolutionary constrained design, it is now necessary to encode the set $\{K_1, K_2, D, \alpha\}$ of controller design parameters in accordance with a system of concatenated, multi parameter, mapped, fixed-point coding. Thus, each set $\{K_1, K_2, D, \alpha\}$ of controller parameters is represented by a string of binary digits. Then, following

any random choice of the initial generation of such strings, successive generations of strings are obtained using the genetic operations of selection and mutation in the case of (λ + μ)-evolution strategies, and of selection, crossover and mutation in the case of genetic algorithms. These operations ensure that the successive generations of error-actuated digital PID controller thus produced by the genetic algorithms and (λ + μ)-evolution strategies tend to exhibit improving trajectory-tracking performance with respect to the performance measure of equation (8.5)

In the design studies, all three evolutionary techniques were used with a 10 bit representation for each of the twenty eight (28) controller parameters; and evolution occurred over 100 generations for the 30 individuals making up the population.

8.3.2 GENETIC ALGORITHM

In the genetic design procedure, it is evident that the time-consuming process of selecting an appropriate crossover probability, p_c = 0.6, and an appropriate mutation probability, p_m = 0.01, is unavoidable. In addition, care must be taken in selecting the values of the intervals within which each individual controller parameter is allowed to vary. Indeed, the bigger the intervals, the bigger is the hyper-space of search and the greater is the tendency for the genetic algorithms to encounter unfit solutions.

It is clear from Table 8.1 which represents results of an average over three runs for the sampling periods T = 0.01s, T = 0.02s, and T = 0.04s that, in the *unconstrained* case, genetic algorithms were successful in improving the trajectory-tracking performance of the robotic manipulator. Thus, Figures 8.2(a), 8.3(a) and 8.4(a) indicate a significant improvement in the trajectory-tracking performance for all three sampling periods T = 0.01s, T = 0.02s, and T = 0.04s. Indeed, the present tracking performance is the best obtained so far. The essential features of the designs for these sampling periods are summarised in Table 8.1, where the values of the performance measure, Γ, shown in bold type are those obtained after the first 50 generations. These *unconstrained* designs thus represent improvements in performance varying between 30.19% to 60.68% in comparison with their *constrained* counterparts of Chapter 4. However, the time-domain behaviour of the torques depicted in Figures 8.2(b), 8.3(b), and 8.4(b) reveals the existence of undesirable high-frequency content. This genetic design is therefore unlikely to be favoured by practical control designers despite the excellence of its tracking performance.

In addition, genetic algorithms failed to meet the time requirement since the average time required to complete the first 50 generations of the 100 generations-long evolutionary process varied between 12 and 16 hours (compared to the 3 to 5 hours taken by the genetic algorithms to complete 50 generations in the *constrained* case of Chapter 4). This is considered to be too long to be attractive for practical use. It is also relevant to mention that an increase in the value of the sampling period, T, was accompanied by an

increase in the time taken by the genetic algorithm to complete the entire evolutionary process. Indeed, this increase in the duration of the evolutionary process can be attributed to the fact that, as the sampling period T is increased, the number of possible 'bad' designs increases within the search space. Hence, the evolutionary process becomes inclined to frequent rejections of undesirable 'bad' designs, which in turn extends the time required to complete the evolutionary process.

	Performance measure, Γ		
Design structure	$T = 0.01$	$T = 0.02$	$T = 0.04$
Unconstrained design	5.78×10^{-4} **$(5.78 \times 10^{-4})^*$**	7.36×10^{-3} **$(7.36 \times 10^{-3})^*$**	4.32×10^{-2} **$(4.32 \times 10^{-2})^*$**
Constrained design	10.47×10^{-4}	10.53×10^{-3}	6.2×10^{-2}
Relative improvement	60.68%	30.1%	30.67%

Table **8.1** Comparison of genetically designed controllers.
(*): Values after 50 generations.

The best-of-generation and average-of-generation plots depicted in Figures 8.11, 8.12, and 8.13 reveal a pattern of premature convergence despite the fact that the genetic algorithms were allowed to evolve for 100 generations. Indeed, the presence of fluctuations in the average-of-generation plots (see Figures 8.11(b), 8.12(b) and 8.13(b)) clearly indicates the difficulties encountered by the genetic algorithms in producing 'good' designs during the evolutionary process.

8.3.3 NON-ADAPTIVE EVOLUTION STRATEGY

In the same way as in the *unconstrained* genetic design of PID trajectory-tracking controllers, it is necessary to proceed by trial-and-error in order to select an appropriate value of the mutation probability, $p_m = 0.005$, for the non-adaptive evolution strategy. The results of an average over three runs for the sampling periods $T = 0.01$s, $T = 0.02$s, and $T = 0.04$s are summarised in Table 8.2, where the values of Γ shown in bold are those obtained after the first 50 generations. These *unconstrained* designs obtained using non-adaptive (30 + 30)-evolution strategies represent improvements in the trajectory-tracking performance in comparison with the *constrained* evolutionary counterparts of Chapter 5. Indeed, Table 8.2 indicates for the three sampling periods $T = 0.01$s, $T = 0.02$s, and $T = 0.04$s improvements that range between 34% and 61.22%. In fact, the non-adaptive (30 + 30)-evolution strategy succeeds in meeting both the tracking and time-running requirements. However, it took on average 4 to 5 hours to complete the first 50 generations of the 100 generations-long evolutionary process, which represents a 30% increase compared with the running time of its *constrained* counterpart.

The trajectory-tracking performance associated with the designs obtained using the non-adaptive (30 + 30)-evolution strategy, as depicted in Figures 8.5(a), 8.6(a) and 8.7(a), clearly indicates a large reduction of the peak error in comparison to that exhibited by the *constrained* designs of Chapter 5. It is also clear in Figures 8.5(b), 8.6(b), and 8.7(a) that the undesirable high-frequency joint torques present in the previous genetic design have been successfully

attenuated. Yet, those high-frequency components do not disappear completely because the performance measure, Γ, as formulated in equation (8.5) is concerned only with the minimisation of the trajectory-tracking performance regardless of the practical systems constraints.

However, it is clear from Figures 8.14, 8.15, and 8.16 that the non-adaptive (30 + 30)-evolution strategy does not suffer from any premature convergence; but rather the opposite, since it keeps improving well beyond the nominal 50 generations-long evolutionary process.

	Performance measure, Γ		
Design structure	$T = 0.01$	$T = 0.02$	$T = 0.04$
Unconstrained design	4.95×10^{-4} **$(5.08 \times 10^{-4})^{*}$**	6.66×10^{-3} **$(6.95 \times 10^{-3})^{*}$**	3.16×10^{-2} **$(3.32 \times 10^{-2})^{*}$**
Constrained design	13.10×10^{-4}	10.53×10^{-3}	6.10×10^{-2}
Relative improvement	61.22%	34.00%	45.57%

Table **8.2** Comparison of non-adaptive evolutionarily designed controllers.
(*): Values after 50 generations.

8.3.4 ADAPTIVE EVOLUTION STRATEGY

The adaptive (30 + 30)-evolution strategy, generates the results presented in

Table 8.3, where the values of the performance measure, Γ, shown in bold are those obtained after the first 50 generations. It is evident from Table 8.3 that in all three cases the adaptively generated *unconstrained* designs outperform their *constrained* counterparts. Indeed, the relative improvement in the trajectory-tracking performance of the *unconstrained* designs over their *constrained* counterparts varies between 23.71% and 60.08%. This performance improvement of the *unconstrained* designs over their *constrained* counterparts is reflected in the time-domain behaviour of the errors and torques depicted in Figures 8.8, 8.9, and 8.10. However, the crucial feature of the adaptive (30 + 30)-evolution strategy is the relative shortness of the time —about 4 hours— taken by the algorithm to complete the first 50 generations of the 100 generations-long evolutionary process. Yet, this still represent a running-time increase of 70% when compared to its *constrained* counterpart.

	Performance measure, Γ		
Design structure	$T = 0.01$	$T = 0.02$	$T = 0.04$
Unconstrained design	4.93×10^{-4} **$(5.06 \times 10^{-4})^*$**	6.57×10^{-3} **$(6.82 \times 10^{-3})^*$**	2.80×10^{-2} **$(3.00 \times 10^{-2})^*$**
Constrained design	12.90×10^{-4}	8.94×10^{-3}	6.09×10^{-2}
Relative improvement	60.08%	23.71%	50.07%

Table **8.3** Comparison of adaptive evolutionarily designed controllers.

The best-of-generation and average-of-generation plots depicted in Figures 8.17, 8.18, and 8.19 clearly attest to the effectiveness of the (30 + 30) adaptive evolution strategy in solving the *unconstrained* design problem. In addition, the higher degree of automation present in the adaptive evolution strategy and the relatively shorter running-time for the completion of a given evolutionary process indicate that the adaptive evolution strategy is the most suitable candidate amongst the three evolutionary algorithms. This suitability and convenience of use of the adaptive evolutiion strategy is confirmed by the comparative study depicted in Table 8.4. Indeed, It is clear from Table 8.4 that in all three cases the performances of controllers obtained using adaptive ES are superior to their genetically and non-adaptively obtained counterpart.

	Performance Measure **(relative improvement)**		
Evolution algorithms	$T = 0.01$	$T = 0.02$	$T = 0.04$
Genetic Algorithm	5.78×10^{-4} **(0%)**	7.36×10^{-3} **(0%)**	4.32×10^{-2} **(0%)**
Non-adaptive ES	4.95×10^{-4} **(14.7%)**	6.66×10^{-3} **(9.5%)**	3.16×10^{-2} **(26.9%)**
Adaptive ES	4.93×10^{-4} **(14.7%)**	6.57×10^{-3} **(10.7%)**	2.80×10^{-2} **(35.2%)**

Table **8.4** Comparison of the performances of the controllers obtained.

It is clear from Tables 8.1, 8.2 and 8.3 that in all cases, the *unconstrained* controller designs outperform their *constrained* counterparts. However, it is

important to remember that in all these cases the *unconstrained* designs required an additional running-time ranging between 70% and 400% of the running-time of their *constrained* counterparts. This is relevant to practising engineers, who usually try to achieve a balance between an acceptable design and a rapidly obtained design.

8.4 CONCLUSION

It has been shown in this chapter that genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies can be conveniently used to solve the non-trivial problem of designing *unconstrained* digital PID controllers for robotic manipulators. This challenging design task has been illustrated by the design of *unconstrained* digital trajectory-tracking PID controllers for the three-degree-of-freedom robotic manipulator previously used in Chapters 3, 4, 5, 6, and 7 for a range of sampling frequencies. The evolutionary unconstrained design approach used in this chapter improved significantly the accuracy of the trajectory-tracking performance in this case. However, this increase in accuracy was accompanied by an increase in the running-time of all three evolutionary algorithms. However, adaptive evolution strategies provide by far the more effective approach amongst all three proposed evolutionary algorithms. Indeed, the effectiveness of adaptive evolution strategies is more evident in the present *unconstrained* case than in the *constrained* case of Chapter 7.

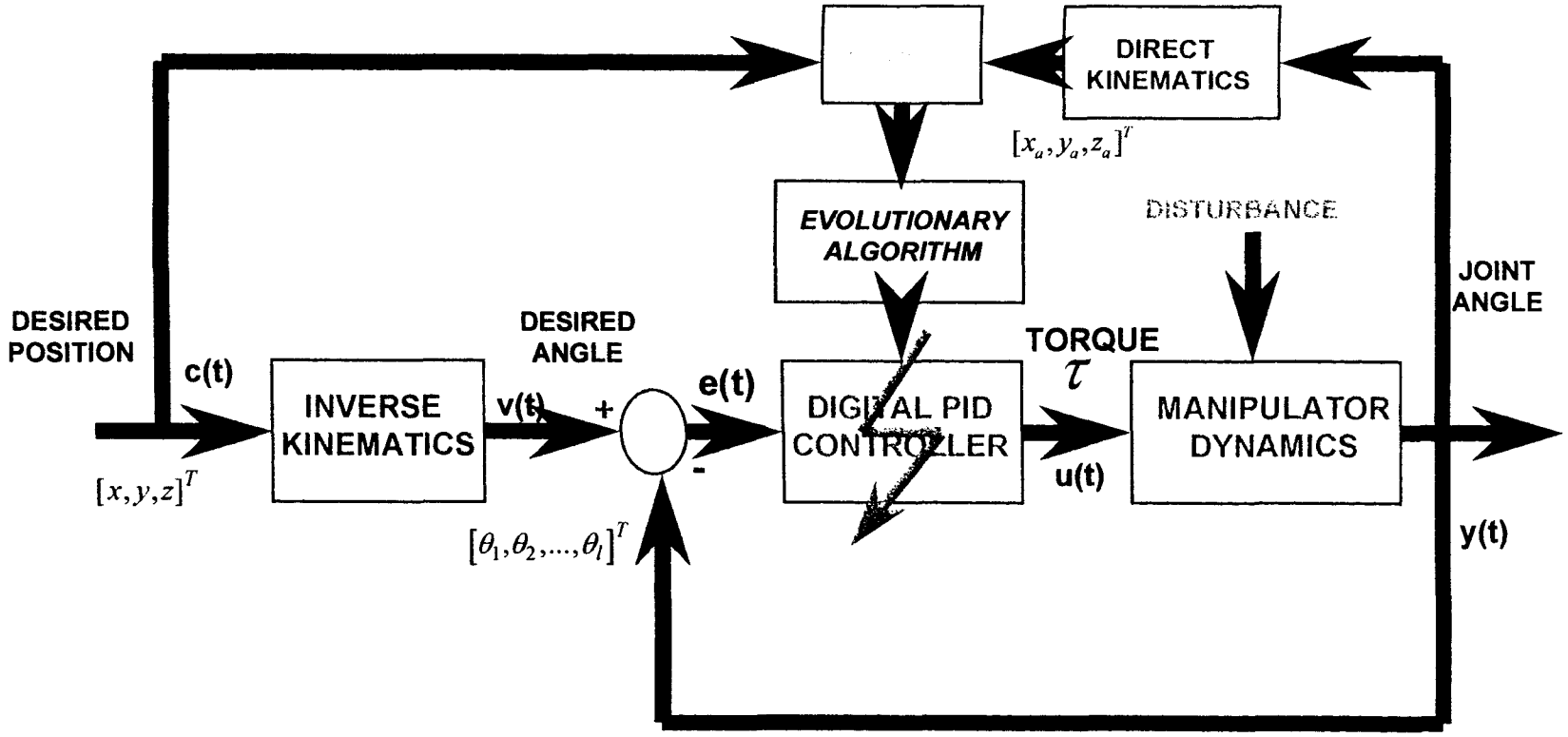

Figure 8.1: Computational implementation of trajectory-tracking system and its associated evolutionary algorithms

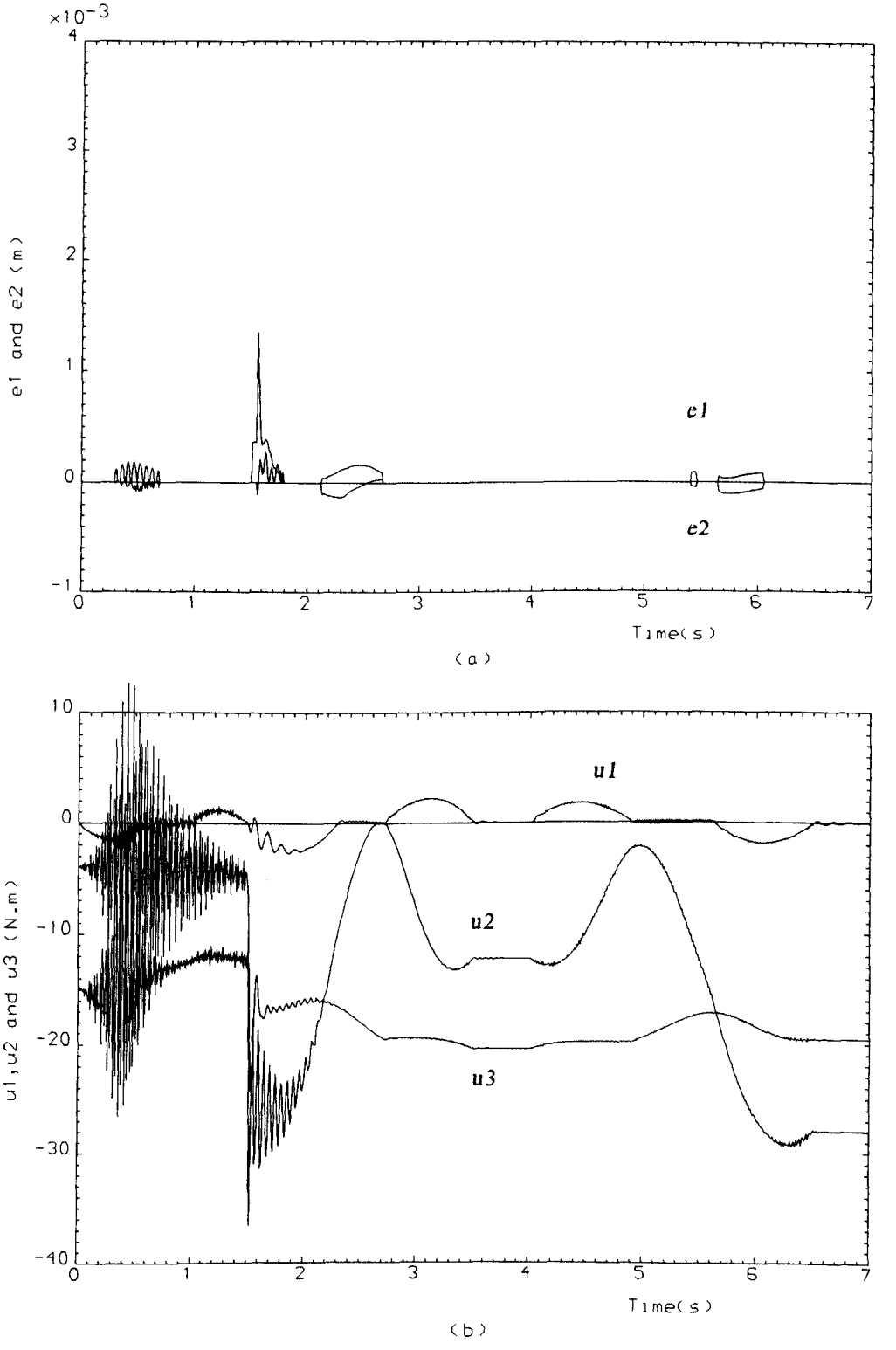

Figure 8.2: Time-domain behaviour of errors and torques for the genetically designed unconstrained controller (p_m = 0.002,T = 0.01).

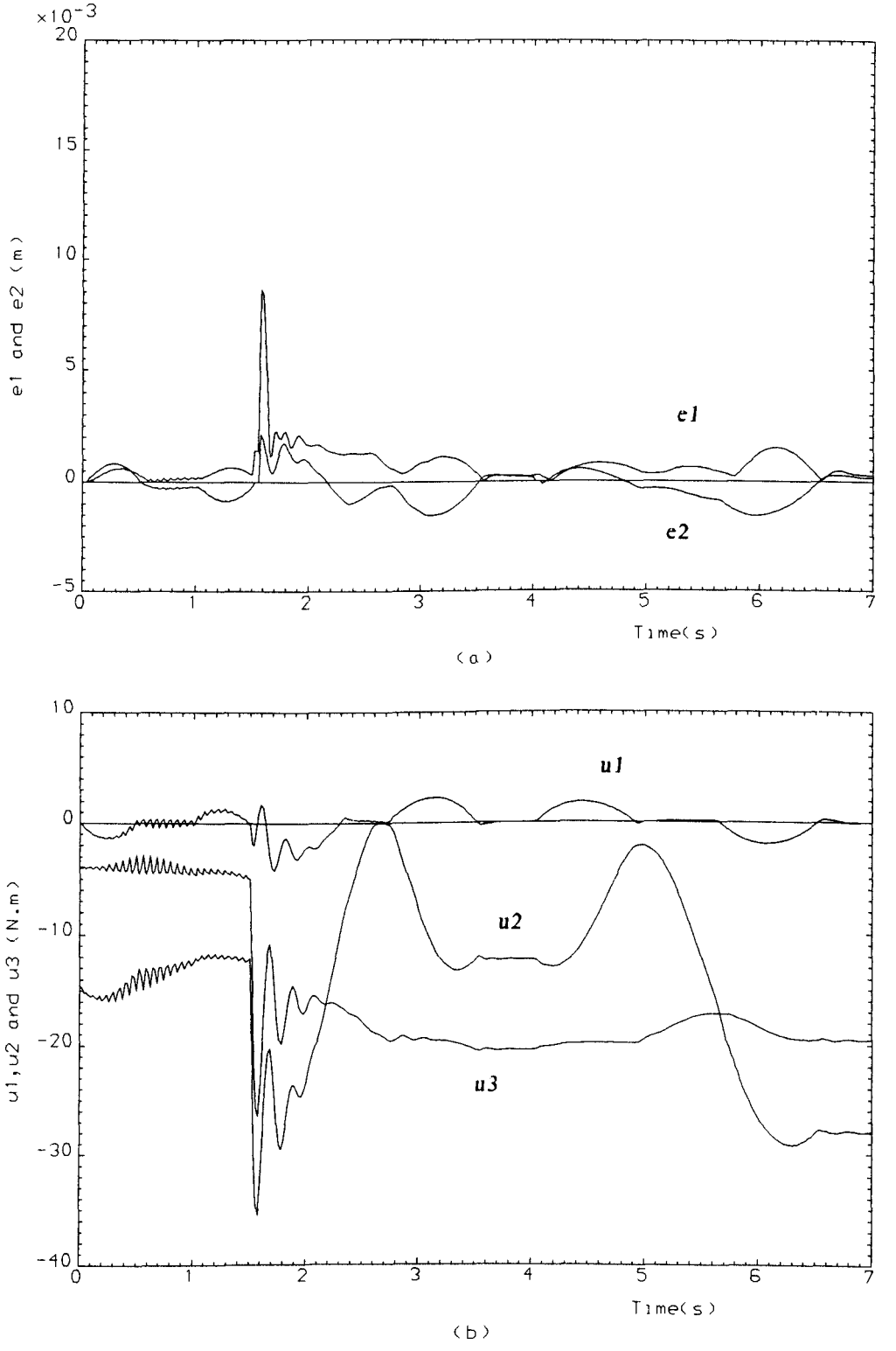

Figure 8.3: Time-domain behaviour of errors and torques for the genetically designed unconstrained controller (p_m = 0.002,T = 0.02).

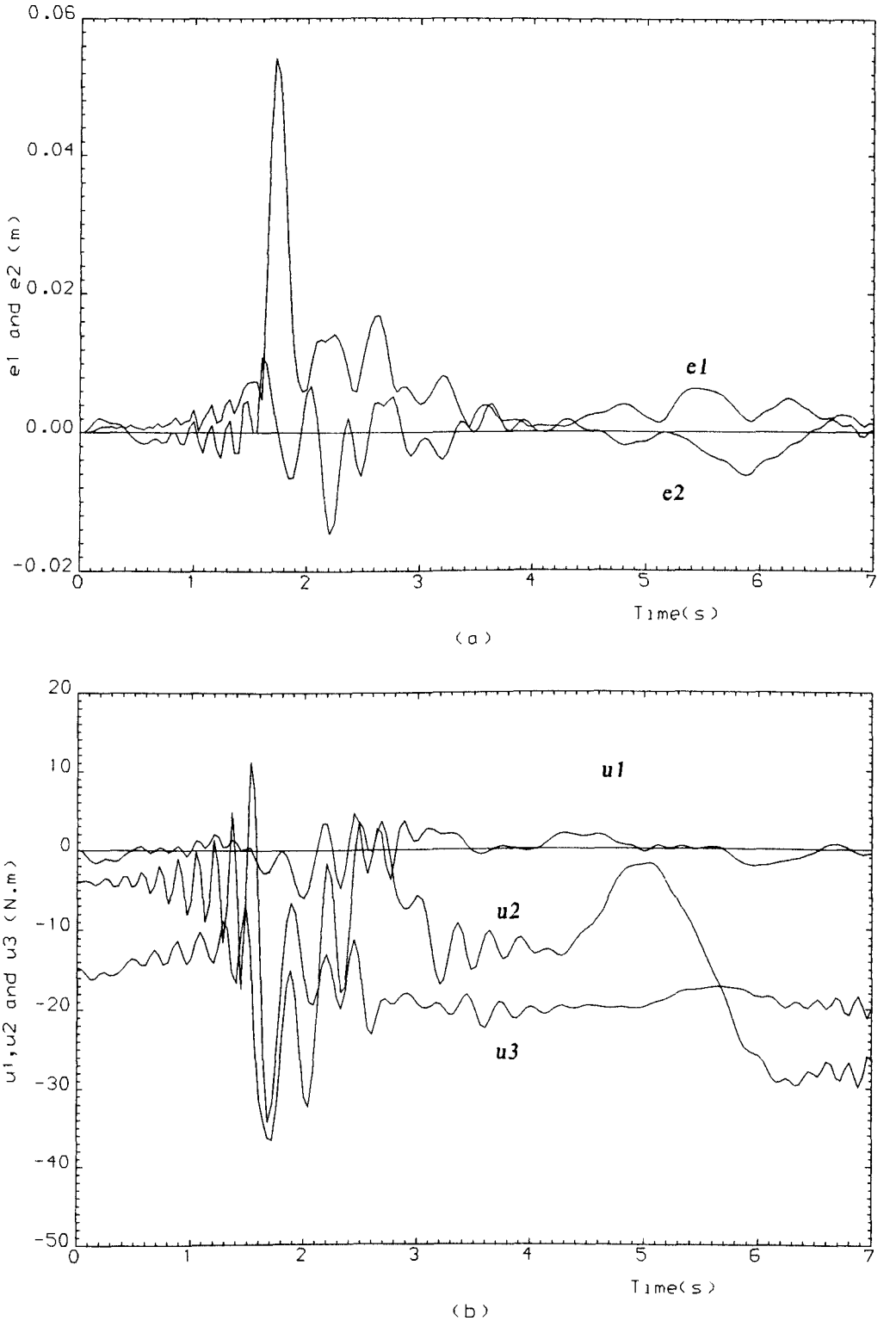

Figure 8.4: Time-domain behaviour of errors and torques for the genetically designed unconstrained controller (p_m = 0.002,T = 0.04).

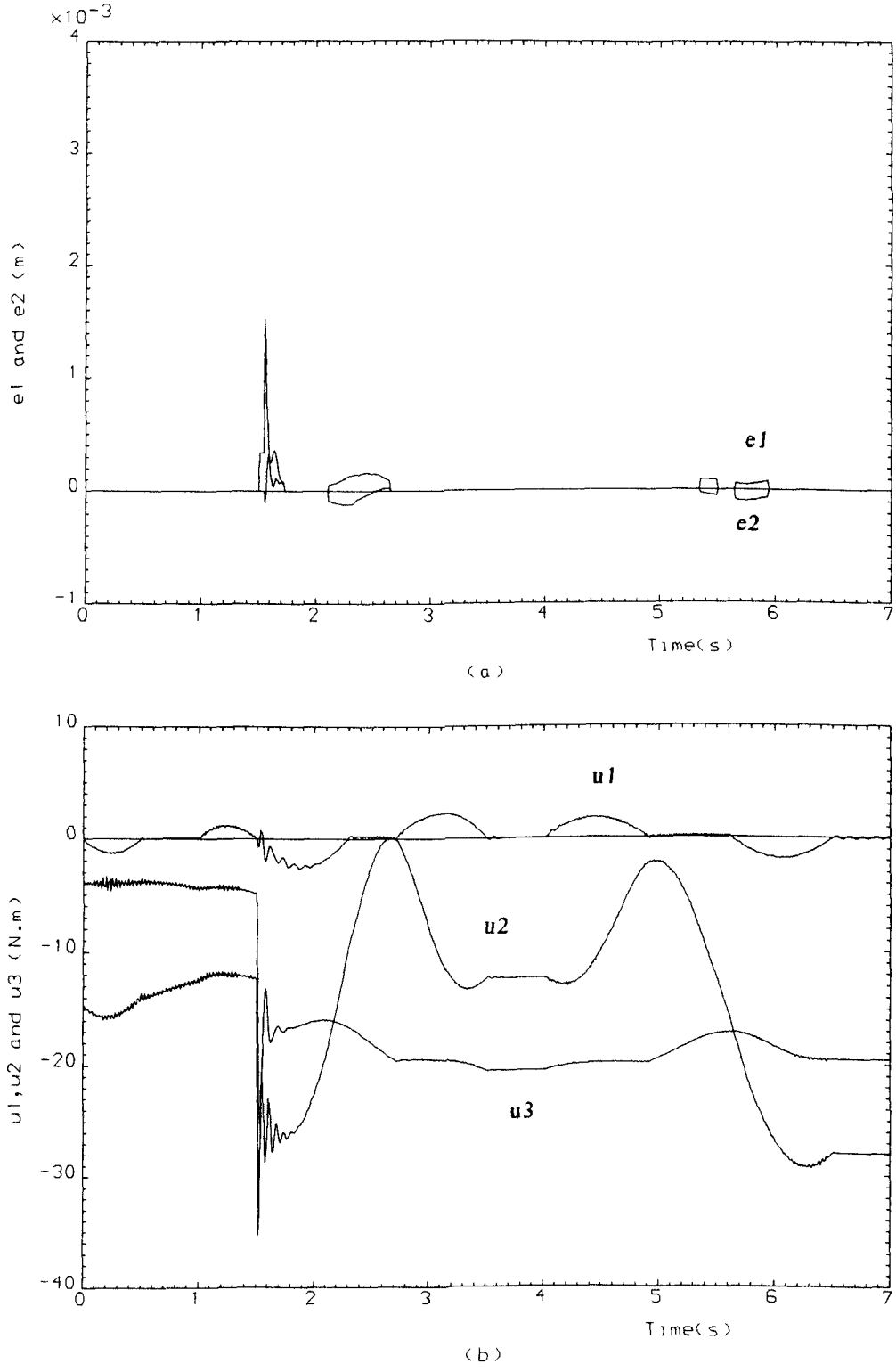

Figure 8.5: Time-domain behaviour of errors and torques for the non-adaptive evolutionarily designed unconstrained controller (p_m = 0.005,T = 0.01s).

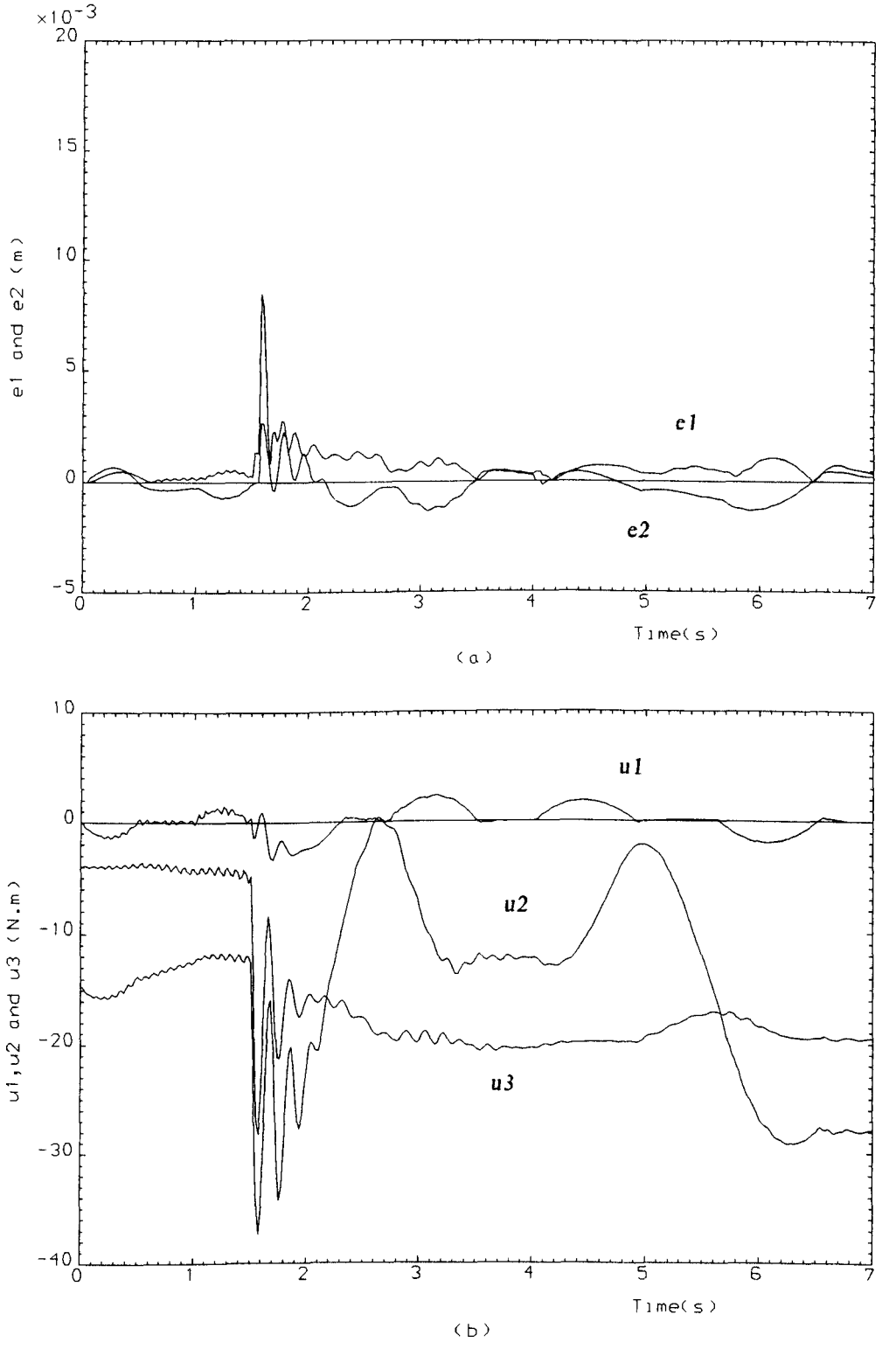

Figure 8.6: Time-domain behaviour of errors and torques for the non-adaptive evolutionarily designed unconstrained controller (p_m = 0.005,T = 0.02s).

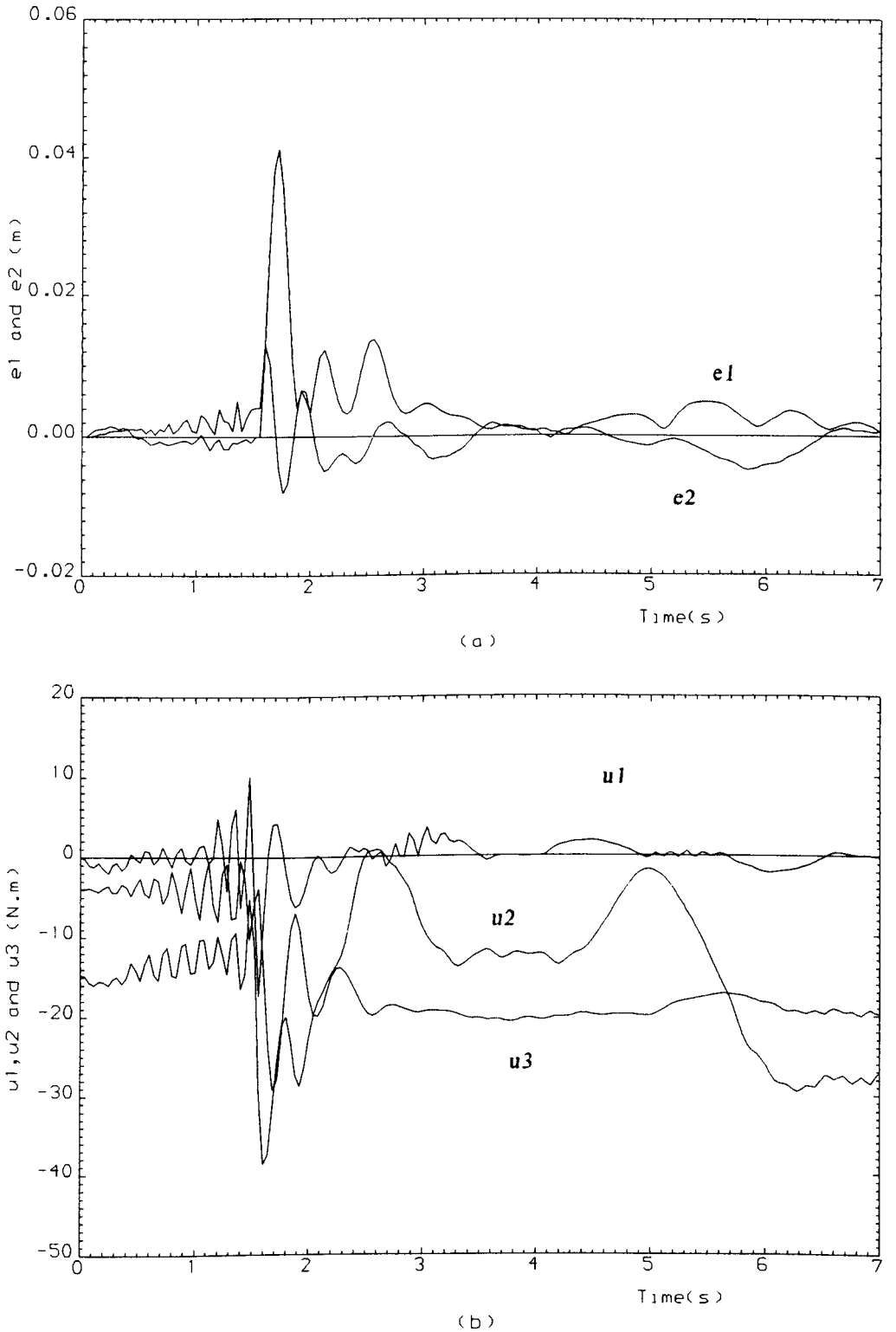

Figure 8.7: Time-domain behaviour of errors and torques for the non-adaptive evolutionarily designed unconstrained controller (p_m = 0.005,T = 0.04s).

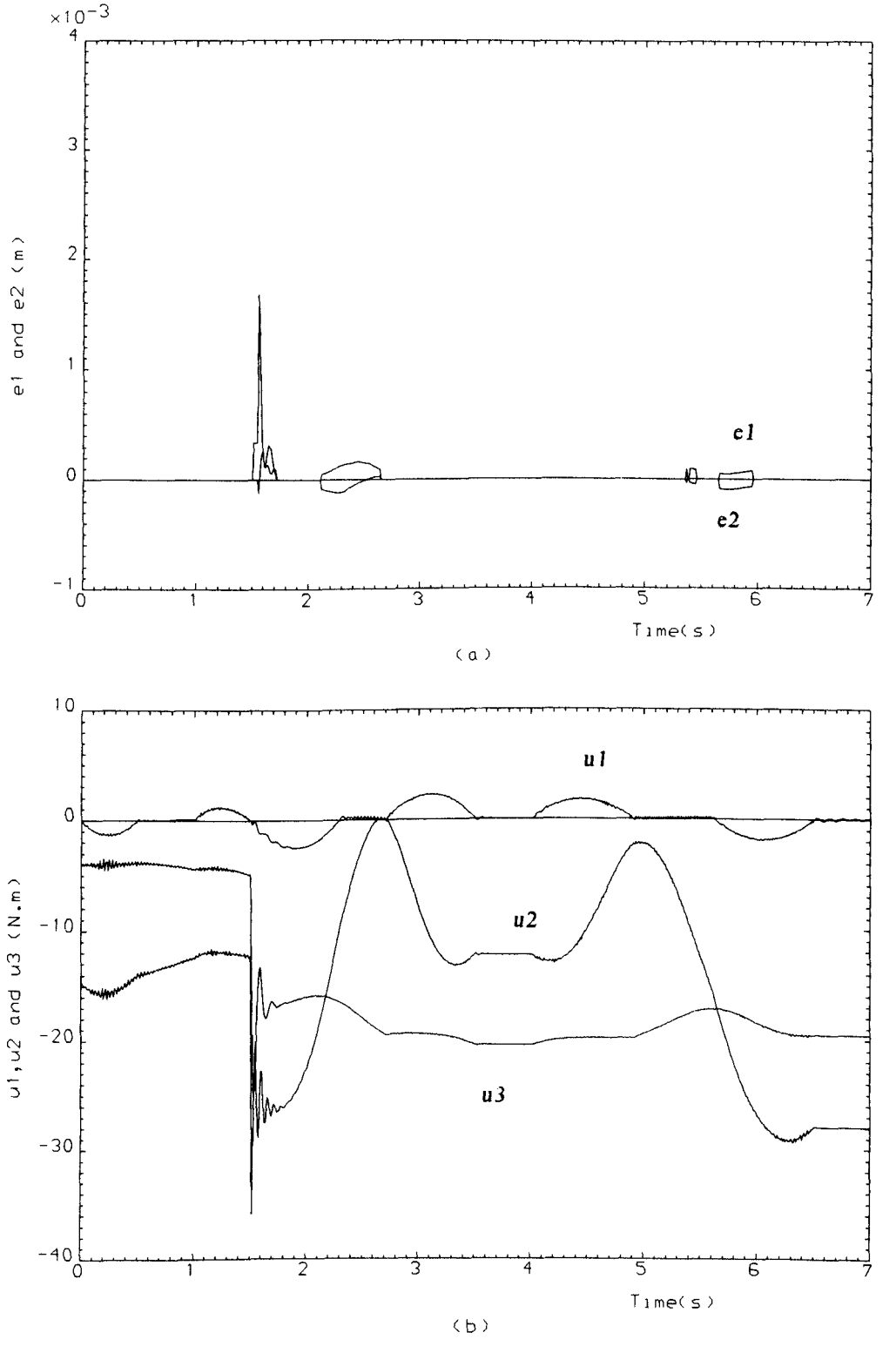

Figure 8.8: Time-domain behaviour of errors and torques for the the adaptive evolutionarily designed unconstrained controller (T = 0.01s).

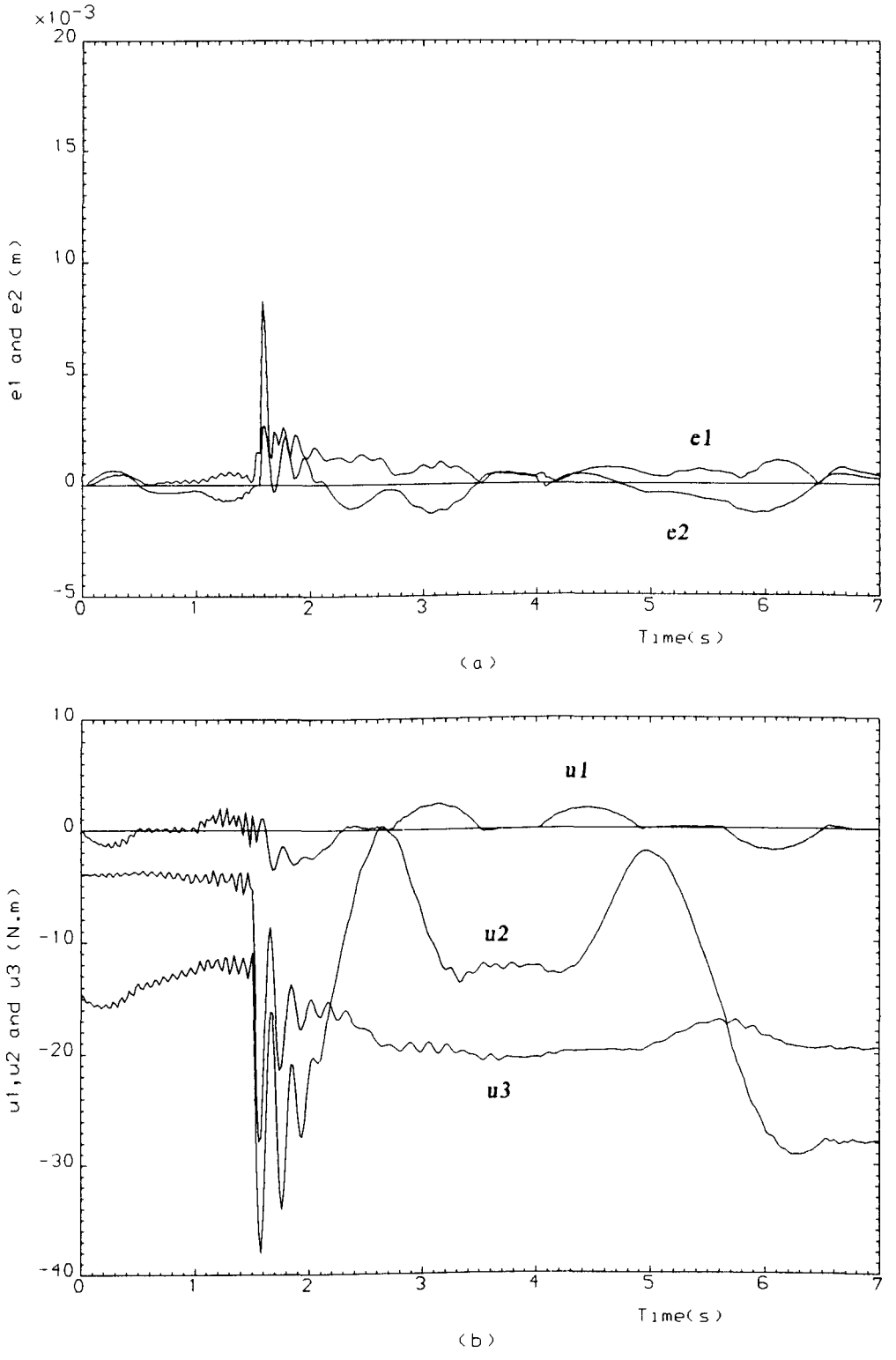

Figure 8.9: Time-domain behaviour of errors and torques for the the adaptive evolutionarily designed unconstrained controller (T = 0.02).

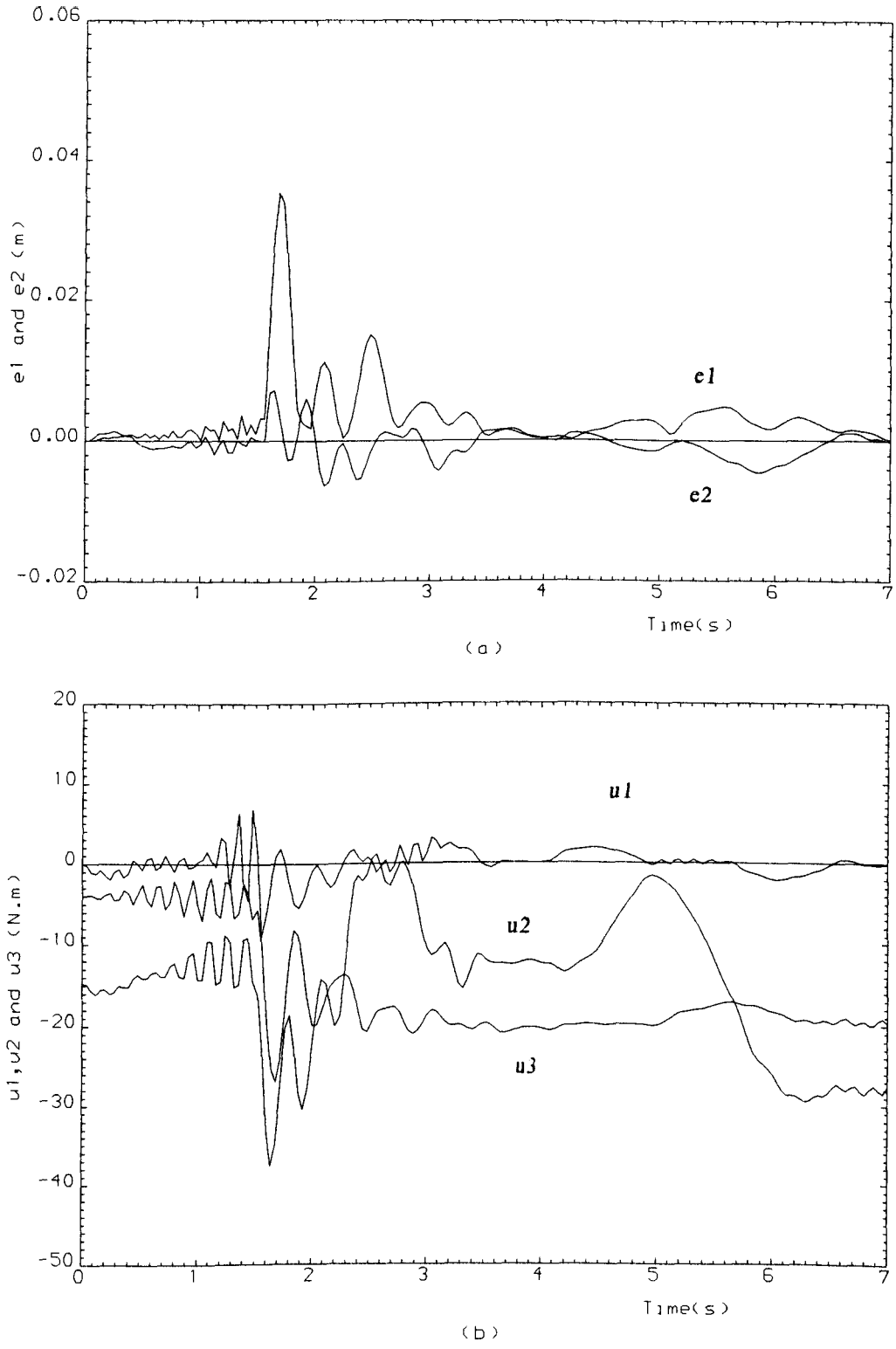

Figure 8.10: Time-domain behaviour of errors and torques for the the adaptive evolutionarily designed unconstrained controller (T = 0.04).

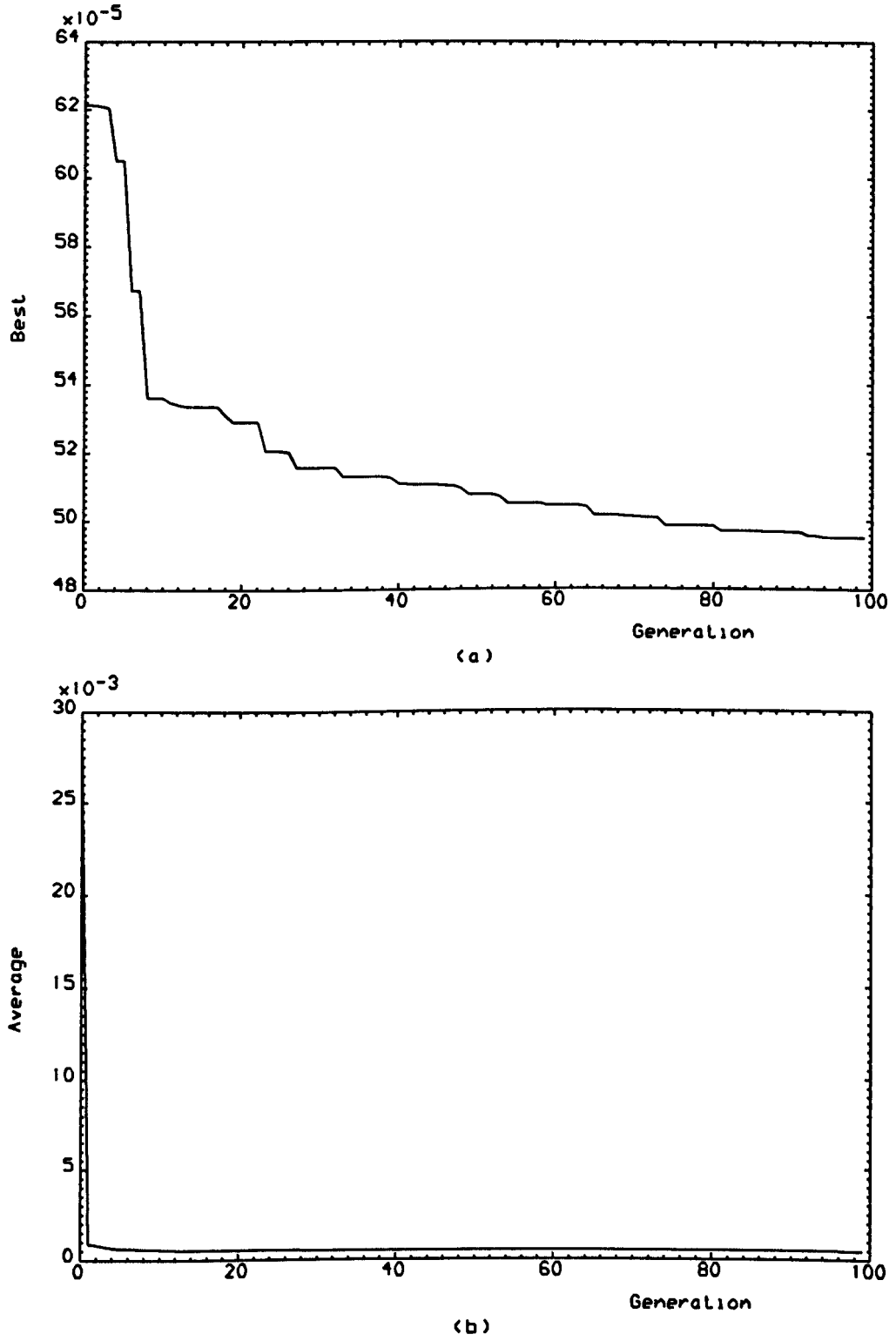

Figure 8.14: Best-of-generation and average-of-generation for the non-adaptive evolutionarily designed unconstrained controller (p_m = 0.005,T = 0.01s).

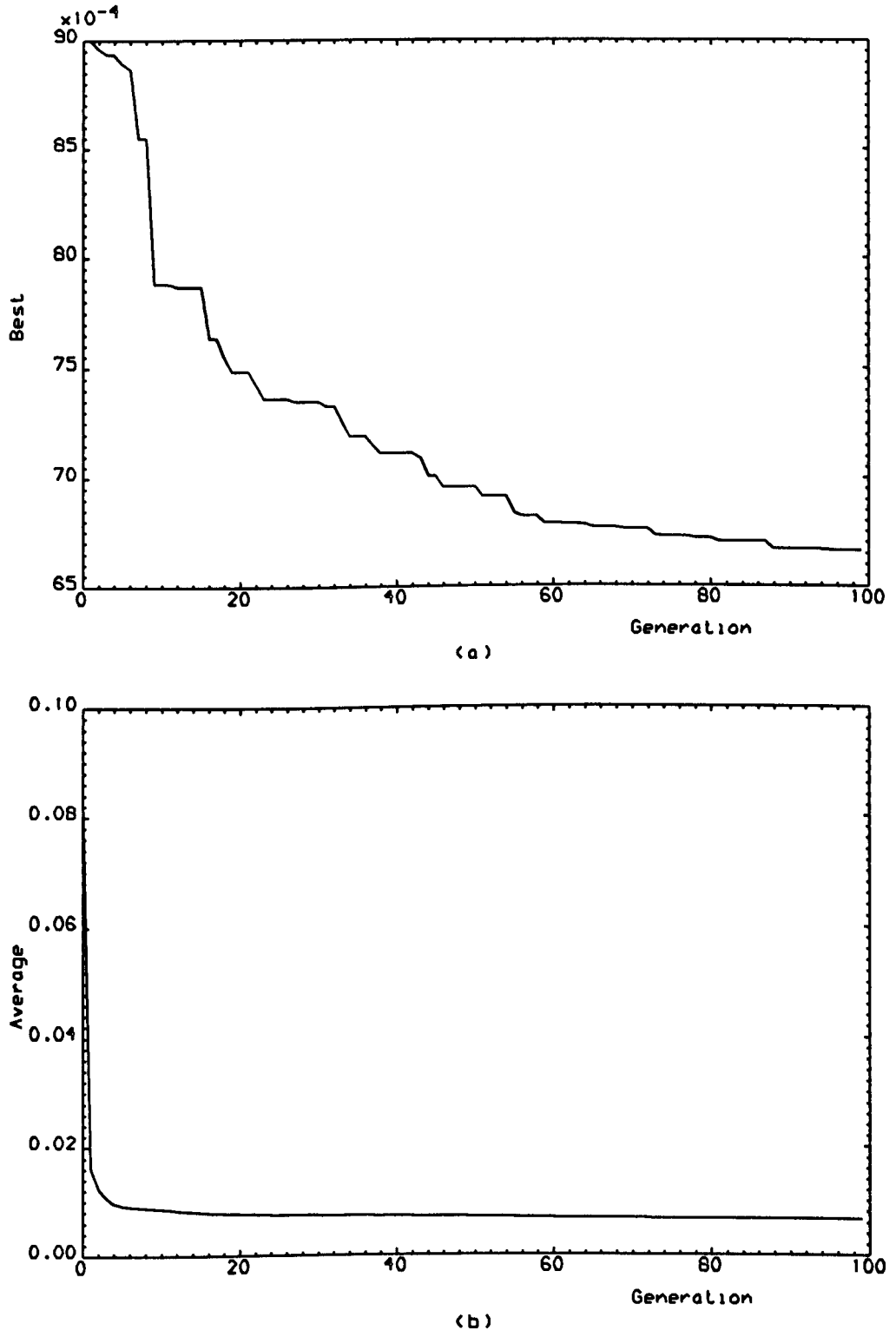

Figure 8.15: Best-of-generation and average-of-generation for the non-adaptive evolutionarily designed unconstrained controller (p_m = 0.005,T = 0.02s).

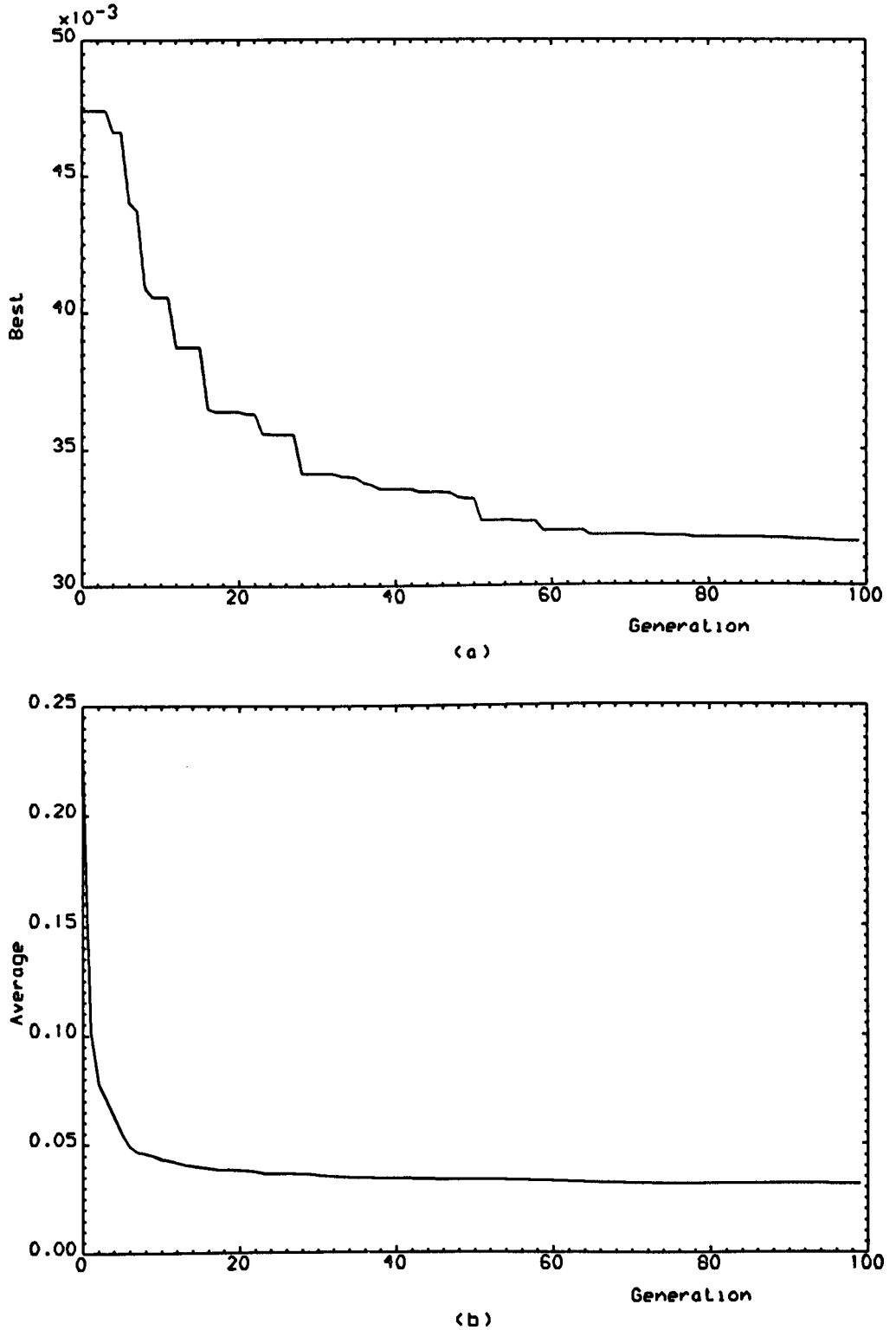

Figure 8.16: Best-of-generation and average-of-generation for the non-adaptive evolutionarily designed unconstrained controller (p_m = 0.005,T = 0.04s).

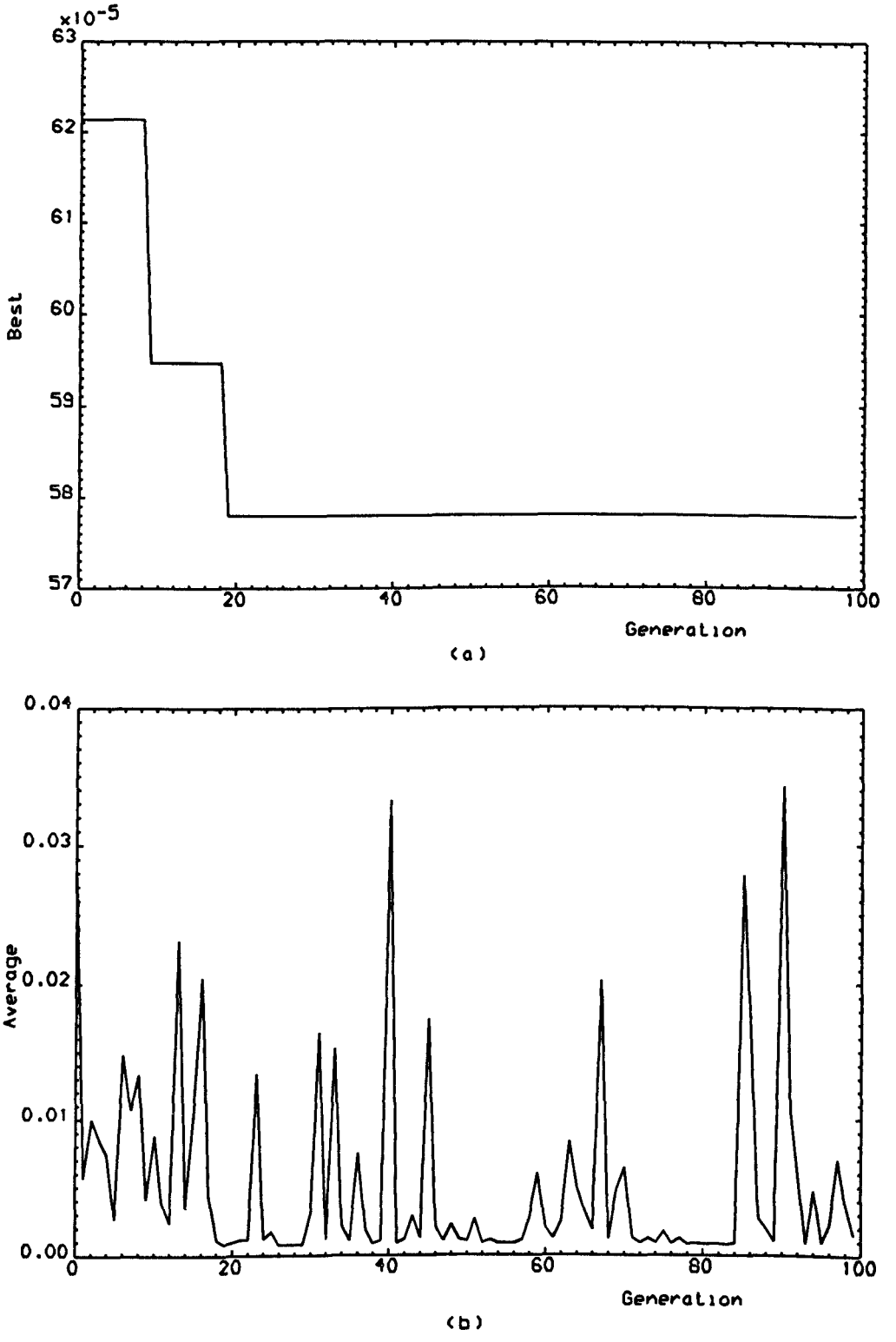

Figure 8.11: Best-of-generation and average-of-generation for genetically designed unconstrained controller (p_m = 0.002,T = 0.01).

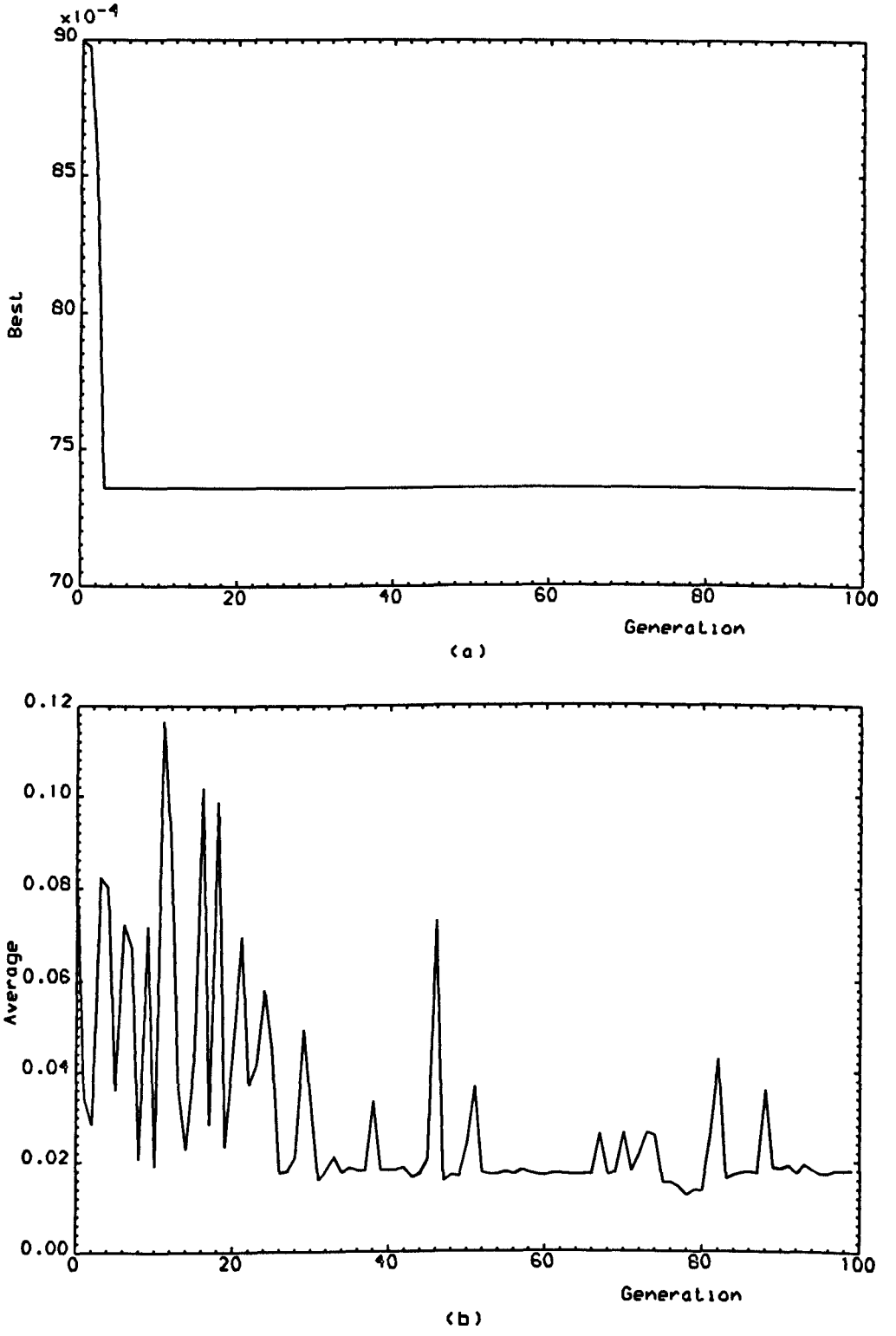

Figure 8.12: Best-of-generation and average-of-generation for genetically designed unconstrained controller (p_m = 0.002,T = 0.02).

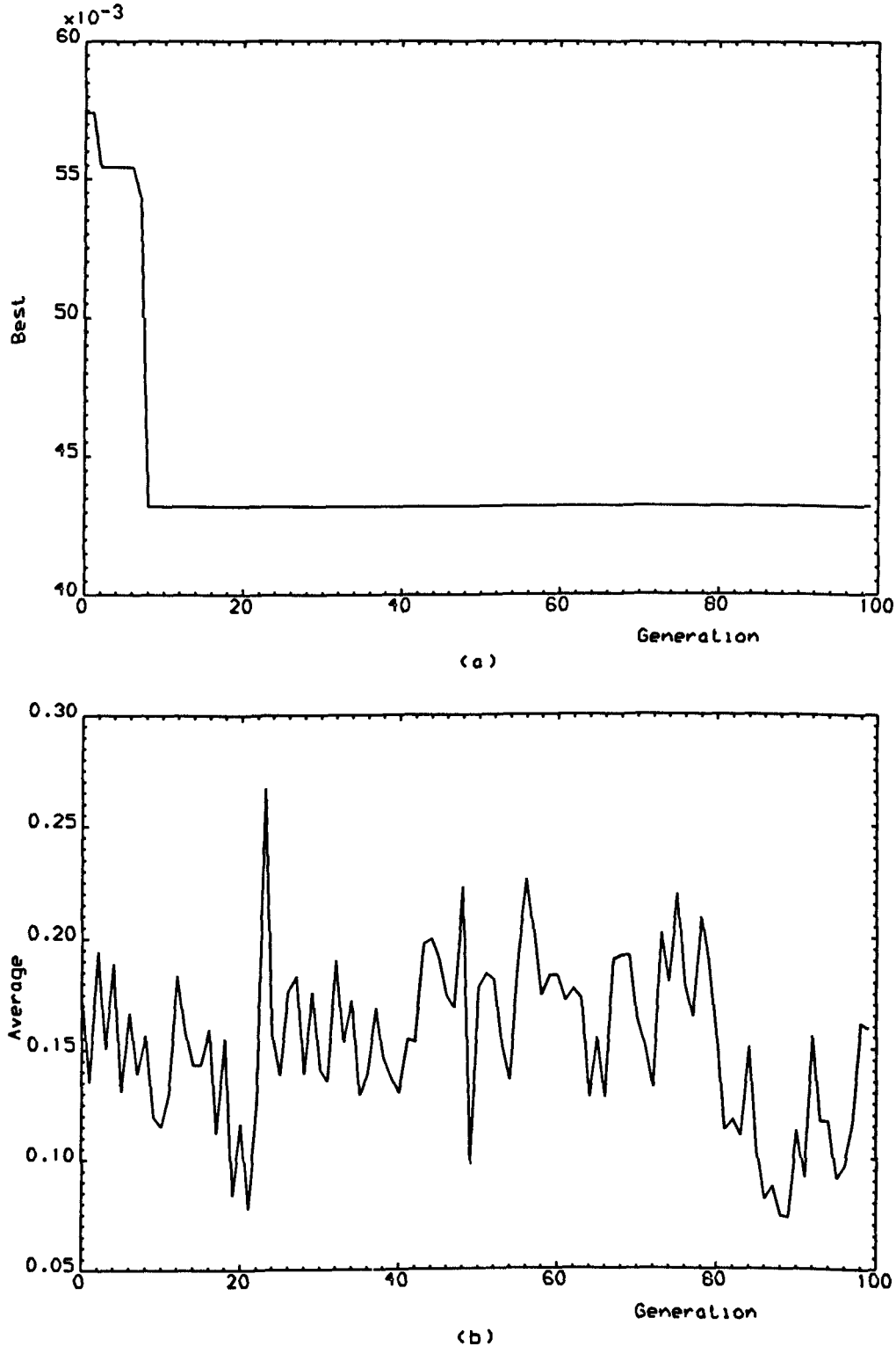

Figure 8.13: Best-of-generation and average-of-generation for genetically designed unconstrained controller (p_m = 0.002,T = 0.04).

PART V

PRACTICAL IMPLEMENTATION OF DIGITAL TRAJECTORY-TRACKING CONTROLLERS

CHAPTER 9

PRACTICAL TESTS OF TWO-LINK DIRECT-DRIVE ROBOTIC MANIPULATOR

9.1 INTRODUCTION

In Chapters 4, 5, 6, and 8, genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies were used for both the *constrained* and *unconstrained* design of robust digital trajectory-tracking PID controllers for robotic manipulators. Such design was previously effected by the singular perturbation methods presented in Chapter 3 in the *constrained* case. It was shown in Chapters 4, 5, 6, and 8 that genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies can be used to provide an effective tuning procedure for such controllers in the non-asymptotic case for which the sampling period is non-zero.

The non-triviality of this design task arises from the complex nature of robotic manipulators while performing wide ranges of trajectory-tracking tasks. Indeed, this design difficulty occurs because robotic manipulators are highly non-linear multivariable plants with time-varying parameters. Furthermore, the design of robotic controllers for trajectory-tracking purposes becomes even more complicated when work-space co-ordinates rather than the joint-space co-ordinates are used. This is because the relationship between these two systems of co-ordinates is given by complex trigonometric transformations.

It is proposed in this chapter to test the performance of evolutionarily designed digital trajectory-tracking PID controllers by conducting practical laboratory tests on a two-link direct-drive robotic manipulator. In these tests, a range of different controllers is used that are designed by genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies

9.2 SYSTEM DESCRIPTION

The dynamical model of the two-link direct-drive robotic manipulator under investigation is derived and presented in Appendix C. Thus, the vector-matrix differential governing equation for such a two-link robotic manipulator has the form

$$M(\theta)\ddot{\theta} + v(\theta,\dot{\theta}) + f = \tau 1 \qquad (9.1)$$

In this equation, $M(\theta) \in \Re^{2x2}$ 2is the inertia matrix, $v(\theta,\dot{\theta}) \in \Re^2$ 3is the vector of centrifugal, coriolis4, and gravitational torques, $f \in \Re^2$ 5is the vector of coulomb friction torques, $\tau \in \Re^2$ 6is the vector of actuator torques, $\theta \in \Re^2$ 7is the vector of joint angles, $\dot{\theta} \in \Re^2$ 8is the vector of joint velocities, and $\ddot{\theta} \in \Re^2$ 9is the vector of joint accelerations. The specific form of equation (9.1) in the case of the particular two-link direct-drive robotic manipulator under investigation is given in Appendix C.

In the design of digital trajectory-tracking PID controllers for robotic manipulators, it is required that the end-effectors of such manipulators exhibit small tracking errors when following a desired trajectory. In order to achieve such a high accuracy, it is proposed to use digital trajectory-tracking PID controllers (see Chapter 3) which

are governed on the discrete-time set $T_T = \{0, T, 2T, ...\}$ by control-law equations of the form [*Porter et al* (1985)]

$$u(kT) = K_1 r(kT) + K_2 z(kT) \quad . \tag{9.2}$$

In equation (9.2), $T \in \Re^+$ is the sampling period, $K_1 \in \Re^{2x2}$ and $K_2 \in \Re^{2x2}$ are the proportional and integral controller matrices, whilst the vectors $r(kT) \in \Re^2$ and $z(kT) \in \Re^2$ are generated in accordance with the difference equations [*Porter* (1987)]

$$s\{(kT + 1)T\} = -\alpha s(kT) + e(kT) , \tag{9.3}$$

$$r(kT) = -\frac{2}{T}(1 + \alpha)Ds(kT) + \left(I_1 + \frac{2}{T}D\right) e(kT), \tag{9.4}$$

and

$$z\{(k + 1)T\} = z(kT) + Tr(kT) . \tag{9.5}$$

Moreover, in equations (9.3), (9.4) and (9.5), $\alpha \in (-1, +1]$, $s(kT) \in \Re^2$, $e(kT) = v(kT) - y(kT) \in \Re^2$ is the error vector, $v(kT)$ is the set-point command vector, and the derivative matrix $D \in \Re^{2x2}$ is such that

$$D = \delta I_2 \, (\delta \in \Re^+) \quad , \tag{9.6}$$

where δ is a positive derivative parameter so that the rank of the derivative matrix $D = 2$.

9.3 EVOLUTIONARILY DESIGNED CONTROLLERS

9.3.1 INTRODUCTION

In practice, it is advisable as indicated in Chapter 6, to use performance measures that are effective in preventing undesirable behaviour of the input torques as well as in producing high-accuracy tracking. These alternative performance indices can therefore embody practical constraints such as amplitude or rate limits on the control effort associated the robotic manipulator.

Indeed, in order to introduce the rate limits on the control torques, it is convenient to consider the performance index

$$\Gamma_2 = \int_0^{\tau} \| \Delta u(t) \| \, dt \tag{9.7}$$

where τ is the duration of the tracking task, $\Delta u(t) \in \Re^l$ is the change in the control vector over a sampling period, and $\|.\|$ denotes the Euclidean norm.

It is also possible to combine this performance index with the trajectory-tracking performance index

$$\Gamma_1 = \int_0^\tau \left\| e(t) \right\| \, dt \tag{9.8}$$

where τ is the duration of the tracking task, $e(t) \in \Re^l$ is the trajectory-tracking error vector in Cartesian space, and $\|.\|$ denotes the Euclidean norm. Such a combination of these two indices results in the performance index

$$\Gamma = \lambda \int_0^\tau \|e(t)\| \, dt + \mu \int_0^\tau \|\Delta u(t)\| \, dt \,, \tag{9.9}$$

where λ and μ are weighting parameters. In the manner described in Chapter 6, evolutionary algorithms can be used to design digital trajectory-tracking PID controllers for robotic manipulators for any choice of such performance measure. Indeed, this design problem is immediately amenable to the non-asymptotic approach presented in Chapter 6.

9.3.2 CONSTRAINED DESIGNS

In the *constrained* case, the evolutionary design process of Chapters 4, 5, and 6 used the design equations derived by *Porter* and *Abidin* (1990) and presented in Chapter 3. Indeed, this methodology indicates that high-accuracy tracking behaviour can be achieved asymptotically if fast-sampling digital PID controllers are designed such that

$$K_1 = T\tilde{H}^{-1}(T)\Sigma(TI_l + 2D)^{-1} \in \Re^{2x2} \tag{9.10}$$

and

$$K_2 = \rho T \tilde{H}^{-1}(T)\Sigma(TI_1 + 2D)^{-1} \in \Re^{2x2} \quad . \qquad (9.11)$$

In these design equations,

$$\Sigma = \sigma I_1 (\sigma \in \Re^{+}) \qquad (9.12)$$

is the positive diagonal tuning matrix, and ρ is the positive tuning parameter integral-to-proportional ratio. In addition, in equations (9.10) and (9.11),

$$\tilde{H}(T) = \int_0^T \tilde{C} e^{\tilde{A} t} \tilde{B} dt \qquad (9.13)$$

is the step-response matrix of the nominal open-loop plant with state-space triple $(\tilde{A}, \tilde{B}, \tilde{C})$ which is used for design purposes in obtaining the controller for the actual open-loop plant with state-space triple (A,B,C) governed by equations (C.12) and (C.13).

In this *constrained* evolutionary design process, genetic algorithms, non-adaptive evolution strategies, and adaptive evolutionary strategies are in turn used to design digital trajectory-tracking PID controllers for the two-link direct-drive robotic manipulator under non-asymptotic conditions. It is evident that, as shown in Chapters 4, 5 and 6, the solution of this problem provides an optimal quadruple, $\{\alpha,\sigma,\rho,\delta\}$, of the controller design parameters in equations (9.3), (9.4), (9.10), and (9.11).

It is clear that the solution of this problem will provide an optimal quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller design parameters which is dependent upon the given manipulator, the given task, and the measure of trajectory-tracking performance used in the optimisation procedure. It is therefore evident from equation (9.9) that the nature of optimality will be determined by the choice of the parameters λ and μ.

In the present tests, the controller is designed so to cause the end-effector of the robotic manipulator to track the straight-line trajectories forming the triangle shown in Figure 9.3. The evolutionary design of the controller for this selected trajectory-tracking task can be readily effected so as to minimise the cost function defined in equation (9.9). The robotic manipulator is, for design purposes, considered to be in the neighbourhood of the median of the triangular trajectory. Hence, in the task-space the selected operating point assigns the end-effector the position (0.509,0.132)m which corresponds to the joint-space coordinates (-0.1657,1.0891)rad.

The four-dimensional evolutionary design problem can be readily solved by using the binary representation of the quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller design parameters presented in Chapter 4. Indeed, the binary concatenated representation of the quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller design parameters in this case has the form

$$|\alpha|\sigma|\rho|\delta| \quad , \tag{9.14}$$

where each parameter is coded as a 15-bit sub-string. It is, therefore, evident that a 60-bit string will represent the quadruple $\{\alpha,\sigma,\rho,\delta\}$ of controller design parameters for the particular two-link robotic manipulator.

In applying genetic algorithm (GA), after a trial-and-error process results were obtained using a population N=10, a crossover probability p_c=0.6, and a mutation probability p_m=0.01, with evolution occurring over 50 generations. The results obtained by genetically minimising the performance measure, Γ, for λ = 0.1, while μ was left to vary within the selected interval [1.5, 6.0], are shown in the second column of Table 9.1. The optimal controller gains and parameters obtained as μ varies are presented in Table 9.2.

Performance Measure	GA	Non-adaptive ES	Adaptive ES
$\Gamma(\mu = 1.5)$	0.2101	0.1897	0.1878
$\Gamma(\mu = 3.0)$	0.3984	0.3508	0.3423
$\Gamma(\mu = 4.0)$	0.6692	0.4421	0.4408
$\Gamma(\mu = 6.0)$	1.2810	0.6678	0.5696

Table 9.1: Simulated performance measured for the evolutionarily designed constrained trajectory-tracking PID controllers .

(μ = 1.5) **constrained**	$\alpha = 0.0013;\ D = \begin{bmatrix} 1.405 & 0.0 \\ 0.0 & 1.405 \end{bmatrix};\ K_1 = \begin{bmatrix} 16.571 & 0.921 \\ 0..921 & 0.559 \end{bmatrix};\ K_2 = \begin{bmatrix} 136.97 & 7.612 \\ 7.612 & 4.624 \end{bmatrix}$
(μ = 3.0) **constrained**	$\alpha = 0.4894;\ D = \begin{bmatrix} 2.564 & 0.0 \\ 0.0 & 2.564. \end{bmatrix};\ K_1 = \begin{bmatrix} 22.407 & 1.245 \\ 1.245 & 0.756 \end{bmatrix};\ K_2 = \begin{bmatrix} 305.80 & 16.995 \\ 16.995 & 10.323 \end{bmatrix}$
(μ = 4.0) **constrained**	$\alpha = 0.416;\ D = \begin{bmatrix} 4.380 & 0.0 \\ 0.0 & 4.380 \end{bmatrix};\ K_1 = \begin{bmatrix} 13.895 & 0.772 \\ 0.772 & 0.469 \end{bmatrix};\ K_2 = \begin{bmatrix} 49.736 & 2.764 \\ 2.764 & 1.679 \end{bmatrix}$
(μ = 6.0) **constrained**	$\alpha = 0.614;\ D = \begin{bmatrix} 2.530 & 0.0 \\ 0.0 & 2.530 \end{bmatrix};\ K_1 = \begin{bmatrix} 110.44 & 6.138 \\ 6.138 & 3.728 \end{bmatrix};\ K_2 = \begin{bmatrix} 1197.1 & 66.524 \\ 66.524 & 40.411 \end{bmatrix}$

Table 9.2: Genetically designed constrained digital trajectory-tracking PID controllers.

In order to explore alternatives to genetic algorithms, a non-adaptive (10 + 10)-evolution strategy (ES) was used while retaining the 60-bit representation of the controller parameters. In applying this non-adaptive evolution strategy, results were obtained using the mutation probability, p_m = 0.03 (which was selected after a time-consuming trial-and-error process), and evolution occurred over 50 generations. In a similar way to the genetic design procedure, this non-adaptive (10 + 10)-evolution strategy was used to minimise the performance measure, Γ, for λ = 0.1 while μ was left to vary within the selected interval [1.5, 6.0]. The attained optimal values of the performance measure are shown in the third column of Table 9.1. The corresponding optimal controllers gains and parameters obtained as μ varies are presented in Table 9.3.

(μ = 1.5) **constrained**	$\alpha = 0.072;\ D = \begin{bmatrix} 0.176 & 0.0 \\ 0.0 & 0.176 \end{bmatrix};\ K_1 = \begin{bmatrix} 209.48 & 11.642 \\ 11.642 & 7.071 \end{bmatrix};\ K_2 = \begin{bmatrix} 2050.2 & 113.92 \\ 113.92 & 69.235 \end{bmatrix}$
(μ = 3.0) **constrained**	$\alpha = 0.251;\ D = \begin{bmatrix} 2.578 & 0.0 \\ 0.0 & 2.578 \end{bmatrix};\ K_1 = \begin{bmatrix} 6.743 & 0.375 \\ 0.375 & 0.228 \end{bmatrix};\ K_2 = \begin{bmatrix} 74.490 & 4.140 \\ 4.140 & 2.515 \end{bmatrix}$
(μ = 4.0) **constrained**	$\alpha = 0.041;\ D = \begin{bmatrix} 4.253 & 0.0 \\ 0.0 & 4.253 \end{bmatrix};\ K_1 = \begin{bmatrix} 3.145 & 0.175 \\ 0.175 & 0.234 \end{bmatrix};\ K_2 = \begin{bmatrix} 34.778 & 1.933 \\ 1.933 & 1.174 \end{bmatrix}$
(μ = 6.0) **constrained**	$\alpha = 0.020;\ D = \begin{bmatrix} 1.245 & 0.0 \\ 0.0 & 1.245 \end{bmatrix};\ K_1 = \begin{bmatrix} 24.450 & 1.359 \\ 1.359 & 0.825 \end{bmatrix};\ K_2 = \begin{bmatrix} 147.31 & 8.187 \\ 8.187 & 4.973 \end{bmatrix}$

Table 9.3: Constrained trajectory-tracking PID controllers designed using non-adaptive ES.

Finally, an adaptive (10 +10)-evolution strategy (ES) was used while retaining the 60-bit representation of the controller parameters. In applying this adaptive evolution strategy, evolution again occurred over 50 generations. This adaptive (10 + 10)-evolution strategy was used to minimise again the performance measure, Γ, for $\lambda = 0.1$ while μ was left to vary within the selected interval [1.5, 6.0]. The attained optimal values of the performance measure are shown in the fourth column of Table 9.1. The corresponding optimal controllers gains and parameters obtained as μ varies are presented in Table 9.4.

(μ = 1.5) **constrained**	$\alpha = 0.003;\ D = \begin{bmatrix} 0.140 & 0.0 \\ 0.0 & 0.140 \end{bmatrix};\ K_1 = \begin{bmatrix} 416.10 & 23.125 \\ 23.125 & 14.046 \end{bmatrix};\ K_2 = \begin{bmatrix} 3698.5 & 205.54 \\ 205.54 & 124.81 \end{bmatrix}$
(μ = 3.0) **constrained**	$\alpha = 0.048;\ D = \begin{bmatrix} 1.300 & 0.0 \\ 0.0 & 1.300 \end{bmatrix};\ K_1 = \begin{bmatrix} 130.396 & 7.247 \\ 7.247 & 4.402 \end{bmatrix};\ K_2 = \begin{bmatrix} 1057.3 & 58.731 \\ 58.731 & 35.747 \end{bmatrix}$
(μ = 4.0) **constrained**	$\alpha = 0.452;\ D = \begin{bmatrix} 2.549 & 0.0 \\ 0.0 & 2.549 \end{bmatrix};\ K_1 = \begin{bmatrix} 6.985 & 0.300 \\ 0.300 & 0.236 \end{bmatrix};\ K_2 = \begin{bmatrix} 67.516 & 3.752 \\ 3.752 & 2.279 \end{bmatrix}$
(μ = 6.0) **constrained**	$\alpha = 0.476;\ D = \begin{bmatrix} 0.644 & 0.0 \\ 0.0 & 0.644 \end{bmatrix};\ K_1 = \begin{bmatrix} 13.217 & 0.735 \\ 0735 & 0.446 \end{bmatrix};\ K_2 = \begin{bmatrix} 99.876 & 5.551 \\ 5.551 & 3.371 \end{bmatrix}$

Table 9.4: Constrained trajectory-tracking PID controllers designed using adaptive ES.

9.3.3 UNCONSTRAINED DESIGNS

In the *unconstrained* case, the evolutionary design process of Chapter 8 was used to determine the individual controller parameters. This requires the incorporation of the controller matrices

$$K_1 = \begin{bmatrix} K_{11} & K_{12} \\ K_{13} & K_{14} \end{bmatrix} \quad , \tag{9.16}$$

$$K_2 = \begin{bmatrix} K_{21} & K_{22} \\ K_{23} & K_{24} \end{bmatrix} \quad , \tag{9.17}$$

and

$$D_1 = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \quad , \tag{9.18}$$

together with the controller parameter, α, into the evolutionary algorithms. Thus, the four-dimensional *constrained* evolutionary design problem presented in section 9.3.1 is upgraded to become a thirteen-dimensional design problem. Indeed, the *unconstrained* design can be readily effected by representing the controller parameter, α, together with the individual elements of the controller matrices K_1, K_2, and D, as a string of concatenated sub-strings of binary digits as presented in Chapter 9. Each such sub-string represents in binary coded form an individual controller design parameter. In this case, the concatenated sub-strings of binary digits are arranged so that the entire binary string has the form

$$|\alpha|D_{11}|D_{12}|D_{21}|D_{22}|K_{11}|K_{12}|K_{13}|K_{14}|K_{21}|K_{22}|K_{23}|K_{24}| \quad , \tag{9.15}$$

where each individual element is coded as a 10-bit sub-string. It is, therefore, evident that a 130-bit string will represent the controller matrices K_1, K_2, and D, together with the controller parameter, α, for the particular two-link direct-drive robotic manipulator.

In a similar way to the constrained case of section 9.3.3, an evolutionary design procedure was deployed using in turn genetic algorithms, non-adaptive evolution strategies, and adaptive evolution strategies.

In applying a genetic algorithm (GA), results were again obtained using a population N=10, a crossover probability, p_c = 0.6, and a mutation probability, p_m = 0.01, over 50 generations. The results of genetically minimising the performance measure, Γ, for λ = 0.1, while μ was left to vary within the selected interval [1.5, 6.0], are shown in the second column of Table 9.5. The optimal designed controllers gains and parameters obtained as μ varies are presented in Table 9.6.

Performance Measure	GA	Non-adaptive ES	Adaptive ES
$\Gamma(\mu = 1.5)$	0.2801	0.1806	0.1699
$\Gamma(\mu = 3.0)$	0.6844	0.3086	0.3123
$\Gamma(\mu = 4.0)$	0.8310	0.3982	0.4330
$\Gamma(\mu = 6.0)$	0.8370	0.6325	0.5438

Table 9.5: Simulated performance measured for the evolutionarily unconstrained designs of trajectory-tracking PID controllers.

(μ = 1.5) **unconstrained**	$\alpha = 0.569;\ D = \begin{bmatrix} 6.087 & 0.049 \\ 0.015 & 3.993 \end{bmatrix};\ K_1 = \begin{bmatrix} 18.102 & 1.941 \\ 3.723 & 1.575 \end{bmatrix};\ K_2 = \begin{bmatrix} 299.86 & 46.973 \\ 6.370 & 12.209 \end{bmatrix}$
(μ = 3.0) **unconstrained**	$\alpha = 0.325;\ D = \begin{bmatrix} 8.474 & 0.001 \\ 0.008 & 12.055 \end{bmatrix};\ K_1 = \begin{bmatrix} 21.624 & 11.656 \\ 6.048 & 1.184 \end{bmatrix};\ K_2 = \begin{bmatrix} 1063.2 & 11.037 \\ 2.053 & 16.651 \end{bmatrix}$
(μ = 4.0) **unconstrained**	$\alpha = 0.018;\ D = \begin{bmatrix} 9.843 & 0.136 \\ 0.021 & 2.427 \end{bmatrix};\ K_1 = \begin{bmatrix} 16.341 & 1.8637 \\ 1.049 & 1.066 \end{bmatrix};\ K_2 = \begin{bmatrix} 1040.1 & 22.072 \\ 27.833 & 16.495 \end{bmatrix}$
(μ = 6.0) **unconstrained**	$\alpha = 0.569;\ D = \begin{bmatrix} 1.136 & 0.069 \\ 0.041 & 0.353 \end{bmatrix};\ K_1 = \begin{bmatrix} 23.738. & 7.895 \\ 3.859 & 17.379 \end{bmatrix};\ K_2 = \begin{bmatrix} 222.76 & 12.307 \\ 16.310 & 3.138 \end{bmatrix}$

Table 9.6: Unconstrained trajectory-tracking PID controllers designed using GA.

In turn, a non-adaptive (10 + 10)-evolution strategy (ES) was used while retaining the 130-bit representation of the controller parameters. In applying this non-adaptive strategy, results were again obtained using a mutation probability, p_m = 0.01 (which was again selected after a time-consuming trial and error process), over 50 generations. The results of evolutionarily minimising the performance measure, Γ, using a non-adaptive evolution strategy, for $\lambda = 0.1$, while μ was left to vary within the selected interval [1.5, 6.0], are shown in the third column of Table 9.5. The optimal designed controllers gains and parameters obtained as μ varies are presented in Table 9.7.

(μ = 1.5) **unconstrained**	$\alpha = 0.073;\ D = \begin{bmatrix} 0.451 & 0.046 \\ 0.049 & 0.549 \end{bmatrix};\ K_1 = \begin{bmatrix} 29.374 & 0.332 \\ 0.468 & 8.577 \end{bmatrix};\ K_2 = \begin{bmatrix} 164.93 & 46.973 \\ 28.810 & 19.416 \end{bmatrix}$
(μ = 3.0) **unconstrained**	$\alpha = 0.343;\ D = \begin{bmatrix} 0.314 & 0.002 \\ 0.025 & 0.314 \end{bmatrix};\ K_1 = \begin{bmatrix} 37.828 & 1.727 \\ 1.088 & 1.849 \end{bmatrix};\ K_2 = \begin{bmatrix} 168.79 & 11.037 \\ 16.701 & 16.417 \end{bmatrix}$
(μ = 4.0) **unconstrained**	$\alpha = 0.004;\ D = \begin{bmatrix} 0.334 & 0.006 \\ 0.040 & 0.314 \end{bmatrix};\ K_1 = \begin{bmatrix} 32.368 & 1.727 \\ 1.708 & 14.665 \end{bmatrix};\ K_2 = \begin{bmatrix} 1153.37 & 29.396 \\ 13.576 & 12.328 \end{bmatrix}$
(μ = 6.0) **unconstrained**	$\alpha = 0.014;\ D = \begin{bmatrix} 0.921 & 0.027 \\ 0.004 & 0.158 \end{bmatrix};\ K_1 = \begin{bmatrix} 18.278. & 4.052 \\ 3.452 & 9.633 \end{bmatrix};\ K_2 = \begin{bmatrix} 110.96 & 5.276 \\ 2.541 & 6.759 \end{bmatrix}$

Table 9.7: Unconstrained trajectory-tracking PID controllers designed using non-adaptive ES.

Finally, an adaptive (10 +10)-evolution strategy (ES) was used while retaining the 130-bit representation of the controller gains matrices and parameters. In applying this adaptive evolution strategy, evolution again occurred over 50 generations. This adaptive (10 + 10)-evolution strategy was used to minimise again the performance measure, Γ, for $\lambda = 0.1$ while μ was left to vary within the selected interval [1.5, 6.0]. The attained optimal values of the performance measure are shown in the fourth column of Table 9.5. The corresponding optimal controllers gains and parameters obtained as μ varies are presented in Table 9.8

(μ = 1.5) **unconstrained**	$\alpha = 0.231;\ D = \begin{bmatrix} 0.666 & 0.006 \\ 0.004 & 1.997 \end{bmatrix};\ K_1 = \begin{bmatrix} 29.022 & 0.177 \\ 1.243 & 1.614 \end{bmatrix};\ K_2 = \begin{bmatrix} 299.86 & 7.814 \\ 9.084 & 19.065 \end{bmatrix}$
(μ = 3.0) **unconstrained**	$\alpha = 0.742;\ D = \begin{bmatrix} 0.999 & 0.024 \\ 0.010 & 0.079 \end{bmatrix};\ K_1 = \begin{bmatrix} 14.579 & 0.933 \\ 0.158 & 13.037 \end{bmatrix};\ K_2 = \begin{bmatrix} 145.66 & 16.994 \\ 3.908 & 18.209 \end{bmatrix}$
(μ = 4.0) **unconstrained**	$\alpha = 0.204;\ D = \begin{bmatrix} 0.725 & 0.012 \\ 0.039 & 0.158 \end{bmatrix};\ K_1 = \begin{bmatrix} 41.879 & 0.332 \\ 5.002 & 19.374 \end{bmatrix};\ K_2 = \begin{bmatrix} 346.13 & 40.137 \\ 6.057 & 13.964 \end{bmatrix}$
(μ = 6.0) **unconstrained**	$\alpha = 0.079;\ D = \begin{bmatrix} 5.147 & 0.244 \\ 0.007 & 0.862 \end{bmatrix};\ K_1 = \begin{bmatrix} 10.881 & 1.282 \\ 0.778 & 4.235 \end{bmatrix};\ K_2 = \begin{bmatrix} 145.66 & 12.502 \\ 8.303 & 11.316 \end{bmatrix}$

Table 9.8: Unconstrained trajectory-tracking PID controllers designed using adaptive ES.

9.4 PRACTICAL PERFORMANCE OF CONTROLLERS

9.4.1 CONSTRAINED DESIGNS

The practical performance of the *constrained* designs obtained in 9.3.2 was assessed during practical tests of the two-link direct-drive robotic manipulator system shown in Figures 9.1 and 9.2, and described in Appendix C. In these tests,

the desired end-effector trajectory was specified as shown in Figure 9.3. The design and implementation results were obtained for this trajectory using the digital PID controller with a sampling period of 1ms.

By implementing the *constrained* GA designs of Table 9.2, the values of the performance measures presented in the second column of Table 9.9 were obtained for the digital PID trajectory-tracking controller as μ varies. The corresponding trajectory-tracking behaviour is depicted in Figure 9.4. In addition, Figure 9.5 shows the associated tracking errors. The developed torques for both joints, corresponding to the different values of the weighting parameter μ, are shown in Figure 9.6.

Performance Measure	GA	Non-adaptive ES	Adaptive ES
$\Gamma(\mu = 1.5)$	0.4352	0.4022	0.3121
$\Gamma(\mu = 3.0)$	0.5814	0.5543	0.4754
$\Gamma(\mu = 4.0)$	0.9321	0.8878	0.5308
$\Gamma(\mu = 6.0)$	2.1120	1.0775	0.6196

Table 9.9: Actual performance measured for the evolutionarily designed constrained trajectory-tracking PID controllers .

Alternatively, by implementing the *constrained* non-adaptive ES designs of Table 9.3, the values of the performance measures presented in the third column of Table 9.9 were obtained for the digital PID trajectory-tracking controller as μ varies. The corresponding trajectory-tracking behaviour is

depicted in Figure 9.10. In addition, Figure 9.11 shows the associated tracking errors. The developed torques for both joints, corresponding to the different values of the weighting parameter μ, are shown in Figure 9.12.

Finally, by implementing the *constrained* adaptive ES designs of Table 9.4, the values of the performance measures presented in the fourth column of Table 9.9 were obtained as μ varies. The corresponding trajectory-tracking behaviour is depicted in Figure 9.16. In addition, Figure 9.17 shows the associated tracking errors. The developed torques for both joints, corresponding to the different values of the weighting parameter μ, are shown in Figure 9.18

It is evident from Table 9.9 that the controllers designed using the adaptive evolution strategy attain a better degree of optimality (as defined in equation (9.9)) than their genetic and non-adaptive counterparts. However, It is also clear that the corresponding trajectory-tracking errors shown in Figures 9.5, 9.11, and 9.17 exhibit the smallest errors in the genetic case whilst the corresponding torques are much higher and often contain a high-frequency signal. Indeed, this disproportion between the tracking errors and torques is caused by the formulation of the performance measure and its associated sense of optimality as defined in equation (9.9). Indeed, the weighting parameters, λ and μ, are selected in such a way that a tradeoff between the tracking errors and developed torques occurs. Thus, by varying the value of the weighting parameter, μ, it is possible to introduce constraints on the developed torques so that smooth practical torques are

produced and tracking errors are minimised. It is therefore clear that, by targeting smaller values of the performance measure, Γ, smoother torques are more likely to be produced.

9.4.2 UNCONSTRAINED DESIGNS

The practical performance of the *unconstrained* designs obtained in 9.3.3 was assessed in practical tests of the two-link direct-drive robotic manipulator system shown in Figures 9.1 and 9.2, and described in Appendix C. In these tests, the desired end-effector trajectory was specified as shown in Figure 9.3. The test results were obtained for this trajectory using the digital PID controller with a sampling period of 1ms.

By implementing the *unconstrained* GA designs of Table 9.6, the values of the performance measures presented in the second column of Table 9.10 were obtained for the digital PID trajectory-tracking controller as μ varies. The corresponding trajectory-tracking behaviour is depicted in Figure 9.7. In addition, Figure 9.8 shows the associated tracking errors. The developed torques for both joints, corresponding to the different values of the weighting parameter μ, are shown in Figure 9.9.

Performance Measure	GA	Non-adaptive ES	Adaptive ES
$\Gamma(\mu = 1.5)$	0.4121	0.3576	0.3090
$\Gamma(\mu = 3.0)$	0.5537	0.5011	0.4511
$\Gamma(\mu = 4.0)$	0.8971	0.7822	0.5245
$\Gamma(\mu = 6.0)$	1.4378	0.9617	0.6023

Table 9.10: Actual performance measured for the evolutionarily designed unconstrained trajectory-tracking PID controllers .

Alternatively, by implementing the *unconstrained* non-adaptive ES designs of Table 9.7, the values of the performance measures presented in the third column of Table 9.10 were obtained for the digital PID trajectory-tracking controller as μ varies. The corresponding trajectory-tracking behaviour is depicted in Figure 9.13. In addition, Figure 9.14 shows the associated tracking errors. The developed torques for both joints, corresponding to the different values of the weighting parameter μ, are shown in Figure 9.15.

Finally, by implementing the *unconstrained* adaptive ES designs of Table 9.8, the values of the performance measures presented in the fourth column of Table 9.10 were obtained as μ varies. The corresponding trajectory-tracking behaviour is depicted in Figure 9.19. In addition, Figure 9.20 shows the associated tracking errors. The developed torques for both joints, corresponding to the different values of the weighting parameter μ, are shown in Figure 9.21

It is evident from Table 9.10 that the controllers designed using the adaptive evolution strategy again attain a better degree of optimality (in the sense defined in equation (9.9)) than their genetic algorithm and non-adaptive evolution strategy counterparts. However, It is also clear that the corresponding trajectory-tracking errors shown in Figures 9.8, 9.14, and 9.20 exhibit the smallest errors in the genetic case whilst the corresponding torques are much higher and again contain a high-frequency signal. However, the unconstrained adaptive designs succeeded to some extent in achieving the best trade-off between small tracking errors and smooth behaviour of the torques as shown in Figures 9.20 and 9.21. It is therefore clear that, by targeting smaller values of the performance measure, Γ, adaptive unconstrained designs not only produced smoother torques but also an acceptable accuracy.

9.5 EVALUATION OF EVOLUTIONARY DESIGNS

In this attempt to evaluate the performance of the digital trajectory-tracking PID controllers for the direct-drive two-link robotic manipulator controllers designed using all three proposed evolutionary algorithms, it is proposed to consider aspects such as the reliability and quality of the results, the convenience associated with the use of these various evolutionary algorithms, and the relative shortness of the running-time to design the controllers. It is evident, even before initiating any form of quantitative comparison of the different evolutionary algorithms, that the adaptive (10 + 10)-evolution strategy is distinctive in not needing any pre-tuning phase as opposed to the genetic algorithm and the non-adaptive(10 + 10)-

evolution strategy. Indeed, both the latter algorithms required much experiment in the selection of mutation probability, p_m, and/or crossover probability ,p_m. Moreover, the running-time of the genetic algorithm was in all cases greater than its evolution strategy counterpart in both adaptive and non-adaptive variants. It was also observed that within a particular algorithm, the unconstrained version of the design procedure had a longer running-time than its constrained counterpart. This extra time cost evidently relates to the higher dimensionality of the unconstrained version of the design procedure. The quantitative results presented in Table 9.3 indicate that the genetic algorithm achieved a lower degree of optimality than the evolution strategies by producing optimal values of the performance measure larger than those associated with the evolution strategies (see Tables 9.6 and 9.9). Indeed, this lack of optimality translated into the generation of torques which exhibited behaviour considered to be undesirable for practical engineering applications as shown in Figures 9.6 and 9.9. It also emerges from Table 9.6 and 9.7 that evolution strategies are more sensitive to changes in the performance measure (see equation 9.14) while varying the weighting parameter μ within the interval [1.5, 6.0]. It is therefore clear, from this evaluation of the performance of the controllers designed using all three evolutionary algorithms that the *unconstrained* design of digital trajectory-tracking PID controllers using adaptive evolution strategies offered the best results in the most effective way.

9.6 CONCLUSION

In this chapter, genetic algorithms, non-adaptive evolution strategies and adaptive

evolution strategies have been used to design digital trajectory-tracking controllers for a direct-drive two-link robotic manipulator. The practical usefulness of such evolutionary design techniques was illustrated by the practical implementation of various PID controllers on a direct-drive two-link robotic manipulator. The evolutionary design procedure involved finding the optimal values of the appropriate controller gains in each case, such that a pre-defined cost function is minimised.

The practical tests results have clearly indicated that these evolutionary methodologies for designing different types of controllers, while considering the practical constraints and limitations present in real systems, are convenient and very effective. Indeed, it has been shown that the evolutionary design of controllers capable of satisfying practical constraints can be readily effected by formulating an appropriate performance measure that embodies such constraints.

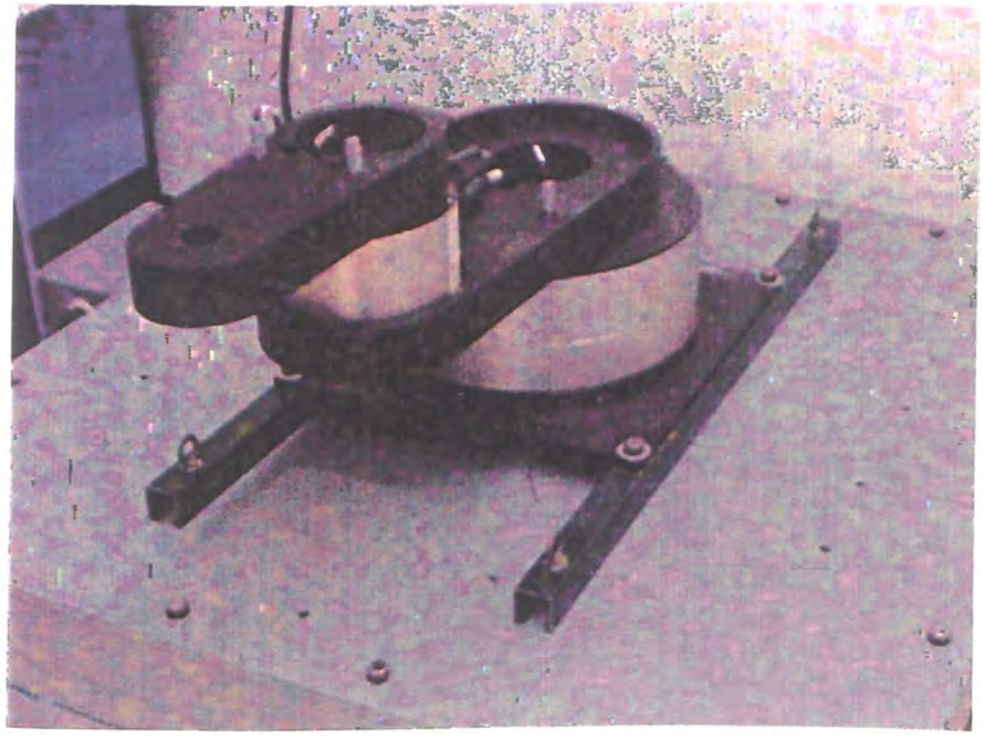

Figure 9.1: Direct-drive two-link robotic manipulator.

Figure 9.2:Control panel of the robotic manipulator.

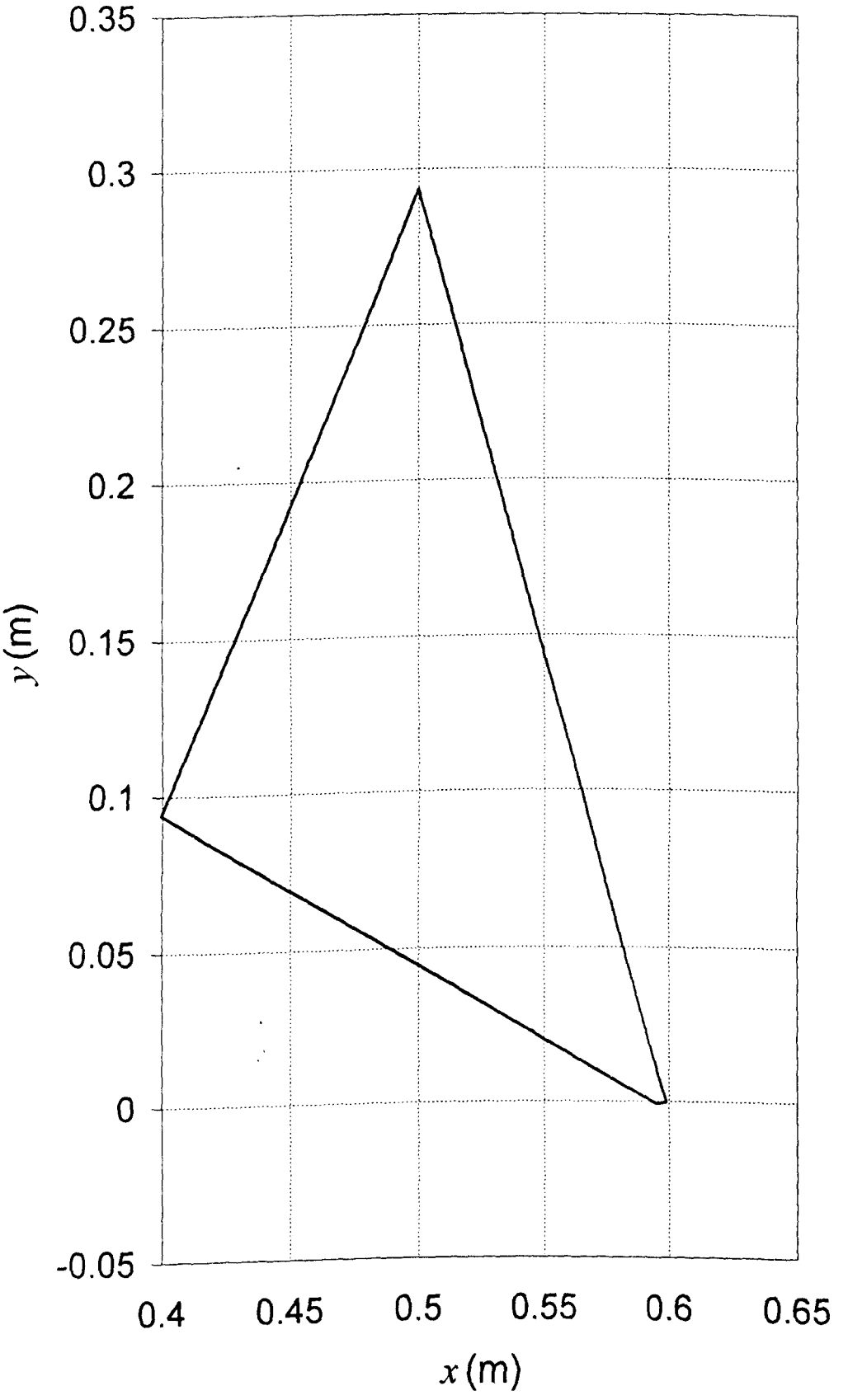

Figure 9.3: Planar desired trajectory.

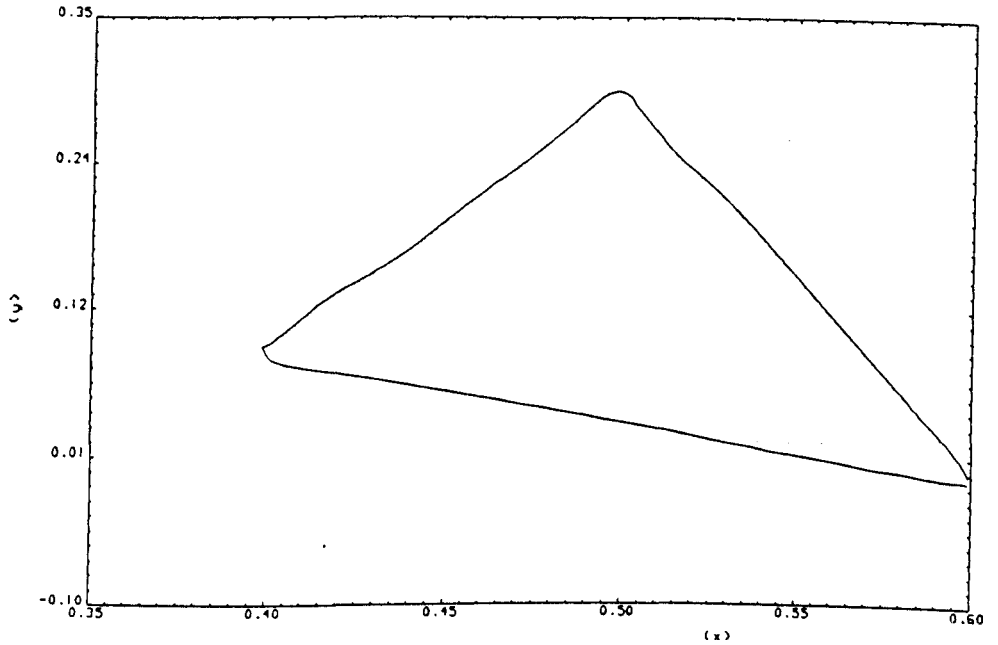

Figure 9.4(a): Trajectory of controlled robot: constrained design, GA (μ = 1.5).

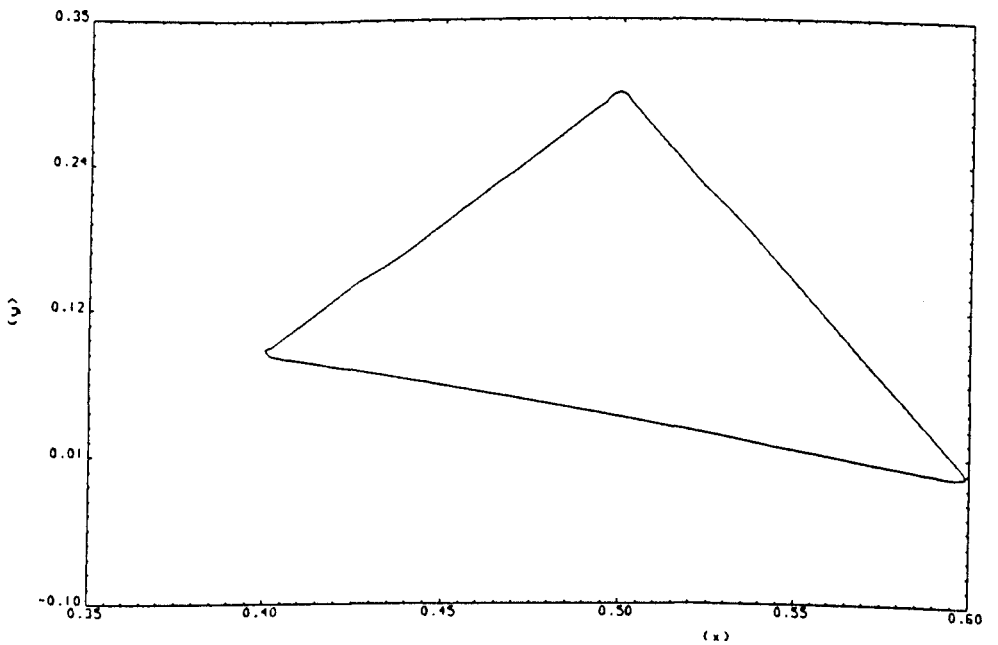

Figure 9.4(b): Trajectory of controlled robot: constrained design, GA (μ = 3.0).

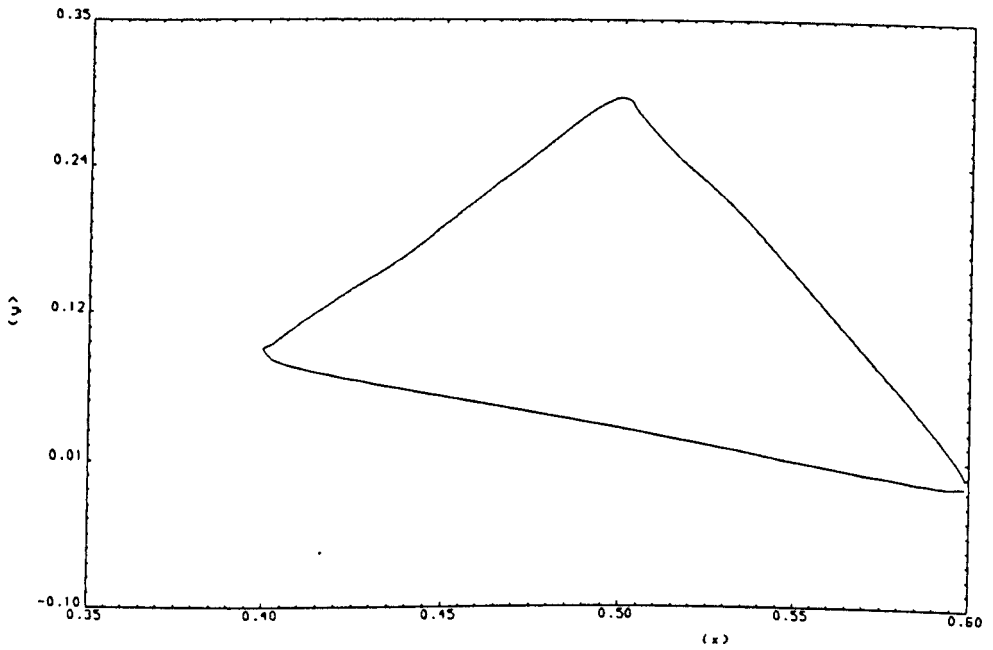

Figure 9.4(c): Trajectory of controlled robot: constrained design, GA (μ = 4.0).

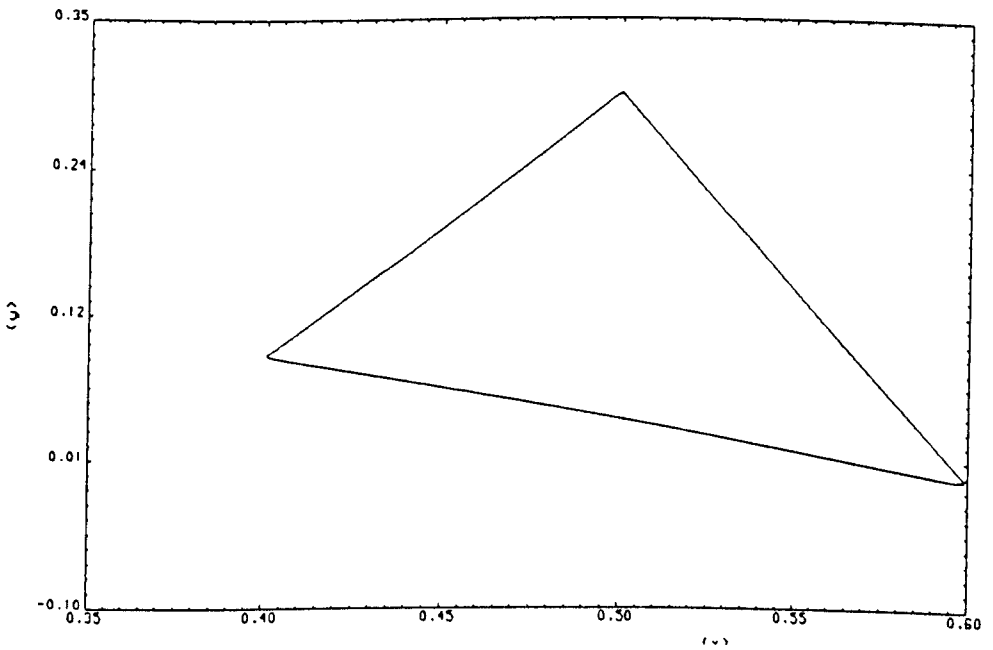

Figure 9.4(d): Trajectory of controlled robot: constrained design, GA (μ = 6.0).

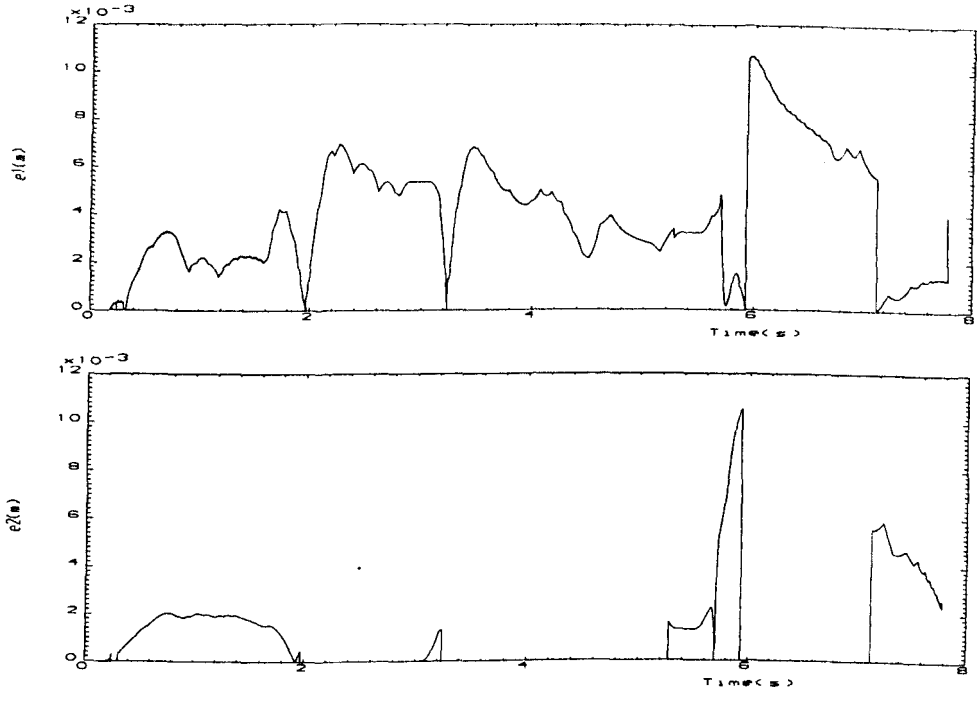

Figure 9.5(a): Time-domain behaviour of errors e1 and e2: constrained design, GA (μ = 1.5).

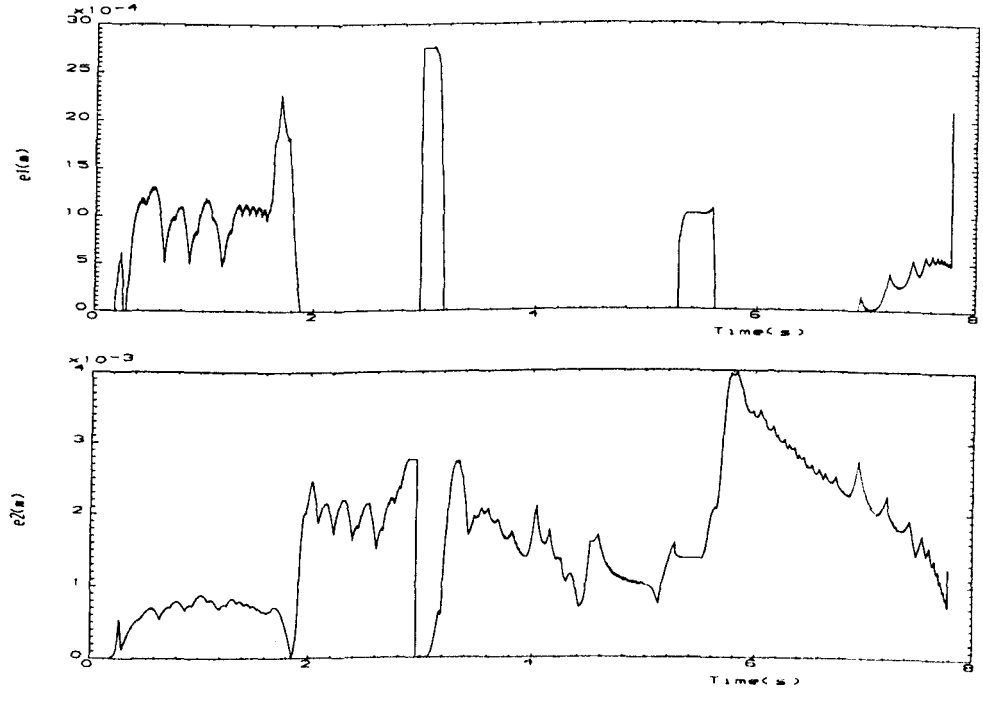

Figure 9.5(b): Time-domain behaviour of errors e1 and e2: constrained design, GA (μ = 3.0).

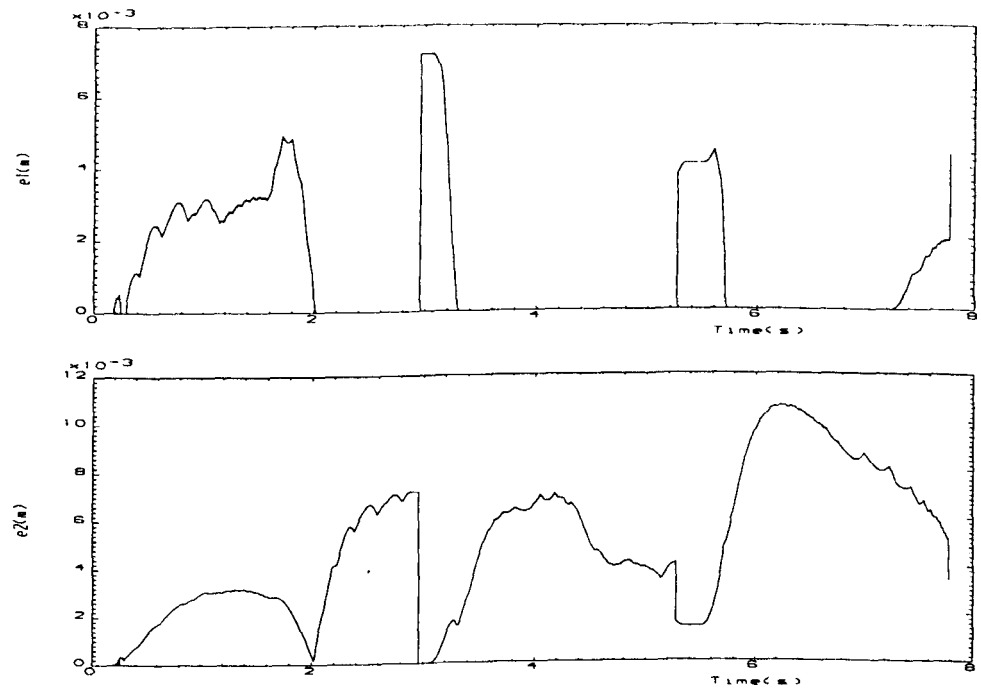

Figure 9.5(c): Time-domain behaviour of errors e1 and e2: constrained design, GA (μ = 4.0).

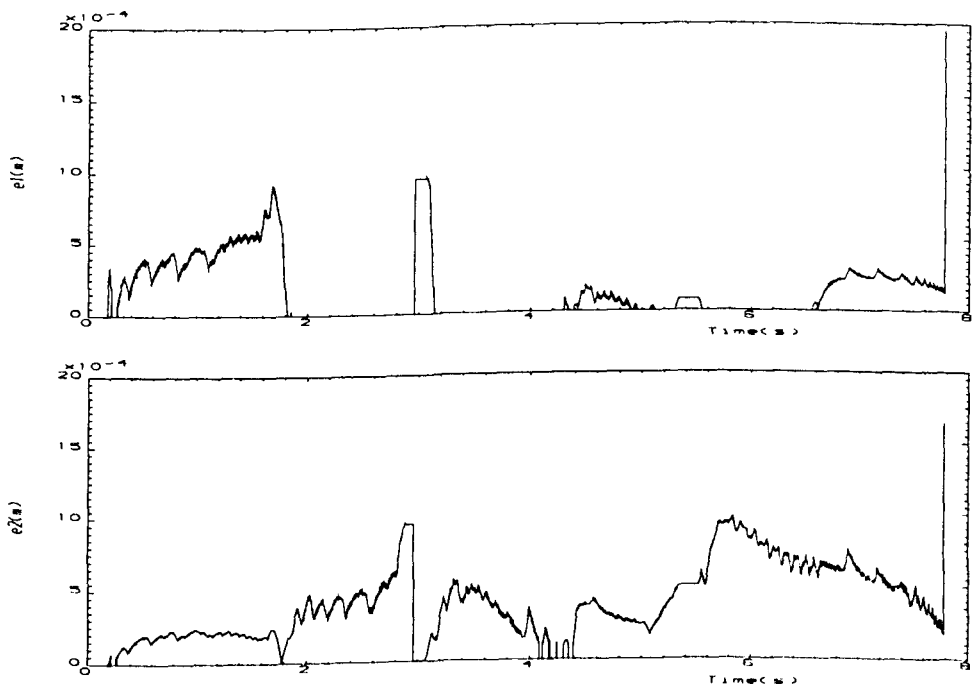

Figure 9.5(d): Time-domain behaviour of errors e1 and e2: constrained design, GA (μ = 6.0).

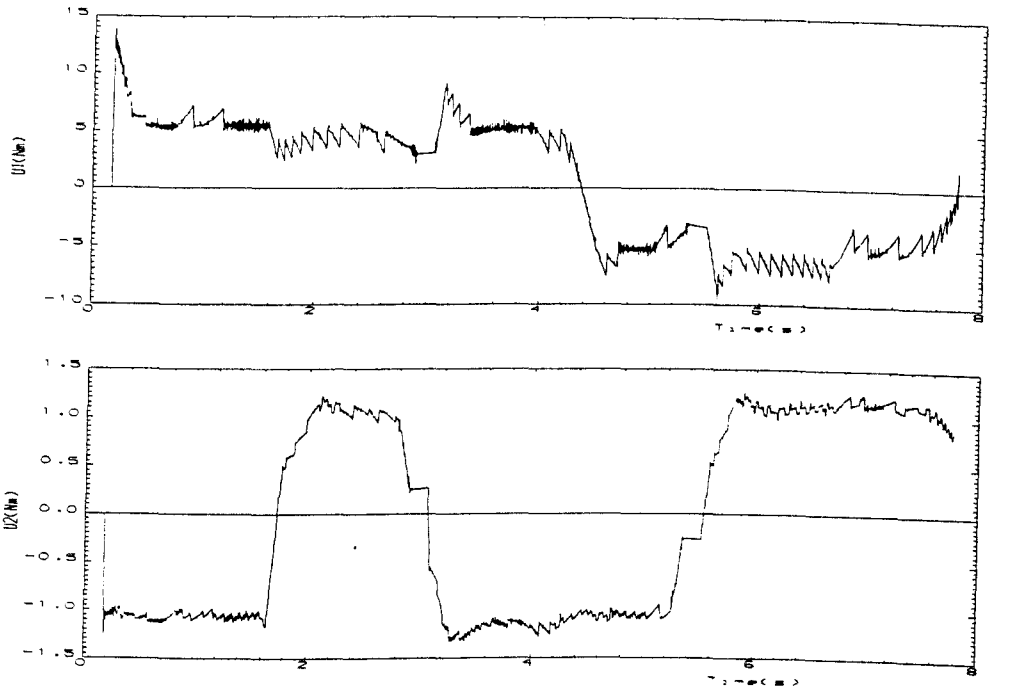

Figure 9.6(a): Time-domain behaviour of torques u1 and u2: constrained design, GA (μ = 1.5).

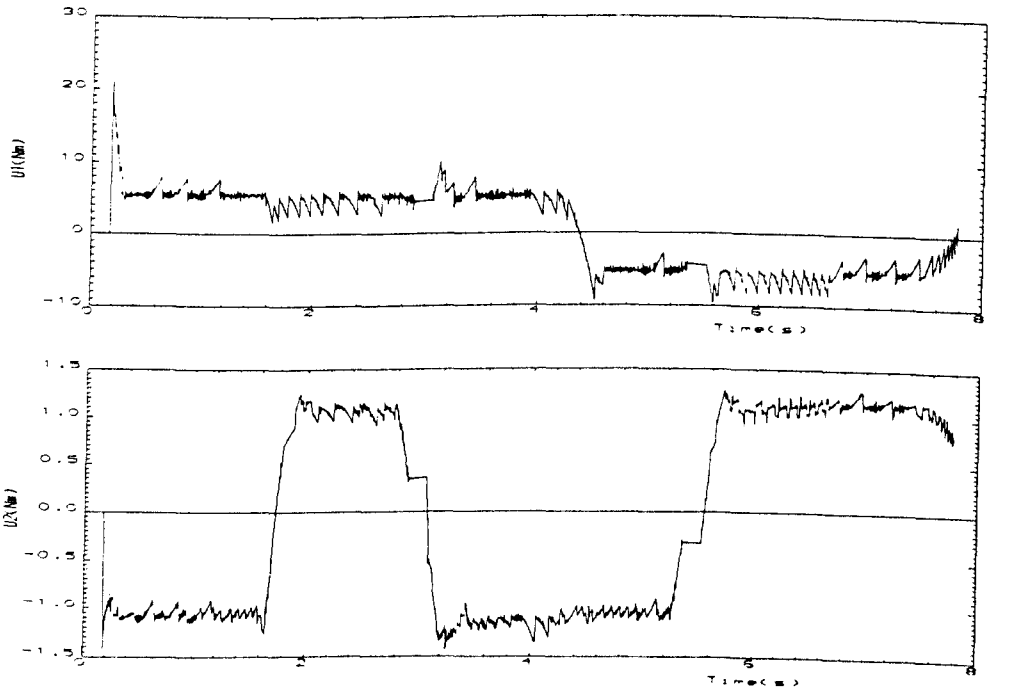

Figure 9.6(b): Time-domain behaviour of torques u1 and u2: constrained design, GA (μ = 3.0).

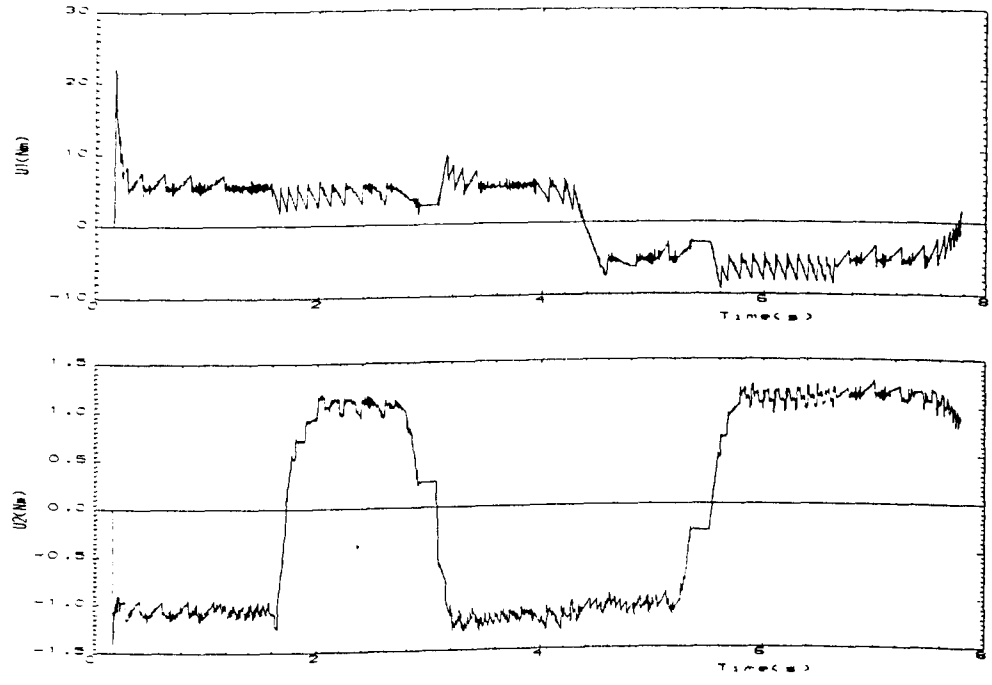

Figure 9.6(c): Time-domain behaviour of torques u1 and u2: constrained design, GA (μ = 4.0).

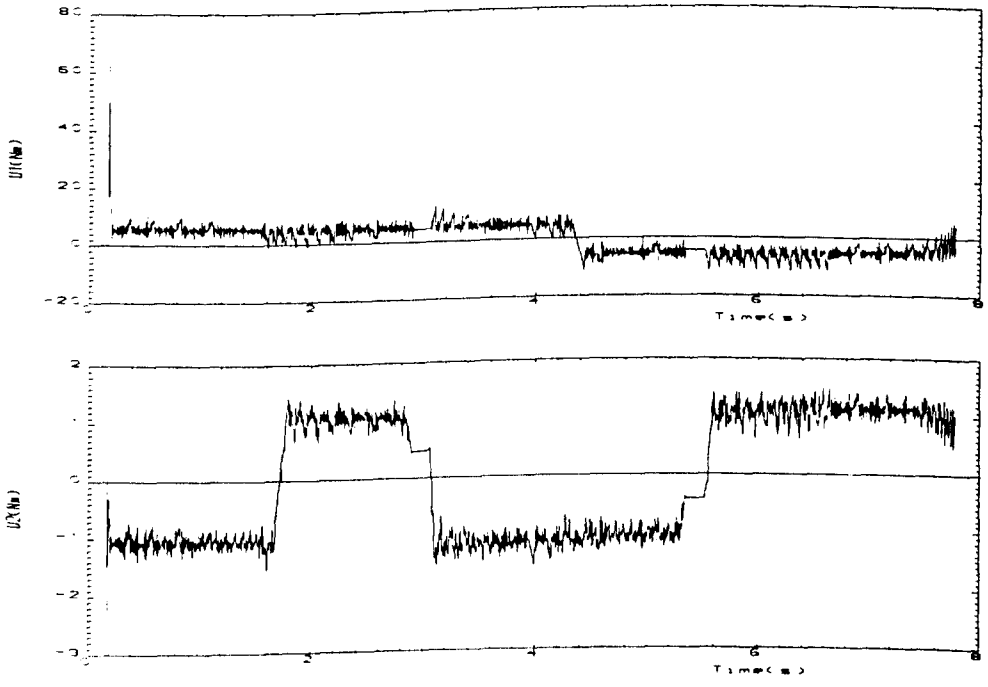

Figure 9.6(d): Time-domain behaviour of torques u1 and u2: constrained design, GA (μ = 6.0).

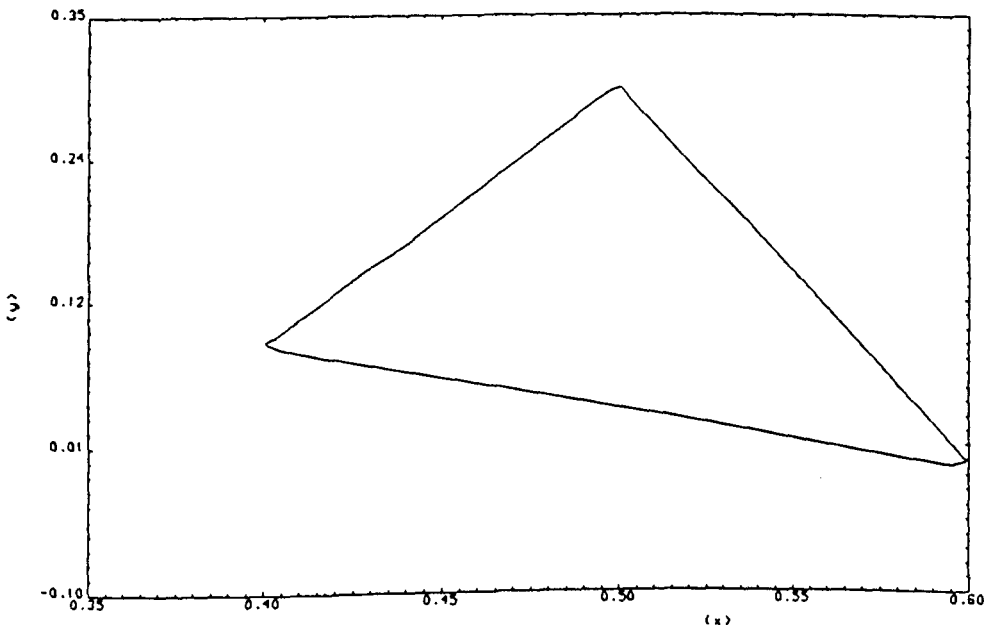

Figure 9.7(a): Trajectory of controlled robot: unconstrained design, GA (μ = 1.5).

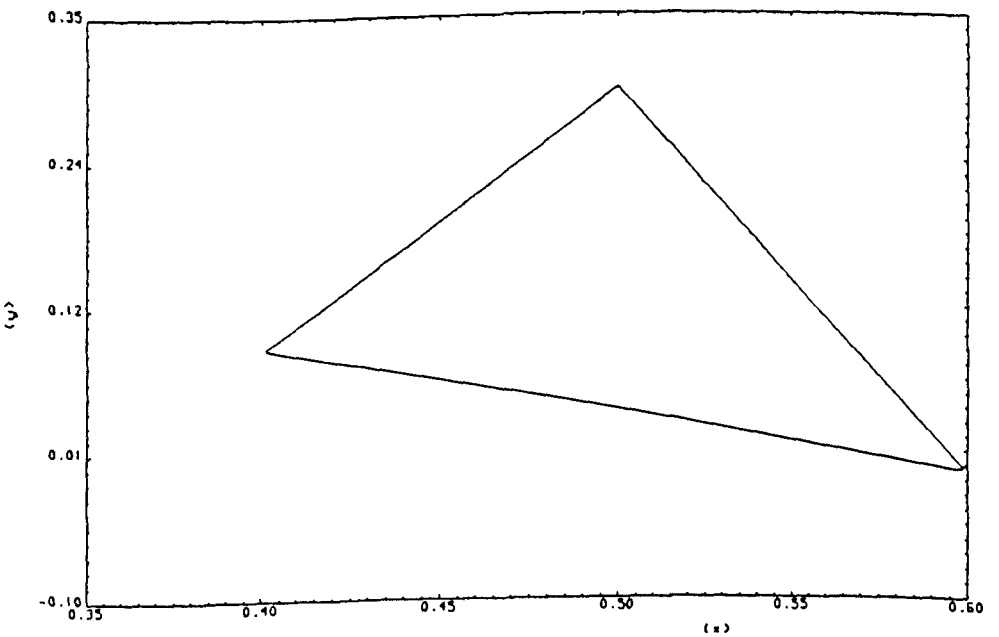

Figure 9.7(b): Trajectory of controlled robot: unconstrained design, GA (μ = 3.0).

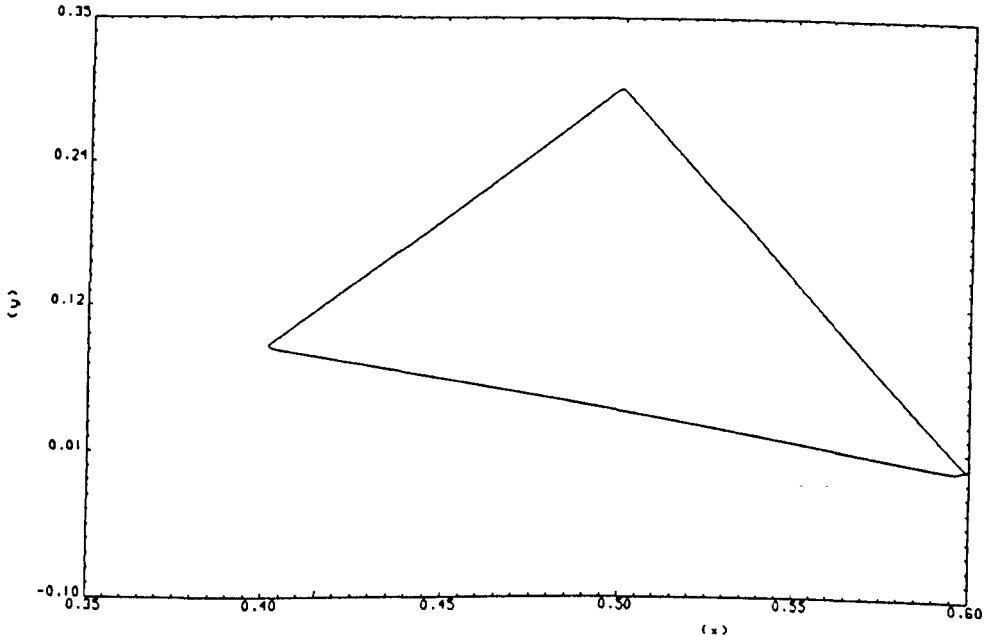

Figure 9.7(c): Trajectory of controlled robot: unconstrained design, GA (μ = 4.0).

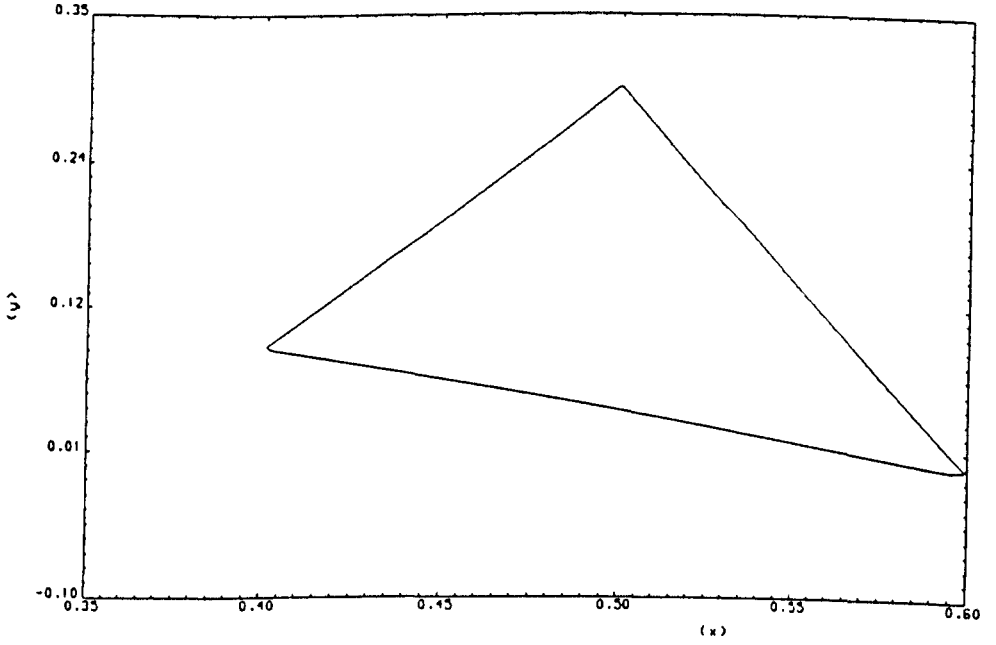

Figure 9.7(d): Trajectory of controlled robot: unconstrained design, GA (μ = 6.0).

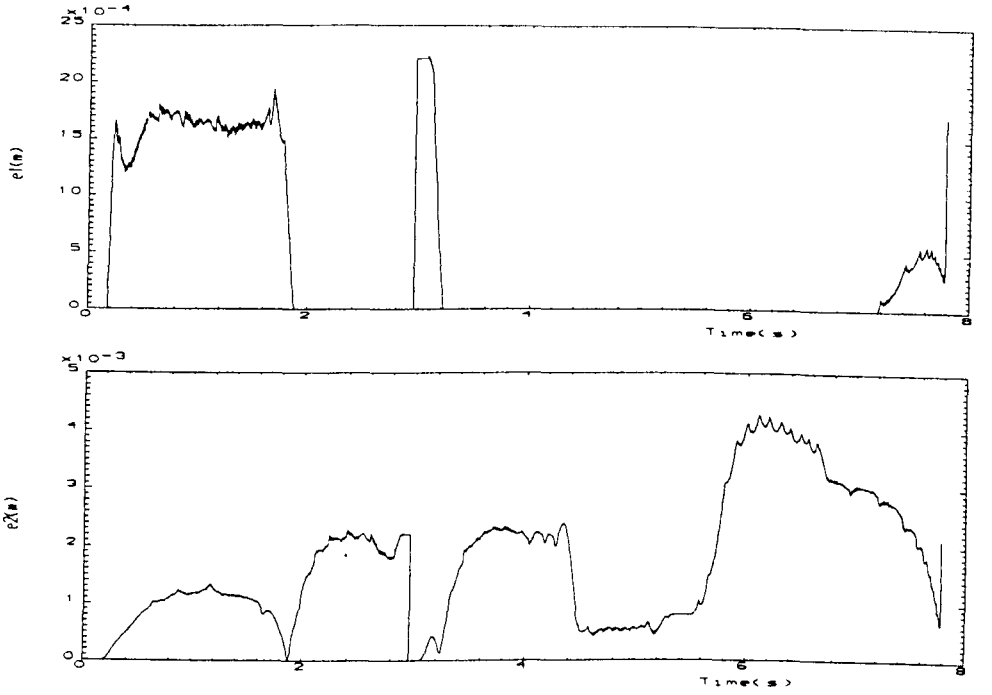

Figure 9.8(a): Time-domain behaviour of errors e1 and e2: unconstrained design, GA (μ = 1.5).

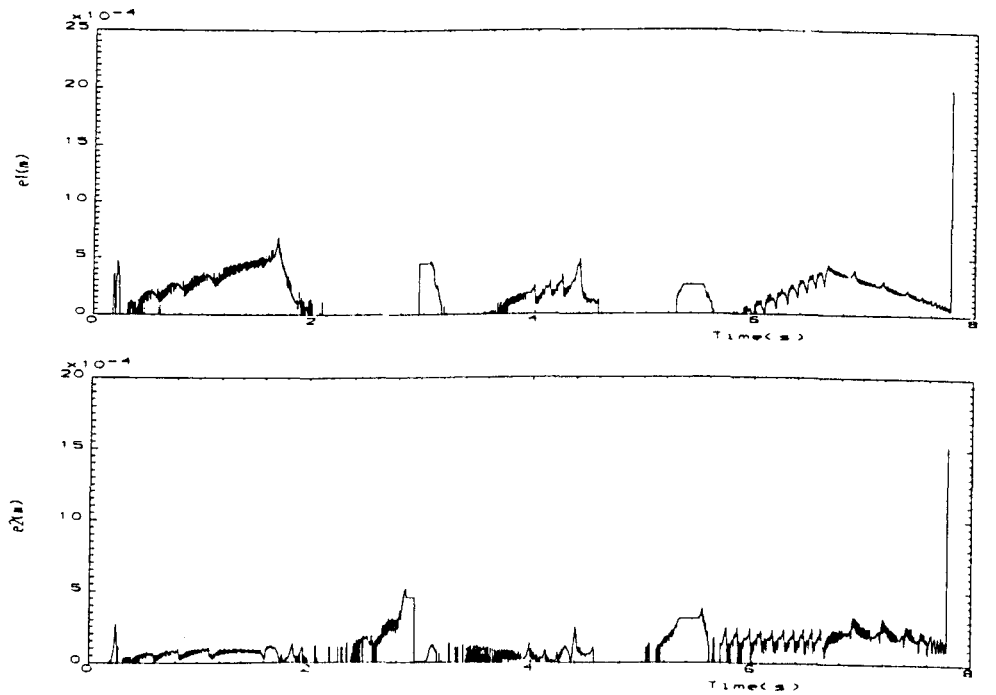

Figure 9.8(b): Time-domain behaviour of errors e1 and e2: unconstrained design, GA (μ = 3.0).

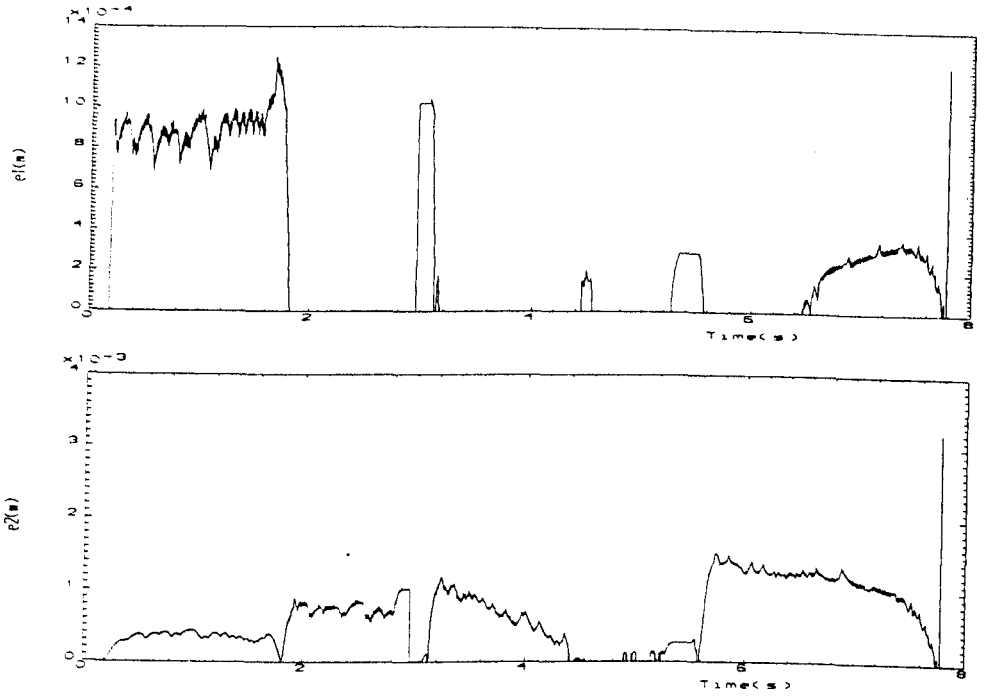

Figure 9.8(c): Time-domain behaviour of errors e1 and e2: unconstrained design, GA (μ = 4.0).

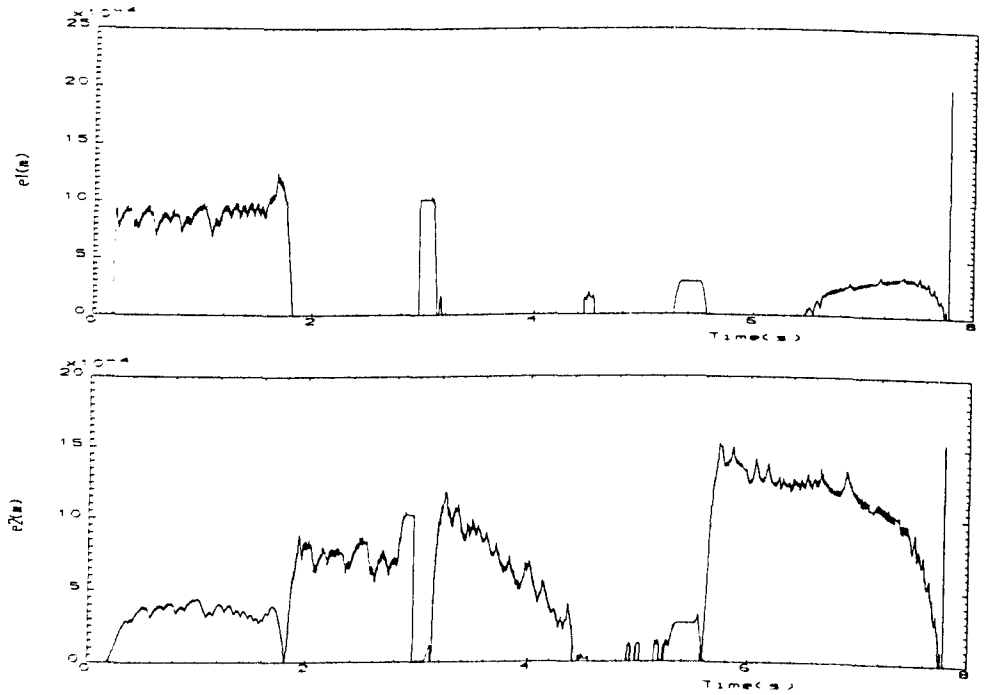

Figure 9.8(d): Time-domain behaviour of errors e1 and e2: unconstrained design, GA (μ = 6.0).

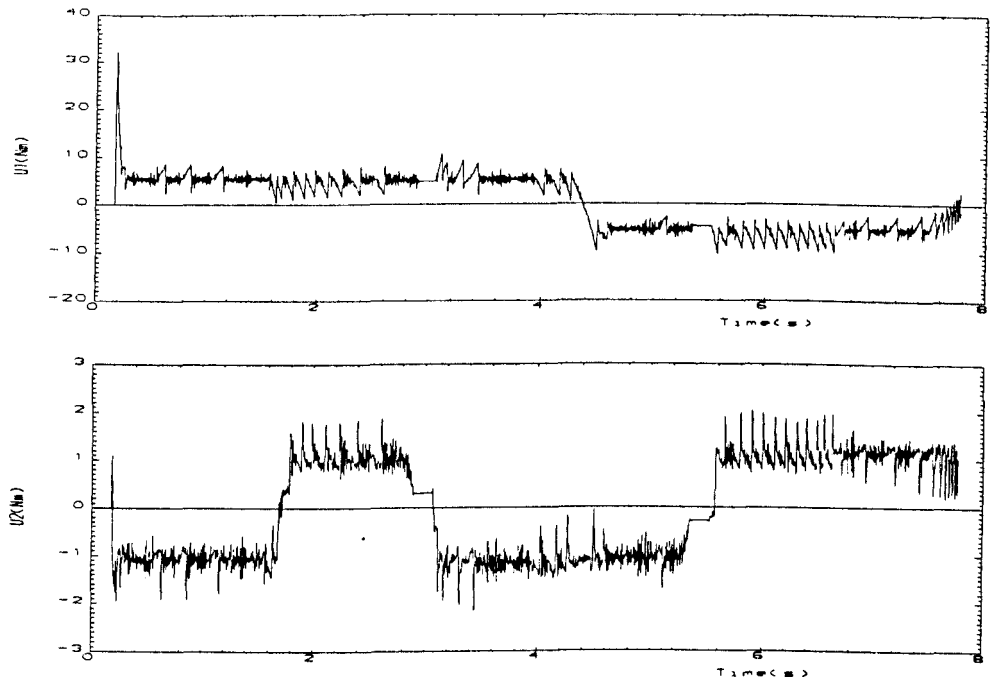

Figure 9.9(a): Time-domain behaviour of torques u1 and u2: unconstrained design, GA (μ = 1.5).

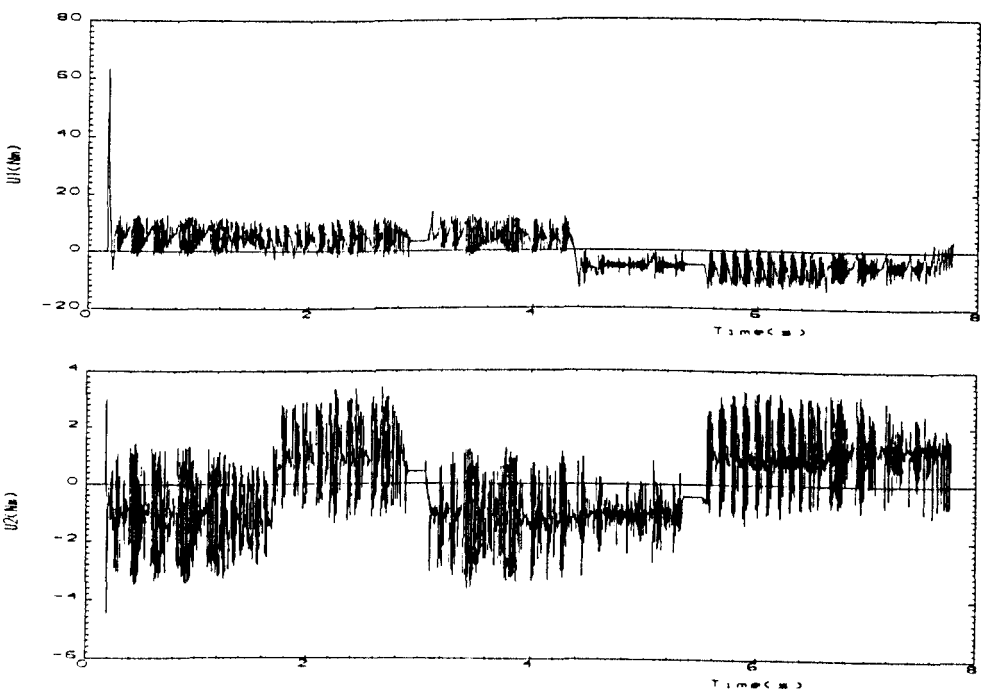

Figure 9.9(b): Time-domain behaviour of torques u1 and u2: unconstrained design, GA (μ = 3.0).

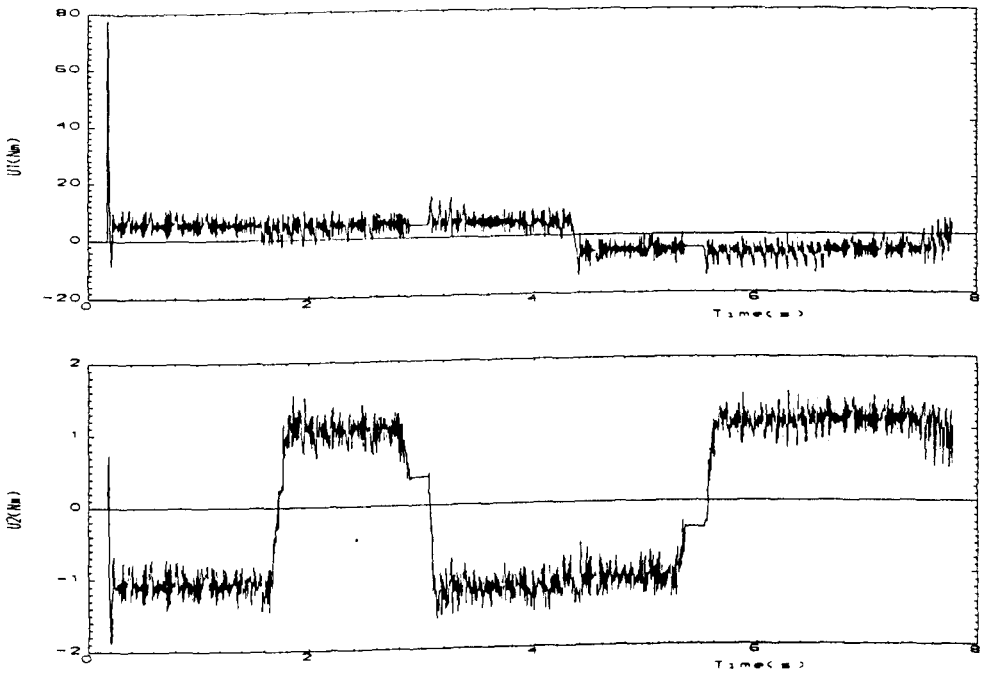

Figure 9.9(c): Time-domain behaviour of torques u1 and u2: unconstrained design, GA (μ = 4.0).

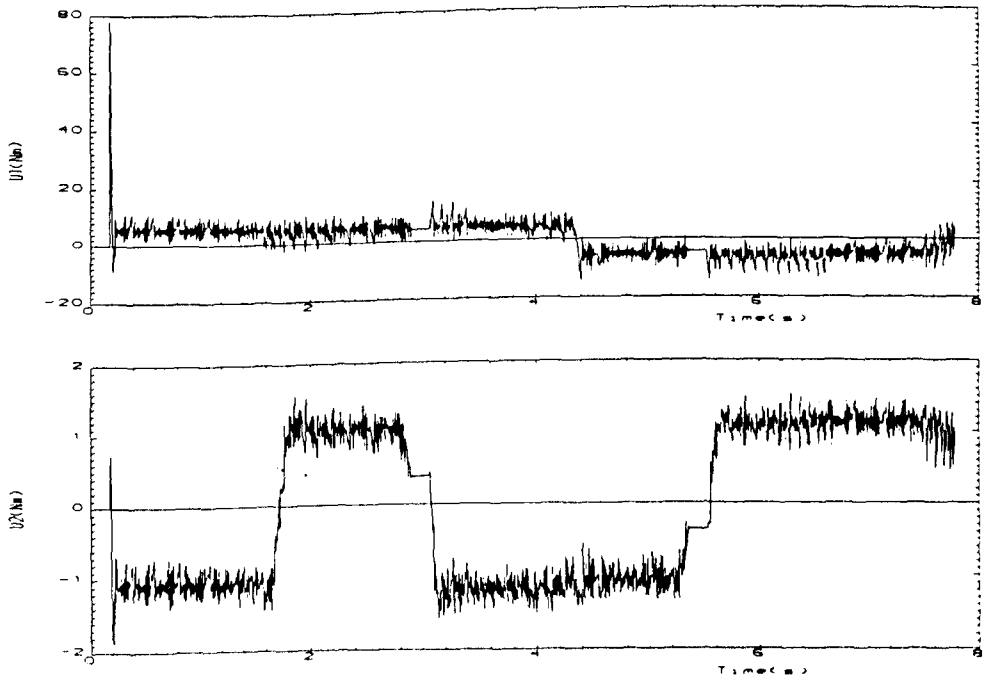

Figure 9.9(d): Time-domain behaviour of torques u1 and u2: unconstrained design, GA (μ = 6.0).

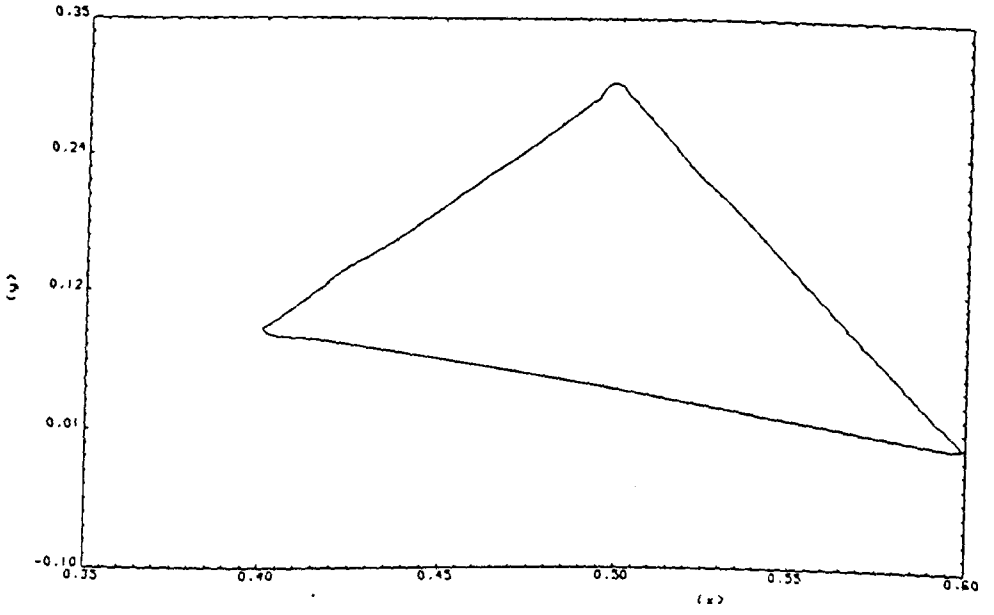

Figure 9.10(a): Trajectory of controlled robot:
constrained design, non-adaptive ES (μ = 1.5).

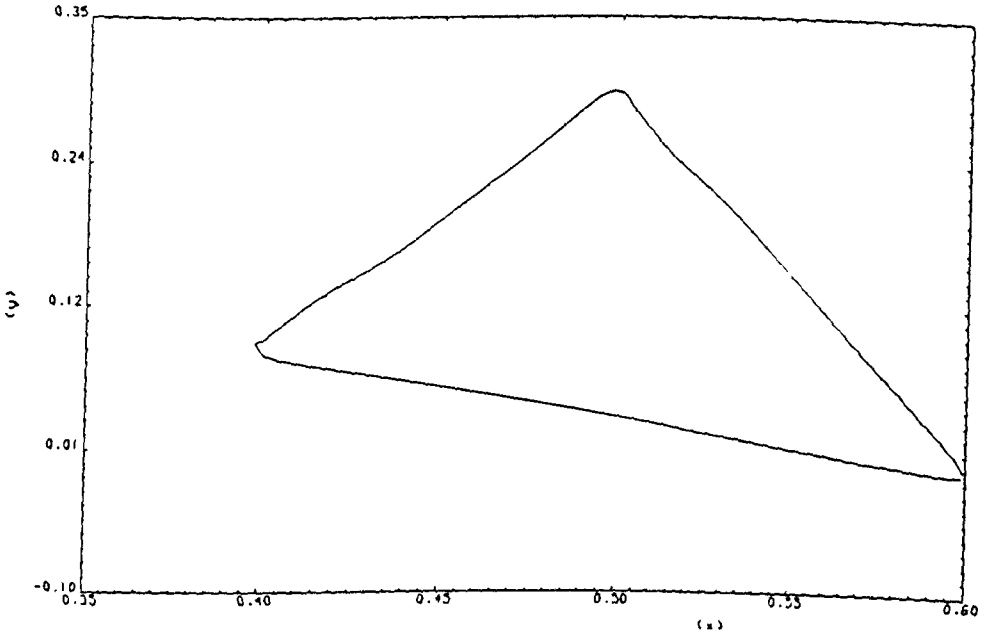

Figure 9.10(b): Trajectory of controlled robot:
constrained design, non-adaptive ES (μ = 3.0).

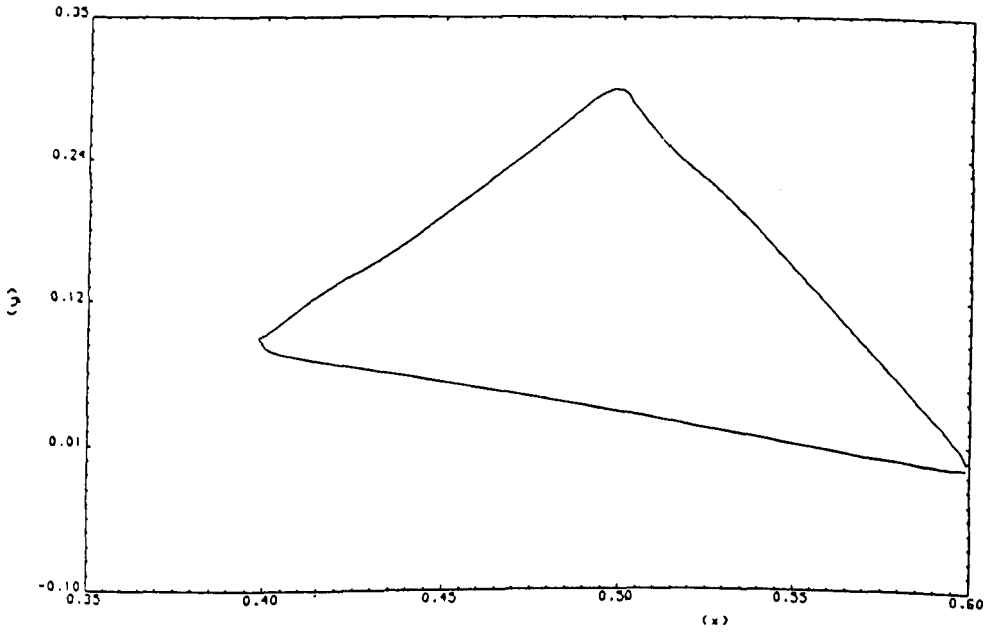

Figure 9.10(c): Trajectory of controlled robot:
constrained design, non-adaptive ES (μ = 4.0).

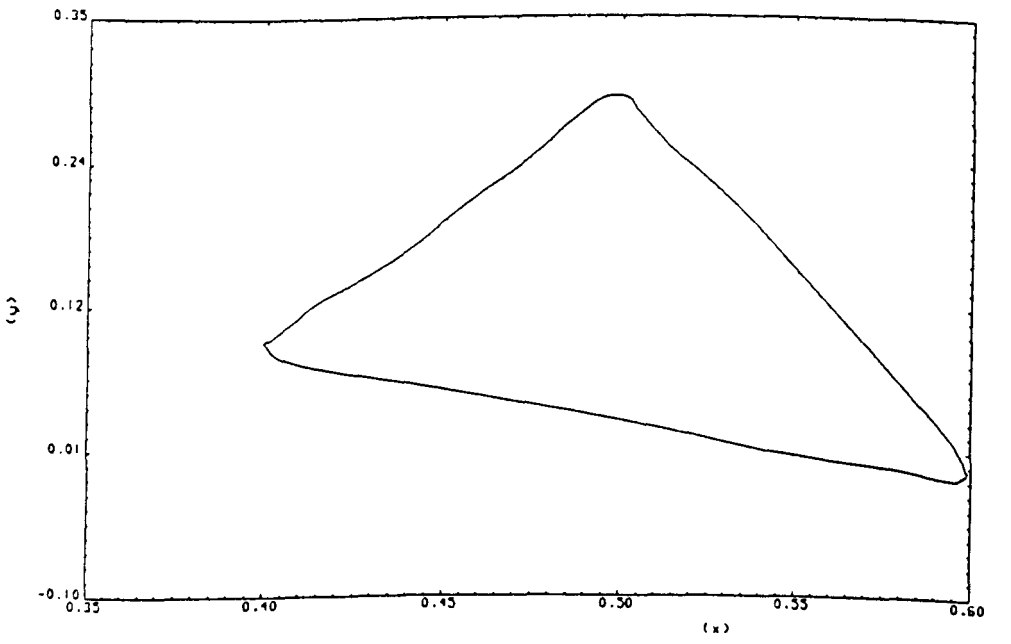

Figure 9.10(d): Trajectory of controlled robot:
constrained design, non-adaptive ES (μ = 6.0).

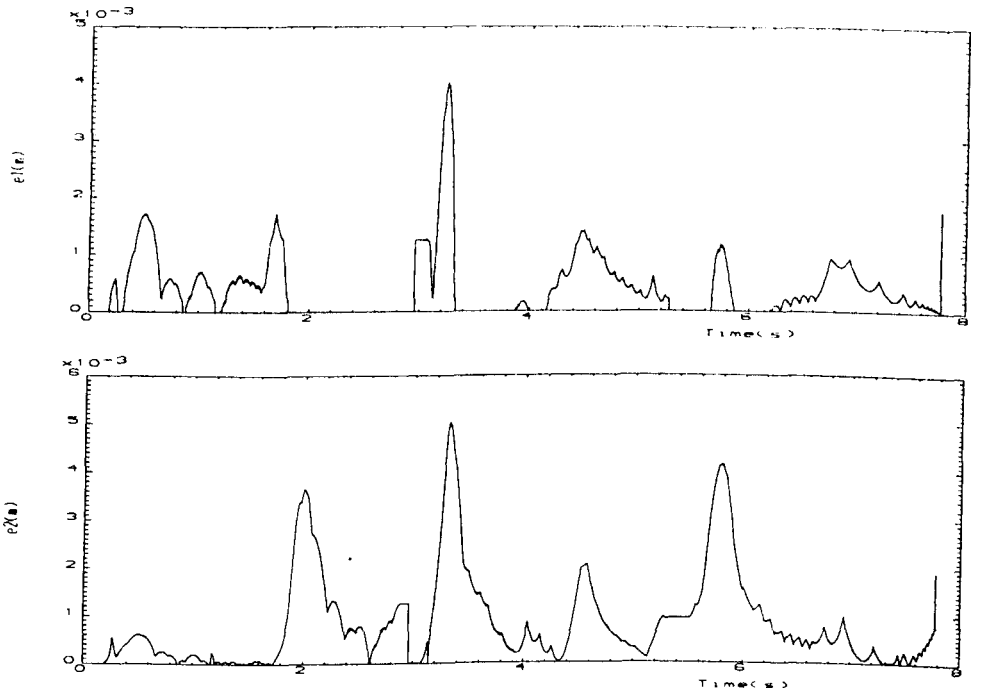

Figure 9.11(a): Time-domain behaviour of errors e1 and e2: constrained design, non-adaptive ES (μ = 1.5).

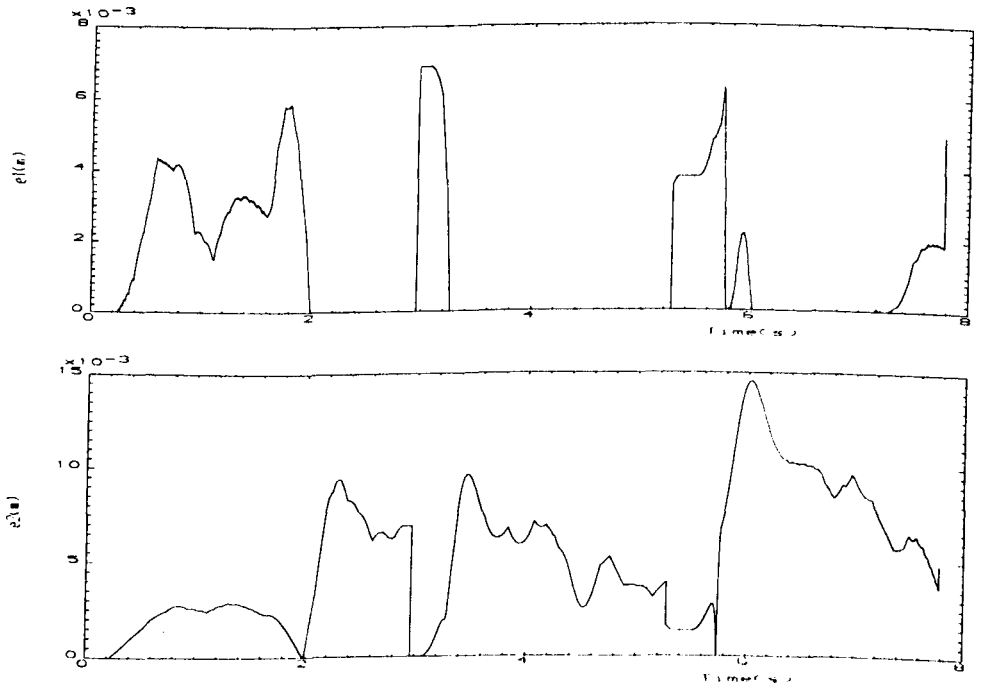

Figure 9.11(b): Time-domain behaviour of errors e1 and e2: constrained design, non-adaptive ES (μ = 3.0).

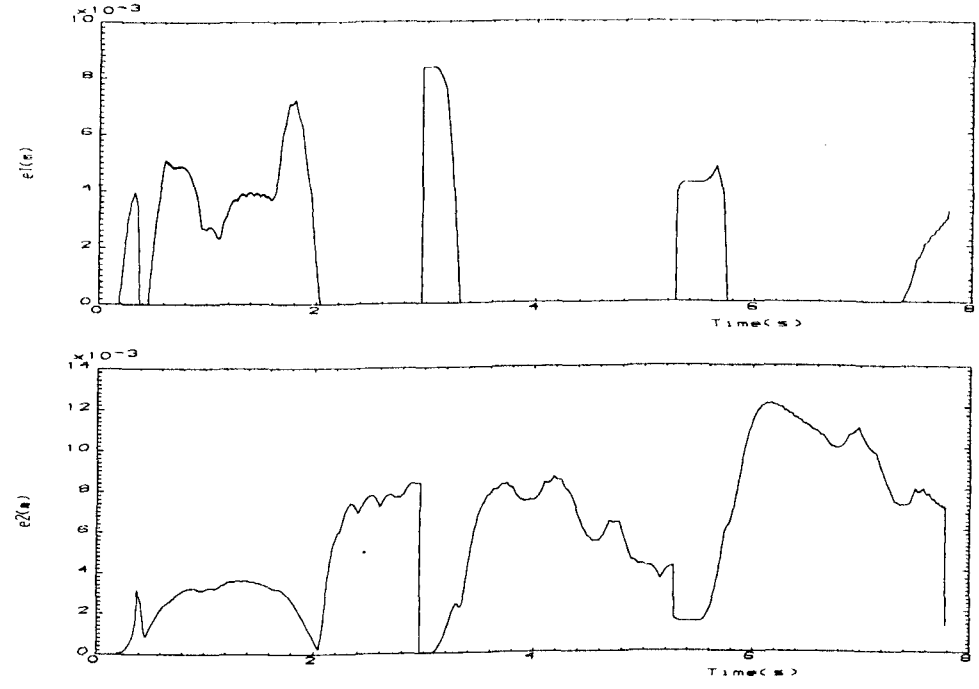

Figure 9.11(c): Time-domain behaviour of errors e1 and e2: constrained design, non-adaptive ES (μ = 4.0).

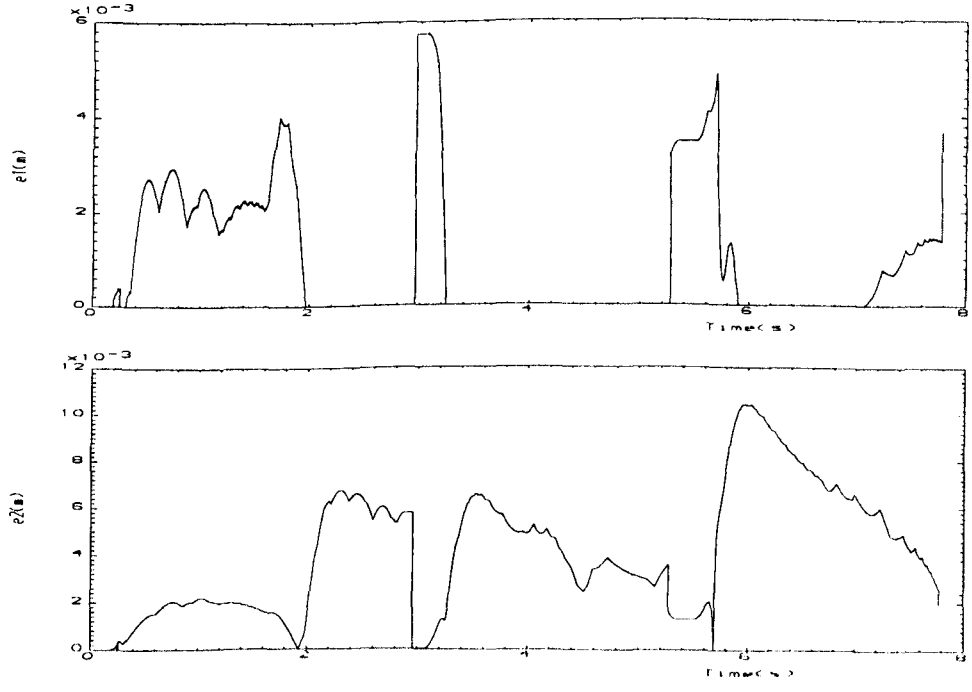

Figure 9.11(d): Time-domain behaviour of errors e1 and e2: constrained design, non-adaptive ES (μ = 6.0).

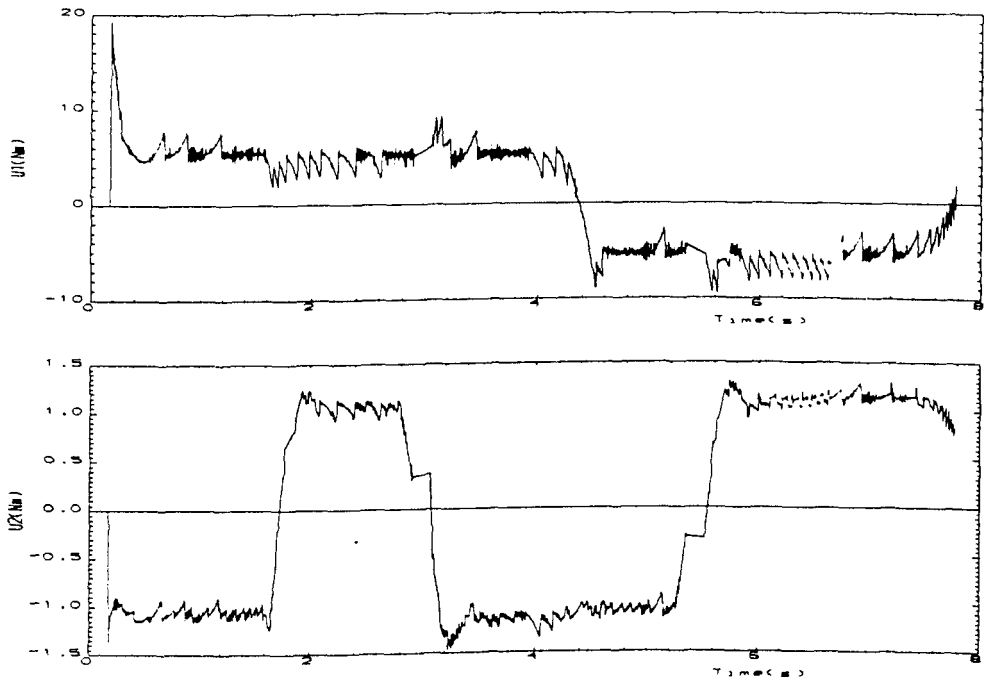

Figure 9.12(a): Time-domain behaviour of torques u1 and u2: constrained design, non-adaptive ES (μ = 1.5).

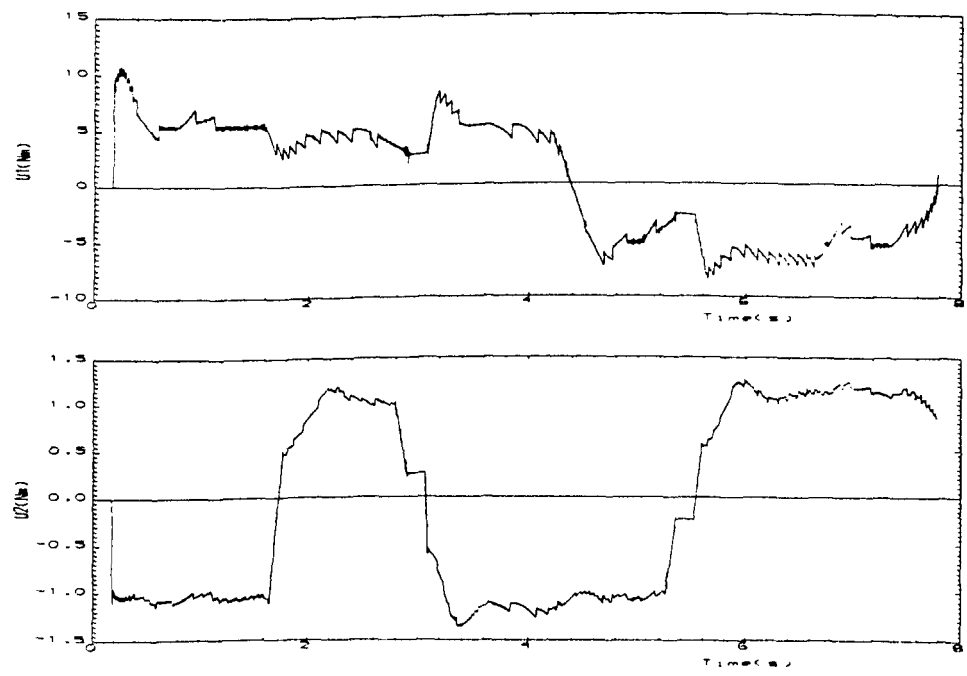

Figure 9.12(b): Time-domain behaviour of torques u1 and u2: constrained design, non-adaptive ES (μ = 3.0).

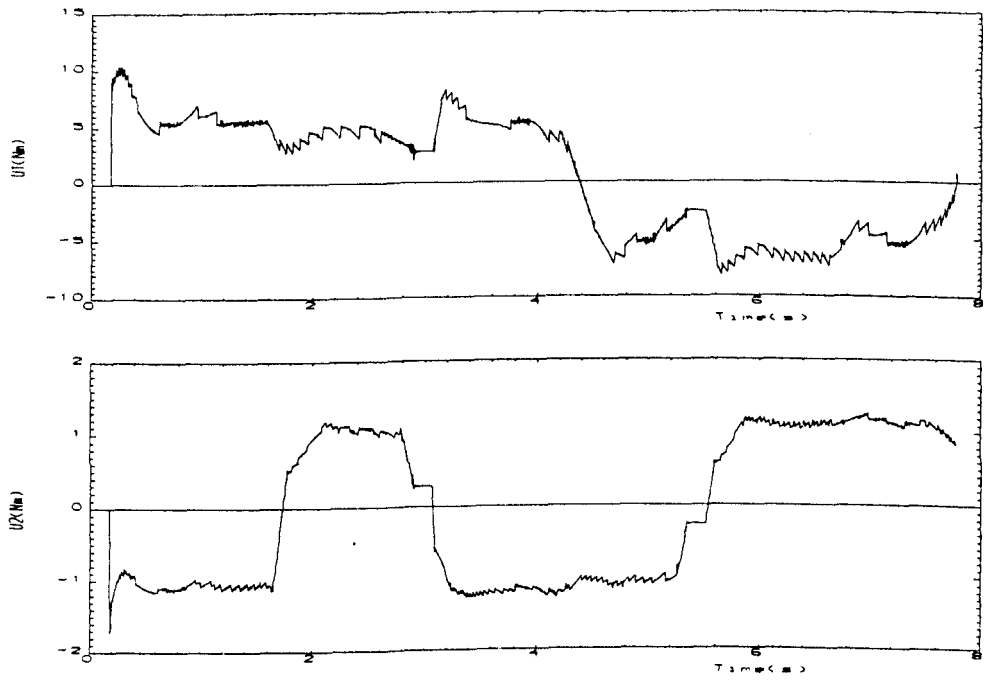

Figure 9.12(c): Time-domain behaviour of torques u1 and u2: constrained design, non-adaptive ES (μ = 4.0).

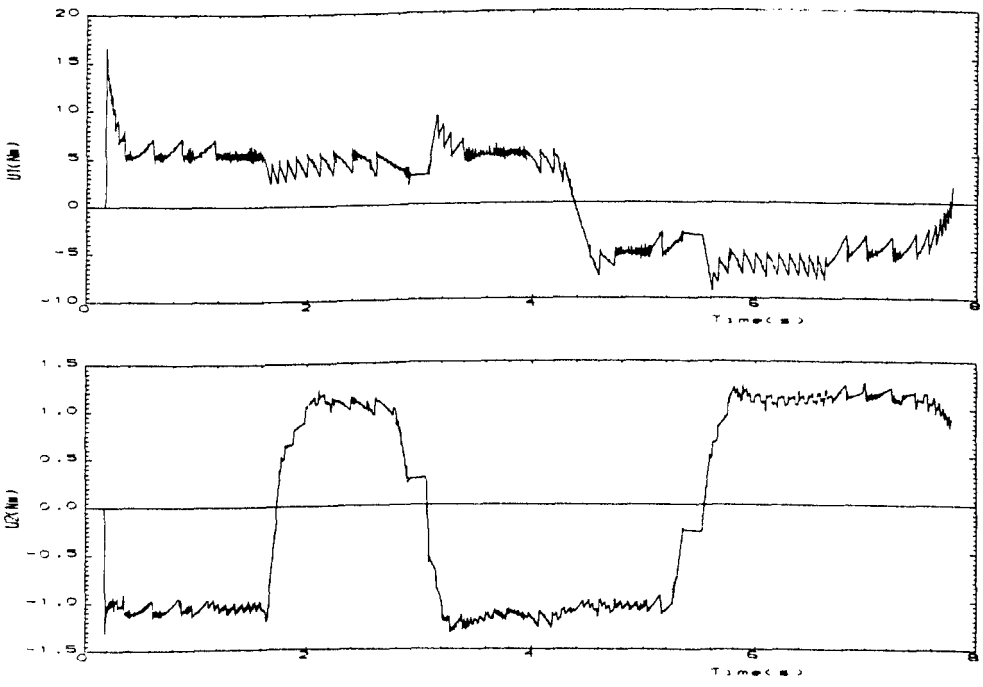

Figure 9.12(d): Time-domain behaviour of torques u1 and u2: constrained design, non-adaptive ES (μ = 6.0).

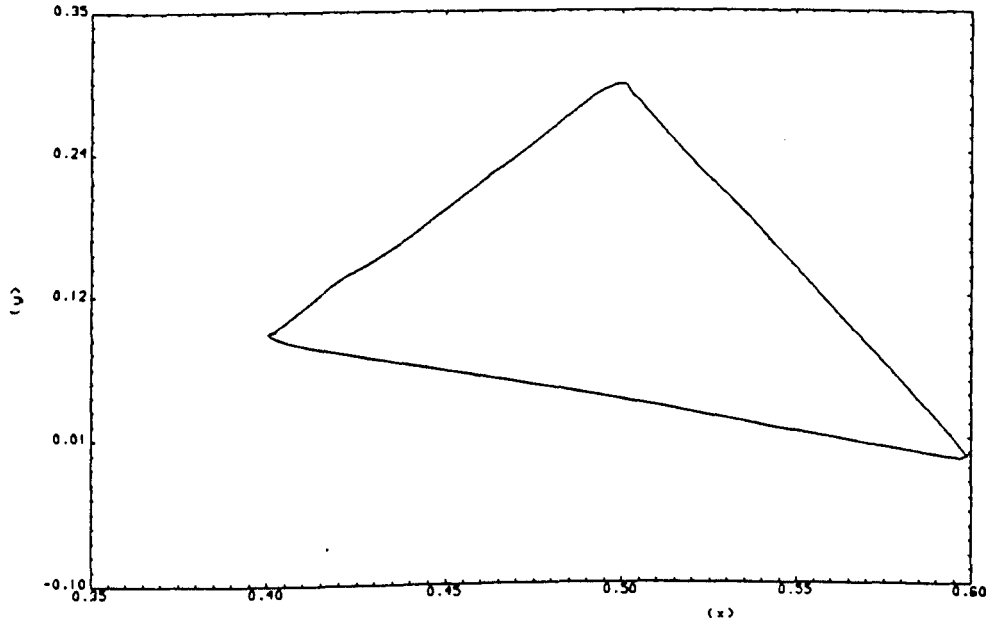

Figure 9.13(a) : Trajectory of controlled robot: unconstrained design, non-adaptive ES (μ = 1.5).

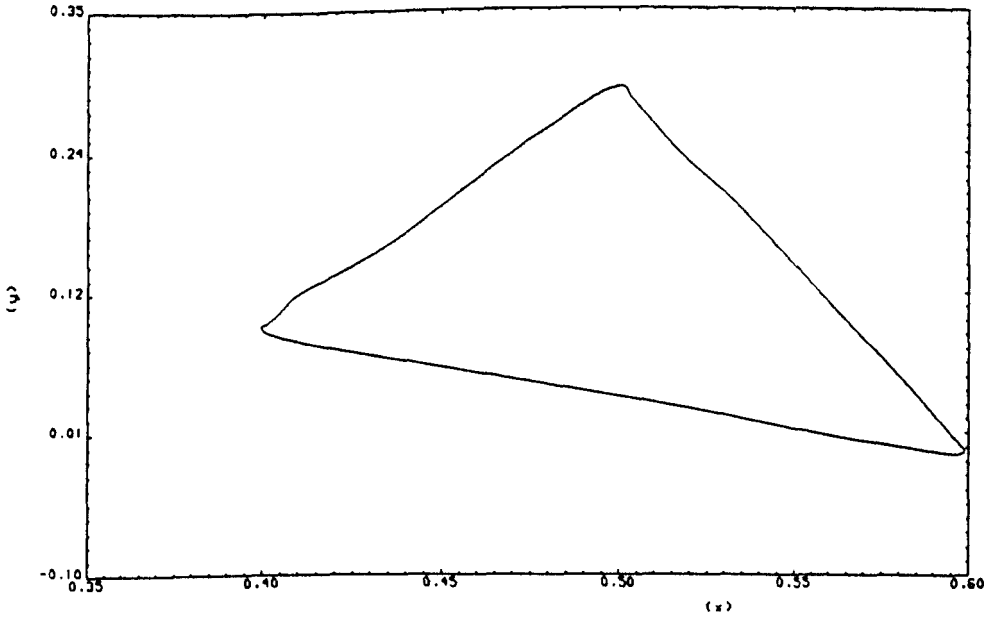

Figure 9.13(b) : Trajectory of controlled robot: unconstrained design, non-adaptive ES (μ = 3.0).

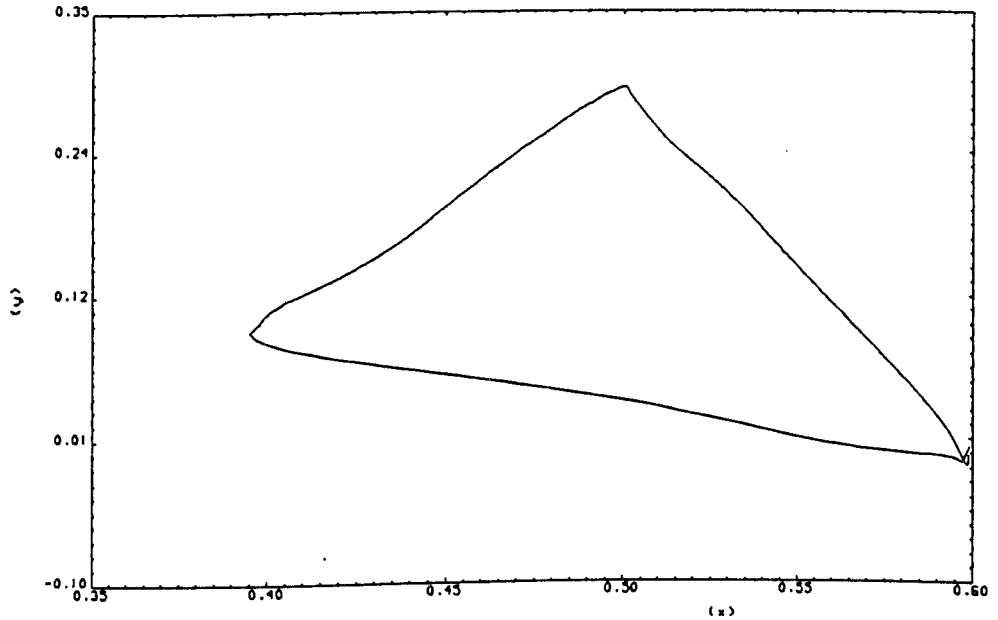

Figure 9.13(c) : Trajectory of controlled robot: unconstrained design, non-adaptive ES (μ = 4.0).

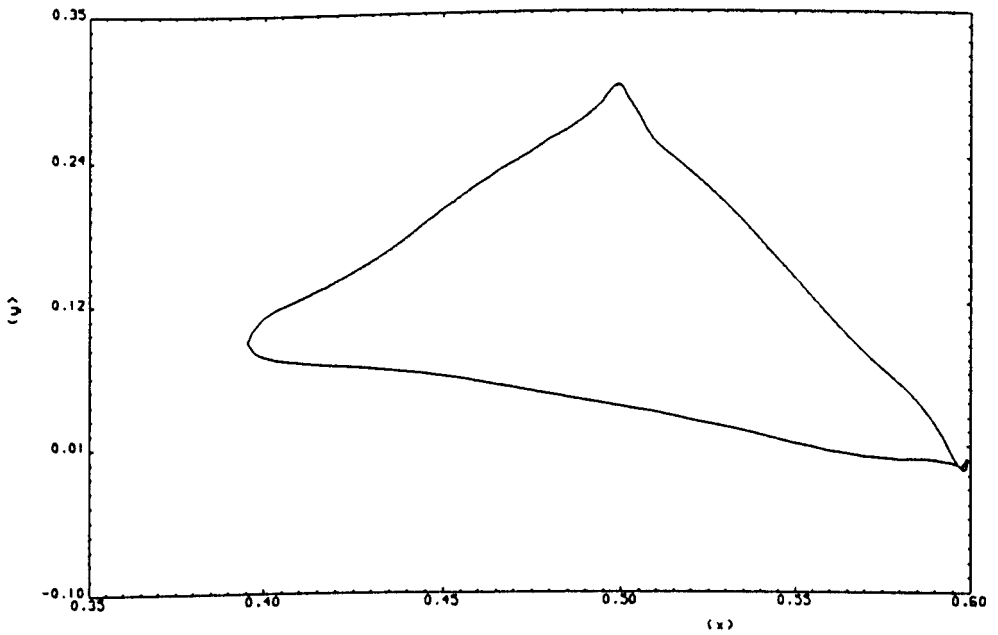

Figure 9.13(d) : Trajectory of controlled robot: unconstrained design, non-adaptive ES (μ = 6.0).

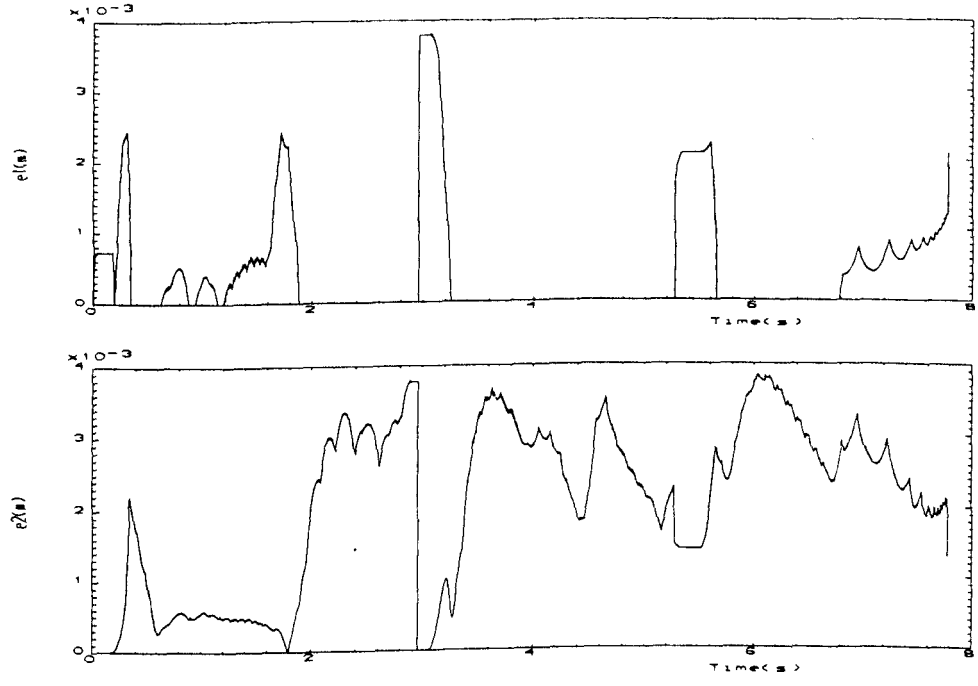

Figure 9.14(a): Time-domain behaviour of errors e1 and e2: unconstrained design, non-adaptive ES (μ = 1.5).

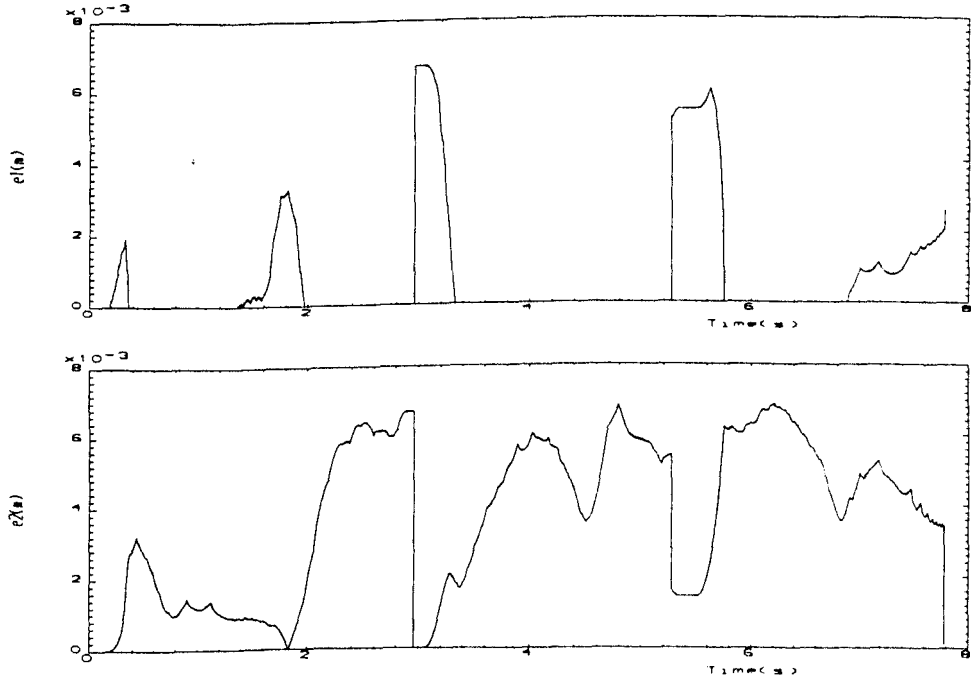

Figure 9.14(b): Time-domain behaviour of errors e1 and e2: unconstrained design, non-adaptive (μ = 3.0).

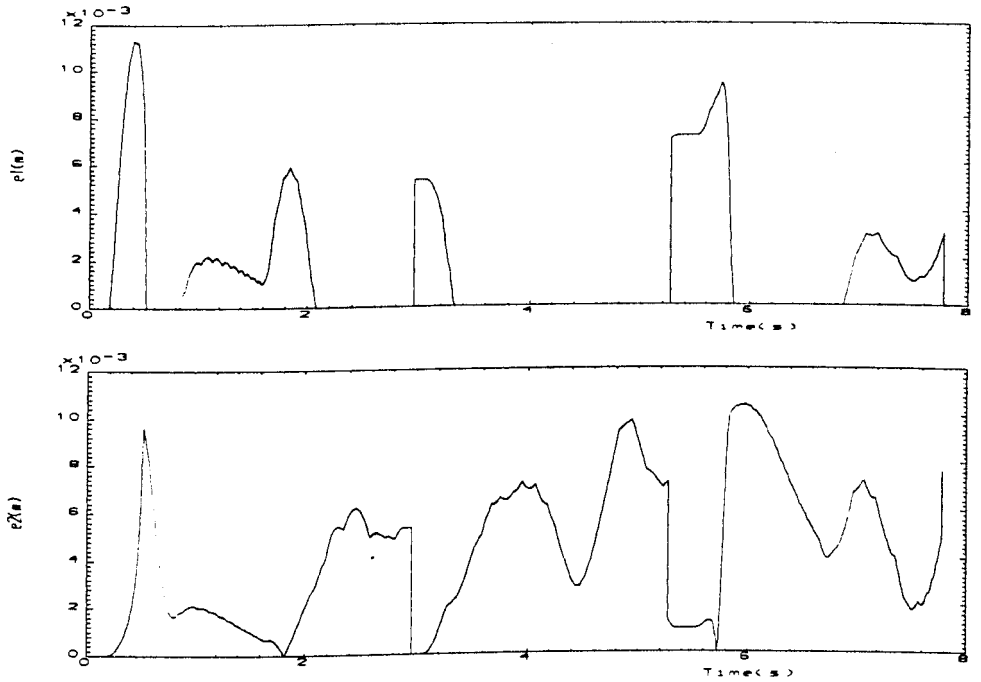

Figure 9.14(c): Time-domain behaviour of errors e1 and e2: unconstrained design, non-adaptive ES (μ = 4.0).

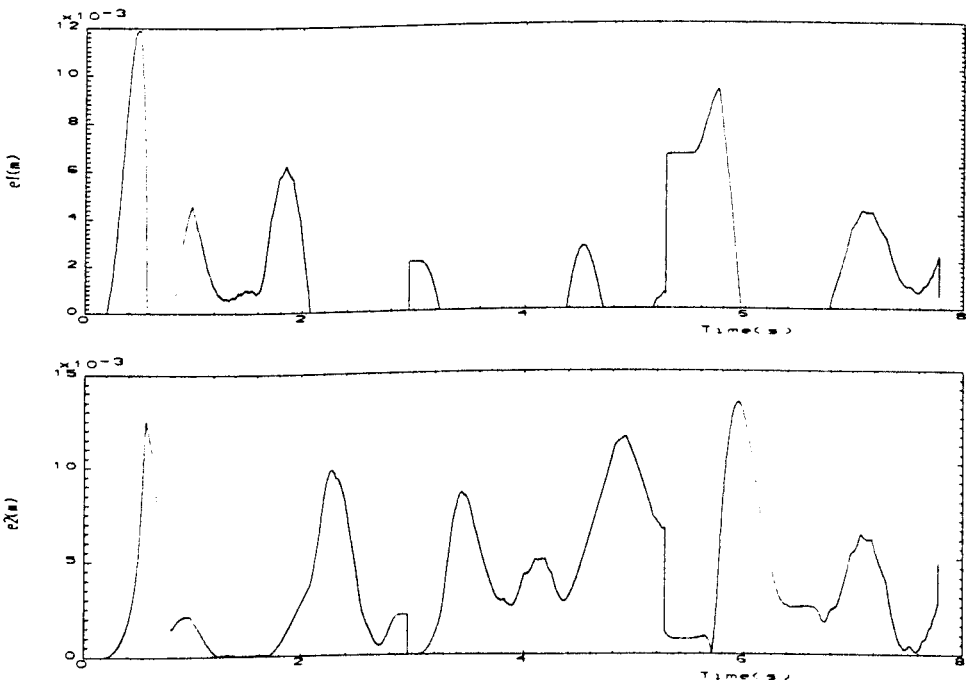

Figure 9.14(d): Time-domain behaviour of errors e1 and e2: unconstrained design, non-adaptive ES (μ = 6.0).

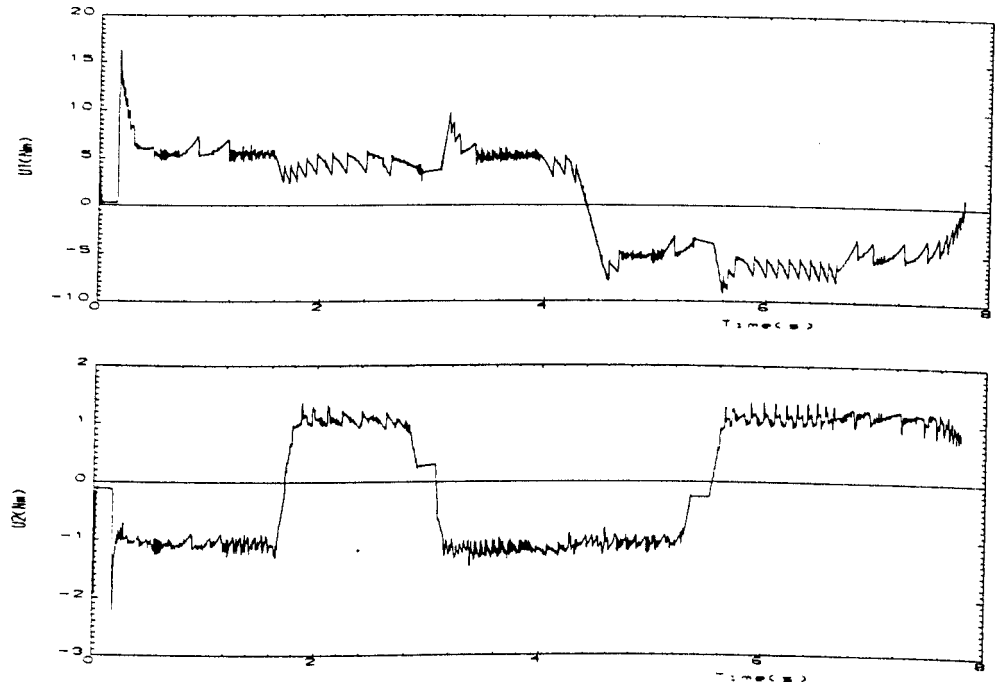

Figure 9.15(a): Time-domain behaviour of torques u1 and u2: unconstrained design, non-adaptive ES (μ = 1.5).

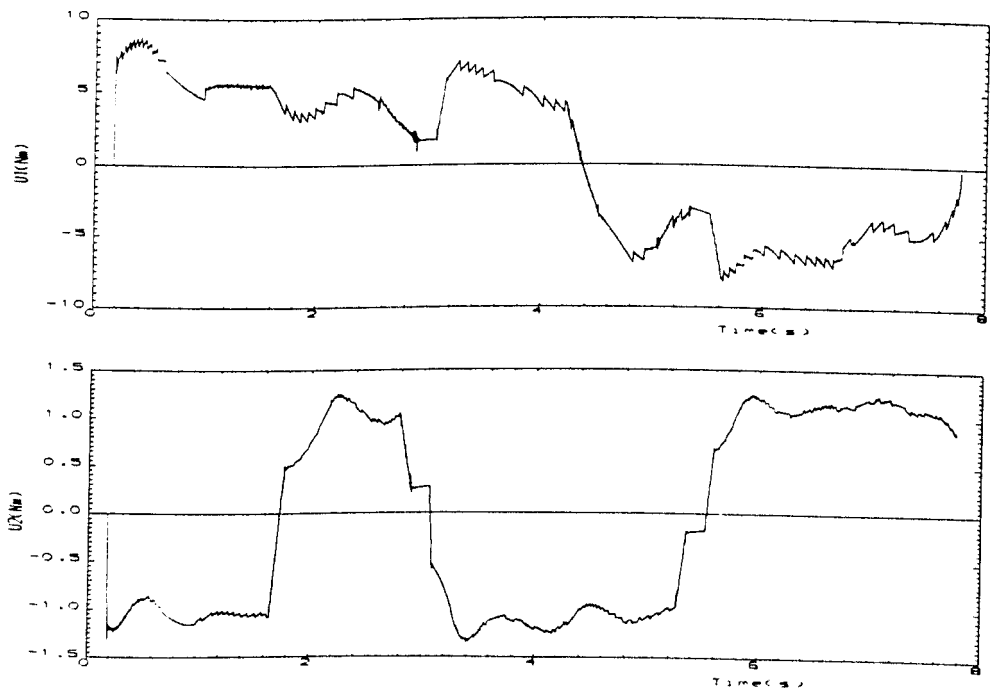

Figure 9.15(b): Time-domain behaviour of torques u1 and u2: unconstrained design, non-adaptive ES (μ = 3.0).

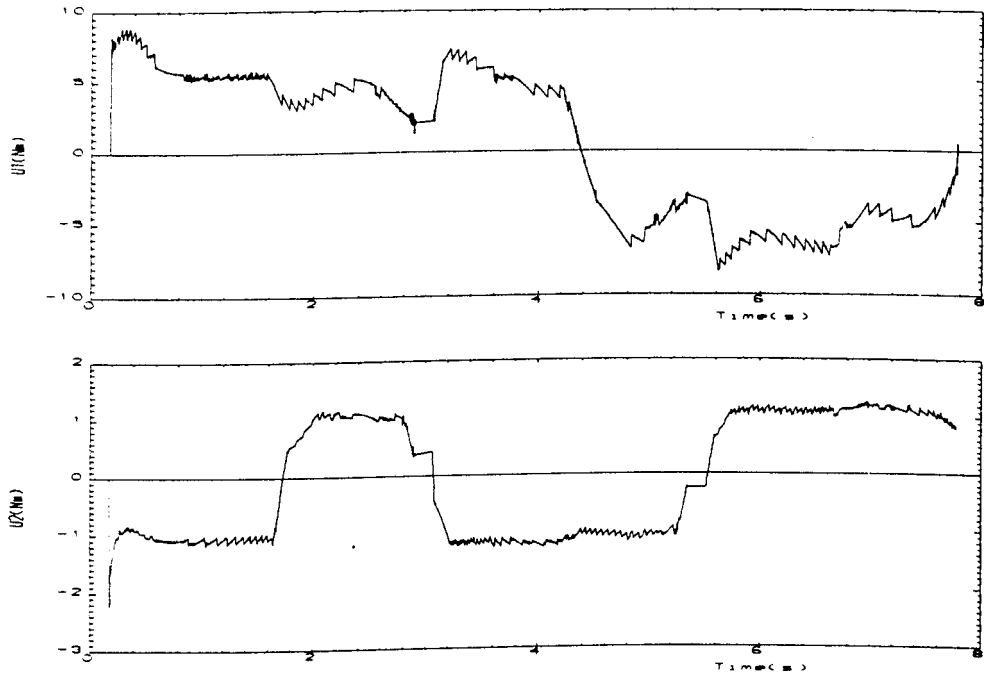

Figure 9.15(c): Time-domain behaviour of torques u1 and u2: unconstrained design, non-adaptive ES (μ = 4.0).

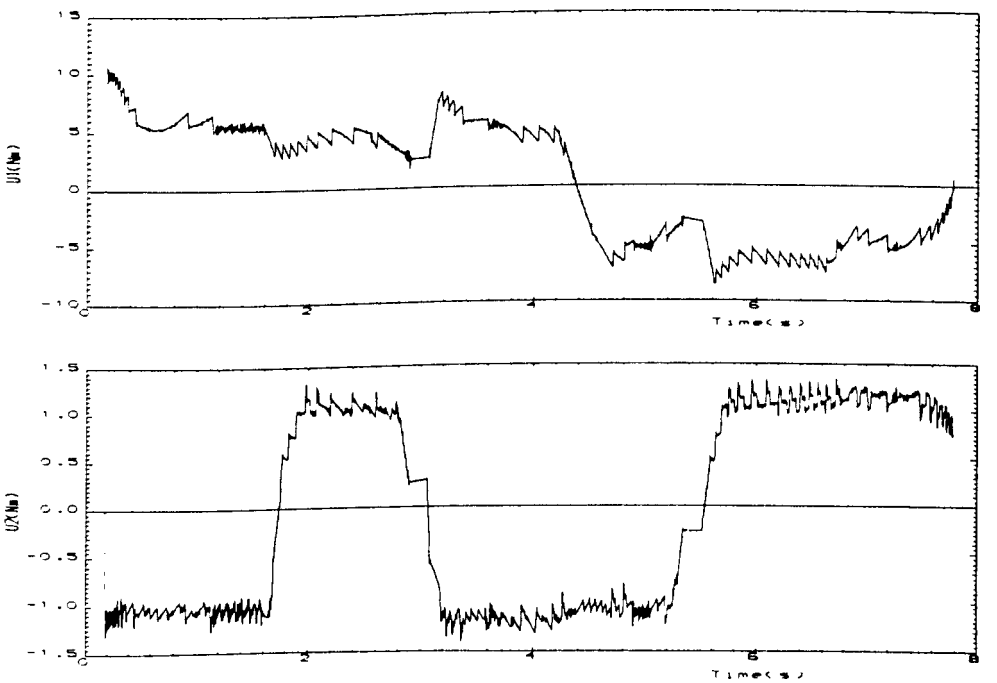

Figure 9.15(d): Time-domain behaviour of torques u1 and u2: unconstrained design, non-adaptive ES (μ = 6.0).

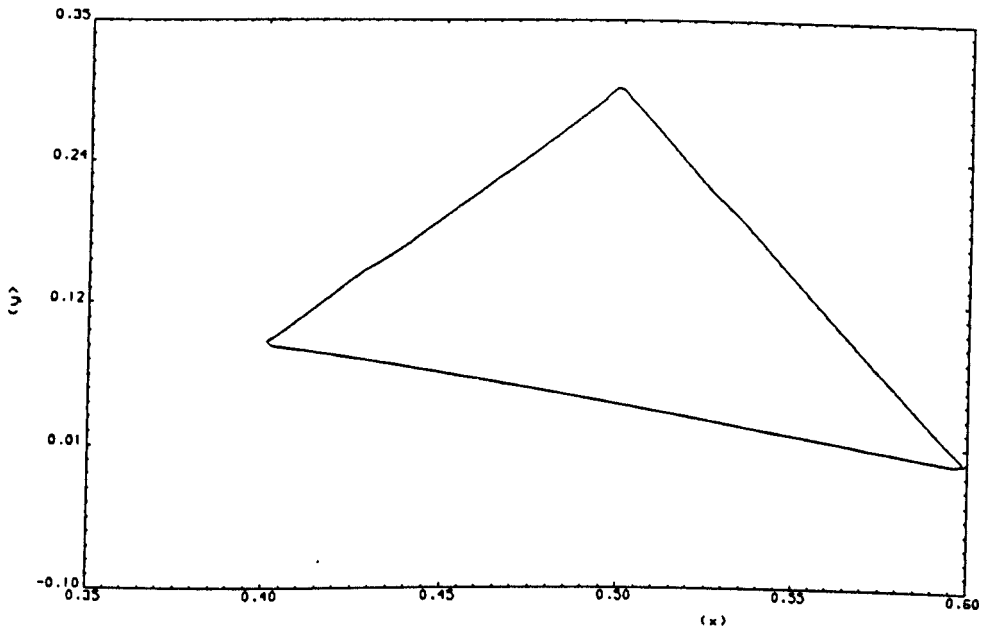

Figure 9.16(a): Trajectory of controlled robot: constrained design, adaptive ES (μ = 1.5).

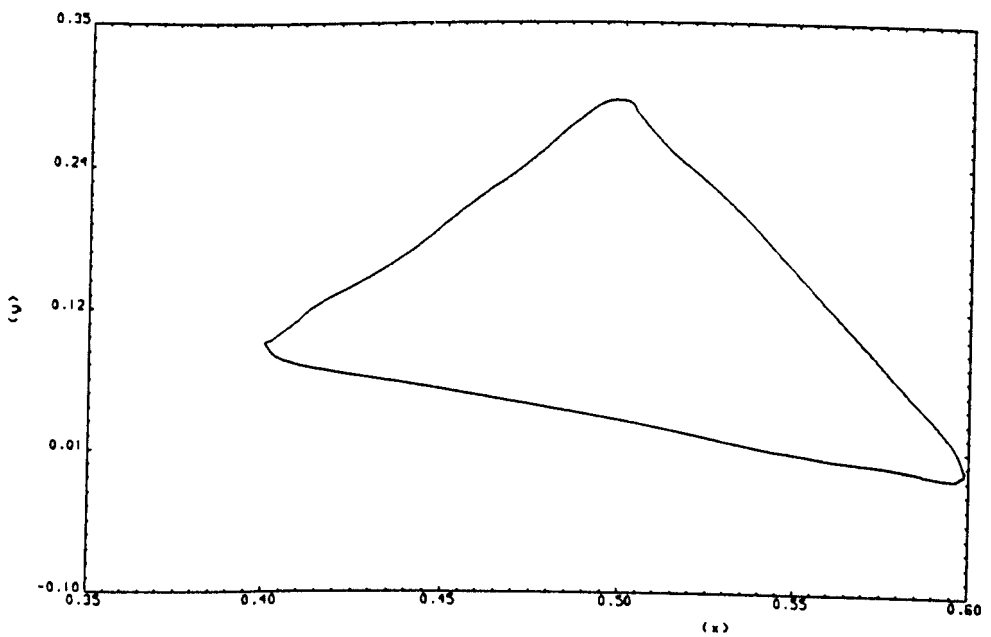

Figure 9.16(b): Trajectory of controlled robot: constrained design, adaptive ES (μ = 3.0).

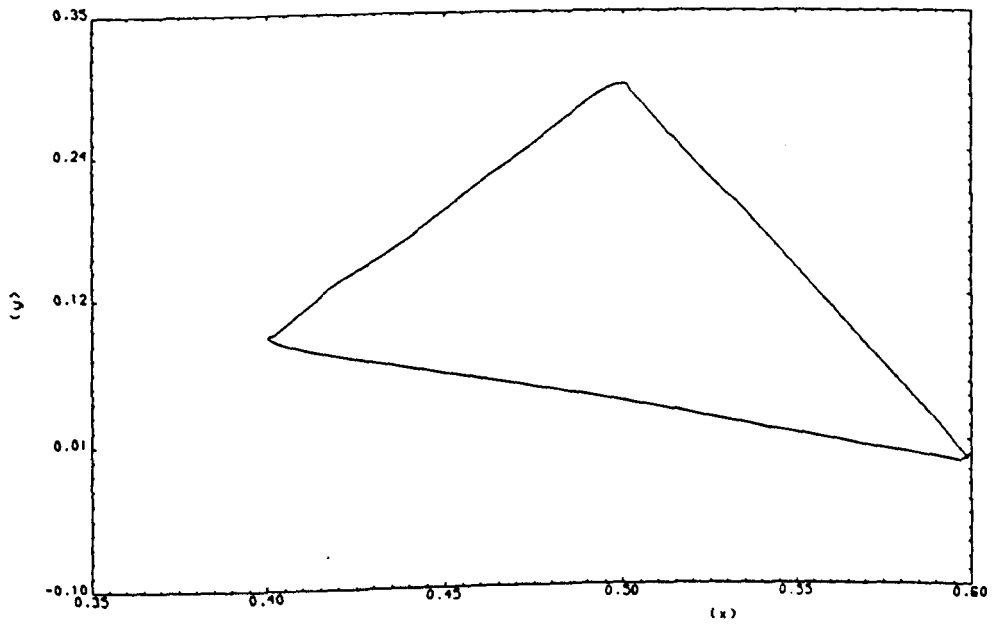

Figure 9.16(c): Trajectory of controlled robot: unconstrained design, adaptive ES (μ = 4.0).

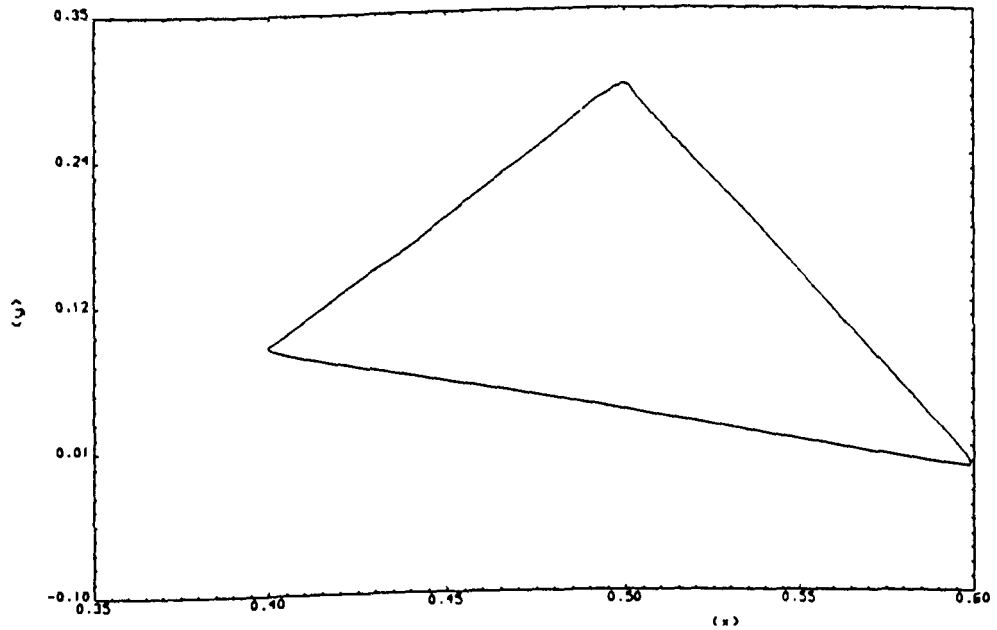

Figure 9.16(d): Trajectory of controlled robot: unconstrained design, adaptive ES (μ = 6.0).

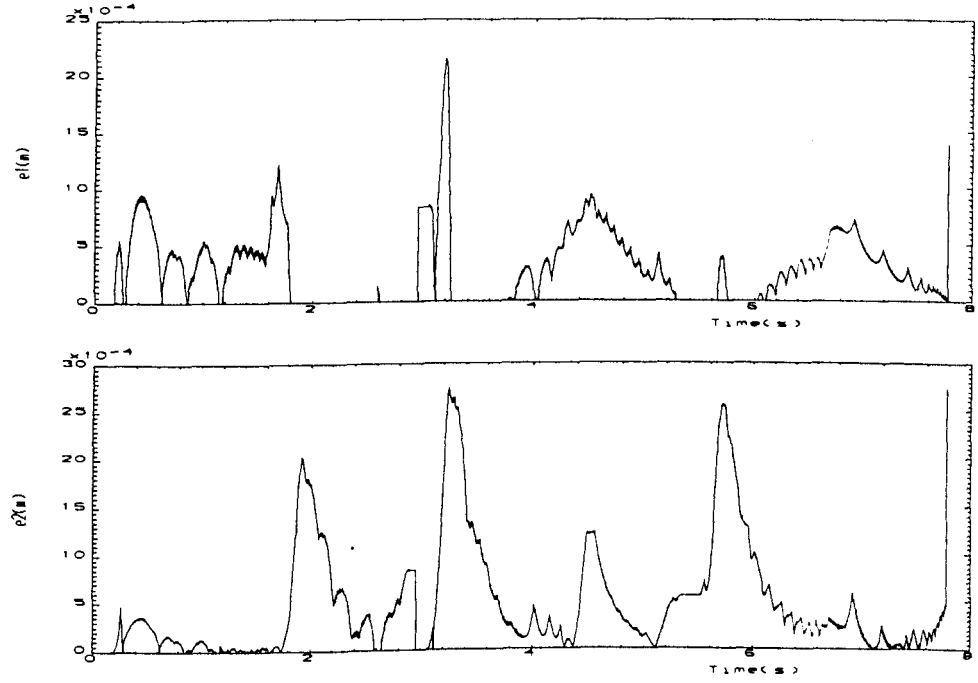

Figure 9.17(a): Time-domain behaviour of errors e1 and e2: constrained design, adaptive ES (μ = 1.5).

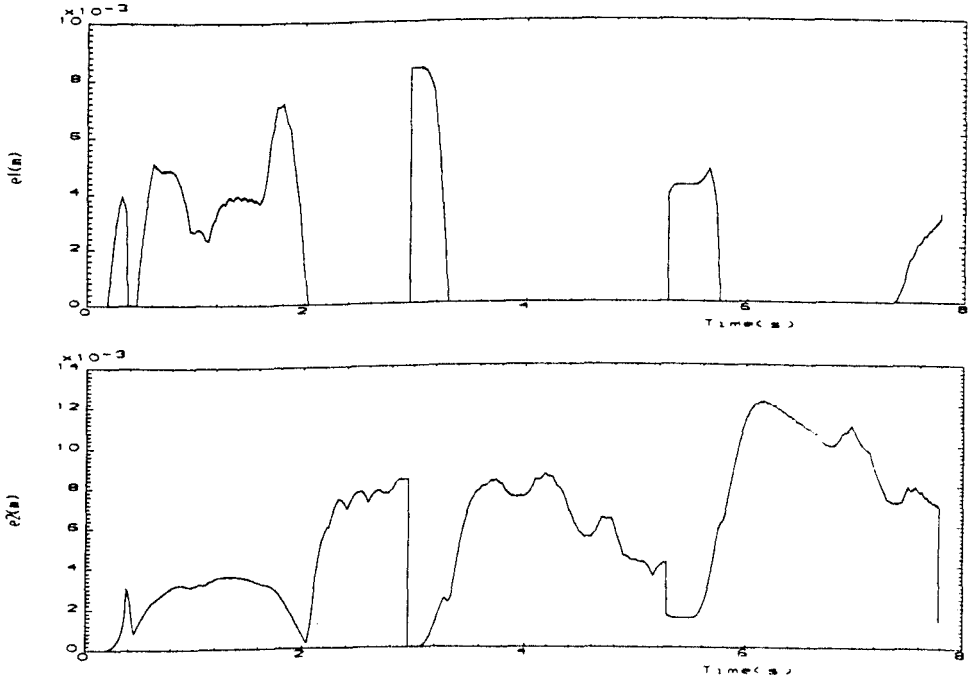

Figure 9.17(b): Time-domain behaviour of errors e1 and e2: constrained design, adaptive ES (μ = 3.0).

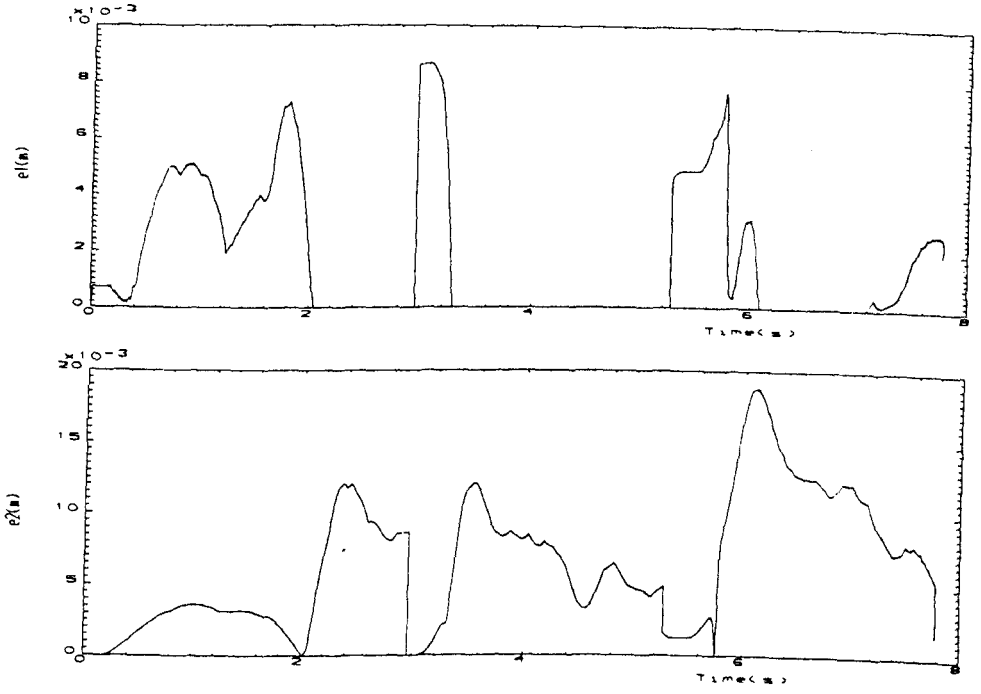

Figure 9.17(c): Time-domain behaviour of errors e1 and e2: constrained design, adaptive ES (μ = 4.0).

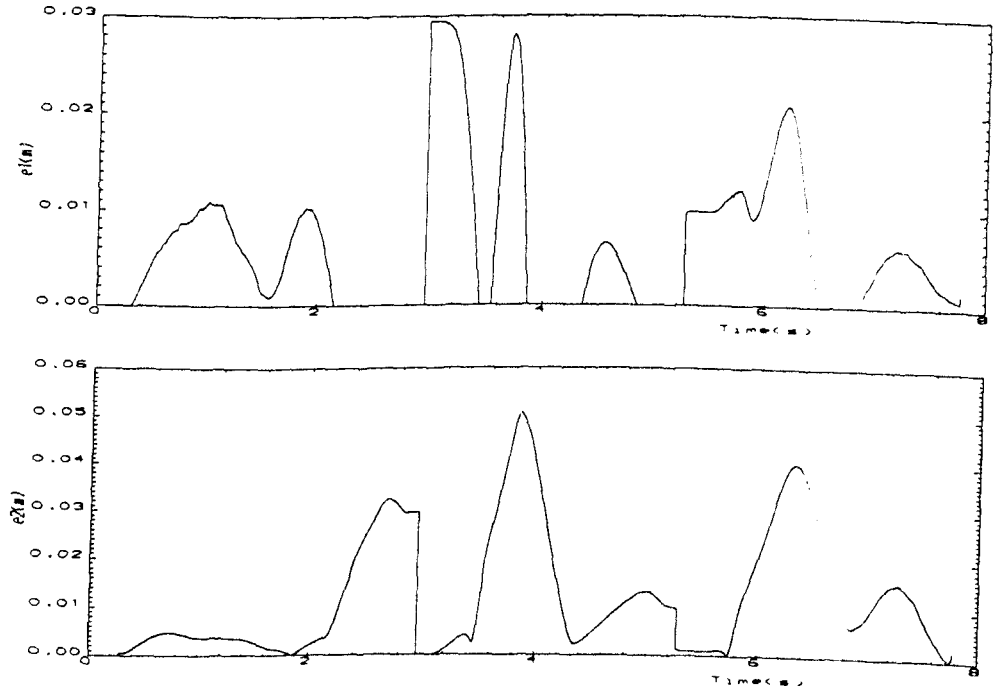

Figure 9.17(d): Time-domain behaviour of errors e1 and e2: constrained design, adaptive ES (μ = 6.0).

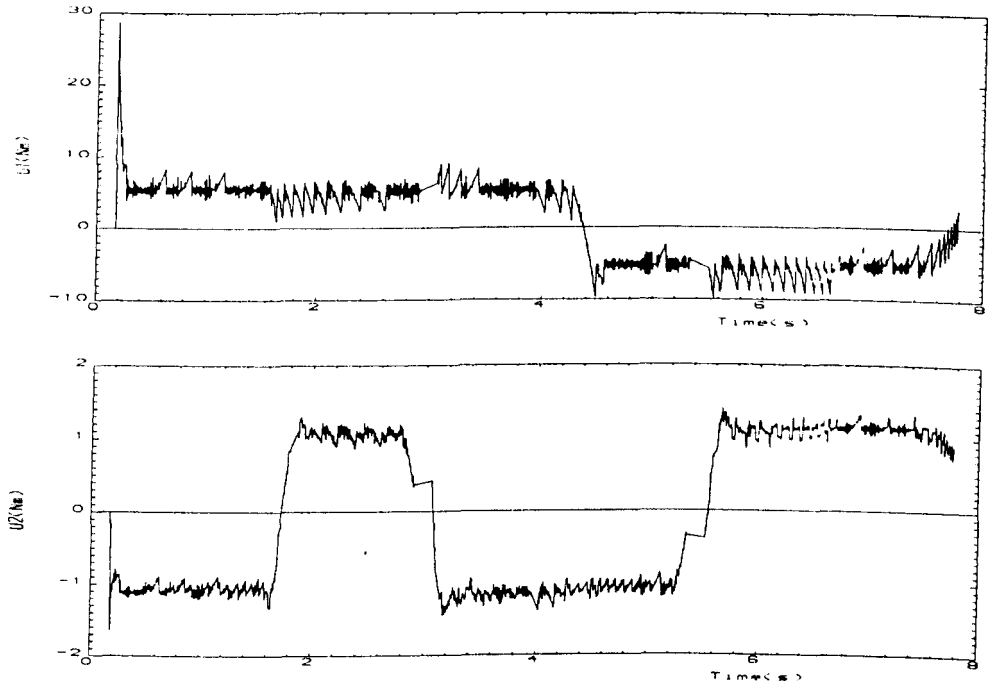

Figure 9.18(a): Time-domain behaviour of torques u1 and u2: constrained design, adaptive ES (μ = 1.5).

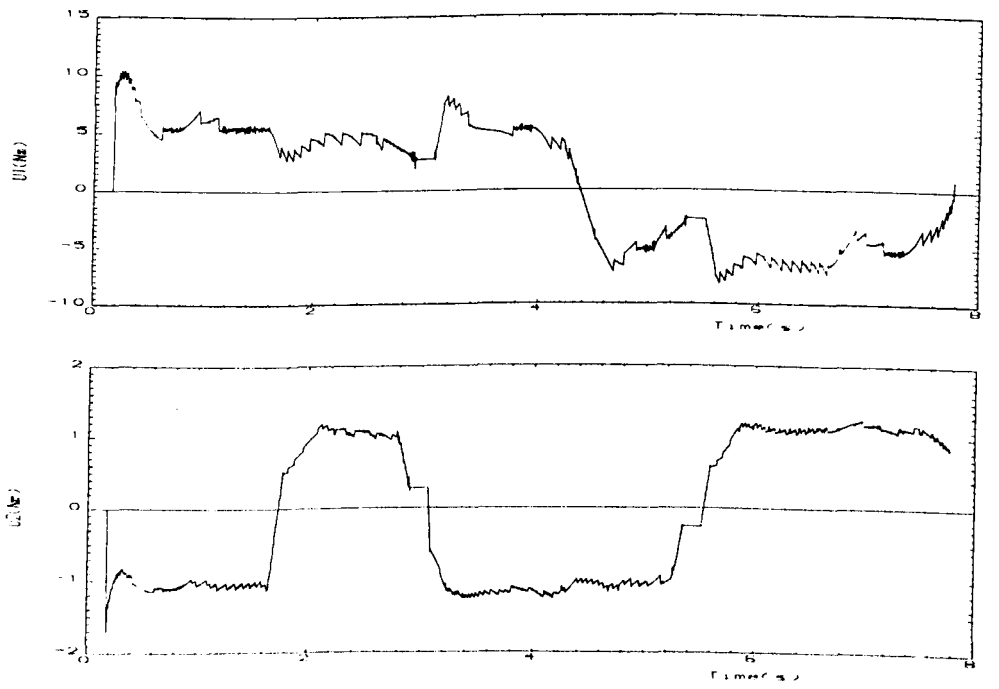

Figure 9.18(b): Time-domain behaviour of torques u1 and u2: constrained design, adaptive ES (μ = 3.0).

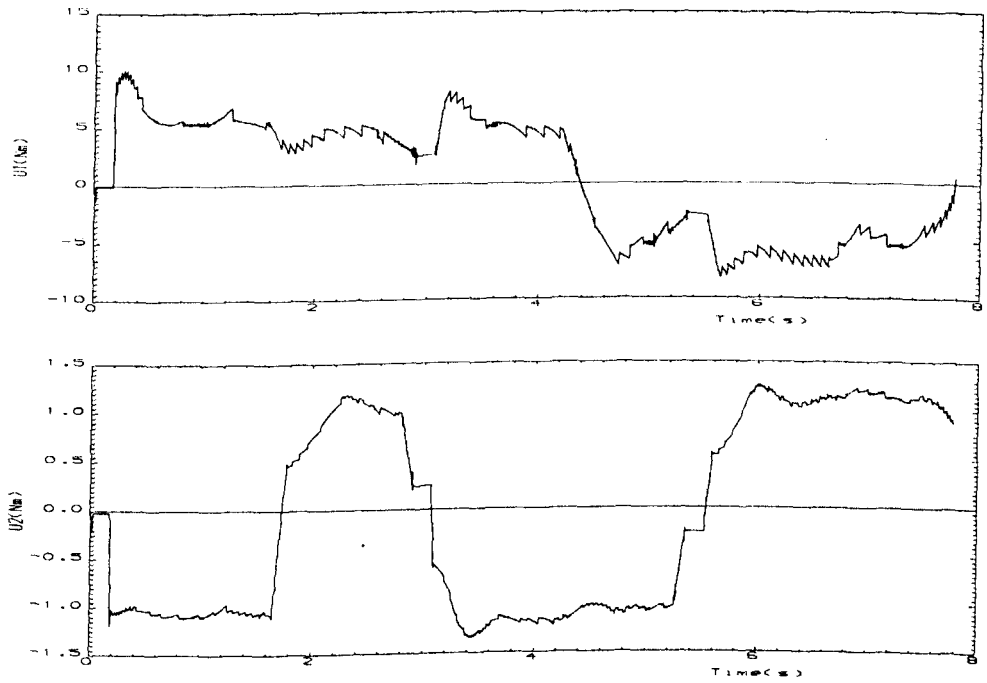

Figure 9.18(c): Time-domain behaviour of torques u1 and u2: constrained design, adaptive ES (μ = 4.0).

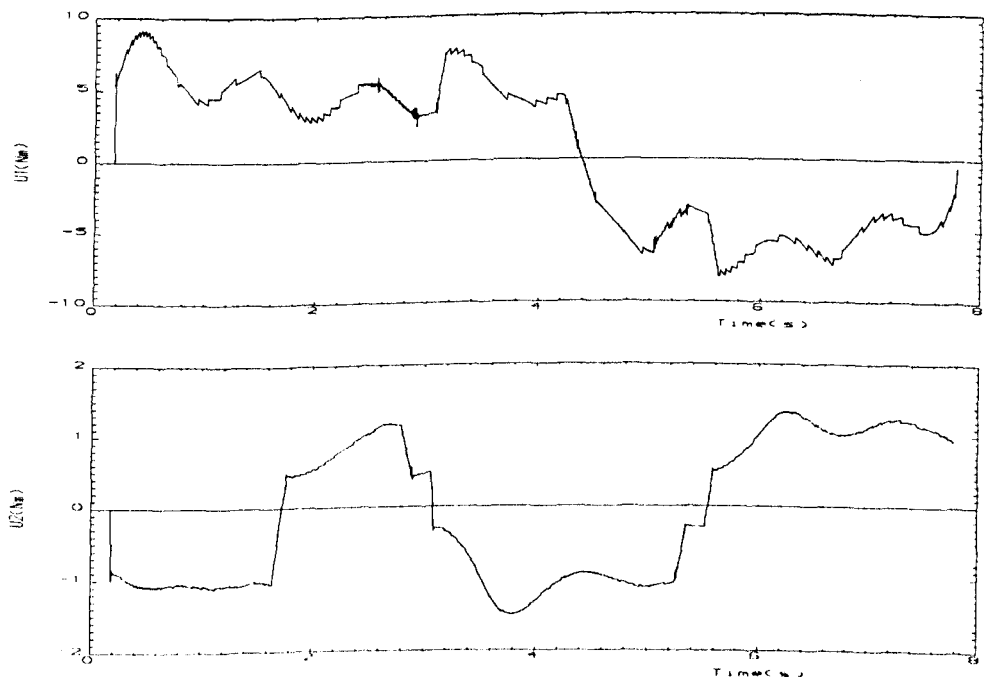

Figure 9.18(d): Time-domain behaviour of torques u1 and u2: constrained design, adaptive ES (μ = 6.0).

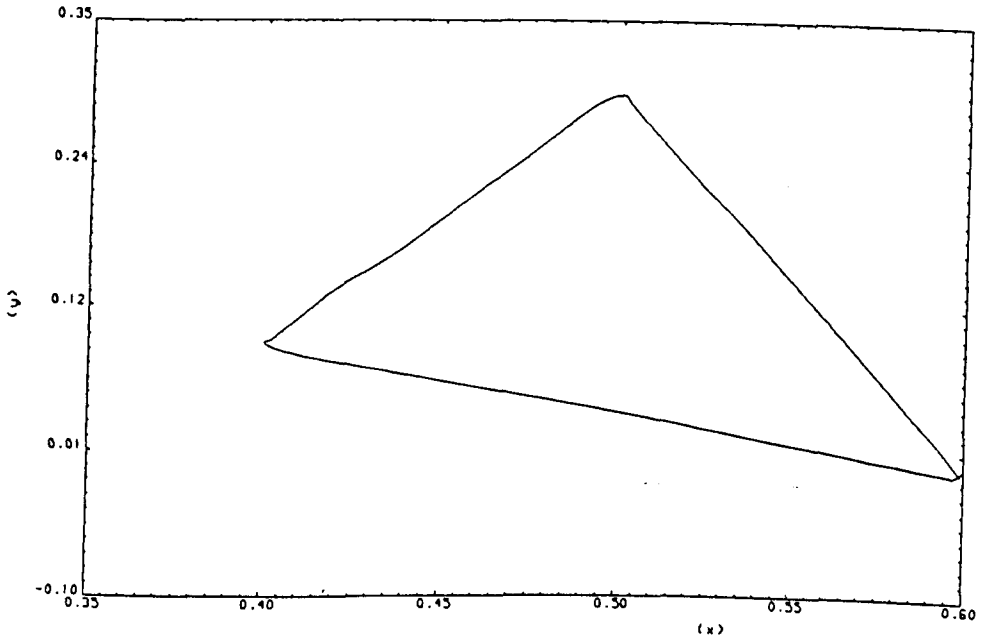

Figure 9.19(c): Trajectory of controlled robot:
unconstrained design, adaptive ES (μ = 4.0).

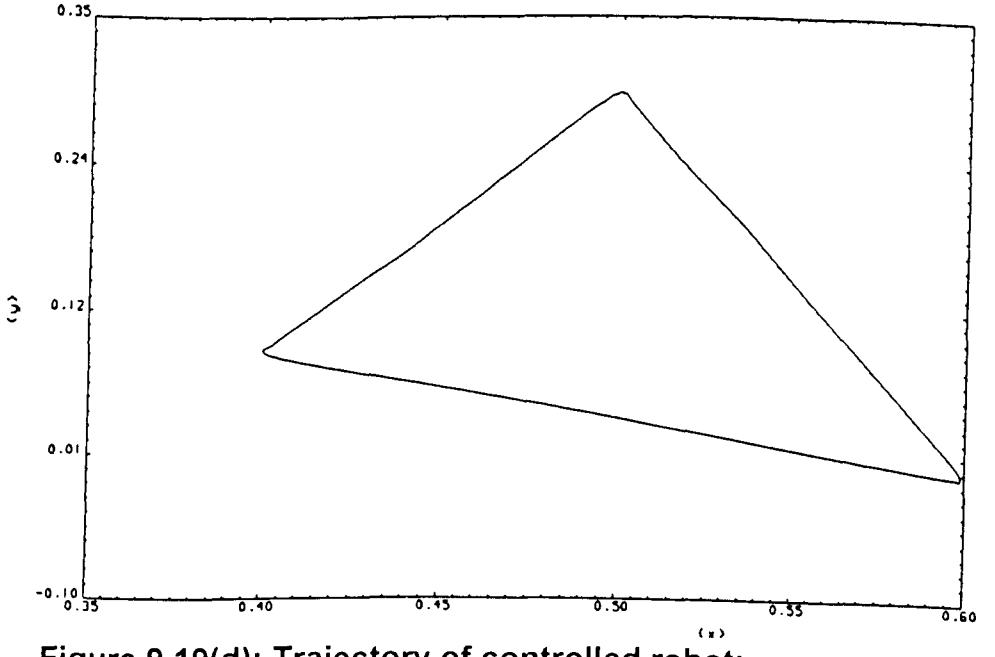

Figure 9.19(d): Trajectory of controlled robot:
unconstrained design, adaptive ES (μ = 6.0).

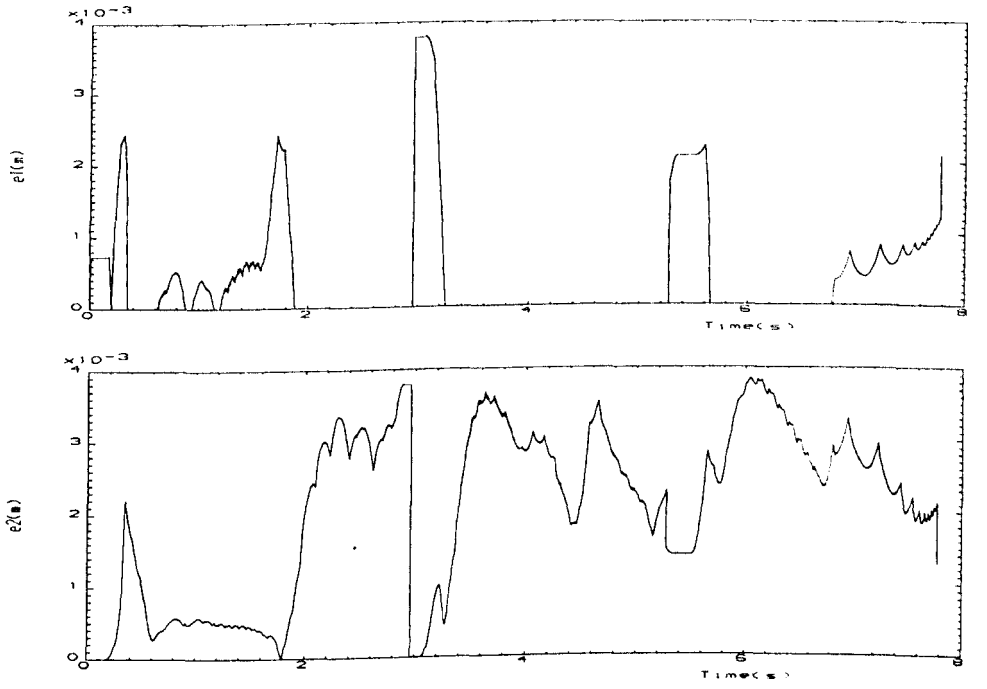

Figure 9.20(a): Time-domain behaviour of errors e1 and e2: unconstrained design, adaptive ES (μ = 1.5).

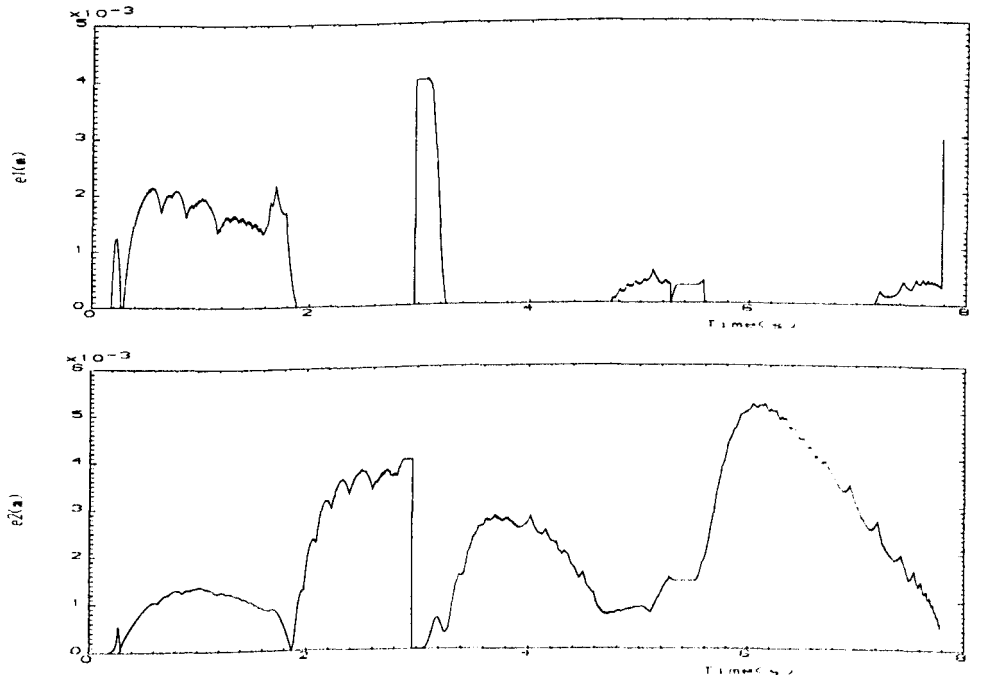

Figure 9.20(b): Time-domain behaviour of errors e1 and e2: unconstrained design, adaptive (μ = 3.0).

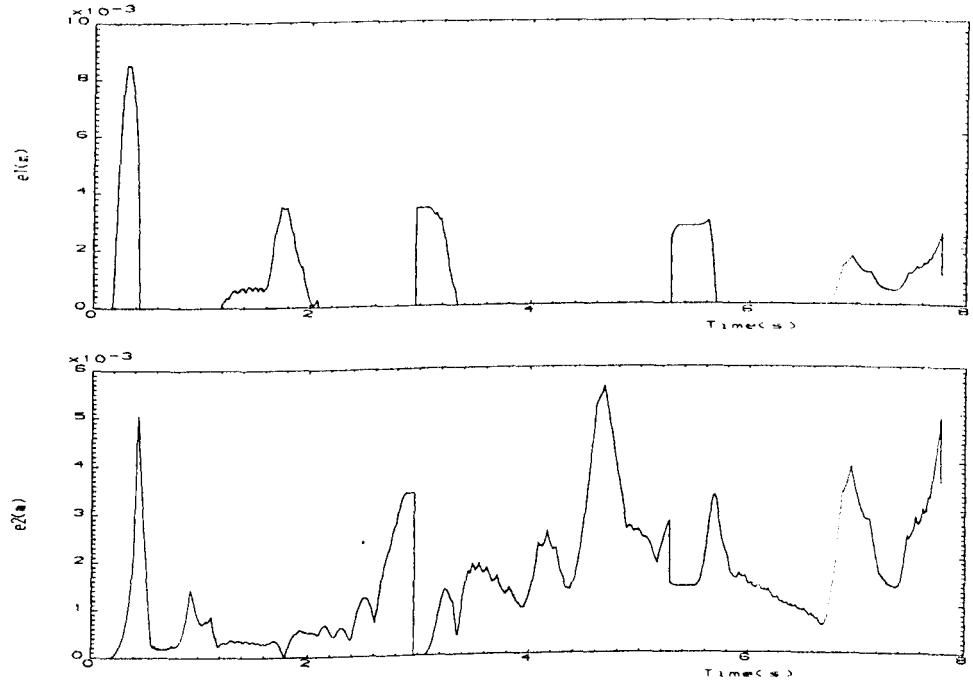

Figure 9.20(c): Time-domain behaviour of errors e1 and e2: unconstrained design, adaptive ES (μ = 4.0).

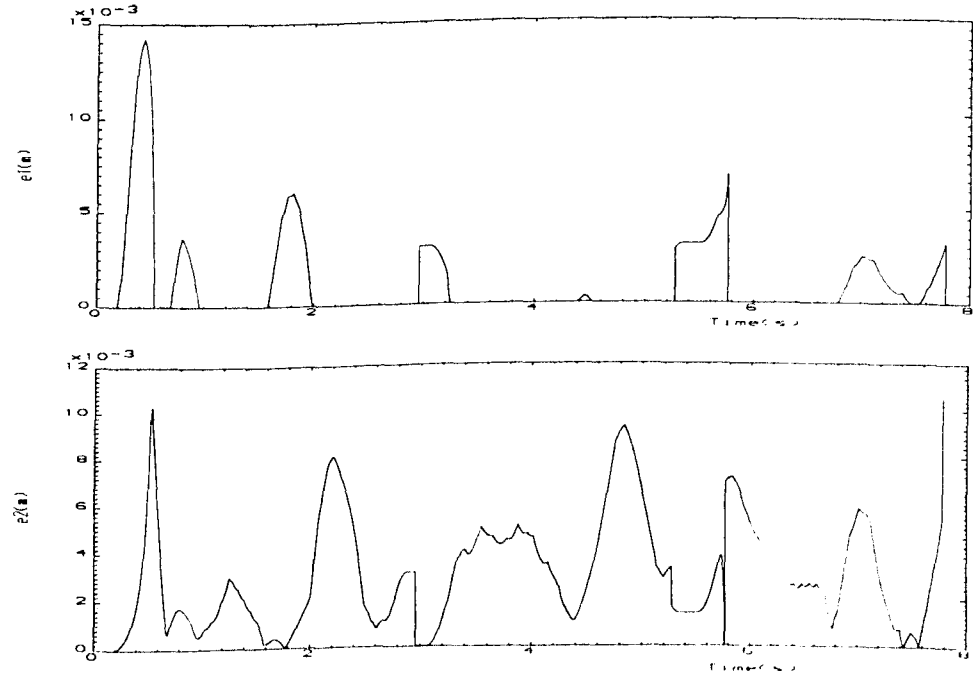

Figure 9.20(d): Time-domain behaviour of errors e1 and e2: unconstrained design, adaptive ES (μ = 6.0).

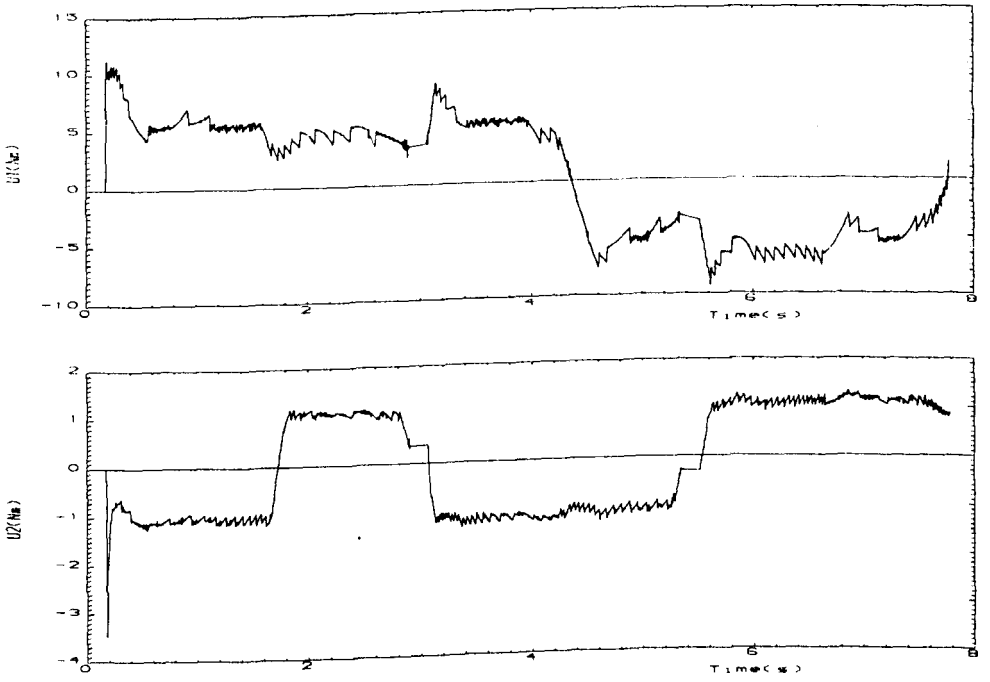

Figure 9.21(a): Time-domain behaviour of torques u1 and u2: unconstrained design, adaptive ES (μ = 1.5).

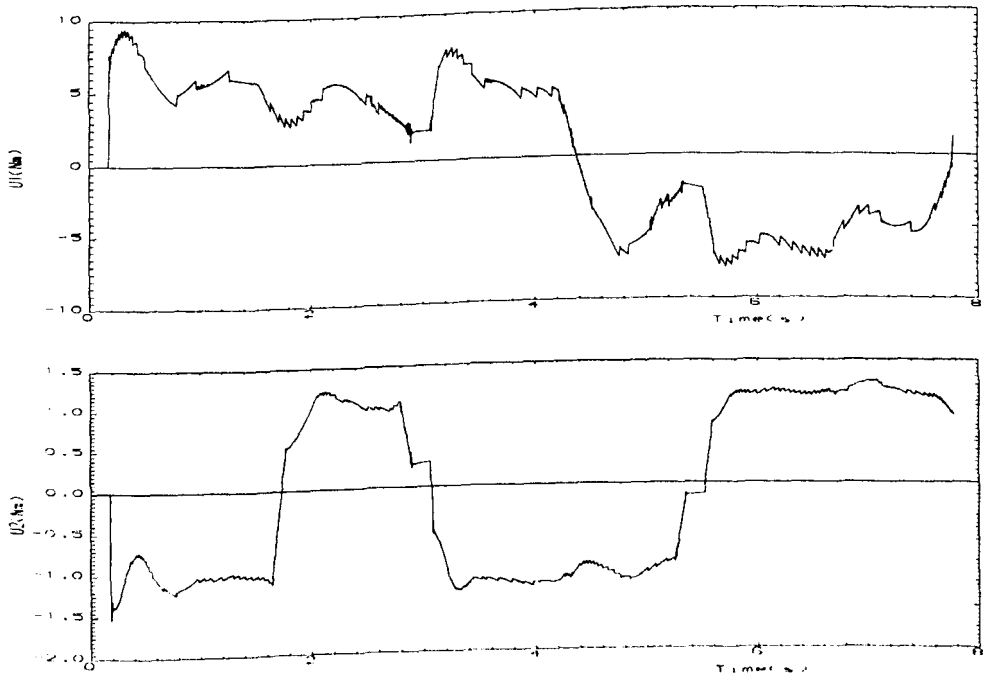

Figure 9.21(b): Time-domain behaviour of torques u1 and u2: unconstrained design, adaptive ES (μ = 3.0).

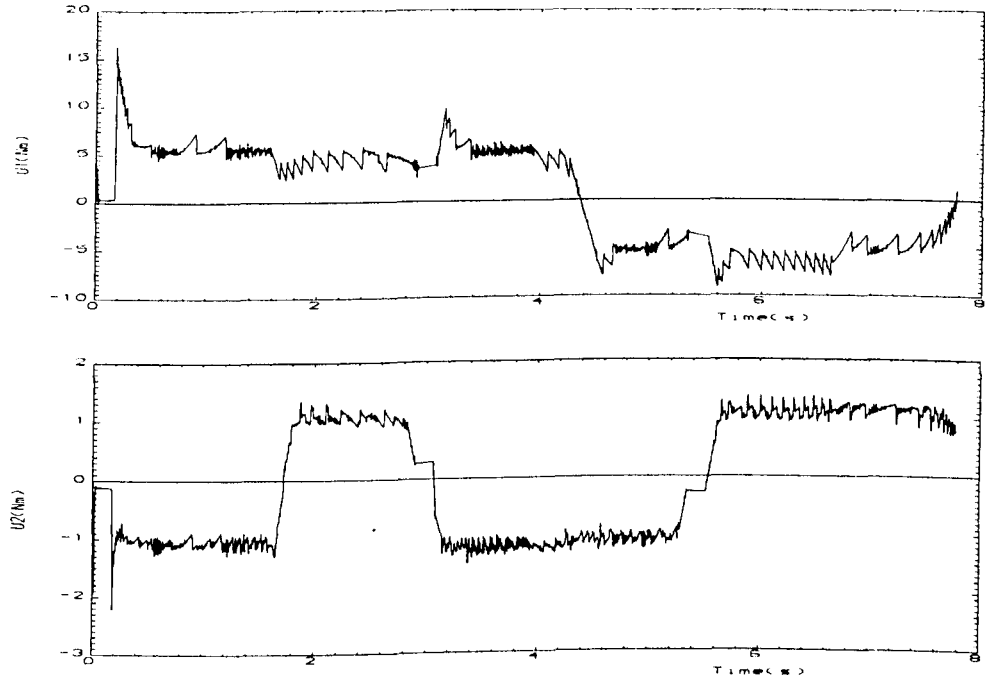

Figure 9.21(c): Time-domain behaviour of torques u1 and u2: unconstrained design, adaptive ES (μ = 4.0).

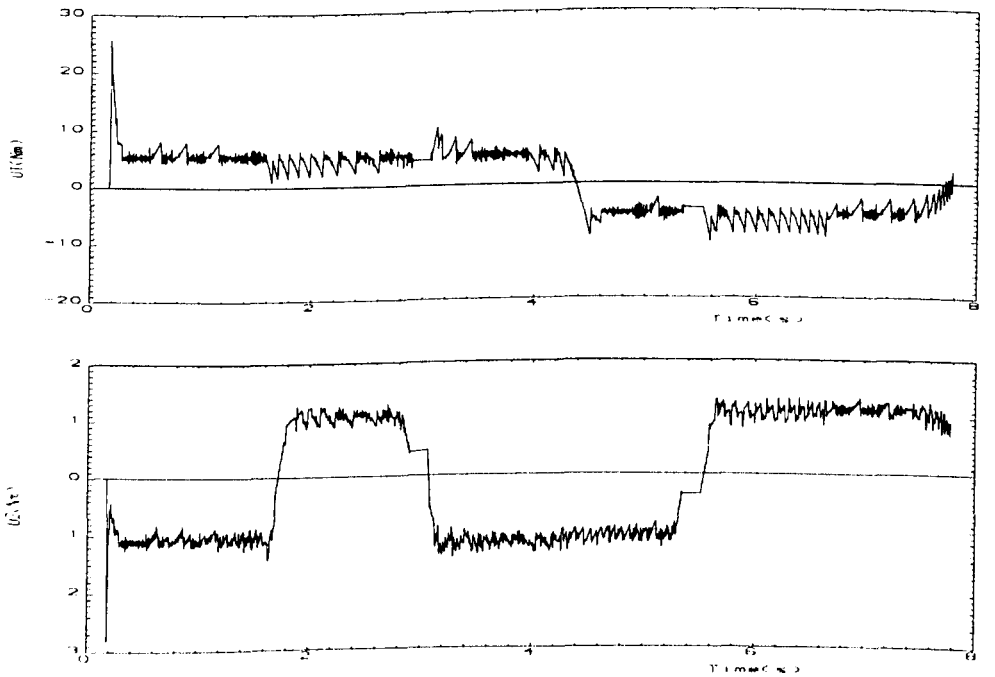

Figure 9.21(d): Time-domain behaviour of torques u1 and u2: unconstrained design, adaptive ES (μ = 6.0).

PART VI

CONCLUSIONS AND RECOMMENDATIONS

Chapter 10

CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK

10.1 CONCLUSIONS

In the modern technological era, increasingly stringent performance criteria are being used in the design of precision-pointing devices such as robotic manipulators. Thus, the design of high-accuracy digital trajectory-tracking controllers for robotic manipulators, which are highly non-linear with time-varying parameters, has posed a serious challenge. Furthermore, the high-accuracy performance demands of such controllers must be achieved even in extremely difficult conditions such as in the presence of unpredictable payload variations. These tough requirements can all be met to some extent by application of the previously developed fast-sampling digital PID controllers for robotic manipulators. Indeed, the use of such a design methodology circumvents the need for detailed mathematical models. Moreover, this methodology relies only on input/output data in the form of step-response matrices in the synthesis of its control laws. Indeed, for such controllers it is possible to prove a series of very reassuring robustness results using only the Markov parameters associated with locally linearised representations of robotic manipulators. However, these theoretical results for digital PID controllers are valid only asymptotically as sampling periods become vanishingly small. In practice, of course, the sampling periods of digital controllers remain non-zero,

in which case no theoretical optimisation results are currently available.

This research has therefore attempted to provide some alternative optimisation procedure that will facilitate the non-asymptotic design of digital PID controllers for robotic manipulators. Indeed, this design need has been addressed in detail in this thesis. In particular, the following evolutionary optimisation techniques have been used to design digital trajectory-tracking controllers for robotic manipulators:

- genetic algorithms,
- non-adaptive evolution strategies,
- adaptive evolution strategies.

These various evolutionary algorithms have been used to optimise digital multivariable PID controllers for a range of finite sampling frequencies. In particular, such evolutionary algorithms have been used to determine the optimal sets of parameter values for digital multivariable PID controllers in both their constrained and unconstrained configurations. Indeed, this latter configuration of the controller is characterised by its high dimensionality associated with the fact that all the elements of the controller matrices are used as design parameters.

It has been shown initially that genetic algorithms can be readily used to tune digital multivariable PID controllers for robotic manipulators performing typical

trajectory-tracking tasks. However, in the case of very complex robotic manipulators performing wide ranges of trajectory tracking tasks, there was a practical need to accelerate the evolutionary process involved in the genetic design methodology. This need prompts the search for evolutionary algorithms that are more powerful than genetic algorithms. In this thesis, the use of both non-adaptive and adaptive evolution strategies without recombination has therefore been investigated. Indeed, it has been shown that highly accurate tracking can be achieved using these evolutionary algorithms in the design of digital PID controllers for robotic manipulators. Furthermore, it has been shown that simple manipulation of the associated performance measure can introduce realistic practical constraints of the type present in typical applications of robotic manipulators.

The effectiveness of the evolutionary design methodologies in providing high-accuracy tracking performance, and in ensuring a high degree of robustness in the face of plant-parameter variations, has been demonstrated through a series of comprehensive simulation studies for a typical robotic manipulator. Indeed, it has been shown that the evolutionary design procedure can be used to optimise high-accuracy digital multivariable PID controllers for a typical direct-drive three-link robotic manipulator with varying plant parameters, unpredicted mass payload variations, and non-linear dynamics for a range of finite sampling frequencies. Furthermore, a special design and an associated implementation study have been presented in order to demonstrate the practical applicability of these evolutionary design techniques for a typical two-link direct-drive robotic

manipulator. In addition, the performance of all three evolutionary algorithms has been compared in various case studies. Hence, it can be concluded that the proposed evolutionary design approach results in a powerful and effective yet relatively simple controller tuning technique capable of alleviating the problems currently encountered in the design of digital PID controllers for robotic manipulators under non-asymptotic conditions.

10.2 RECOMMENDATIONS FOR FURTHER WORK

All three evolutionary design techniques presented and used in this thesis are derived from the wider evolutionary computational methodology which also encompasses evolutionary programming together with genetic algorithms and evolution strategies. Therefore, it would be very instructive to consider the use of hybrid approaches involving a combination of neural networks and evolutionary algorithms or of fuzzy-logic and evolutionary algorithms. Similarly, a combination of traditional optimisation theory and evolutionary algorithms deserve to be investigated in the robotics case, as this was previously done for a non-robotic application by *Dhingra* and *Lee* (1995). In addition, only the off-line approach was used in this thesis. Indeed, with the off-line approach there is nothing evolutionary about the control process itself, since this approach uses the evolutionary algorithm to design a controller, which is then used to control the system. It would be therefore instructive to adopt the on-line approach in which the evolutionary algorithm is an active part of the control process. Indeed,

the adaptive characteristic of evolutionary algorithms qualifies them as suitable candidates to build on-line controllers for dynamic control systems.

These various techniques all belong to the family of intelligent controller design techniques. Indeed, these techniques are being introduced as alternatives to their established traditional counterparts because these latter sometimes perform poorly in modern complex systems. Such, modern complex systems are characterised by poorly defined models, high dimensionality of the design space, high noise levels, multiple performance criteria, and stringent performance requirements.

Finally, a better understanding of natural organic evolution should result in the improvement of all three evolutionary algorithms, which at this early stage are considered to be rather primitive in comparison with the evolutionary mechanisms present in such organic evolution.

APPENDICES

APPENDIX A

INTRODUCTION TO EVOLUTIONARY COMPUTATION TECHNIQUES

A.1 GENETIC ALGORITHMS

Historically, the underlying principles of GAs were first developed by*Holland* (1975) and have been used in many areas for search and optimisation. One of the most useful and excellent references on GAs, together with applications, is the recent book by *Goldberg* (1989). The important themes of GAs which have been emphasized in [*Krishnakumar* and *Goldberg* (1992)] are their globality and robustness over a broad spectrum of problems. However, it has been mentioned in [*Beasley et al* (1993)] that GAs are not guaranteed to find the global optimum, but are nevertheless generally good at finding acceptably good solutions acceptably quickly.

It is evident that GAs as an instance of evolution computation techniques are different from other conventional optimisation and search methods in the following principal respects:

1) GAs work with a population of points not a single point. This reduces the chance of getting stuck on a false optimum.

2) GAs work with a coding of the parameter set, rather than with the parameters themselves. The coding is often binary and, intuitively, it is better to have few possible options for many bits than have many options for few bits.

3) GAs require only values of the objective function values, rather than values of its derivative.This minimal requirement broadens the applications GAs.

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

A.2 EVOLUTION STRATEGIES

As instance of evolution computation, *Evolution Strategies* (ES) were introduced by *Renchenberg* (1973) in the 60ies, and further developed by *Schewfel* (1975a). ES were applied first to experimental optimisation problemswith more or less continuously changeable parameters only. The first numerical application were performed by *Hartmann* (1974) and *Hofler* (1976), and a first attempt towards extending this strategy in order to solve discrete problem or even binary parameter optimisation problems was made by *Schewfel* (1975b). The first version of ES — in 1964—, later called the (1 + 1) ES, was born with discrete, biniomally distributed mutation centered at the ancestor's position, and just one descendant per generation.A first multimembered ES, (μ + 1) ES was later proposed by *Rechenberg* (1973). Much more widespread became the (μ + λ) and (μ, λ) both formulated by *Schewfel* (1975).

The most striking differences exit between GA and ES with respect to the mutation parameter strategy (See Chapter 5 for details), the selection procedure, and the relevance of recombination. Indeed, the mutation parameter strategy can be rendered adaptive, the recombination in GA contains an element which in ES is achieved by mutations only. In addition, only ES operate with a surplus of descendants, the $(\mu + \lambda)$ and (μ, λ) versions with $\lambda > \mu$ childrens from μ parents. This help in handling inequality constraints, the violation which lead to lethal descendants. With proportional selection as well as most other forms like (linear) ranking, all individuals produced during generation N within GA have a chance to have childrem themselves in the next generation N+1. ES, however, discard the $\lambda - \mu$ worst decendants. The remaining μ individuals becom parants on the next generation and do have equal chances to mate and have children.

A.3 CODING AND DECODING

The process of binary coding is that by which each design variable is converted to a finite-length string of 1's or 0's. In the case of multiparameter optimization, all coded variables are concatenated and thus form a long string which is often referred to as a chromosome. Note that, by already knowing the length of each string, it is possible to extract each coded parameter and decode it to obtain its denary value. The first population is generated by the random allocation of 1 or 0 to each bit. If the number of bits in each coded variable is L, it is then obvious that the primary decoded parameter value will be in the interval $\boldsymbol{p} \in [\mathbf{0}, \mathbf{2}^{L} - \mathbf{1}]$. In

order to recognize the fact that the parameter belongs to a specified interval, $p \in [p_{max}, p_{min}]$, the primary decoded integer must be mapped linearly to this interval. Thus, for example, in the case of multiparameter coding, a concatenated string containing five parameters each with a length of L=20 bits can be written as follows:

0 0 1 ...0 1|0 1 0 ... 0 1|1 1 0 ... 0 0|1 0 0 ... 1 1|0 0 0 ... 1 0|

L=20 L=20 L=20 L=20 L=20

Then, each parameter is decoded as

$$p_i = accum_i * [\frac{p_{i_{max}} - p_{i_{min}}}{2^L - 1}] + p_{i_{min}} \quad (i=1,2,...,5) \qquad (A.1)$$

where "$\mathbf{accum}_i$ " is the computed integer value associated with each coded parameter. In addition, p_{min} and p_{max} correspond to their equivalent binary coding as follows:

0 0 0 ... 0 0 p_{min}

L=20

1 1 1 ... 1 1 p_{max}

The fitness function has been defined in [Beasley *et al* (1993)] so that it returns a single numerical "fitness" or "figure of merit" which is supposed to be proportional to the "utility" or "ability" of the individual that a chromosome represents. In other words, a fitness is assigned to each individual in the population where large numbers mean good fitness. The positive fitness function

can be nonlinear, non-differentiable, or discontinuous, because the algorithm needs only the fitness assigned to each string.

A.4 FITNESS FUNCTIONS

In many real problems, expressing a fitness function in terms of a cost function is not difficult yet it is perhaps the most crucial aspect of applying GAs [*Deb* (1991)]. Minimal integral square of error (ISE) or integral absolute of error (IAE) may be used as the cost function in control problems [*Porter* and *Borairi* (1992)] [*Porter* and *Jones* (1992)]. Some practical examples of the minimization of cost functions are presented in [*Goldberg* (1989)],[*Krishnakumar* and *Goldberg* (1992)] and [*Deb* (1991)]. In many problems, it is necessary to minimize a cost function rather than to maximize it. However, the genetic selection procedure is based on maximization, so that the cost function must be mapped to fitness form. In this connection, one simple commonly used transformation is

$$Fitness=1/g(x) \tag{A.2}$$

There are other forms of mapping procedure, but this form has given the best results in the present research. There are also several possible cost functions which are selected according to the design requirements; but the one which has been used for this robotic application either on its own or as apart of a composite cost function which encompasses several design requirements is

$$\Gamma = \int_0^{\tau} \|e(t)\| \; dt \tag{A.3}$$

is minimised, where τ is the duration of the tracking task, $e(t)$ is the trajectory-tracking error vector in Cartesian space, and $\|\cdot\|$ denotes the Euclidean norm.

A.4 EVOLUTIONARY OPERATORS

The mechanics of simple genetic algorithms are based upon the following three operators:

1) Selection,

2) Crossover,

3) Mutation.

The selection operator allows fit strings to be selected from the population and mated. In GAs, the selection is performed randomly based on fitness (i.e., measure of profit or goodness) such that those with high values of the fitness will get more probability of living and producing new offspring by increasing the chance of being chosen by the selection operator. The probability of a string being selected is

$$P_i(select) = \frac{F_i}{\sum_{i=1}^{n} F_i} \;, \tag{A.4}$$

where i is the string index, F_i is the fitness, and n is the number of strings in the population. This strategy, in which good strings get most copies in the next generation, emphasizes the survival-of-the-fittest aspect of genetic algorithms. It is obvious that good individuals will probably be selected several times. The easiest way to implement this selection has been described in *Goldberg* (1989) and is called "roulette wheel" selection.

In the ES case, unlike that of genetic algorithms where the elitist approach affected only the best and worst of each generation, the greater degree of elitism has the potential to affect the entire generation. Indeed, in the case of $(\mu + \lambda)$-evolution strategies, elitism consists in selecting the μ best individuals from the entire set of $\mu + \lambda$ individuals (parents and offspring).

In EA the creation of new individuals is performed by the crossover operator. Two new offspring, representing new candidate solutions, are built by recombining the information from two parents. This crossover operator works on the parents with a certain probability rate, called the crossover rate P_c. The higher the crossover rate, the more quickly new structures are introduced into the population. The crossover operator takes two individuals and cuts their strings at some random position, so that each string will have two "head"and"tail"segments.

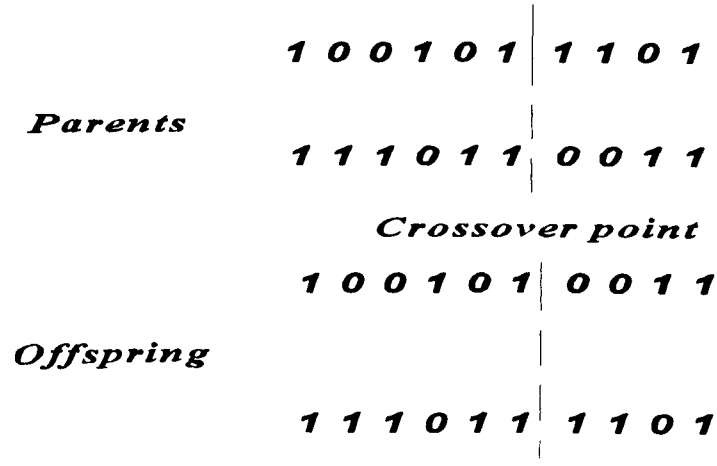

Figure. A.1: Crossover Operation.

The tail segments are swapped and another two new strings are thus produced, as shown in Figure A.1. In the case of a multiparameter search where all parameters are concatenated to form a single string, a multicrossover strategy may be chosen. In this form, each parameter gets its own crossover with the same crossover probability.

Although most EA use mutation along with crossover, mutation is sometimes treated as if it were a background operator for assuring that population of a diverse pool of alleles (individuals) that can be exploited by crossover. For many optimisation problems, however an EA using mutation without crossover can be very effective [*Mathias* and *Whitley* (1994)]. Indeed, throughout this thesis ES have been used without a recombination (crossover) to simplify the computational implementation of the evolutionary algorithm.

The third operator is mutation and simply changes a bit occasionally at a fixed rate in both GA and non-adaptive ES or according to an adaptive mechanism in the adaptive variant of ES (see Chapter 5 for details): it thus occasionally alters each bit from 1 to 0 or vice versa. This produces an amount of random search and assures that no point in the search space has a zero probability of being examined. Mutation increases the variability of populations and prevents the irreversible loss of certain patterns. The mutation operator will thus help to avoid the possibility of mistaking a local optimum for a global optimum. Therefore, it improves the global performance of EA.

APPENDIX B

DYNAMICAL MODEL OF THREE-LINK DIRECT-DRIVE ROBOTIC MANIPULATOR

The three-link direct-drive robotic manipulator which has been used in the simulation studies of Chapter 3, 4, 5, 6, and 8 is shown schematically in Figure. 3.3. All positions of the masses and of the end-effector are measured with respect to a set of orthogonal axes (x, y, z), the origin of which is situated on the second joint of the manipulator arm.

The parameters indicated in this figure are as follows:

I_1 : inertia of link 1

m_1 : mass of link 2

I_2 : inertia of link 2

m_2 : mass of link 3

I_3 : inertia of link 3

M_p : mass of payload

θ_1 : angle of link 1 with respect to the horizontal x-axis in the xy plane

θ_2 : angle of link 2 with respect to the horizontal y-axis in the yz plane

θ_3 : angle of link 3 with respect to the vertical z-axis in the yz plane

r_1 : length of link 2

r_2 : length of link 3

a : distance of CG of link 2 from its axis of rotation

b : distance of CG of link 3 from its axis of rotation

τ_1 : torque applied at joint 1

τ_2 : torque applied at joint 2

τ_3 : torque applied at joint 3

The Cartesian triples specifying the positions of the masses m_1, m_2 and M_p, respectively, are

$$m_1(x,y,z) = \{a \sin \vartheta_2 \cos\theta_1, a \sin\theta_2 \sin\theta_1, a\cos\theta_2\} \quad , \quad \text{(B.1)}$$

$$m_2(x,y,z) = \{(r_1 \sin\theta_2 - b\sin\theta_3)\cos\theta_1, (r_1 \sin\theta_2 - b\sin\theta_3)\sin\theta_1, r_1\cos\theta_2 - b\cos\theta_3\}, \quad \text{(B.2)}$$

and

$$M_p(x,y,z) = \{(r_1 \sin\theta_2 - r_2\sin\theta_3)\cos\theta_1, (r_1\sin\theta_2 - r_2\sin\theta_3)\sin\theta_1, r_1\cos\theta_2 - r_2\cos\theta_3\}. \quad \text{(B.3)}$$

The equations of motion for the manipulator can be derived by using the Euler-Lagrange equations in the form

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} = \tau \quad , \quad \text{(B.4)}$$

where

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \quad , \quad \text{(B.5)}$$

$$\dot{\theta} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \quad , \qquad \text{(B.6)}$$

and

$$\tau = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} \quad . \qquad \text{(B.7)}$$

The kinetic energy of the manipulator in the absence of the payload can then, accordingly, be expressed in the form

$$T = \frac{1}{2} I_1 \dot{\theta}_1^2 + \frac{1}{2} I_2 \dot{\theta}_2^2 + \frac{1}{2} I_3 \dot{\theta}_3^2 + \frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 \qquad \text{(B.8)}$$

where

$$v_1^2 = (-a \sin\theta_2 \sin\theta_1 . \dot{\theta}_1 + a \cos\theta_2 \cos\theta_1 . \dot{\theta}_2)^2 + (a \sin\theta_2 \cos\theta_1 . \dot{\theta}_1 + a \sin\theta_1 \cos\theta_2 . \dot{\theta}_2)^2 + (-a \sin\theta_2 . \dot{\theta}_1^2) \qquad \text{(B.9)}$$

and

$$v_2^2 = \left\{ -(r_1 \sin\theta_2 - b \sin\theta_3) \sin\theta_1 . \dot{\theta}_1 + (r_1 \cos\theta_2 . \dot{\theta}_2 - b \cos\theta_3 . \dot{\theta}_3) \cos\theta_1 \right\}^2 + \left\{ (r_1 \sin\theta_2 - b \sin\theta_3) \cos\theta_1 \dot{\theta}_1 + (r_1 \cos\theta_2 . \dot{\theta}_2 - b \cos\theta_3 . \dot{\theta}_3) \sin\theta_1 \right\}^2 + (-r_1 \sin\theta_2 . \dot{\theta}_2 + b \sin\theta_3 . \dot{\theta}_3)^2 \qquad \text{(B.10)}$$

It follows from equations (B.9) and (B.10) that

$$v_1^2 = a^2 \sin^2\theta_2 . \dot{\theta}_1^2 + a^2 \dot{\theta}_2^2 \qquad \text{(B.11)}$$

and

$$v_2^2 = (r_1 \sin\theta_2 - b\sin\theta_3)^2.\dot{\theta}_1^2 + r_1^2.\dot{\theta}_2^2 + b^2.\dot{\theta}_3^2 - 2r_1 b\cos(\theta_2 - \theta_3).\dot{\theta}_2\dot{\theta}_3 . \qquad \text{(B.12)}$$

Then, substituting from equations (B.11) and (B.12) into equation (B.7) indicates that

$$\begin{aligned} T = {} & \frac{1}{2}I_1\dot{\theta}_1^2 + \frac{1}{2}I_2\dot{\theta}_2^2 + \frac{1}{2}I_3\dot{\theta}_3^2 + \frac{1}{2}m_1\left(a^2\sin^2\theta_2.\dot{\theta}_1^2 + a^2\dot{\theta}_2^2\right) \\ & + \frac{1}{2}m_2\left((r_1\sin\theta_2 - b\sin\theta_3)^2.\dot{\theta}_1^2 + r_1^2.\dot{\theta}_2^2 + b^2.\dot{\theta}_3^2 - 2r_1 b\cos(\theta_2 - \theta_3).\dot{\theta}_2\dot{\theta}_3\right) . \end{aligned} \qquad \text{(B.13)}$$

Furthermore, the potential energy of the manipulator can be expressed as

$$V = m_1 ga\cos\theta_2 + m_2 g(r_1\cos\theta_2 - b\cos\theta_3) \qquad \text{(B.14)}$$

The Lagrangian therefore becomes

$$\begin{aligned} L = T - V = {} & \frac{1}{2}I_1\dot{\theta}_1^2 + \frac{1}{2}I_2\dot{\theta}_2^2 + \frac{1}{2}I_3\dot{\theta}_3^2 + \frac{1}{2}m_1\left(a^2\sin^2\theta_2.\dot{\theta}_1^2 + a^2\dot{\theta}_2^2\right) \\ & + \frac{1}{2}m_2\left((r_1\sin\theta_2 - b\sin\theta_3)^2.\dot{\theta}_1^2 + r_1^2.\dot{\theta}_2^2 + b^2.\dot{\theta}_3^2 - 2r_1 b\cos(\theta_2 - \theta_3).\dot{\theta}_2\dot{\theta}_3\right) \\ & - m_1 ga\cos\theta_2 - m_2 g(r_1\cos\theta_2 - b\cos\theta_3) \end{aligned} \qquad \text{(B.15)}$$

The partial derivatives needed for equation (B.5) are accordingly

$$\frac{\partial L}{\partial\dot{\theta}_1} = I_1\dot{\theta}_1 + m_1 a^2\sin^2\theta_2.\dot{\theta}_1 + m_2(r_1\sin\theta_2 - b\sin\theta_3)^2.\dot{\theta}_1 , \qquad \text{(B.16)}$$

$$\frac{\partial L}{\partial\dot{\theta}_2} = I_2\dot{\theta}_2 + m_1 a^2.\dot{\theta}_2 + m_2 r_1^2.\dot{\theta}_2 - m_2 r_1 b\cos(_1\theta_2 - \theta_3).\dot{\theta}_3 , \qquad \text{(B.17)}$$

$$\frac{\partial L}{\partial \dot{\theta}_3} = I_3\dot{\theta}_3 + m_1 b^2.\dot{\theta}_3 - m_2 r_1 b\cos(_1\theta_2 - \theta_3).\dot{\theta}_2 \quad , \qquad \text{(B.18)}$$

$$\frac{\partial L}{\partial \theta_1} = 0 \quad , \qquad \text{(B.19)}$$

$$\frac{\partial L}{\partial \theta_2} = m_1 a^2 \sin\theta_2 \cos\theta_2.\dot{\theta}_1^2 + m_2(r_1\sin\theta_2 - b\sin\theta_3)r_1\cos\theta_2.\dot{\theta}_1^2 + m_2 r_1 b\sin(\theta_2 - \theta_3).\dot{\theta}_2\dot{\theta}_3 - (m_1 a + m_2 r_1)g\sin\theta_2 \quad , \qquad \text{(B.20)}$$

and

$$\frac{\partial L}{\partial \theta_3} = -m_2(r_1\sin\theta_2 - b\sin\theta_3)b\cos\theta_3.\dot{\theta}_1^2 - m_2 r_1 b\sin(\theta_2 - \theta_3).\dot{\theta}_2\dot{\theta}_3 + m_2 g b\sin\theta_3 \quad . \qquad \text{(B.21)}$$

Finally, substituting from equations (B.16), (B.17), (B.18), (B.19), (B.20) and (B.21) into equation (B.4) indicates that

$$\tau_1 = \left(I_1 + (m_1 a^2 + m_2 r_1^2)\sin^2\theta_2 + m_2 b^2\sin^2\theta_3 - 2m_2 r_1\sin\theta_2\sin\theta_3\right)\ddot{\theta}_1 + \left(2(m_1 a^2 + m_2 r_1^2)\sin\theta_2\cos\theta_2 - 2m_2 r_1 b\sin\theta_3\cos\theta_2\right).\dot{\theta}_1\dot{\theta}_2 + 2m_2 b(b\sin\theta_3 - r_1\sin\theta_2)\cos\theta_3.\dot{\theta}_1\dot{\theta}_3 \quad , \qquad \text{(B.22)}$$

$$\tau_2 = \left(I_2 + (m_1 a^2 + m r_1^2)\right)\ddot{\theta}_2 - m_2 r_1 b\cos(\theta_2 - \theta_3).\ddot{\theta}_3 - \left((m_1 a^2 + m_2 r_1^2)\sin\theta_2\cos\theta_2 - m_2 r_1 b\cos\theta_2\sin\theta_3\right).\dot{\theta}_1^2 - m_2 r_1 b\sin(\theta_2 - \theta_3).\dot{\theta}_3^2 - (m_1 a + m_2 r_1)g\sin\theta_2 \quad , \qquad \text{(B.23)}$$

and

$$\tau_3 = (I_3 + m_2 b^2)\ddot{\theta}_3 - m_2 r_1 b\cos(\theta_2 - \theta_3).\ddot{\theta}_2 + m_2 r_1 b\sin(\theta_2 - \theta_3).\dot{\theta}_2^2 + m_2 b(r_1\sin\theta_2 - b\sin\theta_3)\cos\theta_3.\dot{\theta}_1^2 + m_2 g b\sin\theta_3 \quad \text{(B.24)}$$

Equations (B.22), (B.23) and (B.24) can be conveniently expressed in the matrix form (*Craig* (1986), *Bejczy* (1974))

$$\tau = M(\theta)\ddot{\theta} + c(\theta,\dot{\theta}) + g(\theta) \quad ,\text{(B.25)}$$

where, $c(\theta,\dot{\theta})$ is the coriolis and centrifugal torque vector and $g(\theta)$ is the gravity torque vector. For simplicity, the coriolis, centrifugal, and gravity torques vectors are often combined as the vector $n(\theta,\dot{\theta})$. Equation (B.25) can then be expressed in the form

$$\tau = M(\theta)\ddot{\theta} + n(\theta,\dot{\theta}) \quad ,\text{(B.26)}$$

where $n(\theta,\dot{\theta}) = c(\theta,\dot{\theta}) + g(\theta)$ is the non-linear torque vector.

It is evident from equations (B.22), (B.23) and (B.24) that the inertia matrix has the form

$$M(\theta)=\begin{bmatrix} I_{1+}(m_1a^2+m_2r_1^2)\sin^2\theta_2+m_2b^2\sin^2\theta_3 & & \\ -2m_2r_1\sin\theta_2\sin\theta_3 & 0 & 0 \\ 0 & I_2+(m_1a^2+m_2r_1^2) & -m_2br_1\cos(\theta_2-\theta_3) \\ 0 & -m_2br_1\cos(\theta_2-\theta_3) & I_3+m_2b^2 \end{bmatrix}, \quad \text{(B.27)}$$

whilst the non-linear torque vector is

$$n(\theta,\dot{\theta})=\begin{bmatrix} \left(2(m_1a^2+m_2r_1^2)\sin\theta_2\cos\theta_2-2m_2r_1b\sin\theta_3\cos\theta_2\right).\dot{\theta}_1\dot{\theta}_2 \\ +2m_2b(b\sin\theta_3-r_1\sin\theta_2)\cos\theta_3.\dot{\theta}_1\dot{\theta}_3 \\ -\left((m_1a^2+m_2r_1^2)\sin\theta_2\cos\theta_2-m_2r_1b\cos\theta_2\sin\theta_3\right).\dot{\theta}_1^2 \\ -m_2r_1b\sin(\theta_2-\theta_3).\dot{\theta}_3^2-(m_1a+m_2r_1)g\sin\theta_2 \\ m_2b(r_1\sin\theta_2-b\sin\theta_3)\cos\theta_3.\dot{\theta}_1^2+m_2r_1b\sin(\theta_2-\theta_3).\theta_2^2 \\ +m_2gb\sin\theta_3 \end{bmatrix} \quad \text{(B.28)}$$

The inclusion of the payload, M_p, indicates that equation (B.27) becomes

$$M(\theta)=\begin{bmatrix} I_1+(m_1a^2+m_2r_1^2+M_pr_1^2)\sin^2\theta_2+(m_2b^2+M_pr_1^2) & & \\ \sin^2\theta_3-(2m_2r_1+2M_pr_1^2)\sin\theta_2\sin\theta_3 & 0 & 0 \\ & & \\ 0 & I_2+(m_1a^2+m_2r_1^2+M_pr_1^2) & -(m_2br_1+M_pr_1^2)\cos(\theta_2-\theta_3) \\ & & \\ 0 & -(m_2br_1+M_pr_1^2)\cos(\theta_2-\theta_3) & I_3+m_2b^2+M_pr_1^2 \end{bmatrix} \quad \text{(B.29)}$$

whilst equation (B.28) becomes

$$n(\theta,\dot{\theta})=\begin{bmatrix} \left(2(m_1a^2+m_2r_1^2+M_pr_1^2)\sin\theta_2\cos\theta_2-(2m_2r_1b+2M_pr_1^2)\sin\theta_3\cos\theta_2\right).\dot{\theta}_1\dot{\theta}_2 \\ +2m_2b(b\sin\theta_3-r_1\sin\theta_2)+2M_pr_1^2(\sin\theta_3-\sin\theta_2)\cos\theta_3.\dot{\theta}_1\dot{\theta}_3 \\ \\ -\left(M_pr_1^2+(m_1a^2+m_2r_1^2)\sin\theta_2\cos\theta_2-(m_2r_1b+M_pr_1^2)\cos\theta_2\sin\theta_3\right).\dot{\theta}_1^2 \\ -(m_2r_1b+M_pr_1^2)\sin(\theta_2-\theta_3).\dot{\theta}_3^2-(M_pr_1+m_1a+m_2r_1)g\sin\theta_2 \\ \\ m_2b(r_1\sin\theta_2-b\sin\theta_3)\cos\theta_3.\dot{\theta}_1^2+(M_pr_1^2+m_2r_1b)\sin(\theta_2-\theta_3).\theta_2^2 \\ +M_pr_1^2(\sin\theta_2-\sin\theta_3)\cos\theta_3.\theta_1^2+(M_pgr_1+m_2gb)\sin\theta_3 \end{bmatrix} \quad \text{(B.30)}$$

The numerical values of the parameters in equation (B.15) corresponding to the masses, lengths, and inertias of the links of the simulated model were as follows:

$m_1 = 4.0\,\text{kg}$, $m_2 = 3.0\,\text{kg}$, $r_1 = 0.3\,\text{m}$, $r_2 = 0.3\,\text{m}$, $a = 0.15\,\text{m}$, $b = 0.2\,\text{m}$, $I_1 = 0.05\,\text{kg.m}^2$, $I_2 = 0.1\,\text{kg.m}^2$, $I_3 = 0.1\,\text{kg.m}^2$.

These parametric values correspond to those of an experimental manipulator located in the Department of Mechanical Engineering, Faculty of Engineering Science, Osaka University, Japan, which was used in collaborative studies by Professor B. Porter and Professor S. Arimoto.

APPENDIX C

DYNAMICAL MODEL OF A TWO-LINK DIRECT-DRIVE ROBOTIC MANIPULATOR

The two-link direct-drive robotic manipulator which has been used in simulation and practical implementation studies is shown schematically in Fig. C.1, where the X-Y plane is horizontal.

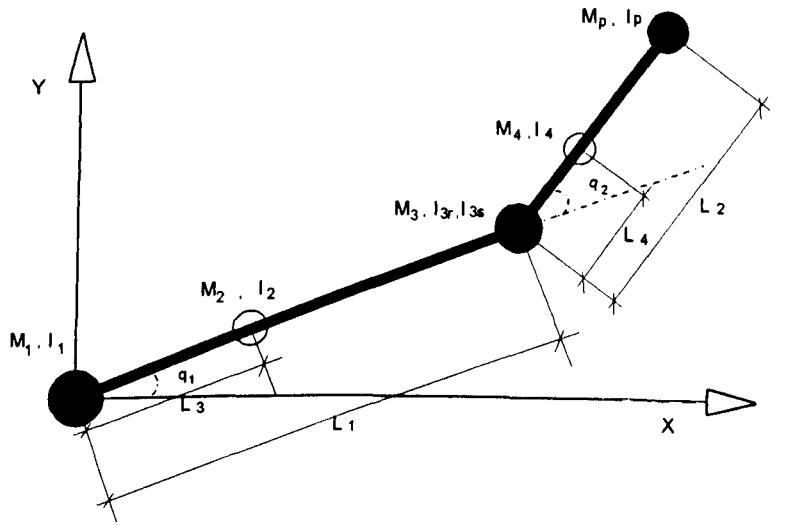

Figure C.1: Schematic diagram of a two-link robotic manipulator.

The parameters indicated in this figure are as follows:

I_1 : rotor inertia of motor 1

M_1 : mass of motor 1

I_2 : centroidal moment of inertia of link 1

M_2 : mass of link 1

I_{3r} : rotor inertia of motor 2

M_3 : mass of motor 2

I_{3s} : stator inertia of motor 2

M_4 : mass of link 2

I_4 : centroidal moment of inertia of link 2

M_p : mass of payload

I_p : moment of inertia of payload

q_1 : angle of link 1 with respect to the horizontal axis

q_2 : angle of link 2 with respect to the link 1

L_1 : length of link 1

L_2 : length of link 2

L_3 : distance of CG of link 1 from its axis of rotation

L_4 : distance of CG of link 2 from its axis of rotation

Other quantities which are used in the derivation of manipulator dynamics are the following:

x_{c2} : vector position of centre of mass of link 1

x_{c3} : vector position of centre of mass of motor 2

x_{c4} : vector position of centre of mass of link 2

x_{cp} : vector position of centre of mass of the payload

v_{c2} : vector velocity of centre of mass of link 1

v_{c3} : vector velocity of centre of mass of motor 2

v_{c4} : vector velocity of centre of mass of link 2

v_{cp} : vector velocity of centre of mass of payload

τ_1 : torque applied by motor 1

τ_2 : torque applied by motor 2

f_1 : friction torque for axis 1

f_2 : friction torque for axis 2

The equations of motion for the manipulator can be derived by using the Euler-Lagrange equations in the form

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right) - \frac{\partial T}{\partial q_i} = Q_i \qquad \text{(i=1,2)} \qquad , \tag{C.1}$$

where $\boldsymbol{q}$ is a vector of generalized coordinates which completely describe the configuration of the manipulator and $\boldsymbol{Q}$ is a corresponding vector of generalized forces which are applied to the manipulator. In this case,

$$\boldsymbol{q} = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \qquad , \tag{C.2}$$

and

$$\boldsymbol{Q} = \tau - \boldsymbol{f} = \begin{bmatrix} \tau_1 - f_1 \\ \tau_2 - f_2 \end{bmatrix} \qquad . \tag{C.3}$$

Note that the Lagrangian of this particular planar robotic manipulator reduces to only the kinetic energy, T , since its motions are confined to a horizontal plane.

In order to derive the dynamical equations for this particular robotic manipulator, it is required that the value of T be worked out. In this way, it is evident, from

elementary dynamics, that the kinetic energy of a single link moving in a plane can be expressed in the form

$$T = \frac{1}{2} I \omega^2 + \frac{1}{2} M v_c^2 \quad , \tag{C.4}$$

where I is the centroidal moment of inertia of the link about axis of rotation, ω is the angular velocity, M is the mass, and v_c is the linear velocity of the centre of mass. Therefore, the total kinetic energy of the manipulator can readily be expressed as the sum of the kinetic energies of each individual part of the manipulator by using Eq. (C.4). In this way, such individual different parts of the total kinetic energy of the manipulator can be specified as follows:

(a) kinetic energy arising from the angular velocity of the rotor of motor 1,

(b) kinetic energy arising from the angular velocity of link 1,

(c) kinetic energy arising from the angular velocity of the stator of motor 2 (installed on link 1),

(d) kinetic energy arising from the angular velocity of the rotor of motor 2,

(e) kinetic energy arising from the angular velocity of link 2,

(f) kinetic energy arising from the angular velocity of payload,

(g) kinetic energy arising from the linear velocity of the centre of mass of link 1,

(i) kinetic energy arising from the linear velocity of the centre of mass of motor 2,

(j) kinetic energy arising from the linear velocity of the centre of mass of link 2,

(k) kinetic energy arising from the linear velocity of the centre of mass of the payload.

Note that the pertinent angular velocity in parts (a), (b), and (c) is the same as the angular velocity of link 1, $\dot{q}_1$, whilst the pertinent angular velocity in parts (d), (e), and (f) is the sum of angular velocities of link 1 and link 2, $\dot{q}_1 + \dot{q}_2$. Therefore, the total kinetic energy of the manipulator can be readily expressed in the form

$$T = \frac{1}{2}[(I_1 + I_2 + I_{3s})\dot{q}_1^{\,2} + (I_{3r} + I_4 + I_p)(\dot{q}_1 + \dot{q}_2)^2 + M_2({v_{c2}}^T v_{c2}) + M_3({v_{c3}}^T v_{c3}) + M_4({v_{c4}}^T v_{c4}) + M_p({v_{cp}}^T v_{cp})] \quad . \tag{C.5}$$

The positions and velocities of the centres of masses are computed as follows:

(1) Position and velocity of the centre of mass of link 1

$$x_{c2} = \begin{bmatrix} L_3\cos(q_1) \\ L_3\sin(q_1) \end{bmatrix} \quad , \tag{C.6}$$

$$v_{c2} = \frac{dx_{c2}}{dt} = \begin{bmatrix} -L_3\dot{q}_1\sin(q_1) \\ L_3\dot{q}_1\cos(q_1) \end{bmatrix} \quad . \tag{C.7}$$

(2) Position and velocity of the centre of mass of motor 2

$$x_{c3} = \begin{bmatrix} L_1\cos(q_1) \\ L_1\sin(q_1) \end{bmatrix} \quad , \tag{C.8}$$

$$v_{c3} = \frac{dx_{c3}}{dt} = \begin{bmatrix} -L_1\dot{q}_1\sin(q_1) \\ L_1\dot{q}_1\cos(q_1) \end{bmatrix} \quad . \tag{C.9}$$

(3) Position and velocity of the centre of mass of link 2

$$x_{c4} = \begin{bmatrix} L_1\cos(q_1) + L_4\cos(q_1+q_2) \\ L_1\sin(q_1) + L_4\sin(q_1+q_2) \end{bmatrix} , \qquad \text{(C.10)}$$

$$v_{c4} = \frac{dx_{c4}}{dt} = \begin{bmatrix} -L_1\dot{q}_1\sin(q_1) - L_4(\dot{q}_1+\dot{q}_2)\sin(q_1+q_2) \\ L_1\dot{q}_1\cos(q_1) + L_4(\dot{q}_1+\dot{q}_2)\cos(q_1+q_2) \end{bmatrix} . \qquad \text{(C.11)}$$

(4) Position and velocity of the centre of mass of the payload

$$x_{cp} = \begin{bmatrix} L_1\cos(q_1) + L_2\cos(q_1+q_2) \\ L_1\sin(q_1) + L_2\sin(q_1+q_2) \end{bmatrix} , \qquad \text{(C.12)}$$

$$v_{cp} = \frac{dx_{cp}}{dt} = \begin{bmatrix} -L_1\dot{q}_1\sin(q_1) - L_2(\dot{q}_1+\dot{q}_2)\sin(q_1+q_2) \\ L_1\dot{q}_1\cos(q_1) + L_2(\dot{q}_1+\dot{q}_2)\cos(q_1+q_2) \end{bmatrix} . \qquad \text{(C.13)}$$

It is now possible to substitute these expressions for v_{c2} , v_{c3} , v_{c4} , and v_{cp} into Eq. (C.5), and then to use several trigonometric substitutions. This yields

$$\begin{aligned} T = \frac{1}{2}[&(I_1 + I_2 + I_{3s} + M_3L_1^2 + M_4L_1^2 + M_pL_1^2 + M_2L_3^2)\dot{q}_1^2 \\ &+ (I_{3r} + I_4 + I_p + M_4L_4^2 + M_pL_2^2)(\dot{q}_1+\dot{q}_2)^2 \\ &+ 2(M_4L_1L_4 + M_pL_1L_2)\dot{q}_1(\dot{q}_1+\dot{q}_2)\cos(q_2) \end{aligned} . \qquad \text{(C.14)}$$

Then, for use in the Lagrange equation (C.1), it follows that

$$\begin{aligned}\frac{\partial T}{\partial \dot{q}_1} &= [(I_1 + I_2 + I_{3s} + M_3 L_1^{\,2} + M_4 L_1^{\,2} + M_p L_1^{\,2} + M_2 L_3^{\,2})\dot{q}_1 \\ &+ (I_{3r} + I_4 + I_p + M_4 L_4^{\,2} + M_p L_2^{\,2})(\dot{q}_1 + \dot{q}_2) \\ &+ (M_4 L_1 L_4 + M_p L_1 L_2)(2\dot{q}_1 + \dot{q}_2)\cos(q_2) \\ &= p_1 \dot{q}_1 + p_2 \dot{q}_2 + p_3 (2\dot{q}_1 + \dot{q}_2)\cos(q_2) \quad , \end{aligned} \tag{C.15}$$

where

$$p_1 = I_1 + I_2 + I_{3s} + I_{3r} + I_4 + I_p + (M_3 + M_4 + M_p)L_1^{\,2} + M_2 L_3^{\,2} + M_4 L_4^{\,2} + M_p L_2^{\,2} \quad , \tag{C.16}$$

$$p_2 = I_{3r} + I_4 + I_p + M_4 L_4^{\,2} + M_p L_2^{\,2} \quad , \tag{C.17}$$

$$p_3 = M_4 L_1 L_4 + M_p L_1 L_2 \quad . \tag{C.18}$$

Similarly, it follows that

$$\frac{\partial T}{\partial \dot{q}_2} = p_2 \dot{q}_1 + p_2 \dot{q}_2 + p_3 \dot{q}_1 \cos(q_2) \quad . \tag{C.19}$$

It is thus evident that

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_1}\right) = p_1 \ddot{q}_1 + p_2 \ddot{q}_2 + p_3 (2\ddot{q}_1 + \ddot{q}_2)\cos(q_2) - p_3 (2\dot{q}_1 + \dot{q}_2)\dot{q}_2 \sin(q_2) \quad , \tag{C.20}$$

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_2}\right) = p_2 \ddot{q}_1 + p_2 \ddot{q}_2 + p_3 \ddot{q}_1 \cos(q_2) - p_3 \dot{q}_1 \dot{q}_2 \sin(q_2) \quad , \tag{C.21}$$

$$\frac{\partial T}{\partial q_1} = 0 \quad , \tag{C.22}$$

and

$$\frac{\partial T}{\partial q_2} = -p_3 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) \sin(q_2) \quad . \tag{C.23}$$

The vector matrix differential equation of motion for the robotic manipulator can now be derived when equations (C.20) through (C.23) are substituted into Eq. (C.1). This procedure indicates that

$$M(q)\ddot{q} + v(q, \dot{q}) = \tau - f \quad , \tag{C.24}$$

where

$$M(q) = \begin{bmatrix} p_1 + 2p_3 \cos(q_2) & p_2 + p_3 \cos(q_2) \\ p_2 + p_3 \cos(q_2) & p_2 \end{bmatrix} \tag{C.25}$$

is the state-dependent inertia matrix of the robotic manipulator and

$$v(q, \dot{q}) = \begin{bmatrix} -\dot{q}_2 (2\dot{q}_1 + \dot{q}_2) p_3 \sin(q_2) \\ \dot{q}_1^2 p_3 \sin(q_2) \end{bmatrix} \tag{C.26}$$

is the vector of coriolis terms.

APPENDIX D

DESCRIPTION OF A PRACTICAL TEST RIG TWO-LINK DIRECT-DRIVE ROBOTIC MANIPULATOR

D.1 BASIC HARDWARE CONNECTIONS

A two-link direct-drive robotic manipulator has been used for the experimental testing of the different methods of motion control developed in this thesis. As a direct-drive system, the manipulator operates in the absence of undesirable factors such as mechanical backlash and gear-train compliance. However, since there are no gear trains between the actuators and the arms, non-linear dynamical effects and cross-coupling become significant in this type of robotic manipulator.

This particular robot is a planar manipulator with two revolute joints (see Figure. 9.1). Each joint is directly connected to a high-torque brushless actuator which eliminates the need for gear reduction.

The hardware components of the robot are as follows:

(I) the robot (the NSK motors+Drives+Arms),

(II) the Texas Instruments Spectrum Signal Processing TMS320C30 Processor Board (see Figure 9.2),

(III) the Integrated Motions Inc. DS2 Motion Control Interface Board,

(IV) the 486 Host Workstation.

The functional description of the manipulator control workstation is shown in Figure D.1. It can be seen that the host computer performs user interface functions, as well as those involving inverse kinematics and the reference (desired) generation of kinematic parameters for the control servo.

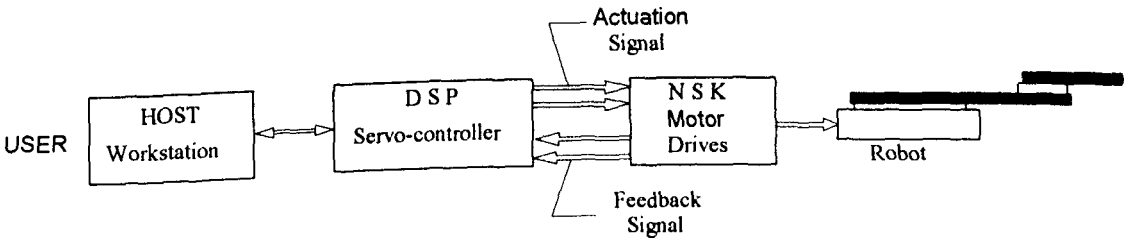

Figure D.1 Functional description of manipulator control workstation.

The Digital Signal Processor (DSP) based servo-controller executes the manipulator servo algorithm and performs all interface functions to the manipulators. A block diagram of the hardware architecture is shown in Figure D.2.

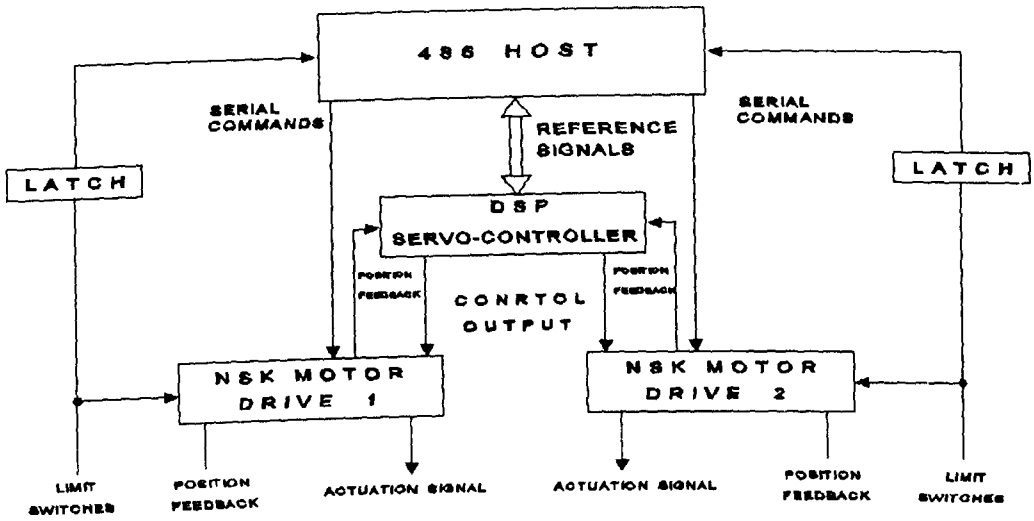

Figure D.2: Hardware Architecture.

D.2 SPECTRUM TMS320C30 DSP BOARD

Under normal operation, the user's servo algorithm is executed on the Spectrum TMS320C30, which is a high-performance CMOS 32-bit DSP floating point device with a clock rate of 33 MHZ. The DSP is installed on the bus of the host computer and interfaced to the NSK motor drives through an Integrated Motions inc DS2 Motion Control Board. The interface of the TMS320C30 DSP Processor Board with the DS2 is shown in Figure D.3.

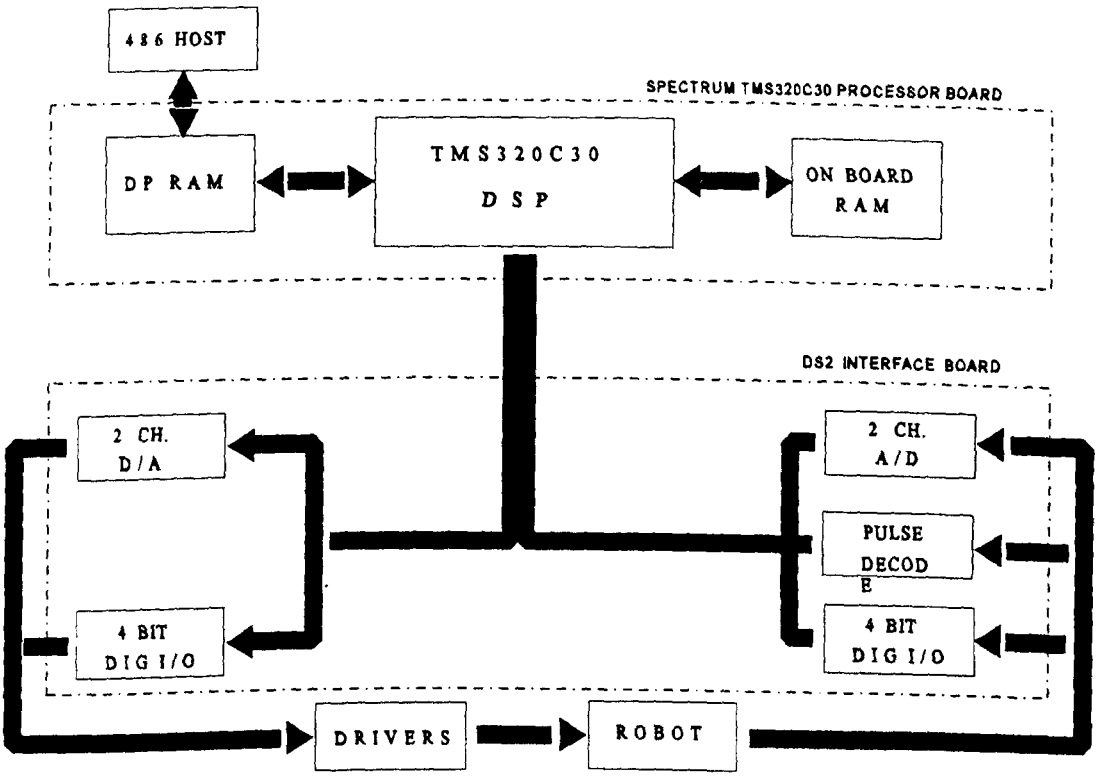

Figure D.3: The TMS320C30 Processor Board with DS2 Interface.

As shown in this figure, the DSP/DS2 combination provides the TMS320C30 DSP, 2 D/A converters, 2 A/D converters, 2 shaft encoders with 153600 counts per actuator revolution, and 4 bits of parallel I/O.

D.3 CONTROLLER SOFTWARE ARCHITECTURE

The control workstation software provides the full functionality of the two-axis robot controller. Figure D.4 is a block diagram of the architecture of the controller software that has been divided into several functional modules such as interpolation, inverse kinematics, and servo controller, each of which performs a special task.

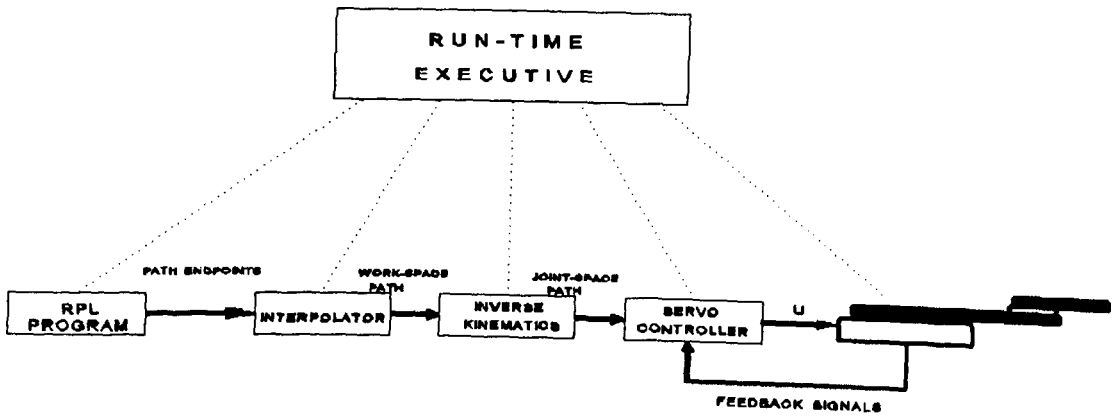

Figure D.4: Controller Software Architecture.

Each of the software modules is actually a separate C-language program. Some of these have been modified (by the author) in order, firstly, to produce the desired velocity and acceleration parameters and, secondly, to communicate with DSP such that these parameters be available to the servo control algorithm in real-time. In Figure D.4, RPL (Robot Programming Language) is the program which specifies the desired path or desired end-points for each joint of the robot for a work-space or joint-space trajectory, respectively. The interpolator connects each set of the end-points of the joints (either in joint-space or in work-space) with a straight line and assigns a trapezoidal velocity to this line. In the case of work-space trajectory, the inverse kinematic transformation converts each point along the work-space path into a corresponding joint-space point, which is sent to the axis servo-controller as the set-point. Note that, if joint space moves are specified in the RPL rather than work-space moves, then the interpolator directly outputs angular set-points and the inverse kinematic module is not used.

The servo algorithm, which generates the appropriate voltage output to drive the

manipulator, is written in the C-language and is compiled by the Texas Instrument 320C30 C compiler. The compiled code is downloaded to the servo processor where it is executed in order to control the robot.

The run-time executive acts as coordinator of the data flow between software modules, performs scheduling and synchronizes the servo-control CPU with the host CPU, and handles real-time communication of data between the host CPU and the servo-controller.

REFERENCES

ABIDIN, Z.: "Design oh Digital High-Accuracy trajectory Tracking System for Multivariable Plants", Ph. D. Thesis, Salford University, England, 1991.

AN, C. H., ATKESON, C. G. and HOLLERBACH, J. M.: " Model-based Control of a Direct Drive Arm, Part II: Control," Proc. IEEE Conf. on Robotics and Automation, Philadelphia, 1988.

ANDERSON, R. J.: "Smart: A Modular Control Architecture For Telerobotics", IEEE Robotics and Automation Magazine, pp 10-18, Sep. 1995.

ARIMOTO, S. et al.: "Iterative Learning Control Theory for Dynamic Systems", In Proceedings of IECON, pp 22-26, 1984.

ARIMOTO, S.: "Learning Control Theory for Robotic Motion", Int. J. Adaptive Contr. Signal Processing, Vol. 4, pp 453-564, 1990.

ARIMOTO, S.: "State-of-the=Art and Future Research Directions of robot Control", In Preprints of the 4th IFAC Symposium on Robot Control, Capri, pp 3-14, 1994.

ARIMOTO, S.: "Another language for describing robot motions: a non-linear position-dependent circuit theory", In Preprints of the 7th International Symposium on Robotics research, Munich, 1995.

ARIMOTO, S.: "Non-linear position-dependent circuits: a language for expressing dynamics of electro-mechanical systems", In Procedeedings of the 1996 Japan-USA Symposium on Flexible Automation, Boston, 1996.

ASADA, H. and SLOTINE, J. J. E.: "Robot Analysis and Control", John Wiley and Sons, 1986.

ASTOLFI, A. and LANARI, L.: "Optimal Tuning of PD Controllers for Rigid

Robots", In Proceedings of the Asian Control Confersnce, Tokyo, pp 141-2., 1994a.

ASTOLFI, A. and LANARI, L.: "H_∞ Control of Rigid Robot", In proceedings of the IFAC Symposiium on Robot Control, Capri, pp 199-204, 1994b.

ASTROM, K. J. and HAGGLUND, T.: "Automatic Tuning of PID Controllers", Research Triangle Park, NC: Instrument Society of America, 1988.

BACK, T.: "Optimal Mutation Rates in Genetic Search", In Proc. 5th Int. Conf. on genetics Algorithms, S. Forrest, Ed. San Mateo, CA: Morgan Kuffmann, pp 2-8, 1993.

BACK, T. et al.: "Evolutionary Programming and Evolution Strategies: Similarities and Differences", Proc. 2nd Ann. Conf. On Evolutionary Programming , San Diego, CA, ed D B Fogel and W Atmar, La Jolla, CA: Evolution Programming Society, pp 11-22, 1993.

BACK, T.: "Heurestics for Parameter-Setting Issues", Handbook of Evolutionry Computation, Oxford University Press, 1997.

BACK, T. et al.: "Evolutionary Computation: Comments on the History and Current State", IEEE Trans. on Evolutionary Computation, Vol. 1:1, pp 3-17, 1997.

BALL. J. A. et al.: "H^∞ Control for Nonlinear Systems with Output Feedback", IEEE Trnsactions on Automatic Control, 38, pp 546-549, 1993.

BEASLY, D. and MARTIN, R. P.: " An Overview of Genetic Algorithms:Part I, Fundamentals", University Computing, 15(1993), pp 58-69.

BEJCZY, A. K.: "Dynamic Models of Control Equations for Manipulators", Jet Propulsion Lab., Tech. Memo 715-719, Jet Propulsion Laboratory, Pasadena, California, 1979.

BEJCZY, A. K.: "Robots Arms Dynamics and Control", TM: 33-69. Jet Propulsion Laboratory, 1974.

BELLMAN, R.: "Dynamic Programming", Princeton University Press, Princeton, 1957.

BERNSTEIN, D. S. et al.: "Minimal Complexity Control law Synthesis- Part 2: Problem Solution Via H_2/H_∞ Optimal Static Output Feedback", in Proc. 1989 Amer. Contr. Conf., Pittsburgh, PA, pp 2506-2511, 1989.

BLACK, H. S.: "Stabilised Feedback Amplifiers", Bell System Technical Jpurnal, Vol. 13, pp 1-18, 1934.

BODE, H. W.: "Network Analysis and Feedback Amplifier Design", Van Nostrand, New York, 1945.

BOX, G. E. P.: "Evolutionary Operation: A Method for Increasing Industrial Productivity", Appl. Stat. 6 pp 81-101. 1957.

BRADSHAW, A. and PORTER, B.: "Design of Linear Multivariable Continuous Time Tracking Systems Incorporating High Gain Decentralized Error-Actuated Controllers", Int. J. Systems Sci., 10, pp 961-970, 1979.

BRADSHAW, A. and PORTER, B.: "Design of Linear Multivariable Discret-Time tracking Systems Incorporating Fast-Sampling Error-Actuated Controllers", Int. J. Systems Sci., Vol. 11, pp 817-826, 1980a.

BRADSHAW, A. and PORTER, B.: "Asymptotic Propreties of Linear Multivariable Discrete Time Tracking Systems incorporating Fast-Sampling Error-Actuated Controllers", Int. J. Systems Sci., Vol.11 pp 1279-1293, 1980b.

BRADSHAW, A. and PORTER, B.: "Singular Perturbation Methods in the Design of Tracking Systems Incorporating Fast-Sampling Error-Actuated Controllers", Int. J. Syst. Sci., Vol. 12, pp 1181-1191, 1981.

BREMERMAAN, H. J.: "Optimization Through Evolution and Recombination", Self-Organizing Systems ed M C Yovits et al. Washington, DC: Spartan. 1962.

CALLENDER, A., HARTREE, D. R.,and PORTER, A.: "Time Lag in a Control System", In Philos. Trans. R. Soc. London A. London: Cambridge University Press, 1936.

CARR, S. A, and GRIMBLE, M. J.: "Comparison of LQG, H_∞, and Classical Designs for the Pitch Rate Control of an unstable Military Aircraft", Proc. IMA/RAe Aerospace Vehicle Dynamics and Control Conference, Cranfield Institute of Technology, September, 1992.

CHEN, B. S. et al.: "A Genetic Approach tp Mixed H_2/H_∞ PID Control", IEEE Control Systems, 1986.

CHOI, Y. K. and BIEN, Z.: "Decentralized Adaptive Control Scheme for Control of a Multi-Arm-Type Robot", Int. J. Control, Vol. 48, No. 4, pp1755-1722, 1988.

CHOW, J. H.: "Asymptotic Stability of a Class of Nonlinear, Singularly Perturbed Systems", J, Franklin Inst., 305, pp 231-252, 1978.

COLAUGH, R. et al.: "Performance-Based Adaptive Tracking Control of Robot Manipulators", J. of Robotic Systems, 12(8), 517-530, 1995.

COMSTOCK, C. and HSIAO, G. C.: "Singular Perturbation for Difference Equations" Rocky Mountain J. Math., 6, pp 561-567, 1976.

CRAIG, J. J.: "Introduction to Robotics", Addison-Wesley, Reading, MA, 1986.

CRAIG, J. J.:" Introduction to Robotics, Mechanics and Control", Addison-Wesley Publishing Company, 1989.

CRAIG, J. J., HSU, P., and SASTRY, S. S.: "Adaptive Control of Mechanical Manipulators", IEEE Int. Conf. on Robotics & Automation, Vol. 1, pp 190-195, 1986.

COLAUGH, R., GLASS, K., and SERAJI, H.,: "Performance-Based Adaptive Tracking Control of Robot Manipulators", J. of Robotic Systems, 12(8), 517-530, 1995.

DAVIDOR, Y.: "Robot Programming With a genetic algorithm", IEEE International Conference on Computer Systems and Software Engineering, 1990.

DAVIDOR, Y.: "Genetic Algorithms and Robotics, A Heuristic Strategy for Optimization", Teaneck, NJ, World Scientific, 1991.

DAVIDSON, E. J, Gesing, W., and Wang, S. H.: "An Algorithm for Obtaining the Minimal Realisation of a Linear Time-Invariant System and Determining if a System is Stabilizable detectable", IEEE Trans. Autom. Control, Vol. Ac-23, No.6, pp 1048-1054, 1978.

DAVIS, L.: "Handbook of Genetic Algorithms", Van Nostrand Reinhold, 1991.

DAWSON, D. M. and LEWIS, F. L.: "Comments on Adaptive Inverse Dynamics

Control of Rigid Robots", IEEE Trans. on Automatic Control, Vol. 36, No. 10, 1991.

D'AZZO, J. J. and HOUPIS, C. H.: "Linear Control System Analysis & Design: Conventional and Modern", 1988

DE JONG, K. A.: "Analysis of Behavior of a Class of Genetic Adaptive Systems", Ph. D Thesis, University of Michigan, 1975.

DEB, K.: "Optimal Design of Welded Beam Via Genetic Algorithms", AIAAJ. Vol. 29, No. 11, pp 2013-2015, 1991.

DESILVA, C. W.: "Intelligent Control: Fuzzy Logic Application", CRC Press, 1995.

DRACOPOULOS, .: "Genetic Algorithms and Genetic Programming in Control Engineering", Invited Chapter, in Evolutionary algorithms in Engineering Applications, ed. by D. Dasgupta and Z. Michalewicz, Springer Verlag, 1997.

EVANS, W. R.: "Graphical Analysis of Control Systems", AIEE Trans., Vol.67, pp 547-551, 1948.

EVANS, W. R.: "Control System Synthesis by Root Locus Method", AIEE Trans., Vol.69, pp 66-69, 1950.

FALB, P. L. and WOLOVICH, W. A.: "Decoupling in the Design and Synthesis of Multivariable Control Systems", IEEE Trans. Autom. Control, AC-12, pp 651-669, 1967.

FISHER, R. A.: "The Genetical Theory of Natural Selection", Clarendon, 1930.

FOGEL, L. J.: "Autonomous Automata", Industrial Res, 1962.

FOGEL, D. B.: "Evolutionary Computation: Toward a New Philosophy of Machine Intelligence", IEEE: New York, 1995.

FOGEL, D. B.: " Why Evolutionary Computation?", Handbook of Evolutionary Computation,Oxford University press, 1997.

FRANCIS, B. A and DOYLE, J. C.: "Linear Control Theaory with an H-Infinity Optimality Criterion", SIAM J. Control, 1986.

FU, K. S. et al.: "Robotics: Control, Sensing, Vision, and Intelligence", McGraw-Hill, New York, 1987.

GILBERT, E. G.: "Controllability and Observability in Multivariable Control Systems", SIAM J. Control, Ser. A., Vol. 1, pp 128-151, 1963.

GLOVER, K. and MACFARLANE, D. C.: "Robust Stabilization of Normalized Coprime Factor Plant Descriptions with H-Infinity Bounded Uncertainty", IEEE Trans. Autom. Control, No. 34, pp 821-830, 1989.

GOLDBERG, D. E.: "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison-Wesley, 1989.

GORINEVSKY, D. et al.: "Learning Approximation of Feedforward Control Dependence on the task parameters with application to Direct-Drive manipulator tracking", Trans. on Robot. Autom., Vol.13, No.4, 1997.

GREFENSTETTE, J. J.: "Optimization of Control Parameters for Genetic Algorithms", IEEE Trans. on Systems Man and Cybernetics, Vol. 16, No. 1, pp 122-128, 1986.

GRUJIC, L. T.: "Singular Perturbation, Large Scale Systems and Asymptotic

Stability of Invariant Sets", Int. J. Syst. Sci., 10, pp. 1323-1341, 1979a.

HALDANE, J. B. S.: "The Cause of Evolution", Longmans, 1932.

HERDY, M.: "Application of the Evolutionstrategien to Discrete Optimization problems", In Manner and Schwefel MS91, pp 188-192, 1991.

HICKS, D. L.: "Optimal Design of Degital Model-Following Systems", Ph. D. Thesis, Salford University, 1994.

HOLLAND, J. H.: "Outline for a Logical Theory of Adaptive Systems", J. ACM 9 pp 297-314, 1962.

HOLLAND, J. H.: "Adaptation in Natural and Artificial Systems", University of Michigan Press, 1975.

HO, B. L. and KALMAN, R. E.: "Effective Construction of Linear State Variable Models from Input/Output Functions", Regelungstechnick, Vol. 14, pp 545-548, 1966.

HOROWITZ, I. M.: "Quantitative Synthesis of Uncertain Multiple-Input Feedback Systems", Int. J. Control, Vol.30, pp 81-106, 1979.

HSIA, T. C.: "Adaptive Control of Robot Manipulators-A Review", IEEE Int. Conf. on Robotics and Automation, San Francisco, 1986.

HUNT, K. J.: "Polynomial LQG and H_∞ Controller Synthesis: A genetic Algorithm Solution", Proc. IEEE Conf. on Decision and Control (Tuscon, AZ) Picataway, NJ: IEEE, 1992b.

KALMAN, R. E.: "On the General Theory of Control Systems", Proc. First IFAC

Congress, Moscow, Vol.1, pp 481-492, London: Butterworth, 1960.

KALMAN, R. E.: "The Theory of Optimal Control and the Calculus of Variations", In Mathematical Optimization Techniques, R. Bellman (ed), Univ. of California Press, Berkeley, 1963.

KARNOPP, D., and ROSENBERG, R. C. : "Analysis and Simulation of Multipart Systems". MIT Press, Cambridge, MA

KAWAMURA, S., MIYAZAKI, F., and D ARIMOTO, S.: " Is a Local Linear PD Feedback Control Law Effective For Trajectory Tracking of Robot Motion? " Proc. IEEE Conf. on Robotics and Automation, Philadelphia, 1988.

KELLY, R.: "A Tuning Procedure for Stable PID control of Robot Manipulators", Robotica, Vol. 13, pp 141-148, 1995.

KODITSHEK, D. E.: "Quadratique Lyapunov Functions for Mechanical Systems", In Proceedings of the IEEE Conference on Robotics and Automation, pp 1-6, 1987.

KOKOTOVIC, P. V. and SANNUTI, P.: "Singular Perturbation Method for Reducing Model Order in Optimal Control Design", IEEE Trans., AC-13, pp 377-384, 1968.

KOKOTOVIC, P. V. and Prkins, W, R.: "Singular Perturbations: Order Reduction in Control Systems Design", American Society of Mechanical Engineers, New York, 1972.

KRISHNAKUMAR, K. and GOLDBERG, D.E.: "Control System Optimization Using Genetic Algorithms", J. of Guidance, Control, and Dynamics, Vol. 15, No. 3, May 1992.

KRISTINSSON, K. and DUMONT, G.A.: " System Identification and Control Using Genetic Algorithms", IEEE Trans. on Sys., Man, and Cyber, Vol 22, No. 5, 1992.

LEE, C. S. G. and CHUNG, M. J.: "An Adaptive Control Strategy for Mechanical Manipulators", IEEE trans. on Automatic Control, Vol. Ac-29, No. 9, 1984.

LEE, C. S. G. and CHUNG, M. J.: "An Adaptive Control Strategy for Computer-Based Manipulators", 21th *Conf. on* Dec. & Control, pp 95-100, Orlando, FL, 1982.

LEVINE, W. R. and ATHANS, M.: "On the determination of the Optimal Constant Control Output Feedback Gains for Linear Multivariable Systems", IEEE. Trans. Autom. Control, Vol. AC-15, pp 44-48, 1970.

LI, C. J.: "A New Adaptive Control Method of Robot", in Recent Trend in Robotics: Modeling, Control, and Education, Jamshidi et al (ed), North Holland Pub., New York, pp 179-184, 1986.

LICHTFUB, H. J.: "Evolution Eines Rohrkrummers Dipl.-Ing. Thesis, Technical University of Berlin, Hermann Fottinger Institute for Hydrodynamics, 1965.

LIN, F. and BRANDT, R. D.: "An Optimal Control Approach to Robust Control of Robot Manipulators", IEEE. Trans. Robot. Autom. Vol 14, No 1, Feb. 1998.

LIN, F.: "Robust Control Design: An Optimal Control Approach", AFI Press, 1997.

LOCATELLI, A. and SCHIAVONI, N.: "Two-Time Scale Discrete Systems", 1st Int. Conference on Information Science and Systems, Patras, Greece, 1976.

LUENBERGER, D. G.: "Observers for Multivariable Systems", IEEE Trans.

Autom. Vol. AC-11, pp 190-197, 1966.

LUH, J. Y. S., WALKER, M. W. and PAUL, R. P. C. : "On-Line Computational Scheme for Mechanical manipulators. Journal of Dynamic Systems, Measurement, and Control. 102, 69-76. 1980.

LUSE, D. W. and KHALLIL, H. K.: "Frequency Domain Results for Systems with Slow and Fast Dynamics", IEEE Trans., AC-30, pp 594-612, 1985.

LYAPUNOV, A. M. : "Probleme General de la Stabilite du Mouvement". Ann. Fac. Sci. Univ. Toulouse Sci. Math. Sci. Phys., vol. 9, original paper pubished in 1893 in Commun. Soc. Math. Kharkow, 1893; reprinted as vol. 17 in Annals of Math Studies. Princeton, NJ: Princeton University Press, 1949

MACLAY, D. and DOREY, R.: " Application of Genetic Search Techniques to Drive train Modelling", Proc. of the 1992 IEEE Int. Symp. on Intelligent Control, 1992, CH. 101, pp 542-547, 1992.

MAHMOUD, M. S.: "Design of Observer-Based Controllers for a Class of Discrete Systems", Automatica, 18, pp. 323-328. 1982a.
MAHMOUD, M. S.: "Stabilization of Discrete Systems with Multiple-Time Scales", IEEE Trans., AC-31, pp 159-162, 1986.

MANGANAS, A.: "Singular Perturbation Methods in the Design of tracking Systems Incorporating Fast-Sampling Error-Actuated Digital PID Controllers", Report USAME-DC-101-85, Department of Aeronautical and Mechanical Engineering, University of Salford, April 1985.

MANGANAS, A.: "Design of High-Performance Tracking Systems for Linear Multivariable Plants Using Input/Output Data", Ph. D. Thesis, Salford University, England, 1987.

MARKIEWICZ, B. R.: "Analysis of The Computed Torque Drive Method And Comparison with Conventional Position Servo for a Computer-Controlled Manipulator", Jet Propulsion Lab., Tech. Memo. 33-601, 1973.

MATHIAS, K. E. and WHITLEY, L. D.: "Changing representation during search: a comparative study of delta coding", Evolutionary Comput. 2, 1994.

THE MATHWORKS INC, Mimo Control Tool Box, Matlab™ , MathWorks Inc, 1994.

MCINNIS, B. C. and LIU, C. K. F.: "Kinematics and Dynamics in Robotics: A Tutorial Based upon Classical Concepts of Vectorial Mechanics", IEEE J. of Robotics and Automation, Vol. RA-2, No. 4, 1986.

MICHALEWICZ, Z.: "*Genetic Algorithms + Data Structures = Evolution* Programs", Berlin: Springer, 1992.

MOORE, B. C.: "On the Flexibility Offered by State Feedback Multivariable Systems Beyond Closed-Loop Eigenvalue Assignment", IEEE Trans., AC-21, pp 689-692, 1976.

MUHLENBEIN, H.: " How Genetic Algorithms Really Work: I. Mutation and Hillclimbing", Proc. 2nd Int. Conf. on Parallel Problem Solving From nature (Brussels), ed R Manner and B Manderick (Amsterdam: Elsevier), pp. 15-25, 1992.

NAIDU, D.S. and RAO, A. K.: " A Singular Perturbation Method for Initial Value Problems with Inputs in Discrete Control Systems", Int. J. Control, 33, pp 953-965, 1981.

NAIDU, D. S. and RAO, A. K.: "Singular Perturbation Analysis of Discrete Control Systems", Lecture Notes in Mathematics, Vol. 1154 (Springer-Verlag, Berlin), 1985.

NYQUIST, H.: "Regeneration Theory", Bell System Technical Journal, Vol.11, pp 126-147, 1932.

ORTEGA, R. and SPONG, M. W.: "Adaptive Motion Control of Rigid Robots: A Tutorial", Proc. of the 27th Conf. on Decision and Control, Austin, Texas, 1988.

PAUL, R. P.: "Robot Manipulators: Mathematics, Programming and Control", MIT Press, Cambridge, MA, 1981.

PAYNTER, H. M.: "Analysis and Design of Engineering Systems", MIT Press, Cambridge, MA, 1960.

PETROPOULAKIS, L.: "Design of Digital Trajectory-Tracking Systems for Robotic Manipulators", Ph.D Thesis: University of Salford, 1986.

PHILLIPS, R. G.: "A Two-Stage Design of Linear Feedback Controls", IEEE Trans. Aut. Control, Vol. AC-25, No.6, pp 1220-1223, 1980.

POLAK, E. and MAYNE, D. Q.: "An Algorithm for Optimization Problems with Functional Inequality Constraints", IEEE Trans., Ac-21, pp 184-193, 1976.

PONTRYAGIN, L. S., BOLTYANSKI, V. G., GAMKRELIDZE, R. V., and MISCHENSKO, Y. F.: "The Mathematical Theory of Optimal Processes", Interscience, New York, 1963.

PORTER, B.: "Singular Perturbation Methods in the Design of Stabilizing feedback Controllers for Multivariable Systems", Int. J. Control, 20, pp 689-692, 1974.

PORTER, B.: "Singular Perturbation Methods in the Design of Full-Order Observers for Multivariable Linear Systems", Int. J. Control, 26, pp 589-594, 1977.

PORTER, B.: "Design of High-Performance Tracking Systems", Report AWFAL-TR82-3032, US Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio, USA, 1982.

PORTER, B.: "Design of set-point tracking systems incorporating fast-sampling error actuated controllers for linear multivariable plants using step-response matrices", Proc. 10th IFAC World Congress, Munich, 1987.

PORTER B.: "Issues in the design of intelligent control systems", IEEE Control Systems Magazine, January 1989

PORTER, B.: "Genetic Design of Control Systems", J. SICE, Vol.34, 1995.
PORTER, B.: "Evolutionary Production Planning: A Cautionary Tale", in QUAGLIERRA, D. et al (Eds), Genetic Algorithms and Evolution strategies in Engineering and computer Scilence, Wiley, 1998.

PORTER B. and ABIDIN, Z.: "Robustness characteristics of fast-sampling digital set-point tracking PID controllers for completely irregular linear multivariable systems", Proc. Inst. MC Int. Symposium on Application of Multivariable Systems Tecniques, Bradford, UK, April 1990a.

PORTER, B. and ABIDIN, Z.: "Robustness characteristics of fast-sampling digital set-point tracking PID controllers for Robotic Manipulators", IASTED International Conference on Control '90, Lugano, Switzerland, 1990b.

PORTER, B. and ALLAOUI, C.: "Genetic Robustification of Digital Trajectory-

Tracking Controllers for Robotic Manipulators", Proc. IEEE International Conference on Systems, Man and Cybernetics. 1995a.

PORTER, B. and ALLAOUI, C.: "Performance Measures in the Genetic Design of Digital Controllers for Robotic Manipulators", Proc. IEEE International Conference on Evolutionary Computing, 1995b.

PORTER, B. and ALLAOUI, C.: "Genetic Synthesis of Optimal Control Policies for Manufacturing Systems", Proc. World automation Congress, 1996.

PORTER, B. and ALLAOUI, C.: "Evolutionary Robustification of Digital Trajectory-Tracking Controllers for Robotic Manipulators " , International Journal of Neural and Soft Computing, In Press (1998).

PORTER, B., SANGOLOLA, B.A. and ZADEH, N.N. : "Genetic Design of Computed-Torque Controllers for Robotic Manipulators", IASTED Int. Conf. on Systems and Control, Switzerland, June 1994.

PORTER, B. et al.: "Design of Fast Non-Interacting Digital Fight-Mode Control Systems for High-Performance Aircraft", Proc. AIAA Guidance, Navigation, and Control Conference, Snowmass, Colorado, USA, 1985.

PORTER, B. and BORAIRI, B.: "Genetic design of linear multivariable feedback control systems using eigenstructutre assignment", *Int. J. Systems Sciences*, Vol. 23, pp. 1387-1390, 1992.

PORTER, B. and BRADSHAW, A.: "Design of Linear multivariable Continuous-Time Tracking Systems Incorporating High-Gain Error-Actuated Controllers", Int. J. Systems Sci., Vol.10, pp 461-469, 1979a.

PORTER, B. and BRADSHAW, A.: "Asymptotic Propreties of Linear

Multivariable Continuous Time Tracking Systems incorporating High-Gain Error-Actuated Controllers", Int. J. Systems Sci., Vol.10, No.12 pp 1433-1444, 1979b.

PORTER B. and J.J. D'AZZO: "Transmission zeros of linear multivariable continuous time systems", Electronic Letters, Vol. 5, pp. 1155-1164, 1974.

PORTER, B. and HICKS, D. L.: "Genetic Design of Digital Model-Following Control Systems for Regular Linear Multivariable System Techniques", March 29-30, Univers. Of Bradford, UK, 1994

PORTER B. and JONES, A. H.: "Genetic tuning of digital PID controllers", Electronics Letters, Vol. 28, pp. 843-844, 1992.

PORTER, B. et al.: "Design of digital model-following flight-mode control systems for high performance aircraft", Proc. AIAA Guidance, Navigation and Control Conference, Minneapolis, USA, August 1988.

PORTER, B. and MANGANAS, A.: "Design of Adaptive Fast-Sampling Digital Set-Point Tracking Controllers Incorporating Recursive Step-Response Matrix Identifiers for Unstable Multivariable Plants", IFAC, Workshop on Adaptive Control of Chemical Process, Unbehaum, Frankfurt/Main, pp 206-211, 1995.

PORTER, B. and MANGANAS, A.: "Design of Adaptive Direct Digital Control System Incorporating Recursive Step-Response Matrix Identifiers for High Performance Aircraft with Noisy Sensors", IEEE, National Aerospace and Electronics Conference, Dayton, Ohio, 1987.

PORTER, B. and MERZOUGUI, T.: "Evolutionary Synthesis of Optimal Control Policies for Manufacturing Systems", Proc. IEEE International Conference on Emerging Technologies and factory Automation, 1997.

PORTER, B., MOHAMED, S. S. and JONES, A. H.: "Genetic Tuning of PID Controlllers", Electronics letters, 28, 843/844, 1993.

PORTER, B. and OTHMAN, M. Z.: "Robustness Characteristics of Fast-Sampling Digital PI Controllers for regular Linear Multivariable Plants", 4th International Symposium on Application of Multivariable System Techniques, Bradford, UK, 1990a.

PORTER, B. and SHENTON, A. T.: "Singular Perturbation methods of Asymptotic Eigenvalue Assignment in Multivariable Linear Systems", Int. J. Systems Science, Vol.6, No.1, pp 33-37, 1975a.

PORTER, B. and SHENTON, A. T.: "Singular Perturbation Analysis of the Transfer-Function Matrices of a Class of Multivariable Linear Systems", Int. J. Control, Vol.21, No 4, pp 655-660, 1975b.

PORTER, B. And ZADEH, N. N.: "Genetic Design of Fuzzy-Logic Controllers for Robotic Manipulators", Proc. IEEE Int. Conf. on Evolutionary Computing, Perth, Western Australia, November 1995a.

PORTER, B. and ZADEH, N. N.: "Genetic Design of Computed-Torque/Fuzzy-Logic Controllers for Robotic Manipulators", Proc. IEEE Int. Symp. on Intelligent Control, Monterey, CA, USA, August 1995b.

PORTER, B. and ZADEH, N. N.: "Evolutionary Design of Fuzzy-Logic Controllers for Manufacturing Systems", Annals of the CIRP, Vol 46/1, pp 425-428, 1997.

PORTER, B. and ZADEH, N. N.: "Solution of Some Classsical Job-Shop Scheduling Problems Using Evolution Strategies", Proc. World Automation

Congress, Anchorage, USA, May 1998.

RECHENBERG, I.: "Cybernetic Solution Path of an Experimental Problem", Royal Aircraft Establishment Library Translation 1122, 1965.
RECHENBERG, I.: "Evolutionistrategie: Optimierrung Technischer Systeme Nach Prinzipien der Biologischen Evolution", Frommann-Holzboog, 1973.

RECHENBERG, I.: "Evolution Strategies in Computation of Intelligence: Imitating Life", IEEE Press 1994.

REINHARDT, H. J.: "On Asymptotic Expansions in Nonlinear Singularly Perturbed Difference Equations", Num, Funct. Anal. Opt., 1, pp 565-587, 1979.

ROSENBERG, R.S.: "Simulation of Genetic Populations with Biochemical Properties", Doctoral Dissertation, University of Michigan. Dissertation Abstracts International, 28(7), 2732B, 1967.

ROSENBERG, R. and KARNOPP, D.: " Introduction to Physical System Dynamics", McGraw-Hill, New York,

ROSENBROCK, H. H.: "State Space and Multivariable Theory", Nelson, London, 1970.

ROSENBROCK, H. H.: "Relatively Prime Polynomial Matrices", Electronics Letters, Vol.4, pp 224-228, 1968.

SADEGH, N. and HOROWITZ, R.: "Stability and Robustness Analysis of a Class of Adaptive Controllers for Robotic Manipulators", The Int. J. of Robotics Research, Vol. 9, No. 3, 1990.

SALOMON, R.: "Improved Convergence Rate of back-Propagation with

Dynamic Adaptation of the Learning Rate", Proc. 1st Int. Confer. Parallel Problem Solving From nature. Portland 90, Berlin, Springer, 1991.

SCHAFFER, J. D. et al.: "A Study of Control Parameters Affecting Online Performance of Genetic algorithms for Function Optimization", Proc. 3rd Int. Conf. on Genetic Algorithms (Fairfax, VA, June 1989) ed J D Schaffer (San Mateo, CA: Morgan Kaufmann) pp 51-60, 1989.

SCHEWEFEL, H. P.: "Kybernetische Evolution als Strategie der Experimentellen Forschung in der Stromungstechnik", Dipl.-Ing. Thesis, Technical University of Berlin, Hermann Fottinger Institute for Hydrodynamics, 1965.
SCHEWEFEL, H. P.: "Experimentelle Optimierung Einer Zweiphasenduse Teill" AEG Research Institute Project MHD-Staustrahlrohr 11034/68, 1968.

SCHEWEFEL, H. P.: "Binare Optimierung Durch Somatische Mutation", Working Group of Bionics and Evolution Techniques at the Institute of Measurement and Control Technology technical Report Re 215/3, 1975.

SCHEWEFEL, H. P.: "Numerische Optimierung Von Computer-Modellen Mittels der Evolutionstrategie", (Interdisciplinary Systems Research 26), Basle: Birkhauser, 1977.

SCHEWEFEL , H. P.: "Advantages (and Disadvantages) of Evolutionary Computation Over other Approaches ", Handbook of Evolutionry Computation, Oxford University Press, 1997.

SCHULTZ W.C. and RIDEOUT V.C.: "Control System Performonce Measures: Past, Present, and Future", IRE TRANSACTION ON AUTOMATIC CONTROL, pp. 22-35 February 1961

SEPHERI, N., WAN F. L. K., LAWRENCE, P. D. and DUMONT, G.: "Hydraulic Compliance Identification Using a Parallel Genetic Algorithm", Mechatronics, Vol. 4, No. 6, pp 617-633, 1994.

SHELDON, S. N. and RASMUSSEN, S. J.: "Development and First Successful Flight Test of A QFT Flight-Control System", IEEE National Aerospace and Electronics Conference, Dayton, Ohio, USA, 1994.

SLOTINE, J. J. E.: "The Robust Control of Robot Manipulators", Int. J. Robotics Research, Vol. 4, No. 2, pp 49-64, 1985.

SLOTINE, J. J. E. and LI, W., "Adaptive Manipulator Control: A Case Study", IEEE Trans. on Automatic Control, Vol. 33, No. 11, 1988.

SLOTINE, J. J. E. and LI, W.: "On the Adaptive Control of Robot Manipulators", The Int. J. of Robotic Research, Vol. 6, No. 3, 1987.

SOLANO, J. and JONES, D. I.: "Parameter Determination for a Genetic Algorithm Applied to Robot Control", IEEE Int. Conf. on Control 94, Ch. 277, pp 765-770, 1994.

SPONG, M.W. and ORTEGA, R.: "On the Adaptive Inverse Dynamics Control of Rigid Robots", IEEE Trans. on Automatic Control, Vol. 35, No.1, 1990.

SU, C. and LEUNG, T.: "A Sliding Mode Controller with Bound Estimation for Robot Manipulators", IEEE Trans. on Robotics and Automation, Vol. 9, No. 2, 1993.

TAKEGAKI, M. and ARIMOTO, S.: "An Adaptive Trajectory Control of Manipulators", Int. J. Control, Vol. 34, pp 219-230, 1981.

VAN DER SHAFT, A. J.: "On a State Space Approach to Nonlinear H_∞ Control", Systems and Control Letters, 16, 1-8, 1991.

VAN DER SHAFT, A. J.: "L_2-Gain Analysis of Nonlinear Systems and Nonlinear state Feedback H_∞ Control", IEEE Transactions on automatic Control. 37, pp 770-784, 1992.

VOIGT, H. M. et al.: "Fuzzy Recombination for the Breeder Genetic Algorithm", In L. Eshelman, Editor, Genetic Algorithms: procedings of the 6th International Conference, pp 104-111. Morgan Kaufmann Publishers, san Francisco, CA, 1995.

VUKOBRATVIC, M. and STOKIC, D.: "Non-Adaptive Dynamic Control Manipulation Robots-Invited Survey Paper", Theory and Practice of Robots and Manipulators, Ch 45, pp 109-122, 1985.

WIENER, N.: "Extrapolation, Interpolation and Smoothing of Stationary Time Series", MIT and Wiley, New York, 1949.

WINTER, G. et al.: "Genetic algorithms in Engineering and Computer Science", Chichester: Wiley, 1995.

WRIGHT, S.: "The Roles of Mutation, Inbreeding, Crossbreeding and Selection in Evolution", Proc. Six International Congress of Genetics, 1932.

YANIV, O.: "Multiple-Input Single-Output (MISO) QFT CAD Package", Tel Aviv University, Tel Aviv 69 978, 1992.

YOUNG, K. K. D.: "Controller Design for a Manipulator Using Theory of Variable Structure Systems", IEEE Trans. on Systems, Man, and Cybernetics, Vol. Smc-8, No. 2, 1978.

ZAMES, G.: "Feedback and Optimal Sensitivity: Model Refence transformations, Multiplicative Semi-Norms, and Approximate Inverses", IEEE Trans. Autom. Control, Vol. AAC-26, pp 301-320, 1981.

ZHOU, K. et al.: "Mixed H_2 and H_∞ Control", In Proc. Amer. Contr. Conf., San Diego, CA, pp. 2505-2507, 1990.