CONTROL OF ROBOTIC SYSTEMS BY USING ADAPTIVE CONTROL AND LEARNING METHODS

A MASTER'S THESIS

In

Mechanical Engineering

University Of Gaziantep

By

Mehmet Topalbekiroglu

January 1996

Approval of the Graduate School Of Natural And Applied Science

[illegible]

Prof. Dr. Mazhar ÜNSAL
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science in Mechanical Engineering Department.

[illegible]

Assoc. Prof. Dr. Sedat BAYSEÇ
Chairman of the Department

We certify that we have read this thesis and in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science in Mechanical Engineering Department.

[illegible]

Asist. Prof. Dr. Ali KİREÇCİ
Supervisor

Examining Committee in Charge

Prof. Dr. İ. Hüseyin FİLİZ	[illegible]
Assoc. Prof. Dr. Sedat BAYSEÇ	[illegible]
Asist. Prof. Dr. Ali KİREÇCİ	[illegible]

ABSTRACT

CONTROL OF ROBOTIC SYSTEMS BY USING ADAPTIVE CONTROL AND LEARNING METHODS

TOPALBEKIROGLU Mehmet

M.S. in Mechanical Engineering

Supervisor: Assist. Prof. Dr. Ali KIRECCI

January 1996, 138 pages

In this study, advanced control methods such as adaptive and learning control methods are aimed to investigate and apply experimentally which ensure the stability of robotic systems under every conditions. Also, performance of these control methods and conventional control method are compared.

Adaptive control can be classified into two groups, namely, self-tuning and model reference adaptive control. Additionally, self-tuning adaptive control can be divided into many groups theoretically, however, two of them can be practically used, which are explicit and implicit self-tuning adaptive control methods. The general strategy for all adaptive control methods is the systematic estimation of system parameters, which minimise the output error of the system at the next step. Learning control is based upon systematic tuning the control input to reduce the error using the information of previous cycles.

Adaptive control and learning control methods are applied to a hydraulic robot which has three DOF, and a DC servo system. The experimental results show that these control methods can be applied to ant robotic system which can provide very good performance with these control method under any circumstances.

Keywords: Adaptive control, implicit self-tuning, explicit self-tuning, model reference adaptive control, learning control and parameter estimation methods.

ÖZET

ROBOTİK SİSTEMLERİN ADAPTİF VE ÖĞRENME KONTROL METODUYLA KONTROLU

TOPALBEKİROĞLU Mehmet

Yüksek Lisans Tezi Makina Mühendisliği Bölümü

Tez Yöneticisi Yrd. Doç. Dr. Ali KİRECÇİ

Ocak 1996 138 Sayfa

Bu çalışmada robotik sistemlerin kararlılığını her şart altında sağlayabilecek ileri düzey kontrol metodlarından adaptif ve öğrenme kontrol metotlarının incelenmesi, deneysel olarak uygulanması ve bu metodlarla klasik bir kontrol metodunun performans açısından karşılaştırılması amaçlanmıştır.

Adaptif kontrol, self-tuning ve model referans adaptif kontrol olmak üzere başlıca iki gruba ayrılır. Ayrıca, self-tuning kontrol pek çok alt gruba ayrılmakla beraber uygulamalı olarak kullanılabilecek indirek ve direk self-tuning kontrol metodu olmak üzere iki gruba ayrılır. Tüm adaptif kontrol metotlarında genel strateji bir sonraki adımda meydana gelecek hatayı minimuma indirgeyecek sistem parametrelerinin sistematik bir şekilde tahmini olarak hesaplanmasına dayanır. Öğrenme metodu ise sistemin bir önceki yörünge takibinde yapmış olduğu hataları kullanarak bir sonraki yörünge için kontrol sinyalinin hataların sistematik bir şekilde ayarlanması esasına dayanır.

Adaptif kontrol ve öğrenme kontrol metodları üç serbestlik dereceli bir hidrolik robota ve bir DC servo sisteme uygulanmıştır. Deney sonuçları bu kontrol metodlarının herhangi bir sisteme kolaylıkla uygulanabileceğini ve her şart altında sistemin iyi bir performans sergileyeceğini göstermiştir.

Anahtar Kelimeler: Adaptif kontrol, direk self-tuning adaptif kontrol, indirek self-tuning adaptif kontrol, model referans adaptif kontrol, öğrenme kontrol ve parametre tahmin metodları.

ACKNOWLEDGEMENTS

I would like to express my sincerest gratitude to the perfect teacher and supervisor Assist. Prof. Dr. Ali Kirecci for his inspiring and illuminating supervision, guidance, suggestions, valuable criticisms and continual encouragement throughout the preparation of this thesis.

I wish to thank Mechanism and Machine Dynamics Laboratory staff members who offered their help, especially to, Okkes Iyiekici and Muhtar Diken.

Also, I wish to thank to my family for their continuous moral support during all my studies and the progress of this work. Therefore, my very special thanks are due to them.

Some part of experimental study is carried out on a hydraulic robot which is entitled "Robotlarda Gorsel Bilgi Analizi" supported by the State Planning Organization of Turkey amounting to 67480 $ under the grant number 91 K 120830.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The control scheme for robotics system is trajectory tracking for which some control techniques are applied in order to obtain acceptable results from a system. When a desired signal (reference signal) is applied to a robotic system it responds in a characteristic fashion. The physical features of the actuators and the gain setting of the controller are the main parameters that determine the response of the system (systems output). The optimum gain values can be determined by developing a mathematical model of the system or by applying an appropriate experimental method. Controllers with fixed gain values are effective for many conventional processes such as pick and place tasks using slow speed manipulators. However, there are several cases where precise tracing of a trajectory under different payloads require more advanced control technique to ensure stability of the process. In such applications the control of a system with fixed parameter is completely inadequate.

Adaptive and learning controls are two important methods of advanced control. The main idea of adaptive control method is based on continuous tuning of the gain values of the controller choosing a time-series difference equation model which to describe the input-output measurements in the system. Since the system dynamic is assumed unknown, the parameters in such model can be computed recursively and on-line at each sampling instant. The gain values of the controller are adjusted according to the updated parameter estimates and measurements so as to make the system achieve the desired goals that are given as the design specification.

Many robotic systems like as industrial robots are often used to perform sequence of repetitive operations. A learning control method takes advantage of the repetitive nature of the system motions to improve trajectory tracking. The basic idea of this control method is based on tuning the input for the system using the experience of past responses of the system when executing repetitive tasks. This control method obtains the system information during iterative operations of system and uses certain type of learning methodology to modify the system input so that the error between desired trajectory and system output becomes more closer at the end of each cycle.

Advances in microprocessor technology have made digital control more attractive then before. Digital control methods offer many advantages compared with analogue control. They allow complicated control has to be implemented and the resulting system performance to be much more accurate [1].

1.1 Adaptive Control

The term "adaptive" has become one of the most fashionable words in the vocabulary of modern automatic control theory. In everyday language, to adapt means to change a behaviour to conform to new circumstances. Intuitively, an adaptive controller can alter its behaviour in response to changes in process and disturbances' dynamics [2, 3]. To avoid confusion, adaptation is taken to mean "the process of changing" the parameters structure and possibly the controls of a system on the basis of information obtained during the control period. So, as to optimise the state of a system when the operating conditions are either, incompletely defined initially or changed.

Many definitions of adaptive control have been suggested. A summary of definitions is given in Saridis et la (1973). Most of the definitions of adaptive controllers are very vague and it is difficult to draw the boundary-lines between the different types of adaptive controllers. It is even difficult to determine if a controller adaptive or not, since many adaptive controllers can be regarded as non-linear and time-varying controller. The main idea is, however, that an adaptive controller has the ability to modify its behaviour depending on the performance of the closed-loop system. Most definitions of adaptive controllers point out some basic functions that are common for most adaptive controllers [4].

1) Identification of unknown parameters or measurement of a performance index.
2) Decision of the control strategy.
3) On-line modification of the parameters of the controller.

Much research in adaptive control is very mathematical, it is sometimes difficult to transform theoretically correct adaptive controller into a practically operable one. Nevertheless, many theoretical developments in the last decades have brought adaptive control theory much closer to practise, which has, however been successfully hidden by the control theory community [2].

It is generally acknowledged that the problem of designing a good control system is basically that of matching the dynamic characteristics of a system by those of the controller. In other words, if the dynamics of the system and the characteristics of the disturbances affecting it are known, then the controller which will yield the desired performance can be designed. Thus controller design can be broadly separated into the following stages.

1) Determine dynamics of the process to be controlled.
2) Identify nature of major disturbances.
3) Specify the desired characteristics of the control system.
4) Design of controller.

From the above, it can easily be deduced that if the dynamic characteristics of the system and associated disturbances are unknown, or if assumed known are incorrect, then the resulting controller will not produce the desired dynamics performance. For example, industrial robots are based on high-precision servo technology. These robots are non-linear system because of the non-linear mechanical coupling between the different motions, the variations in the moment of inertia and compliance with geometry. The dynamics of these manipulator can be characterised by a set of highly coupled non-linear differential equations. On the basis of such dynamic models, the control of manipulators has been extensively studied in recent years and two control design approaches have been proposed, non-adaptive and adaptive control [5]. Non-adaptive control is based on an exact knowledge of the complex system dynamic equations. However, this model usually needs more complicated control structures, thus incurring higher costs when put to practical use. Also many parameters effect the accuracy of the dynamic model such as link inertias, mass centres, friction and air resistance, etc. Some of these are uncertain and change with time and therefore cannot be obtained exactly in advance. The explicit use of these parameters may degenerate the control performance and may lead to instability [5].

Also, it was realised that a fixed controller cannot provide acceptable system behaviour in all situations during the development of modern control theory. Particularly if the process to be controlled has unknown or time-varying parameters, the design of a fixed controller that always satisfies the desired specification is not straightforward.

Model reference adaptive control and self-tuning adaptive control are the two basic methods of adaptive control. The use of these adaptive control techniques are motivated by the need to automatically adjust the parameters of the controller when system parameters and disturbances are unknown or change with time. This is done to achieve an acceptable level of performance of the control system.

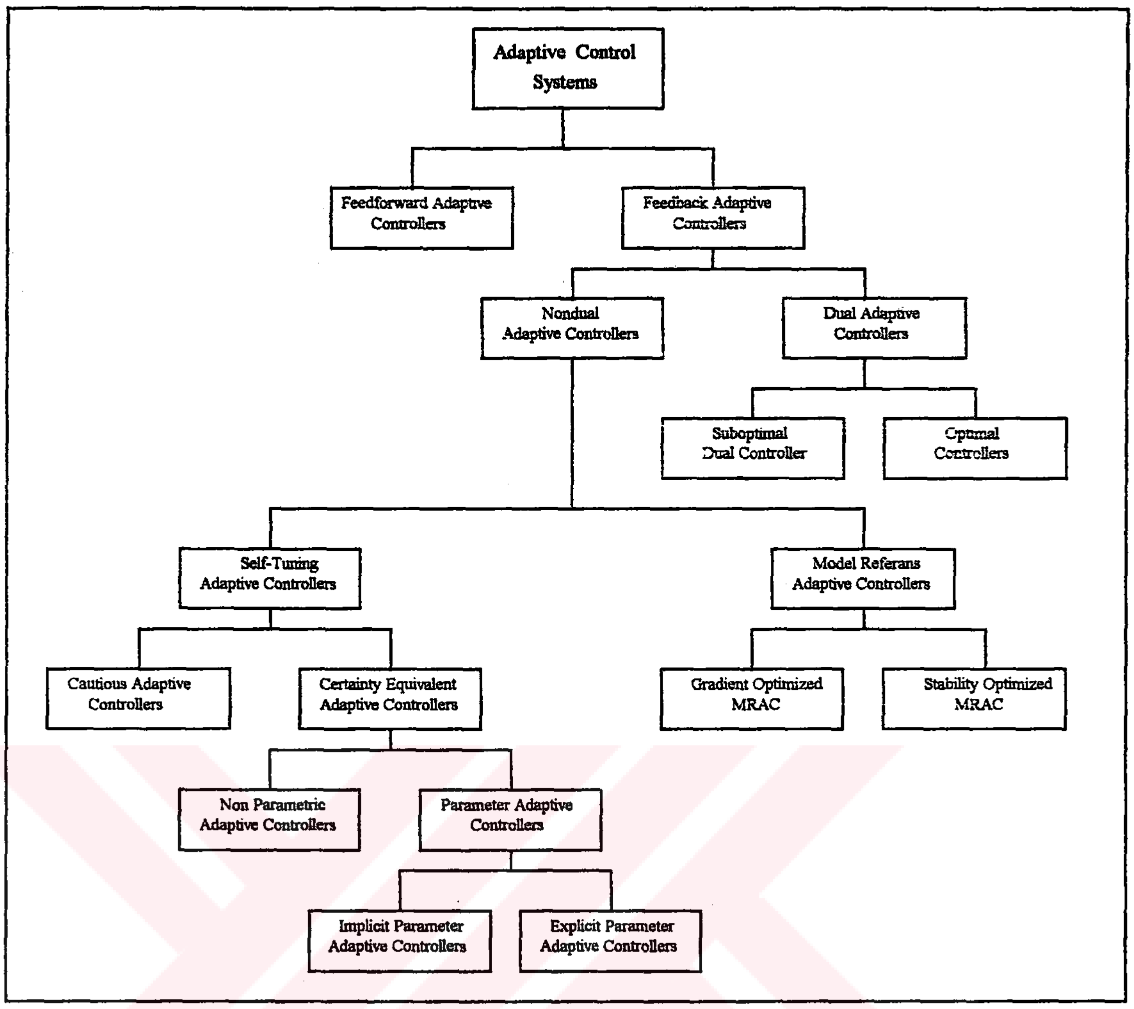

Adaptive Control Systems
Feedforward Adaptive Controllers
Feedback Adaptive Controllers
Nondual Adaptive Controllers
Dual Adaptive Controllers
Suboptimal Dual Controller
Optimal Controllers
Self-Tuning Adaptive Controllers
Model Referans Adaptive Controllers
Cautious Adaptive Controllers
Certainty Equivalent Adaptive Controllers
Gradient Optimized MRAC
Stability Optimized MRAC
Non Parametric Adaptive Controllers
Parameter Adaptive Controllers
Implicit Parameter Adaptive Controllers
Explicit Parameter Adaptive Controllers

Figure 1.1 General structure of adaptive control

Most adaptive control systems can be classified into two main groups feedforward adaptive controllers and feedback adaptive controllers (Figure 1.1) [6].

1.1.1 Feedforward Adaptive Controllers

Feedforward adaptive control systems are based on the fact that the changing properties of the system can be grasped by measurement of signals acting on the process. If it is known how the controller must be changed dependent on these measurable signals, a feedforward adaptation mechanism can be realised (see Figure 1.2). A special feature of feedforward adaptive controllers is that there is no feedback of inner closed-loop signals to adapt the controller parameters. In order to realise the open-loop adaptation algorithm, the influence of the measurable signals on the process behaviour and on the control loop behaviour must be known.

One feedforward adaptive control scheme is the so-called gain scheduling, which was used for the first time in the early 1950 s in aircraft control. For adaptive control with gain scheduling the controller gain is usually designed in advance for measurable signals describing the operating conditions, with the assumption that the external signals change slowly in comparison to the process dynamics. The same procedure can be applied for the other controller parameters. This may be called parameter scheduling. The calculated parameters are stored in tables of characteristic curves and used for control when the operating conditions match those for which the controller parameter set was designed.

One advantage of feed forward adaptive control is a fast reaction to process changes because the process behaviour is known in advance and need not be identified with measurable process input and output signals. Disadvantages are the neglect of all effects based on unmeasured signals or disturbances, unpredictable changes of the process behaviour and the amount of parameter storage that may be necessary to accommodate many operating conditions and the limitations to slow process parameter changes

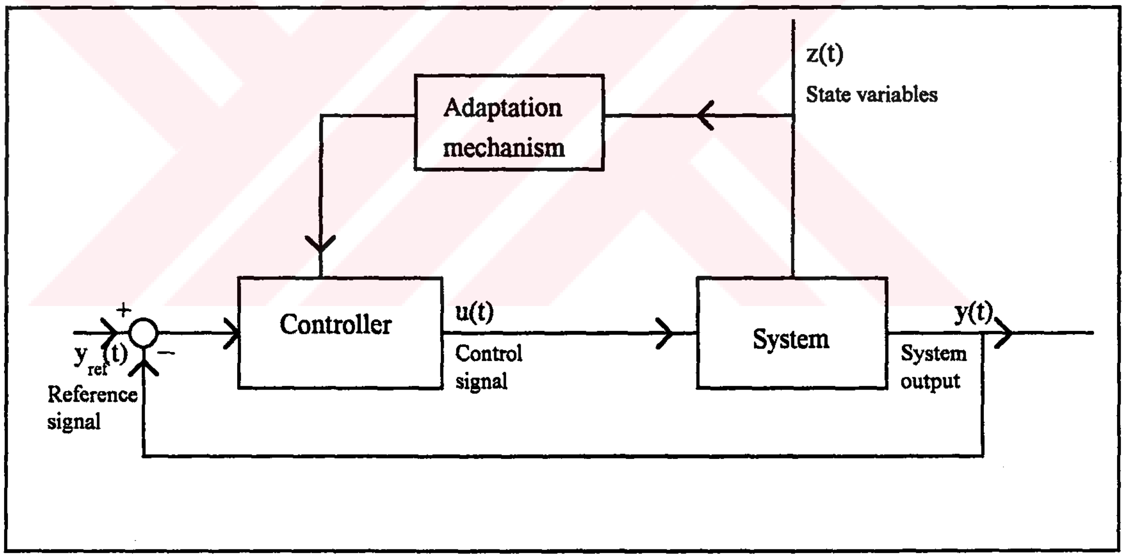


Figure 1.2 Basic structure of feedforward adaptive control (open loop adaptation)

1.1.2 Feedback Adaptive Controllers

If the changes in the process behaviour cannot be determined directly by measurement of the signal acting on the process, feedback adaptive controllers can be used. The structure of adaptive controllers of this type is shown in Figure 1.3.

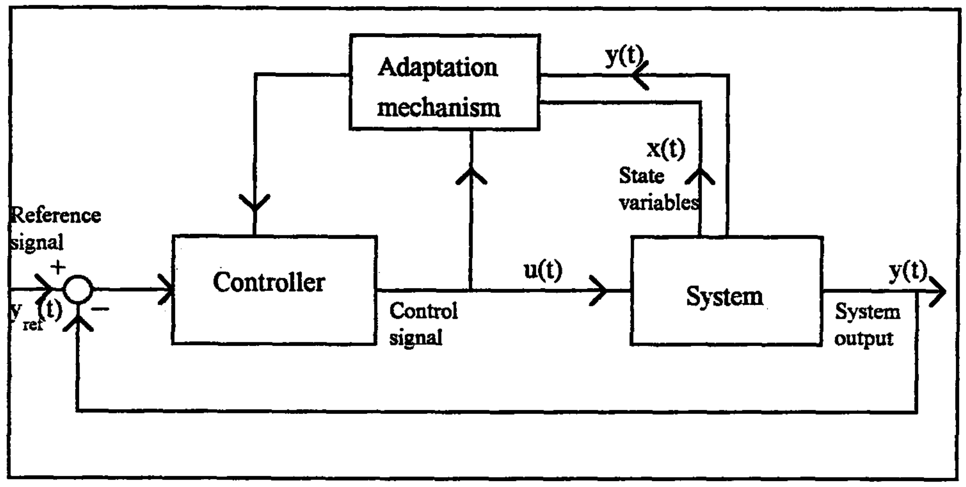


Figure 1.3 Basic structure of feedback adaptive control (closed-loop adaptation)

They are characterised by the following facts:

1) The changing properties of different internal control loop signals.
2) In addition to the basic control loop feedback, the adaptation mechanism results in an additional feedback level.
3) The closed-loop signal flow path yields a non-linear second feedback level.

The most common feedback adaptive control systems, as opposed to open-loop adaptive systems, can be distinguished as shown in Figure 1.2. Feedback adaptive controllers are subdivided into nondual adaptive and dual adaptive controllers.

1.1.2.1 Nondual Adaptive Controllers

The nondual adaptive type of controller minimises a design performance criterion considering only present and past values of the control loop signal, and the current information about the process, state or signal estimates. No information about future estimates is taken into account. The design strategy of nondual adaptive controllers is closely associated with the separation and certainty equivalence principles. Based on these principle of controller design the model reference and self tuning adaptive controller.

1.1.2.1.1 Self-Tuning Adaptive Controllers (STAC)

Self-tuning adaptive control (STAC) is one of important basic control method of adaptive control. The designs of self-tuning adaptive controllers are based upon systems whose parameters have been described by stochastic models. Basic structure of STAC method is shown in Figure 1.4. This figure implies that there is unknown parametric model (stochastic model) underlying the system control response, an estimator for system model parameters based on real system data and control design procedure.

The basic idea of this approach is to construct an algorithm that automatically changes controller parameters to meet a particular requirement or situation. That is, if the STAC is applied to time-varying systems and the methods are modified to enable the tracking of changing system parameters. This is done by adaptation mechanism. The adaptation mechanism consists of system parameter identification and controller design procedure (see Figure 1.4). Unknown parameters of the system model can be estimated using a parameter estimation methods. The controller parameters are calculated from the estimated system model, frequently by neglecting the uncertainties of the estimates. Finally control signal is calculated from a redefined control law, normally derived from some form of performance criterion or cost function and applied to manipulate the system. That's why the improving of the control algorithm used for system model identification and controller designs are executed in closed loop, on-line and in real time.

Note that Figure 1.4 shows the system identifier and the controller that are two distinct blocks. This division can be justified either through the use of the certainty equivalence principle or separation principle.

The certainty Equivalence principle is theoretically justified if the estimated parameters are accurate representations of the true parameters. In other words the control signal is calculated as if the controller has been designed from known system parameters (deterministic model). One well-known class of problem for which the certainty equivalence principle holds the linear quadratic-gaussian control problems. In adaptive control there are very few cases where the certainty equivalence principle is applicable. One exception is when the unknown parameters are stochastic variables that are independent between different sampling intervals. This is the underlying assumption that is being made in the design and application of self-tuning algorithms.

The separation principle is valid if it is possible to make a separation between system identifier and design of controller. For the calculation of the controller parameters, only the controller design criterion is minimised, without consideration of the information performance criterion. The parameters of the controller are also allowed to be function of the uncertainties in the estimated system parameters.

Therefore the separation principle is weaker than the certainty equivalence principle in that design follows a less strict set of procedure. Detailed discussion of these two concepts can be found in Bar-Shalom and Tse (1974) [7] and Patchell and Jacobs (1971) [8].

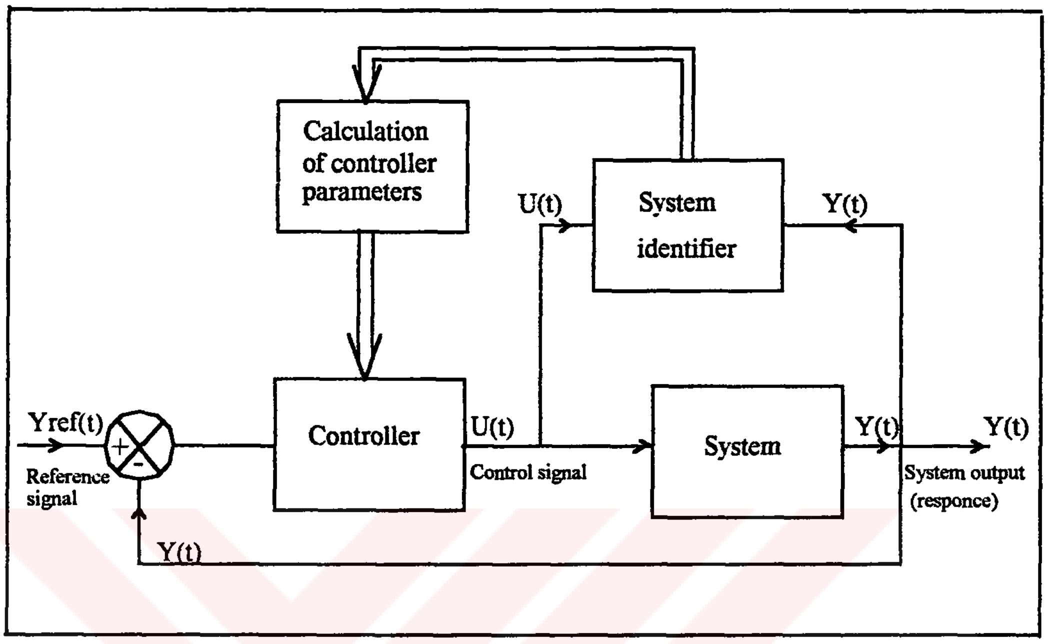


Figure 1.4 Basic structure of the self-tuning adaptive controller

Self-tuning adaptive control method can be classified into cautious adaptive controllers and certainty adaptive controllers.

a) Cautious Adaptive Controller

Cautious adaptive controller design is characterised by the fact that the separation principle used but the uncertainties of the estimates are considered in the controller design. In the case of estimate uncertainties the controller applies a cautious action on the process in the form of smaller values, or smaller change in the process input signal, result in worse estimation and this yields a much smaller value of the manipulated variable. This positive feedback leads to the possibility of a drop off the control loop. In this case the controller is aware of the errors in the estimates and takes a more conservative control action.

b) Certainty Equivalent Adaptive Controllers

Certainty equivalent adaptive controllers are based on the separation principle and also on the assumption that the estimated model parameters are identical with the true process parameters. For controller design estimate uncertainties are not taken into account. The true process parameters in the theoretical design equations are replaced by their estimates for on line realisation.

Certainty equivalent adaptive controllers can be subdivided into parameter adaptive controllers and non-parameter adaptive controllers. In the latter case, the identification of the system is performed by using a nonparameteric system model. A parametric system model is used in parameter adaptive controller (Iserman, 1982) [6, 9].

The structure of a parameter adaptive control loop is the same as shown in Figure 1.4, where the coefficients of the parameters for system model are explicitly estimated, so that the system model parameters are available as an intermediate result. This is called an explicit parameter adaptive controller.

The certainty equivalence principle mentioned above is only theoretically justified for this type of adaptive controller if the system parameters are constant, a quadric performance criterion is used for system model estimation and only white noise is acting on the process. For unknown stochastic parameter variations, the certainty equivalence principle is valid theoretically only if the parameter variations are statistically independent. For parameter adaptive control the certainty equivalence principle is not satisfied in general, although it is frequently used as an Ad Hoc design principle (Wittenmark, Bar-Shalom and Tse, 1974)[4, 7].

If the controller parameters are directly identified because the system model is inserted in the controller design equation, the parameter adaptive controller may be called implicit parameter adaptive controller (Astrom, 1983) [10].

The idea of self-tuning is not new, Kalman (1958) [11] derived the fist recognisable self-tuner and attempted to realise the algorithm in a special purpose computer. Parameter-adaptive control algorithms based on parameter estimation algorithms has taken place in recent years. These adaptive controllers are characterised by combining a recursive parameter estimation algorithm for a difference equation system and noise model and a suitable control algorithm. The parameter estimates are calculated after each sampling and used for the calculation of the parameters of the control algorithm. In the work of Astrom and Wittenmark (1973) [12] the recursive least-squares parameter estimation method is combined with a minimal variance controller. In this case the model structure is chosen so that the estimated parameters coincide with the controller parameters, i.e. the controller parameters are calculated directly. In the work of Cegrell and Hedqvist (1975) [13] and Borisson and Syding (1976) [14] successful applications of this "self-tuning regulator" are reported. In the work by Clarke and Gawthrope (1975) [15] the recursive least square's method is combined with an extended minimal variance controller, and in Saridis and Lobbia (1972) [16] stochastic approximation with a state feedback controller is regarded.

The application of self-tuning adaptive control covered a range of industries. These are ore-crushing, paper-making, titanium-dioxide kilns, distillation columns, and others [17]. For the application of self-tuning adaptive control methods to the control of manipulator arms, it was introduced by Koivo and Guo [18, 19] in a form based on minimum-variance self-tuning (Astrom and Wittenmark 1973) [12]. In addition, M. B. Zarrop [20] proposed the pole placement self-tuning control to a manipulator arm. Elliot (1984) [21] also suggested a non-linear adaptive control to a two link manipulator arm. The other some applications of self-tuning adaptive control method have been reported [22, 23, 24, 25]. All applications of self-tuning adaptive control to control of manipulator arms and other systems remain in computer simulation studies.

1.1.2.1.2 Model Reference Adaptive Controller (MRAC)

Model reference adaptive control method is a class of adaptive controls and belongs to the modern control theory. Model reference adaptive control, that is an explicit adaptive technique has been very attractive from the very beginning of the adaptive control era. Basic structure of a model reference adaptive control is shown in Figure 1.5. In the MRAC, a reference model specifies the desired control performance and a control input is generated to drive the controlled system to track the response of the reference model. The problem to be considered here is to design an adaptive control system that will cause the error between the model and the system outputs to approach zero without using any anticipative values of the system output. That is, on the basis of the desired closed-loop behaviour the unknown controller parameters can be adapted in such a way that a suitable error signal, $e(t)$, will be minimised. In order to diminish the error signal, a performance criterion may be defined which can be optimised analytically. The minimisation, together with consideration of possible constraints and additional demands, yields the necessary adaptation algorithm creating an adaptive system that is both non-linear and time-varying.

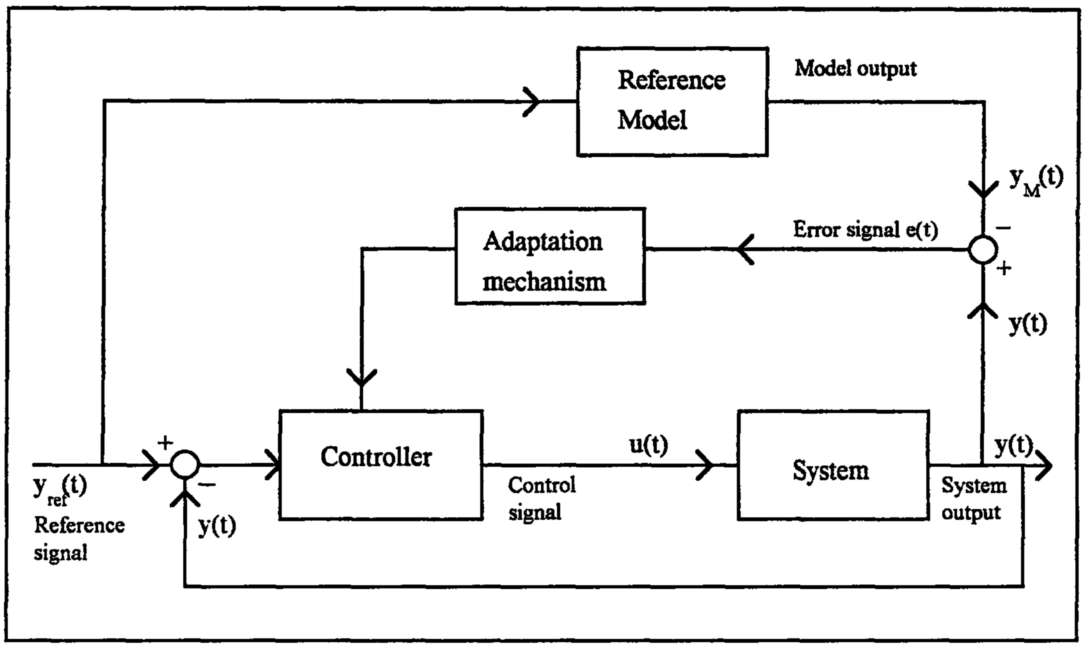


Figure 1.5 Basic structure of a model reference adaptive control

For practical applications the adaptation can be divided into three stages:

1) Comparison of closed loop behaviour.
2) Calculation of the controller parameters based on the adaptation law.
3) Adjustment of the controller.

The differences between the various model references adaptive controllers are due to the design procedure of the adaptation law. Common to all model reference adaptive control method is the rapid adaptation to defined input signals and straightforward treatment of stability using non-linear stability theory. However when using a fixed reference model, the model reference system approaches the given reference models but not necessarily an optimal one. Additionally, the control loop only adapts if the input signal of the reference model changes. The presence of significant noise signals in the measured signal may cause problems for the adaptation.

Model reference adaptive controller was one of the methods originally suggested for aerospace problems in the 1950s (Whitaker et al, 1958). Further work was done by Parks (1966), Hang and Parks (1973), Monopoli (1973), Landau (1974) and Ionescu and Monopoli (1977). There has been a steady interest in the method (Hang and Parks, 1973). Laudau's book (Laudau, 1979) gives a comprehensive treatment of work up to 1978. It also includes many references [10]. Duboswsky (1979) [26] proposes a model reference adaptive control method in which a second order dynamic system was used as the reference model for manipulator arm.

Since the 1960s, MRAC theory has been extended to the case where only output feedback is used. Several specific discrete times MRAC algorithms have been developed. Various MRAC algorithms have been proposed. However, they are rather complicated and only a few examples of the practical applications of MRAC have been reported in the process control. Almost all applications of the MRAC to system remain in computer simulation study for this reason [27, 28, 29].

In the following sections, some methods are mentioned which are regarded in the literature as being MRAC, and the connection of MRAC with other approaches to controller design is touched upon.

A combination of H_∞ and MRAC is done by Hwang and Chen (1988) [30], who designed an MRAC system with on-line minimisation of an H_∞ criterion of the output error, based on least-squares process identification. The combined analysis of the complete adaptive system allows this scheme to be regarded as an MRAC scheme.

Optimal control strategies, such as Linear Quadratic Gaussian methods (LQG), have been linked to MRAC designs by Hwang and Chen (1989) [31]. Here, the difference between the reference model and the process is minimised in an LQG sense, which is accomplished by using a least-square algorithm for process identification and calculating the controller parameter accordingly. This approach has the advantage that the perfect model-matching condition need not be satisfied.

The MRAC approach has evolved from deterministic servo problem and is still used mainly for this type of application (see, for a broad but overview, Butlerr, 1990) [2].

Model reference adaptive controllers can be subdivided into gradient optimised and stability optimised MRAC.

a) Gradient Optimised MRAC

In this method, the classical feedback closed-loop controller is used in the basic loop, but in some cases feedback controller and prefilter can be used. The actual output of the controlled system is compared with the reference model output and the parameters of the controller are modified in such a way that the response of the controlled system follows that of the reference model. Gradient-type model reference systems are treated for the continuous-time. The model reference technique may be used for other purposes (identification) and with other types of model (series, series and parallel).

b) Stability Optimised MRAC

The basic idea of this approach is to introduce a global (Lyapunov or Popov) stability criterion into the design procedure and to choose the adaptive control law in such a way that the requirements of the stability criterion are fulfilled [6].

1.1.2.2 Dual Adaptive Controllers

The principle of dual adaptive controllers is the optimisation of a performance criterion using maximum information gathering for estimation, and generation of optimal control behaviour when designing the adaptive controller. This design principle results in an active learning. The design of optimal dual controllers is very complex and can only be performed numerically i.e. by using the principle of dynamic programming. This principle of controller design implies that the quality of future information has to be taken into account and that the stochastic properties of the process are known a priori. This type of dual adaptive controller is, therefore, not suitable for practical implementation.

1.2 Learning Control Method

Learning control is another method of advanced control. It is based on tuning the input for the system using the experience of previous responses when executing repetitive tasks. Learning, which is defined in [32] as "changes in the system that are adaptive in the sense that they enable the system to do the same task more efficiently and more effectively the next time," Learning control is an iterative approach. These schemes are utilised in application where the system is required to execute a given motion, with a known periodicity. The basic idea behind this technique is to have the system execute the required motion repeatedly. Every time the system executes the motion, the tracking performance of the system is improved by automatic adjusting the control input based on the error signals from previous cycle. With consecutive iterations, the system is expected to eventually "learn" the task, and execute the motion without error.

Learning control has been suggested by several authors. It was originally developed by Arimota and his colleagues for PID type controllers [34]. Some of these are Gu and Loh [35] proposed a multi-step learning control scheme for robotic system where the desired output should be given in the form of generalised momentum instead of position and velocity. Bondi [36] studied the iterative learning control from high-gain feedback point of view, based on which a domain of attraction around the desired trajectory is generated by setting sufficiently high linear feedback gains. Gery, Lee, Carroll and Xie [37] proposes a novel form of iterative learning control system, this approach is successfully designed and applied to the trajectory tracking control problem of parallel link robotic manipulator.

Learning control method has some advantages over adaptive control method, for example, it has a simple algorithm therefore it can be easily applied for high speed application. However, it may suffer from saturation problem for the actuation's after a few cycles depending on the complexity of the trajectory being followed. Kirecci [32] proposes a new leaning control method which is based on Arimoto's iterative method, but, the method is modified in order to prevent saturation of the actuators occurring. In this study Kirecci's method is used for learning control.

1.3 Layout of This Dissertation

Chapter 2 concern with the experimental rigs. A hydraulic robot and a DC servo motor driven system are used for experimental study. The properties of these systems are given in this chapter.

Chapter 3 concern with the system model identification methods and mathematical representation of systems. Within the chapter parameter estimation methods such as recursive least squares method, U-D factorisation method and recursive extended least squares method are discussed.

Chapter 4 presents the design of explicit and implicit self-tuning adaptive control for robotic system. The control laws are established by minimising cost functions. The unknown parameters in a chosen time-series model are estimated by recursive parameter estimation methods and controller parameter are determined. The experimental results for both methods of self-tuning adaptive control are illustrated in this chapter.

Chapter 5 includes the designing and application of model reference adaptive control method. The problem is described and the control methods are developed by minimising the output error (difference between the model output and actual system output) and weighting control input in H^2 – norm space. This model reference adaptive control algorithm is applied to a hydraulic robot and a DC servo system.

Some examples are presented in this chapter to illustrate the performance of the method.

Chapter 6 presents the development of iterative learning control algorithm and the application of this control method to a DC servo drive system. Several examples are given. The experimental results show that despite uncertainties and nonlinearities of the system the proposed controller can achieve high tracking accuracy.

Chapter 7 contains the conclusion and recommendation for further study.

CHAPTER 2

EXPERIMENTAL SYSTEMS

2.1 Introduction

The results of a control study can be analysed in two ways: using simulation program or experimental set-up. Simulation sometimes may cause to misleading because of some assumptions that may not match with the real environment, such as friction etc., however using experimental set-up give more realistic conclusion. Therefore, the proposed control methods are tested on two different experimental set-ups which are readily available in the mechanism laboratory. One of the systems is a hydraulic robot which is designed and manufactured by the members of Mechanical Engineering Department of Gaziantep University. The second one is a DC servo motor driven system that is constructed for the studies of a research supported by Gaziantep University.

Self-tuning and model reference adaptive control methods given in Chapter 4 and 5 respectively are implemented on the hydraulic robot and DC servo system. The learning control method given in Chapter 6 is only implemented on DC servo system.

2.2 Hydraulic Robot

In general, a robot is a multifunctional machine that is composed of links connected by joints into an open kinematic chain. A more technical definition for a robot is given by Robot Institute of America as "A robot is a programmable multifunctional manipulator designed to move material, parts, tools or specialised device through variable programmed motions for the performance of variety of tasks". The actuating systems for robots are energy conversion devices converting electrical, hydraulic or pneumatic power to mechanical power. The basic drive elements are known as actuators.

A manipulator must be able to reach work pieces and tools. This requires a combination of an arm and wrist subassembly, plus a hand, commonly called an end-effector. The robot's workspace of influence is based upon the volume into which the manipulator's arm can deliver the end-effector. A variety of geometric configuration have been studied, however, they are classified into the following four groups:

1) Rectangular manipulator (PPP).

2) Cylindrical manipulator (PRP).
3) Spherical manipulator (RRP).
4) Revolute manipulator (RRR).

The manipulator in our experiment arrangement is in the class of spherical manipulators. The hydraulic robot consists of :

1) Hydraulic actuated spherical manipulator.
2) Macintosh Centris 650 computer.
3) (I/O) interface card.
4) Joint Actuator drives.
5) Potentiometers.

The position of the end-effector of manipulator can be represented in terms base rotation y_1, elevation angle y_2 and reach y_3 (see in Figure 2.1). Range of each joint is gives as $-160^\circ \le y_1 \le 163^\circ$, $-39^\circ \le y_2 \le 15^\circ$ and $42.95\ cm \le y_3 \le 142.75\ cm$ respectively. The hydraulic robot is shown in Figure 2.1.

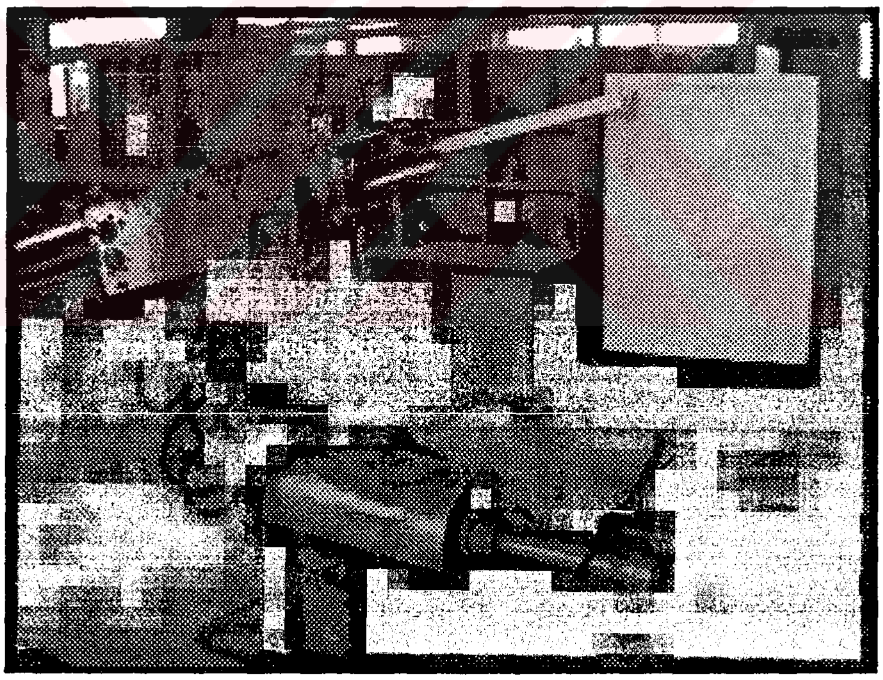


Figure 2.1. Laboratory Set-up for Hydraulic Robot

Each joints of the robot is driven by hydraulic actuators controlled by electro-hydraulic servo valves. Signals for these values are produced by a 25 MHz Apple Macintosh computer and downloaded through the I/O interface card.

The supply of hydraulic pressure and flow to the servo valves are provided by 22 kW hydraulic power unit capable of supplying up to 95 *lit* / min at 0.12 MP. The Apple Macintosh Centris 650 computer is based on a 25 MHz 32 bit Motorala 68040 central processor unit (CPU) and the computer has 8 MB of ram, a harddisc of 230 MB. The computer consists of a high speed, 12 bit D/A converter used for sending analogue control voltages to the multichannel voltage to current (V/A) converter which drives the servo valves.

The control programming was performed in Macintosh Programmer's Workshop (MPW) using C programming language.

2.2.1 (I/O) Interface Card

In order to establish the communication between the computer and the robot a MacADIOS II interface card is used. This card is manufactured by GW Instruments company for Macintosh II series computers to perform general purpose, analog or digital input and output (I/O) operations. The card is installed to one of the NuBus slots located on the computer's main board.

The card has 16 single ended analog input channels which can be converted to 8 differential inputs. The analog voltages between ±10 volts can be read when the card is configured to bipolar mode. The analog to digital (A/D) converter of the card has a resolution of 12 bits. This enables the card to sense analog voltages as small as ± 4.88 mvolts. The A/D conversion speed is 12 ms. The card originally has two digital to analog (D/A) converters. Two more D/A converters is added by installing a secondary card which is called "daughter board" by the manufacturing company. Here, analog voltages between ±10 volts can be produced. The original pair of D/A converters have a resolution of 12 bits, while D/A converters of the daughter board have a resolution of 16 bits. The D/A conversion speed is 9 ms [38, 39].

2.2.2 The Joint Actuator Drives

Each joint of the robot is driven by hydraulic actuators controlled by electro-hydraulic servo valves. Hydraulic actuators are supplied from Mert Akiskan Gücü Ltd (MAG). The actuators are powered by a hydraulic power unit which is also manufactured by the same company. Model Regenventile size NG6 closed loop proportional electro-hydraulic servo valves of BOSCH Ltd are used to control the hydraulic fluid flow to actuators. Specifications of this valve is listed in the following table.

Table 2.1 Specifications of the Bosch electro-hydraulic servo valve model Regenventile size NG6, [38, 40].

General

Construction	Spool valve with steel sleeve.				
Actuation	Proportional solenoid with position control.				
Nominal flow l/min at Dp=35 bar per notch	2	4	12	24	40
Leakage cm^3/min at 100 bar	180	180	350	700	1100
Max. operating pressure	250 bar				
Max. flow	40 l/min				

Electrical

Solenoid current	Max. 2.7 A
Coil resistance R_{20}	2.5 - 2.8 Ω
Power consumption	Max. 25 VA
Position transducer	Supply +15 V/35 mA ; -15 V/25 mA Signal ±10V, Rl 3 10 W

Static/Dynamics

Hysteresis	< 0.2%
Range of inversion	< 0.1%
Response time for signal	< 10 ms (change 0-100%)
Thermal drift	< 1% , at T= 40°C

2.2.3 Position Transducers

Three rotary position potentiometers are used to provide the position feedback. Model RCP 18 of Penny & Giles Company is used.

2.3 DC Servo Drive System

In application, DC motors are extremely versatile drives, capable of reversible operation over a wide range of speeds, with accurate control of the speeds at all times. They can be controlled smoothly from zero speed to full speed in both direction. The system is a direct drive system which means the motors are coupled directly to the load without the use of belts or gears. This enables high speed to be achieved and avoids the problems of backlash it geared drives. DC servo motors are used as actuators in many industrial robots for many reasons, they generate very high torques for their size and are easily controlled by a microprocessor.

The servo motor is controlled by a multi-axis servo motor controller module. This is interfaced by 33 MHz personal computer (PC/AT) using the DOS operating system to perform I/O operations and provide communication interfaces. This kind of control system is known as a bus-based control system. For on-line control, the programming was performed in PASCAL language.

The laboratory set-up is shown in the photo of Figure 2.2 for DC servo system. DC servo system has a Brushless servo motor. The BRU-200 is a digital drive employing a 16-bit microprocessor. Features include high dynamic performance, full tuning / diagnostic. A plug-in personality module matched to the motor and drive unit characteristics fixed parameters for immediate start-up ready for time tuning, if required.

DC Servo system consists of:

1) DC brushless servo drive (BRU-200) which control circuitry utilises a 16 bit microprocessor.
2) A (PC / AT) compatible 80486 16 bit 33 MHz computer.
3) A servo motor controller module (Warwick Comuter Designs).
4) An Electro Craft S-series permanent magnet synchronous motor which has integral mounted optical encoder.

Some specification of this type servo motor is listed in the following table.

Table 2.2 Specifications of DC brushless servo drives [41].

Terminal voltage	110 V
Continuous stall torque	10.2 Nm
Peak torque	19.2 Nm
Rotor moment of inertia	$6.8*10^{-4}\, kgm^2$
Maximum continuous operation speed	3000 rpm
Motor weight, including feedback	13.2 kg
Radial load 12.7mm from shaft end	20 kg
Axial load	10 kg
Thermal resistance	0.48 C° / W

Winding resistance, phase to phase	0.8 (25 $C^{\circ}W$)
Ke voltage constant	90 V / rpm
Kt torque constant	0.76 Nm / A

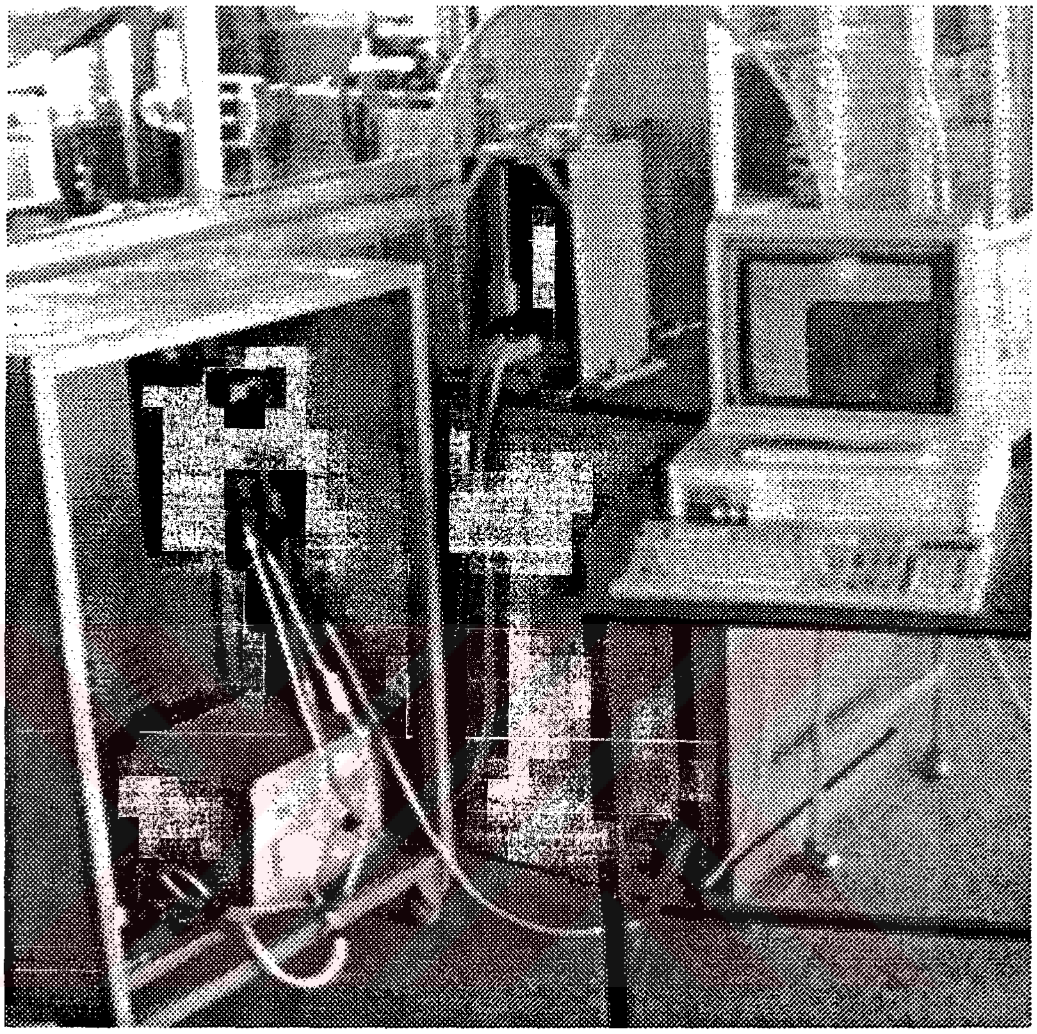


Figure 2.2. Laboratory Set-up for DC servo motor

The following part is discussed the control circuit of the DC servo motor.

2.3.1 Control Circuit

A typical motion control system can be represented in block diagram form as shown in Figure 2. 3. In such systems, the control algorithm can be divided into two levels. The first level of control involves profile selection, the generation of the motion commands and the co-ordination of the motion with overall process control. The next level is the trajectory control. The motion controller closes the position loop around the servo motors by sensing the positions of the motor shafts with incremental encoders. The positions are decoded and compared with the command positions, the difference forming the position errors. The error signals are processed by a stabilising filter. The outputs of the filter are then amplified by the servo drivers before they are applied to the motors.

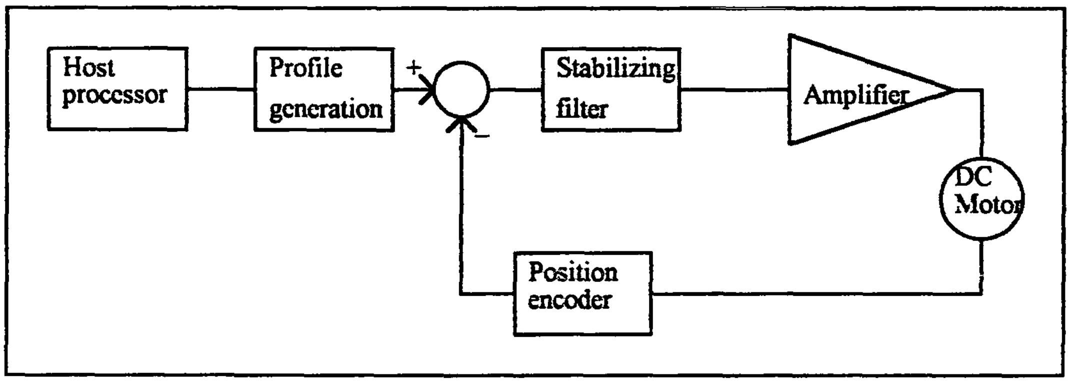


Figure 2.3. Block diagram for a motion control system

In the following sections, some of basic components which constitute the hardware of a general servo system are described. The main part of the system is the controller which converts the digital information for each point of a trajectory into an analog signal. These signals are in the form of voltages which are sent to the servo drives as pulses which are transmitted at regular intervals in order to control them.

The job of the servo drive is to draw electrical energy from the mains (at constant voltage and frequency) and supply electrical energy to each motor at whatever voltage and frequency is necessary to achieve a desired motor output. A servo motor is actually a DC-motor but with the inertia of the rotating parts minimised to provide for rapid changes in acceleration. The feedback element is the other important part of a servo system and this is usually attached to the servo motor. The controller can read the position or velocity from the feedback element and modify the command signal to prevent errors between desired position and actual position or the rotor.

2.3.1.1 Servo Motor Controller

The controller module is the most functional part of a servo system. It converts a desired set of data, usually the positions or velocities of a trajectory, to analog signals mostly in the range of (-10, +10) volts. The outputs from the controller module are used as command signals for the servo motors after amplification by the amplifier.

Modern controller modules generally interface with a computer. They can be inserted into slots available on most computer expansion boards. There are numerous controller modules available commercially most employing proportional-derivative (PD) or proportional-integral (PID) control methods. In the following sections the controller module used for the experimental system is described.

The controller card is a general purpose motor controller capable of controlling up to three axes. It provides position and velocity control for DC, DC-brushless or stepper motors. The system block diagram is given in Figure 2.4. The controller receives the input command from the host processor and position feedback from an incremental encoder with quadrature output. The controller then compares the desired position to the actual position and computes compensated motor commands using a programmable digital filter which continues trying to achieve the desired position until a new command is received from the host processor.

a) (I / O) Interface

The controller card assumes that the host processor is a bank of 8-bit registers and the values of these registers can be changed by the control software at any time. The data in these registers controls the operation of the card. The host processor communicates to the controller over a bi-directional multiplexed 8-bit data bus. Four I/O control lines execute the data transfer.

It is important that the host computer does not attempt to perform too many I/O operation in a single sampling period of the controller. Each I/O operation interrupts the execution of the controller card is internal code for one clock cycle. However, for every unit increase in the sample time register above the minimum limit the user may perform 16 additional I/O operations per sample time.

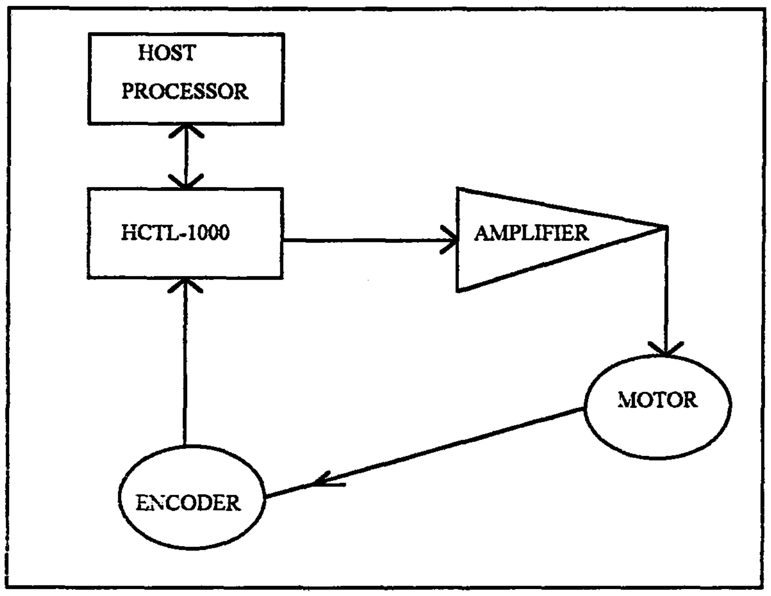


Figure 2.4. System block diagram

b) Encoder Interface

The position of the motor is measured by an optical encoder connected to the shaft. It emits 8000 electrical pulses per revolution. The controller card employs an internal 3-bit state delay filter to remove any noise spikes from the encoder inputs to remain stable for three consecutive clock raising edges for an encoder pulse to be considered valid by the controller is actual position counter. The user should therefore generally avoid creating encoder pulses of less than 3 clock cycles.

c) Digital Filter

The controller card is not a PID controller, however, it uses a digital filter $D(z)$ to compensate for closed loop system stability. The compensation $D(z)$ has the form:

$$D(z) = \frac{K * \left(z - \frac{A}{256}\right)}{4 * \left(z + \frac{B}{256}\right)}$$

where

z = the digital domain operator

K = digital filter gain

A = digital filter zero

B = digital filter pole

The compensation is a first-order lead filter which in combination with sample timer T affects the dynamic step response and stability of the control system. The sample timer T determines the rate at which the control algorithm is executed. All parameters A, B, K and T are 8-bit scalars that can be changed by the user at any time. The implementation of the digital filter in the time domain is according to the rule below:

d) Sampling Period

The content of this register sets the sampling period of the controller. The sampling period is

$$t = 16(T+1)\left(\frac{1}{\text{frequency of the external clock}}\right)$$

where T = register value

There are two options for the frequency of the external clock. Its jumpers can be set either to 1-MHz or 2-MHz. The sample timer has a limit on the minimum allowable sample time depending on the control mode being executed.

With a 2-MHz cock, the sample time can vary from 64 micro seconds to 2048 micro seconds. With 1-MHz clock the sample time can vary 128 micro seconds to 4096 micro seconds. Digital closed systems with slow sampling times have lower stability and lower bandwidth than similar systems with faster sampling times. To keep the system stability and bandwidth as high as possible the controller should be programmed with the fastest sampling time possible.

2.3.2 Pulse Width Modulated (PWM) Amplifier.

The armature voltage of the DC motor is adjusted by a 4 quadrant PWM converter employing servo motor control module. A PWM chip designed for motor drives, in conjunction with few logic chips are used for this purpose. The PWM chip has the ability to convert an analog signal between (-5, +5) volts acquired from the PC into a proportional duty ratio of PWM cycle. Motor current is sensed by the controller module for overcurrent protection- excessive current results in reduction of the duty ratio of PWM cycle.

CHAPTER 3

PROCESS MODEL IDENTIFICATION

3.1 Introduction

In the majority of physical sciences of primary importance in the study of system behaviour, prediction and design is the formulation of a mathematical model which describes the system under consideration. The class and accuracy of a particular model are, however, dependent on it's required application [42]. A model may be obtained by examining the internal structure of the system, although it is often the case that a complete picture cannot be achieved due to an unknown factor, an element which is not directly measurable or an extremely complicated process.

The most common approach taken to find such a model is to make use of data from the system input(s) and output(s) when operating dynamically in its normal range. Unfortunately this data is invariably noisy because of random fluctuations or disturbances, some of which are due to the measuring devices, despite the added noise, to obtain estimates of parameter within a model of the system is called system identification. That is, system identification is the experimental determination of the dynamically behaviour of processes. The system behaviours within a class of mathematical models are determined by using measured signals. The error between the real system and its mathematical model must be as small as possible. Therefore system identification is the field of modelling dynamic systems from experimental data [42].

System identification can be performed off-line or on-line with regard to the application of general purpose digital computers and process computers.

For off-line identification, the process data is first stored in a data storage. Later on the data is transferred to a computer and evaluated. For this type of identification mostly batch processing of the data is applied, that means, the whole data set is evaluated at once.

In the case of on-line identification, the system identification is performed in on-line operation with system. This situation is the real-time processing. Therefore, the data can be distinguished by batch and real processing. Real-time processing, i.e. the data is evaluated immediately after each sample instant, however batch processing, i.e. the data set is evaluated at once after all the measurements have been made.

For adaptive control systems on-line identification in real-time is of primary interest. Since, the parametric process models can be used for controller design. Parametric models involve a finite number of parameter and allow the application of advanced controller design procedures with relatively little computational effort for a wide class of processes. In this chapter three recursive parameter estimation algorithm for parameter adaptive control methods are proposed. The parameter adaptive control methods are based on recursive parameter estimation algorithms. These adaptive controllers are characterised by combining a recursive parameter estimation algorithm for a difference equation process and suitable control algorithm.

In this chapter, the information about the type of mathematical models, how system identification can be done on the system at off-line identification and parameter estimation methods for adaptive control methods are given.

3.2 Model Parametrization

The types of models can be classified as ;

1) Analog Models, which are based on analogies between processes in different areas. For example, a mechanical and an electrical oscillator can be described by a second order linear differential equations constituting a model of some system are solved by using an "analog" or "equivalent" electrical network.

2) Physical Models, which are mostly laboratory-scale units that have the same essential characteristics as the processes they model. In the systems science mathematical models are useful because they can provide a description of a physical process.

Mathematical models can be divided in two ways.

1) Mathematical Modelling:

This is an analytic approach. Basic laws from physics (such as Newton's law and balance equation) are used to describe the dynamic behaviour of a process. For example, exact mathematical model of a mechanical system is derivable by Lagrange, Hamilton and Newton-Euler formulation or by Energy methods. In order to do that all system parameters like masses, inertias, dimensions etc. should be known.

2) System Identification:

This is an experimental approach. It refers to the determination of dynamic models from experimental data. That is, some experiments are performed on the system; It includes the set-up of the identification experiment, i.e. data acquisition, the determination of a suitable form of the model as well as of its parameters, and a validation of the model. We can consider above example for mathematical modelling. In some cases, it can be very difficult to compile these parameters and hence system identification techniques become very useful tools to generate an empirical mathematical model by probing the system with a variety of inputs and relating the output motion to the input signals.

3.2.1 Classification of Mathematical Models

Mathematical models of dynamic system can be classified in various ways. Such models describe how the effect of an input signal will influence the system behaviour at subsequent times. A summary on the ways of classifying dynamic models is given below:

1) Single Input / Single Output (SISO) models and Multi-variable models :

SISO models refer to processes where a description is given of the influence of one input on one output. It should be noted that multi input/single output (MISO) models are in most cases as easy to derive as SISO models, whereas multi input/multi output (MIMO) models are more difficult to determine.

2) Linear model and Non-linear models:

A model is linear if the outputs depends linearly on the input and possible disturbances; otherwise it is non-linear.

3) Parameter models and Non parameter models;

Models may be divided into two main classes, parametric and nonparametric. With parametric models the system order must be specified such that errors are eliminated, in nonparametric models, however, order specification is unnecessary. But nonparametric models are infinite dimensional and hence it becomes virtually impossible to match the output from a model to that of the system. Similar difficulties arise with noise signals, which are not averaged out in the nonparametric case.

4) Time Invariant models-Time Varying models;

Time invariant models are certainly the more common. For time varying models special identification method are needed. In such cases where a model has parameters that change with time, one often speaks about tracking or real-time identification when estimating the parameters.

5) Discrete time models-Continuous time models;

A discrete time model describes the relation between inputs and outputs at discrete time points. It will be assumed that these points are equidistant and the time between two points will be used as time unit. Therefore the time *t* will take values' *1,2,3*... for discrete time models. Note that a continuous time model, such as a differential equation, can very well be fitted to discrete time data.

6) Deterministic models-Stochastic models;

For a deterministic model the output can be exactly calculated as soon as the input signal is known. In contrast, a stochastic model contains random term that makes such an exact calculation impossible. The random terms can be seen as a description of disturbances.

7) Time domain models-Frequency domain models;

Typical examples of time domain models are differential and difference equations, while a spectral density and a bode plots are examples of frequency domain models.

8) Lumped models-Distributed parameter models;

Ordinary linear or non-linear differential equations are used to describe the behaviour of the system in lumped models. In other hand, the dynamic behaviour of the system is described by partial differential equations of parabolic, hyperbolic and elliptic type in distributed parameter models.

3.3 How System Identification is Applied

An identification experiment is performed by exciting the system and observing its input and output over a time interval. These signals are normally recorded in a computer mass storage for subsequent "information processing". We then try to fit a parameter model of the process to the recorded input and output sequences. The first step is to determine an appropriate form of the model (typically a linear difference equation of a certain order). As a second step some statistically based method is used to estimate the unknown parameters of the model (such as the coefficients in the difference equation). In practice, the estimations of structure and parameters are often done iteratively. The model obtained is then tested to see whether it is an appropriate representation of the system.

The result of an identification experiment will be influenced by the following four factors.

1) The system:

The physical reality that provides the experimental data will generally be referred to as the process. In order to perform a theoretical analysis of an identification it is necessary to introduce assumptions on the data. In practice, where real data are used, the system is unknown and can even be an idealisation. For simulation data, however, it is not only known but also used directly for the data generation in the computer. We will use the system concept only for investigation of how different identification methods behave under various circumstances. For that purpose the concept of a "system" will be most useful.

2) The Model Structure:

Sometimes we will use non-parametric models. Such a model is described by a curve function or table. A step response is an example. However, in many cases it is relevant to deal with parametric models. Such models are characterised by a parameter vector.

3) The Identification Method:

A large variety of identification methods have been proposed in the literature[43, 44].

4) The Experimental Condition:

It is a description of how the identification experiment is carried out. This includes the selection and generation of the input signal, possible feedback loops in the process, the sampling interval, prefiltering of the data prior to estimation of the parameter.

3.4 Process Model

The main aim of identification is to accurately model the system under consideration. Certain model structures must therefore be defined and estimations made of the parameters contained within each particular model. It must also be flexible and lead to simple parameter estimation procedures.

A single input / single output (SISO) model is chosen to represent the input-output measurement for system in order to implement the proposed adaptive control method. A linear discrete time-series model of Auto Regressive Moving Average (ARMAX) can be assumed for variables.

$$A(z^{-1})y(t) = z^{-d}B(z^{-1})u(t) + C(z^{-1})e(t) \tag{3.1}$$

Where the sampling period (T) has been omitted in the arguments; the argument t refers to the sampling instant $t = n,\ n+1, \ldots$ integer $n > 0$, n specifies the order of the model, $u(t)$ is control input for the system and $y(t)$ is system output at time tT. $e(t)$ represents Gaussian white noise sequence. Moreover, residual $e(t)$ represents independent zero-mean samples that are not influenced by the input. The positive integer d represents the time delay dT between the input and output of the system. Time delay is primarily the delay between the change in $u(t)$ and its effect on $y(t)$. By introducing a delay operator z^{-1} that is $z^{-1}y(t) = y(t-1)$. $A(z^{-1})$, $B(z^{-1})$ and $C(z^{-1})$ of Equation (3.1) are polynomials of z^{-1} defined as follows;

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} +, \ldots, + a_{n_a} z^{-n_a}$$

$$B(z^{-1}) = b_0 + b_1 z^{-1} +, \ldots, + b_{n_b} z^{-n_b} \qquad b_0 \neq 0$$

$$C(z^{-1}) = 1 + c_1 z^{-1} +, \ldots., + c_{n_c} z^{-n_c}$$

The constant n_a, n_b and n_c are the orders of polynomials $A(z^{-1})$, $B(z^{-1})$ and $C(z^{-1})$ respectively. The roots of the polynomials $C(z^{-1})$ lie within the unit circle in the z-plane. The degree of $C(z^{-1})$, the noise model, is less easy to determine from physical considerations. If there are very low levels of noise, then assume $n_c = 0$. Otherwise if the noise appears coloured, then begin by choosing $n_c = 1$. It may be necessary to use $n_c = 2$ but this is the maximum value that should be needed in practice [47]. Once these constant and delay times are known, then measurements are used to estimate the unknown parameter a_1 to a_{n_a}, b_0 to b_{n_b} and c_1 to c_{n_c}. An accurate model, however depends on the proper choice of n_a, n_b, n_c and d.

If the same input $u(t)$ is then applied to both the system and mathematical model of the system (see Figure 3.1). The outputs of the system and model can be compared giving rise to an error which is dependent on how poor the parameter estimates are, or rather the error gives an indication of how close to the actual system the mathematical representation. It must be remembered that of primary importance is the choice of mathematical model structure, or how many unknown parameters are included. If the structure is incorrect or not enough parameters are used, no matter how comprehensively the remaining parameters are estimated a large error between system and model outputs will most likely occur. Unfortunately, in practice the system output is corrupted by noise due to disturbances, measurement devices and interfacing. This means that extremely complicated models are not necessarily of great use because variations in some of the parameters may have less effect on the error value than the noise signal.

If $\varepsilon(t)$ is the error value (the prediction error), an attempt to obtain a better mathematical model, i.e. better estimates of unknown parameters by trying to minimise the error would be of little direct use as such a procedure would result in a large "negative" error. The square of the error $\varepsilon(t)^2$ rather than the error itself is used as the basis for the vast majority of identification procedures. The general idea is to sum a number of samples of the error signal squared and to use this sum of squares as a cost function to be minimised by selection of unknown parameter estimates. The effect of noise on the model can by this mean be reduced considerably [44].

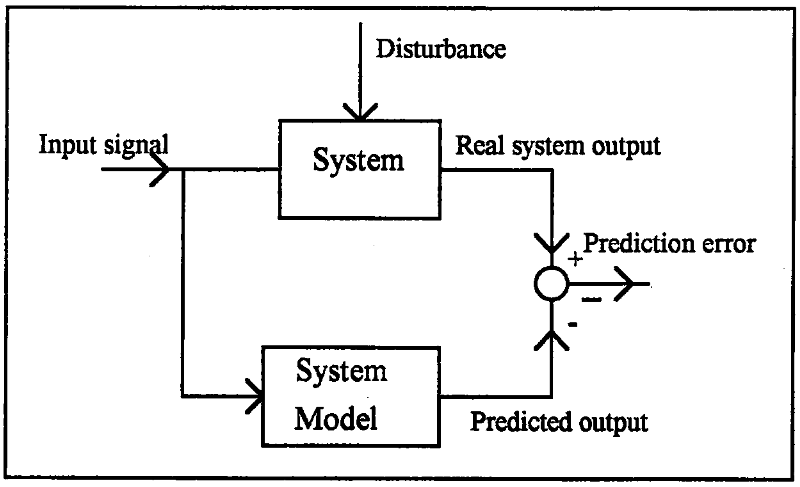


Figure 3.1 Propagation of prediction errors in estimation

3.5 Parameter Estimation Methods

Several powerful methods to estimate model parameters have been described in the literature [44, 45, 46, 47, 48]. These are least square method, recursive least square method, instrumental variable, recursive extended least squares method and U-D factorisation method and etc. The first and perhaps simplest parameter estimation method is the least-squares method. Least-squared error method may give biased estimates. This is inconsequential when the model is used with adaptive controller that inherently compensates for the bias. However, if some physical parameters such as masses, inertias, or friction coefficients in the system are estimated, then the bias must be corrected to avoid erroneous estimates. Recursive least square method, recursive extended least square and U-D factorisation method are used in this work.

3.5.1 Recursive Least Squares Method (RLS)

Assuming that the structure of the system to be controlled is either known or can be found, the objective of the identification exercise is then to find values for the parameters within a model of similar structure to the system, such that it behaves as nearly as possible like the real system for a set of inputs signals.

Variations of the system parameters occur over a period of time, possibly due to unmodelled variation or system modification where parameter variations are small in magnitude or take place over a long period of time, the controller operating on the system need only be retired occasionally to ensure reasonable performance. However, the variations can be large in magnitude and occur quite rapidly, and almost they are unpredictable. In these cases it is often necessary and/or desirable to modify the controller with the changes that have occurred in the system, thus providing an adaptive control scheme. The controller modifications need an updated version of the system model and hence new estimated parameters are required at any time instant, t.

A basic requirement of the recursive algorithm is that the information stored be only of a finite amount. Hence, once the data is of a certain age i.e. once a number of time periods have elapsed, it is automatically discarded. At any one time instant therefore, a window of past and present input/output data is taken into account. Indeed this is true for recursive forms of all the identification techniques previously considered.

In this scheme new input/output data become available at each sample interval. The model based on past information is used to obtain an estimate $y(t)$ of the current output. This is then compared with the observed output $y(t)$ to generate an error $e(t)$. This generates an update to the model that corrects $(t-1)$ to the new value (t). This recursive "predictor-corrector" form allows significant saving in computation. Instead of recalculating the least squares estimate in its entirety, requiring the storage of all previous data, it is both efficient and elegant to merely store the "old" estimate calculated at time $t-1$, denoted by $(t-1)$, and to obtain the "new" estimates (t) by an updating step involving the new observation.

Consistent parameter estimates for RLS can be obtained if the noise condition satisfies the relation $C(z^{-1})=1$. In order to further consider the recursive form of least-square (RLS) parameter estimation it must be remembered that the system output at time instant t can be rewritten as in Equation (3.1).

$$y(t)=\phi^T(t-1)\theta+e(t) \tag{3.2}$$

$$\phi^T(t-1)=\left[-y(t-1)\ -y(t-2),\dots,-y(t-n_a),\ u(t-d)\ u(t-d-1),\dots,u(t-d-n_b)\right] \tag{3.3}$$

$$\theta^T=\left[a_1\ a_2,\dots,\ a_{n_a},\ b_0\ b_1,\dots,\ b_{n_b}\right] \tag{3.4}$$

Where $\phi^T(t-1)$ is a data vector θ^T is a parameter vector. Also, let the vector of parameter estimates obtained at time instant t be denoted by $\hat{\theta}$, such that at that instant the model prediction error is

$$\varepsilon(t)=y(t)-\phi^T(t-1)\hat{\theta}(t-1) \tag{3.5}$$

The idea behind this is that if the estimated parameter vector $\hat{\theta}$ is very close to its true value θ, the error $\varepsilon(t)$ should be very small. A better $\hat{\theta}$ can however be obtained by taking into account the magnitude of difference, i.e. just how large $\varepsilon(t)$ is, such that a new version of $\hat{\theta}$ can be found from;

$$\hat{\theta}(t)=\hat{\theta}(t-1)+K(t)\varepsilon(t) \tag{3.6}$$

$K(t)$ can be chosen to improve the estimated values, hopefully by means of an optimal path. The error between the true and the estimated parameter values is then defined as;

$$\eta(t)=\theta-\hat{\theta}(t) \tag{3.7}$$

such that

$$\eta(t) = \left[I - K(t)\phi^T(t-1)\right]\eta(t-1) - K(t)e(t) \tag{3.8}$$

Where I is the unity matrix. If $K(t)$ is chosen to contain elements of high magnitude, although this results in rapid updating it also leads to the disturbance term $e(t)$ causing large perturbations. A choice of $K(t)$, the Kalman gain, to contain elements of small magnitude, result in better noise rejection but a slow response in terms of parameter updating. In practice $K(t)$ is selected in terms of the covariance matrix of the error, as a time varying gain.

Let the covariance matrix formed by the parameter estimates at time instant t, be $P(t)$. The elements of the covariance matrix are minimised by choosing the Kalman gain vector as;

$$K(t) = \frac{P(t-1)\phi(t-1)}{\gamma + \phi^T(t-1)P(t-1)\phi(t-1)} \tag{3.9}$$

$$P(t) = \left[I - K(t)\phi^T(t-1)\right]P(t-1) \tag{3.10}$$

It is common practice with many adaptive controllers to use γ as an exponential forgetting factor such that new data is relatively important and its importance declines exponentially. For this purpose the value for $P(t)$ must also be divided by γ;

$$P(t) = \left[I - \frac{P(t-1)\phi(t-1)\phi^T(t-1)}{\gamma + \phi^T(t-1)P(t-1)\phi(t-1)}\right]\frac{P(t-1)}{\gamma} \tag{3.11}$$

However, where data is not persistently exciting enough the algorithm may "blow up" i.e. The parameter estimates tend to infinity, therefore the factor γ can be made variable and depend on the prediction error in order to "wake up" the estimator when necessary.

The RLS algorithm is follows:

At time step t

Step 1: Form $\phi(t-1)$ using new data.

$$\phi^T(t-1) = \left[-y(t-1)\ \ -y(t-2),\ldots,-y(t-n_a),\ u(t-d)\ u(t-d-1),\ldots,u(t-d-n_b)\right]$$

Step 2: Form $\varepsilon(t)$ as before:

$$\varepsilon(t)=y(t)-\phi^T(t-1)\hat{\theta}(t-1) \tag{3.12}$$

Step 3: Form $K(t)$

$$K(t)=\frac{P(t-1)\phi(t-1)}{\gamma+\phi^T(t-1)P(t-1)\phi(t-1)} \tag{3.13}$$

Step 4: Update $\hat{\theta}(t-1)$ to obtain $\hat{\theta}(t)$

$$\hat{\theta}(t)=\hat{\theta}(t-1)+K(t)\varepsilon(t) \tag{3.14}$$

Step 5: Form $P(t)$

$$P(t)=\left[I-\frac{P(t-1)\phi(t-1)\phi^T(t-1)}{\gamma+\phi^T(t-1)P(t-1)\phi(t-1)}\right]\frac{P(t-1)}{\gamma} \tag{3.15}$$

Step 6: Wait for the next time step to elapse and loop back to Step 1.

A final point to note is that the equations given for the update of $K(t)$ and $P(t)$ may result in divergent values because of numerical problems. Hence, it is best to use a numerically stable updating method such as square root extraction or U-D factorisation in order to achieve a working algorithm.

3.5.2 U-D Factorisation Method

The recursive least square (RLS) algorithm is based upon the Matrix Inversion Lemma [46, 47] and represents the basic formulation of the least squares recursion. However, when the basic form of the recursion is inadequate due to either

1) Restricted computational precision.
2) Restricted computational time.

The matrix inversion lemma version of RLS can be numerically ill-conditioned. In particular, rounding errors can accumulate and cause errors to occur in both the estimated parameters and the covariance matrix. Apart from the loss of accuracy, the calculated covariance matrix may become negative semi-definite with large. In a similar vein, numerical problems can occur during estimator runs involving many tens of thousands of recursions in which normally negligible round-up errors are allowed to accumulate.

The key idea in numerically robust RLS algorithms is to replace the recursive calculation of a square matrix by a recursive calculation of a factor of the square matrix. For example, the covariance matrix $P(t)$ is symmetric and can be written in the factored form

$$P(t) = S(t)S^T(t) \tag{3.16}$$

Where $S(t)$ is a triangular matrix and corresponds to the "square root" of the $P(t)$ matrix.

There are numerous factorisation algorithms for computing the RLS recursion. A widely used version in adaptive control is the Bierman U-D factorisation algorithm [47]. This method operates upon a factored form of the covariance matrix $P(t)$. The Bierman algorithm emerged from Kalman filtering work.

The main idea in the U-D algorithm is to assume that $P(t)$ can be factored as

$$P(t) = U_p(t)D_g(t)U_p^T(t) \tag{3.17}$$

where $U_p(t)$ is an upper triangular matrix of the form

$$U_p(t) = \begin{bmatrix} 1 & u_{12}(t) & . & . & u_{1m}(t) \\ 0 & 1 & u_{23}(t) & . & u_{2m}(t) \\ . & . & . & . & . \\ . & . & . & . & u_{m-1,m}(t) \\ 0 & 0 & . & . & 1 \end{bmatrix} \tag{3.18}$$

and $D_g(t)$ is a diagonal matrix given by

$$D_g(t) = \begin{bmatrix} d_1(t) & 0 & . & . & 0 \\ 0 & d_2(t) & 0 & . & 0 \\ . & . & . & . & . \\ . & . & . & . & 0 \\ 0 & . & . & . & d_m(t) \end{bmatrix} \tag{3.19}$$

Where m represents the number of the unknown model parameters. The essence of the U-D factorisation is to recursively compute $D_g(t)$ and $U_p(t)$ from $D_g(t-1)$ and $U_p(t-1)$. The Kalman gain $K(t)$ is then computed directly and the parameter estimates $\hat{\theta}(t-1)$ are updated in the normal way. The algorithm is as follows:

Algorithm: U-D Factorisation

If the data vector is denoted by the vector $\phi(t-1)$ with m entries corresponding to unknown parameters, the U-D update at time t is given by;

Step 1: Compute the m vectors f and g

$$f = U_p^T(t-1)\phi(t-1) \tag{3.20}$$

$$g = D_g(t-1)f \tag{3.21}$$

and set $\beta_0 = \gamma$, the forgetting factor.

Step 2: For $j = 1$ to m repeat (i) and (ii)

(i) Compute

$$\beta_j = \beta_{j-1} + f_j g_j \tag{3.22}$$

$$d_j(t) = \frac{\beta_{j-1} d_j(t-1)}{\beta_j \gamma} \tag{3.23}$$

$$v_j = g_i \tag{3.24}$$

$$\mu_j = \frac{-f_j}{\beta_{j-1}} \tag{3.25}$$

(ii) For $i = 1$ to $j-1$ $(j > 1)$ compute

$$u_{ij} = u_{ij}(t-1) + v_i \mu_j \tag{3.26}$$

$$v_i = v_i + u_{ij}(t-1)v_j \tag{3.27}$$

Where u and d refer the upper triangle and diagonal matrix.

Step 3: Compute

$$\hat{K}(t) = [\, v_1, \ldots, v_m \,]^T \tag{3.28}$$

$$K(t) = \hat{K}(t) / \beta_m$$

Step 4: Update the parameter estimates

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\varepsilon(t) \tag{3.29}$$

Where $K(t)$ is the Kalman gain vector $P(t)\phi(t-1)$ and as usual

$$\varepsilon(t) = y(t) - \phi^T(t-1)\hat{\theta}(t-1) \tag{3.30}$$

Step 5: $t \to t+1$ and loop back to step 1.

3.5.3 Recursive Extended Least Squares Method

In recursive Least Squares estimation the model assumed is

$$A(z^{-1})y(t) = z^{-d}B(z^{-1})u(t) + e(t) \tag{3.31}$$

when the system output is corrupted by coloured noise, however, a more suitable model is in Equation (3.1). In general, $C(z^{-1})$ cannot be assumed to be unity. In order to estimate $C(z^{-1})$ in the general case, knowledge of $\hat{e}(t-1), \ldots, \hat{e}(t-n_c)$ is required. However, $\hat{e}(t)$ is an unobservable error process. A number of estimation procedures exist, however, which replace $\hat{e}(t)$ by an estimate, usually taken to be the prediction error $\varepsilon(t)$.The recursive extended least square's estimation algorithm is the same as recursive estimation algorithm, but data and parameter vector are different. The algorithm is as follows:

Algorithm: Recursive Extended Least Squares

At time step t

Step 1: Form $\phi(t-1)$ using new data.

$$\phi^T(t-1) = [-y(t-1), \ldots, -y(t-n_a),\ u(t-d)\ u(t-d-1), \ldots, u(t-d-n_b),\ e(t-1), \ldots, e(t-n_c)]$$

Step 2: Form $\varepsilon(t)$ as before:

$$\varepsilon(t)=y(t)-\phi^T(t-1)\hat{\theta}(t-1) \tag{3.32}$$

Step 3: Form $K(t)$

$$K(t)=\frac{P(t-1)\phi(t-1)}{\gamma+\phi^T(t-1)P(t-1)\phi(t-1)} \tag{3.33}$$

Step 4: Update $\hat{\theta}(t-1)$ to obtain $\hat{\theta}(t)$

$$\hat{\theta}(t)=\hat{\theta}(t-1)+K(t)\varepsilon(t) \tag{3.34}$$

Step 5: Form $P(t)$

$$P(t)=\left[I-\frac{P(t-1)\phi(t-1)\phi^T(t-1)}{\gamma+\phi^T(t-1)P(t-1)\phi(t-1)}\right]\frac{P(t-1)}{\gamma} \tag{3.35}$$

Step 6: Wait for the next time step to elapse and loop back to Step 1.

3.6 Starting Parameter Adaptive Controllers and Choice of Free Design Parameters

3.6.1 Pre-identification

As the parameter adaptive controllers are based on process parameter estimation, the parameter estimation method and free parameters must be chosen properly. For an unknown process it is therefore recommended that a process identification is first performed.

3.6.2 Choice of Design Parameters

To start the parameter adaptive control algorithms the following parameters must initially be specified.

1) Sample time (T).
2) Process model order (n).
3) Process model dead time (d).
4) Forgetting factor (γ).

3.7.3 Initialising the Estimator

The choices of values for the data vector, parameter vector and covariance matrix for RLS or diagonal and upper triangle matrix for U-D factorisation method can be performed as follows;

1) For data vector at the first time step $\phi(0)$:

The usual way to fill the data vector with initial values is to commence sampling the information for a few time steps before the recursive estimator is started.

2) Initial values of the covariance matrix for RLS or diagonal matrix and upper triangle matrix for U-D factorisation method ($P(0) = rI_m$, $D_g(0) = rI_m$ and $U_p(0) = I_m$):

The size of $P(t)$, $D_g(t)$ and $U_p(t)$ reflects our uncertainty concerning the unknown parameters. If we have no prior knowledge of the system parameters in the model, then a large initial covariance, diagonal and upper triangle would reflect this. Conversely, if the initial parameters $\hat{\theta}(t)$ are known to be close to the true values, then a small covariance, diagonal and upper triangle matrix should be used.

If $P(0)$, $D_g(0)$ and $U_p(0)$ are chosen large, therefore its influence upon $P(t)$, $D_g(t)$ and $U_p(t)$ are much less than the information in the data. Conversely, If $P(0)$, $D_g(0)$ and $U_p(0)$ are small, its influent upon $P(t)$, $D_g(t)$ and $U_p(t)$ are correspondingly greater. In general, the initial values of the upper triangle matrix is chosen as $U_p(0) = I_m$ (I_m is the unity matrix). Typically,

For large $P(0)$ and $D_g(0) \rightarrow \quad 100 \le r \le 1000$
For small $P(0)$ and $D_g(0) \rightarrow \quad 1 \le r \le 10$

3) Initial parameter values:

a) Sometimes prior knowledge will exist which enables good in to be selected for the estimated coefficients of $A(z^{-1}), B(z^{-1})$ and $C(z^{-1})$.

b) A technique that is useful when no prior knowledge exists is to assume that the system is a single integrator with unit gain. At time $t=0$ initial value for parameters;
$a_i = 0, \quad b_0 = 1, \quad b_i = 0 \quad (i \neq 0)$

The initial values of other parameters (i.e. noise and disturbance coefficients) are usually set at zero. In general however, the choice of initial parameter estimates is not crucial to convergence behaviour.

4) Forgetting factors (γ):

Most popular way of getting a recursive estimator to adapt is the forgetting factor technique. A forgetting factor is a number among 0 and 1 that is used to progressively reduce the emphasis placed on past information. The idea of a forgetting factor can best be understood by considering the way in which information is weighed in the least squares cost function. The forgetting mechanism, therefore, uses the influence of γ to progressively reduce the importance given to old data. Values commonly used in practice lie between 0.95 and 1.

CHAPTER 4

SELF-TUNING ADAPTIVE CONTROL

4.1 Introduction

In an adaptive system it is assumed that the controller parameters are adjusted all the time. This implies that the controller parameters follow changes in the process. However, it is difficult to analyse the convergence and stability properties of such systems. To simplify the problem it can be assumed that the process has constant but unknown parameters. When the process is known, the design procedure specifies a set of desired controller parameters. The adaptive controller should converge to these parameter values even when the process is unknown. A controller with this property is called self-tuning, since it automatically tunes the controller to the desired performance.

Self tuning control is one approach to the automatic tuning problem. It can be viewed either as a tuning aid for control laws that are more complex than PID but which have fixed parameters, or as a means by which a time-varying process can be controlled in a consistent way. The idea behind the self-tuning is to adjust the controller settings automatically based on the measured input/output behaviour of real system.

The controller of a real system for adaptive control can be divided into to two parts [49]. These are primary and secondary controller. The primary controller is constructed so as to cause the cancellation of non-linear terms in the dynamic's equation of the system which renders the overall system linear. On the other hand, secondary controller is constructed to compensate for the imperfect cancellations of non-linear terms and the effects of modelling errors and disturbance.

The complete dynamic equation for the system is not known accurately. If some physical parameters in the dynamic model of the system are exactly known and can be computed. They should be used in the construction of the primary controller. If no accurate a priori information about the dynamics is available. A controller with self-tuning can be designed so that the system will track the specified desired trajectory. It should be used in the construction of secondary controller.

Self-tuning adaptive controller perform three basic tasks: information gathering of the present process behaviour (modelling); control performance criterion optimisation (design of a controller); and adjustment of the controller (implementation).

1) Information gathering of the process implies the continuous determination of the actual condition of the process output. Suitable ways are identification and parameter estimation of a process model and signal estimation of non measurable signals for example, noise signals in the stochastic case. Various types of model identification adaptive controller can be distinguished, depending on the information gathered and the method of estimation.

2) Performance criterion optimisation implies the calculation of the control loop performance and the decision as to how the controller parameter set and the replacement of the old parameters in the control loop.

3) Adjustment of the controller implies the calculation of the new controller parameter set and the replacement of the old parameter in the control loop. The implementation of a self-tuning controller will be assumed to be carried out in terms of a digital algorithm. The parameters produced by the design procedure are then inserted directly into the digital algorithm.

The modelling, design and implementation tasks that are automated within a self-tuning adaptive control is shown in Figure 4.1. The self-tuning controller is based on the idea of separating the estimation of unknown parameters from the design of the controller. The unknown parameters are estimated on-line, using a recursive parameter estimation method. The estimated parameters are treated as if they are true: i.e., the uncertainties of the estimates are not considered. This is called the certainty equivalence principle [4]. Many different estimation schemes can be used such as stochastic approximation, least squares, extended and generalised least squares, instrumental variable, maximum likelihood and U-D factorisation method.

Self-tuning adaptive control has two main forms, explicit (indirect) and implicit (direct) control method [6, 50]. Both schemes require a valid digital model to represent the system under control.

In this chapter both methods of self-tuning adaptive control have been investigated and experimentally tested on hydraulic robot and on the DC servo system to examine their effect on trajectory tracing error. Very successful results have been obtained for each system. Different trajectories are planned using MOTDES [32] for implementation .

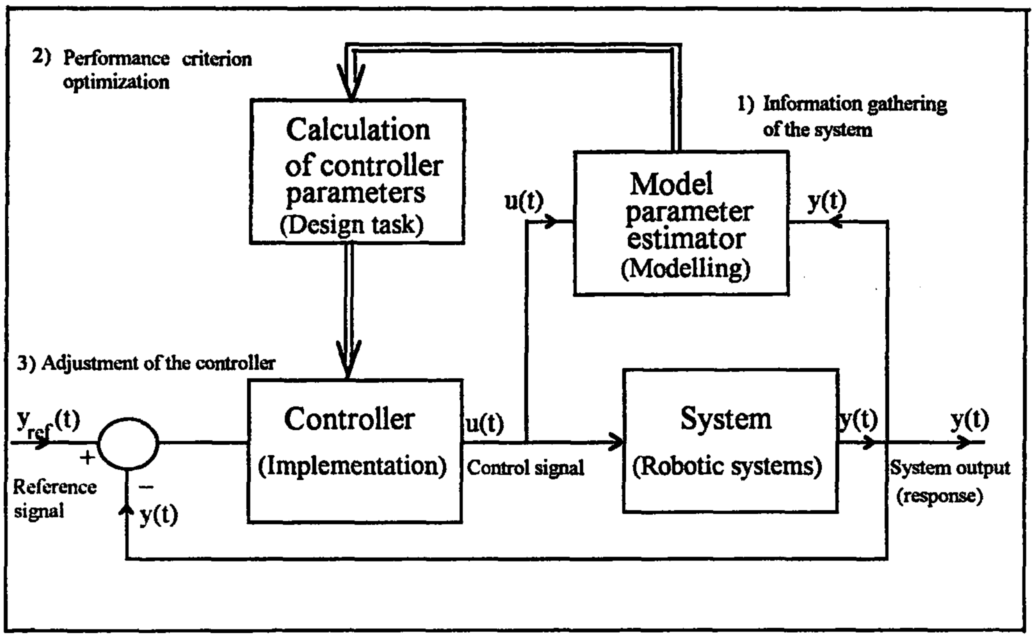

2) Performance criterion optimization
Calculation of controller parameters (Design task)
1) Information gathering of the system
Model parameter estimator (Modelling)
u(t)
y(t)
3) Adjustment of the controller
$y_{ref}(t)$
+
−
Reference signal
y(t)
Controller (Implementation)
u(t)
Control signal
System (Robotic systems)
y(t)
y(t)
System output (response)

Figure 4.1 Self-tuning adaptive controller structure

The theory of the performance optimisation criterion is based on the principle of Generalised Minimum Variance Adaptive Control [24, 49, 51, 52]. For these reasons, the self-tuning adaptive controller is constructed as a linear quadratic criterion with Gaussian noise (LQG) controller by minimising a well defined single-stage criterion. However, the control theory presented in this chapter is an enhancement to that reported in [24] with the following sailed modifications:

1) To determine sample time, forgetting factor, the model order and delay of the difference equation describing these systems, experimentally.

2) The system can achieve excellent tracking performance even when the process under control exhibits non-minimum phase property.

3) The control signal is included in the variances of generalised cost functions. Appropriate weighting factors are introduced to restrain the magnitude and oscillations of the control signal.

In order to implement any method of the proposed self-tuning adaptive control, a digitised model is required and, also a single-input / single-output (SISO) model is chosen to represent the input-output measurements of the system. Under a stochastic measurement environment with uncertainties, such as random disturbances etc., a linear discrete time-series of an Auto regressive Moving Average Exogenous (ARMAX) model is used to represent the system under the control. This model is given Chapter 3, such as

$$A(z^{-1})y(t) = z^{-d}B(z^{-1})u(t) + C(z^{-1})e(t) \tag{4.1}$$

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} +,\dots,+ a_{n_a} z^{-n_a}$$

$$B(z^{-1}) = b_0 + b_1 z^{-1} +,\dots,+ b_{n_b} z^{-n_b} \qquad b_0 \neq 0$$

$$C(z^{-1}) = 1 + c_1 z^{-1} +,\dots,+ c_{n_c} z^{-n_c}$$

Where n_a, n_b and n_c is the order of $A(z^{-1}), B(z^{-1})$ and $C(z^{-1})$ polynomials respectively.

4.2 Self-Tuning Adaptive Controller Design by the Explicit Method

The structure of explicit self-tuning adaptive control loop is shown in Figure 4.2, where the parameters of the system model are directly identified. This gives an indirect adaptive algorithm. The controller parameters are not updated directly, but rather indirectly, which are computed based on the selected control law (such as minimum variance control, pole assignment control etc.) , and on the digital system model.

4.2.1 Recursive Parameter Estimation

A recursive parameter estimator is at the heart of the self-tuning algorithms. This parameter estimation method is given in detail in Chapter 3. The main idea is to update continuously (recursively) the parameters in the digital system model. A basic requirement for this algorithm is that the information is stored only a finite amount. And to be useful in self-tuning control, the parameter estimation scheme should be iterative, allowing the estimated model of the system to be updated at each sample interval as data become available.

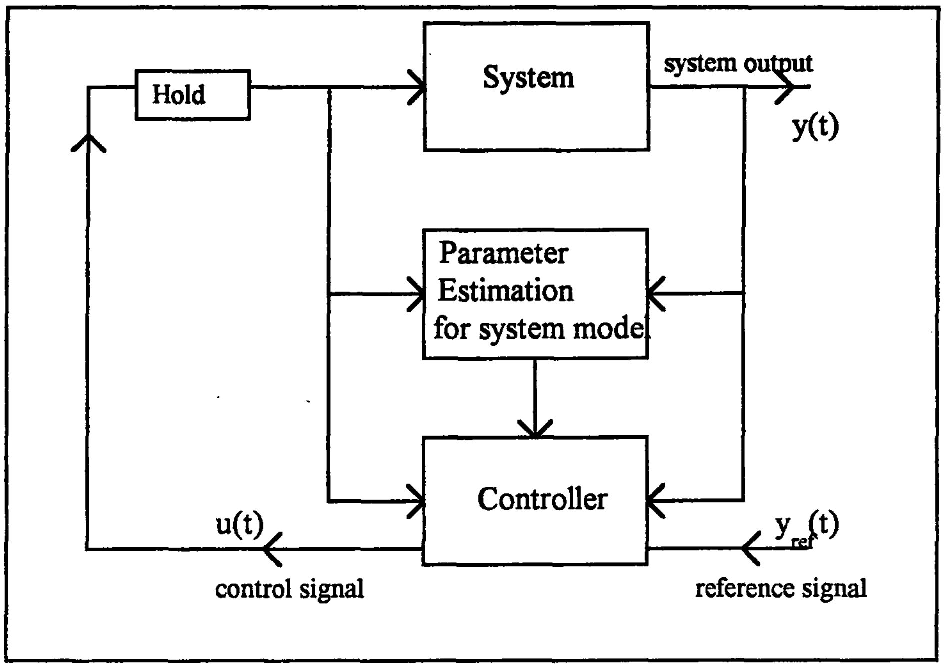


Figure 4.2 Self-tuning adaptive control by explicit method

The parameter estimates for RLS can be obtained if the noise condition satisfies the relation $C(z^{-1}) = 1$ in Equation (4.1). Under this condition, Equation (4.1) can be rewritten explicitly in the following form:

$$y(t) = -a_1 y(t-1) -,\dots,-a_{n_a} y(t-n_a) + b_0 u(t-1) +,\dots,+ b_{n_b} u(t-1-n_b) + e(t) \qquad (4.2)$$

In Equation (4.2), residual $e(t)$ signifies equation error that represents independent, zero-mean samples that are not influenced by the control input signal. This is called an auto regressive exogenous difference equation model (an ARX-model) [44]. This model structure can be viewed as a linear regression. To see this, rewrite the model as

$$y(t) = \phi^T (t-1)\theta + e(t) \qquad (4.3)$$

where

$$\phi^T (t-1) = [-y(t-1) \ -y(t-2),\dots,-y(t-n_a) \ u(t-1) \ u(t-2),\dots,u(t-1-n_b)]$$

$$\theta^T = [a_1 \ a_2,\dots,a_{n_a} \ b_0 \ b_1,\dots,b_{n_b}]$$

Where θ^T is a vector consisting of the parameters to be estimated, $\phi^T(t-1)$ is a vector that includes previous control inputs and system outputs. The unknown parameters of model in Equation (4.3) can be estimated recursively by on-line calculations using RLS method or U-D factorisation method. Let the vector of parameter estimates obtained at time instant t be denoted by $\hat{\theta}(t)$. After that, these parameter estimates are substituted into Equation (4.3) to obtain a model for the controller design in the explicit self-tuning adaptive control. Such as:

$$\hat{A}(z^{-1})y(t) = z^{-1}\hat{B}(z^{-1})u(t) + e(t) \tag{4.4}$$

Above the difference equation model with known numerical values for the parameters is next used to design the proposed control method.

4.2.2 Controller Design

The goal of controller is to get the control signal that is able to drive the system to follow the desired trajectory. The explicit self-tuning adaptive control is designed on the basis of the linearized model that may exhibit a non minimum phase or minimum phase type behaviour. If the system model is linear and the criterion (minimum of performance criterion) is quadratic subject to gaussian disturbances, the problem is often called the linear quadratic gaussian problem [49, 50]. The controller can be constructed by solving a LQG-problem. A good tracking of the desired trajectory can be achieved by using this controller method.

4.2.2.1 Controller Design Based on Minimum of Performance Criterion

The explicit self-tuning adaptive controller is designed so as to make the output of the system track the desired trajectory by minimising a performance criterion while the energy associated with the input is kept at a minimum. These goals are achieved by choosing the following convenient quadratic function which is the variance of generalised cost function in the position error and energy of the control input appropriately weighted [24, 49, 51, 52]:

$$J = E\left\{\left[y(t+1) - y_{ref}(t+1)\right]^2 + \lambda u^2(t) \middle| \Sigma(t)\right\} \tag{4.5}$$

Where J refers the performance index, $y_{ref}(t)$ is the desired trajectory of the system in discrete points. λ is the weighting factor which can be pre specified such as non negative number. The weighting factor can be selected to adjust the trade off between the tracking accuracy and the magnitude of the control signal. $E\{.|\Sigma(t)\}$ indicates a conditional expectation operation when the system outputs with past values $y(j),\ j \le t$ and the past control inputs $u(j),\ j < t$ indicated by $\Sigma(t)$ are given. If the equation $e(t) \equiv 0$, then the expectation operation is not needed.

The performance index is minimised relative to the admissible inputs while satisfying the system model equation. The control input is admissible when it is a function of system output $y(j),\ j \le t$ and the past control inputs $u(j),\ j < t$.

To minimise performance's index (J) with respect to admissible control input, the expression of $y(t+1)$ is required in terms of $y(j)$ and $u(j),\ j \le t$ under conditional expectation operation. Equation (4.4) is solved for $y(t+1)$ such as,

$$y(t+1) = -\hat{a}_1 y(t) -,\dots,-\hat{a}_{n_a} y(t+1-n_a) + \hat{b}_0 u(t) +,\dots,+\hat{b}_{n_b} u(t-n_b) + e(t+1) \quad (4.6)$$

The last term in Equation (4.6) is future disturbance that can not be predicted. That's why it is not considered in the minimisation of performance index under the conditional expectation operation. The following expression is valid for this case.

$$y(t+1|t) = -\hat{a}_1 y(t) -,\dots,-\hat{a}_{n_a} y(t+1-n_a) + \hat{b}_0 u(t) +,\dots,+\hat{b}_{n_b} u(t-n_b) \quad (4.7)$$

Where $y(t+1|t)$ is a one-step ahead predictor of $y(t+1)$ given the information up to $(y(t+1|t) = E[y(t+1)|y(j),u(j),\ i \le t])$ and including time tT. Under this consideration, the $y(t+1|t)$ is used instead of $y(t+1)$ in the performances index.

$$J = \left[y(t+1|t) - y_{ref}(t+1)\right]^2 + \lambda\, u^2(t) \quad (4.8)$$

Substituting Equation (4.7) into (4.8) gives

$$J = \left[-\hat{a}_1 y(t) -,\dots,-\hat{a}_{n_a} y(t+1-n_a) + \hat{b}_0 u(t) +,\dots,+\hat{b}_{n_b} u(t-n_b) - y_{ref}(t+1)\right]^2 \\ +\lambda\, u^2(t) \quad (4.9)$$

To minimise the performance's index, its derivative with respect to the control signal at time t is set equal to zero.

$$\frac{dJ}{du(t)} = 2\left[-\hat{a}_1 y(t) -,\dots,-\hat{a}_{n_a} y(t+1-n_a) + \hat{b}_0 u(t) +,\dots,+\hat{b}_{n_b} u(t-n_b) - y_{ref}(t+1)\right]\hat{b}_0 \\ +2\lambda\, u(t) = 0 \quad (4.10)$$

The admissible control input is obtained from Equation (4.10) as,

$$u(t)=\frac{\hat{b}_0}{\hat{b}_0^2+\lambda}\left[y_{ref}(t+1)+\hat{a}_1 y(t)+,\ldots,+\hat{a}_{n_a} y(t+1-n_a)-\hat{b}_1 u(t-1)-,\ldots,-\hat{b}_{n_b} u(t-n_b)\right] \quad (4.11)$$

Equation (4.11) specifies the controller that makes variable $y(t)$ track the desired trajectory $y_{ref}(t)$. In the case $\lambda = 0$, the controller of this equation is called a minimum variance controller. If this minimum variance control is substituted into Equation (4.6) the resulting equation assumes the following form

$$y(t+1)=y_{ref}(t+1)+e(t+1) \quad (4.12)$$

Thus the mean value of $y(t)$ is equal $y_{ref}(t)$ since the mean value of residual $e(t)$ is zero. The minimum variance controller corresponding to $\lambda = 0$ in Equation (4.11) is often varying fast in time and assumes large values. It causes difficulties in the implementation, for example, due to saturation. By appropriately choosing the weighting factor $\lambda > 0$ the magnitude of input $u(t)$ can advantageously be reduced.

4.2.2.2 Algorithm For Explicit Self-Tuning Adaptive Control

Step 1: Plan trajectory and set initial values.
Step 2: Read $y_{ref}(t)$ and $y(t)$.
Step 3: Update $\hat{A}(z^{-1})$ and $\hat{B}(z^{-1})$ by using recursive parameter estimation method such as U-D factorisation or RLS method.
Step 4: Compute the optimal control input $u(t)$ by using Equation (4.11).
Step 5: Apply the computed values $u(t)$ to the robotic systems. Obtain the current measurements from the system.
Step 6: Replace t by $t+1$ go to step 2 in every sampling period for all the points on the trajectory.

4.3 Self-Tuning Adaptive Controller Design by the Implicit Method

It is often possible to reparameterize the system model using the controller parameters, such that the controller parameters can be estimated directly. That is, the controller parameters are directly identified. This gives a direct adaptive algorithm. Hence the implicit method requires less computational time as compared to the explicit method.

4.3.1 Concept of Generalised Minimum Variance

Figure 4.3 shows three main components of the proposed controller: system, the parameter estimation block and a controller input and the measured output of the system at any sampling instant. The function of the estimation block is to recursively identify the parameters of the controller based on its input and output measurements. The parameters in conjunction with the desired reference track, and the input/output histories of the system are used to calculate the optimal control signal $u_0(t)$. Any changes in the system, such as load torque variations, noisy measurements or other disturbances are adaptively included in the coefficients of the polynomial $A(z^{-1}), B(z^{-1})$ and $C(z^{-1})$ in Equation (4.1) by the recursive identification method.

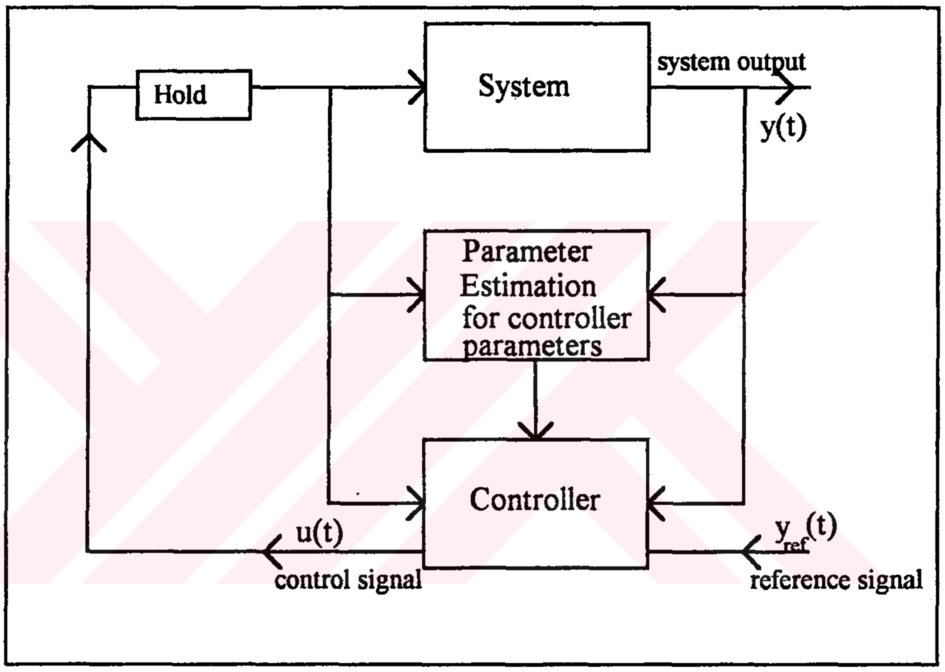


Figure 4.3 Implicit Self-tuning Control Strategy

Under the generalised minimum variance control, the following cost index (J) is minimised in order to calculate the optimal control signal:

$$J = E\left\{\left[y(t+1) - y_{ref}(t+1)\right]^2 + \lambda u^2(t) \middle| \Sigma(t)\right\} \tag{4.13}$$

In this performance criterion, the problem is solved by expressing the predicted value $y(t+1)$ in terms of the available information up to time tT, substituting it into Equation (4.13). Under this condition cost function in Equation (4.13) can be rewritten as explicit self-tuning adaptive control design.

$$J = \left[\left[y(t+1|t) - y_{ref}(t+1) \right]^2 + \lambda u^2(t) \right] \tag{4.14}$$

Where $y(t+1|t)$ is one step ahead predicted output value of $y(t+1)$. To minimise this cost function its derivative with respect to the control signal at time t is set equal to zero.

$$\frac{dJ}{du(t)} = 2\left[y(t+1|t) - y_{ref}(t+1) \right] \left[\frac{dy(t+1|t)}{du(t)} \right] + 2\lambda u_0(t) = 0 \tag{4.15}$$

$u_0(t)$ is the optimal control signal that minimises the cost index of Equation (4.15). On the basis of Equation (4.1), the derivative $dy(t+1|t)/du(t)$ is equal to b_0; Therefore,

$$y(t+1|t) - y_{ref}(t+1) + \alpha u_0(t) = 0 \tag{4.16}$$

where $\alpha = \dfrac{\lambda}{b_0}$

Let Equation (4.1) be rewritten in the following form:

$$y(t+1) = \left[\frac{C(z^{-1})}{A(z^{-1})} \right] e(t+1) + \left[\frac{B(z^{-1})}{A(z^{-1})} \right] u(t) \tag{4.17}$$

By expanding $1/A(z^{-1})$ as a power series in z^{-1}, it can be seen that the first term of Equation (4.17) (the disturbance term) has two components: one related to future disturbances and the other related to past disturbance [17]. The past can be reconstructed using input-output measurements, but the future disturbances can not be predicted. In order to separate the two disturbance components, the following identify (known as a Diphontine equation) is introduced;

$$\frac{C(z^{-1})}{A(z^{-1})} = F(z^{-1}) + z^{-d} \frac{G(z^{-1})}{A(z^{-1})} \tag{4.18}$$

Where $F(z^{-1})$ is a polynomial of order $(d-1)$ and $G(z^{-1})$ is a polynomial of order $(n_a - 1)$. The open forms of the polynomials for given $d = 1, n_a$ and n_b can be written as

$$F(z^{-1}) = f_0 = 1 \tag{4.19}$$

$$G(z^{-1}) = g_0 + g_1 z^{-1} +, \ldots, + g_{n_g} z^{-n_g} = c_1 - a_1 + c_2 z^{-1} - a_2 z^{-1} +, \ldots, + c_{n_a} z^{-(n_a-1)} - a_{n_a} z^{-(n_a-1)} \tag{4.20}$$

Substituting $C(z^{-1})$ Equation (4.18) into (4.17) gives the following:

$$y(t+1) = e(t+1) + \left[\frac{G(z^{-1})}{A(z^{-1})}\right]e(t) + \left[\frac{B(z^{-1})}{A(z^{-1})}\right]u(t) \tag{4.21}$$

Substituting the value of $e(t)$ Equation (4.1) into (4.21) yields

$$y(t+1) = e(t+1) + \left[\frac{G(z^{-1})}{C(z^{-1})}\right]y(t) + \left[\frac{B(z^{-1})}{C(z^{-1})}\right]u(t) \tag{4.22}$$

$e(t+1)$ is future disturbance that can not be predicted since the noise sequence is uncorrelated. Therefore, it is not considered in the minimisation process of J. Hence, if $y(t+1|t)$ is a "one-step-ahead predictor" of $y(t+1)$, given the input/output history up to time t, the following expression is valid [24];

$$y(t+1|t) = \left[\frac{G(z^{-1})}{C(z^{-1})}\right]y(t) + \left[\frac{B(z^{-1})}{C(z^{-1})}\right]u(t) \tag{4.23}$$

Substituting Equation (4.23) into (4.16) gives

$$G(z^{-1})y(t) + \left[B(z^{-1}) + \alpha C(z^{-1})\right]u_0(t) - C(z^{-1})y_{ref}(t+1) = 0 \tag{4.24}$$

The closed loop poles of the controller described by Equation (4.24) are the roots of the following Equation [24, 53];

$$\left[B(z^{-1}) + \alpha C(z^{-1})\right] = 0 \tag{4.25}$$

Equation (4.25) is not entirely dependent on the roots of the polynomial $B(z^{-1})$; thereby a stable controller of non-minimum phase plants [53] can be achieved by on-line adjustment of α. Expanding Equation (4.24) as a difference equation gives the optimal control signal $u_0(t)$ as,

$$u_0(t) = \frac{C(z^{-1})y_{ref}(t+1) - G(z^{-1})y(t)}{\left[B(z^{-1}) + \alpha C(z^{-1})\right]} \tag{4.26}$$

The function of the recursive estimation process is to identify the parameters of polynomials $B(z^{-1}), C(z^{-1})$ and $G(z^{-1})$ by using on-line measurements of the input and output of the robotics system. The system model is rearranged for parameter identification by multiplying both sides of Equation (4.23) by $C(z^{-1})$

$$C(z^{-1})y(t+1|t) = G(z^{-1})y(t) + B(z^{-1})u(t) \tag{4.27}$$

Hence,

$$\begin{aligned} y(t+1|t) = g_0 y(t)+,\ldots,+g_{n_g} y(t-n_g) + b_0 u(t)+,\ldots,+b_{n_b} u(t-n_b) \\ -c_1 y(t|t-1)-,\ldots,-c_{n_c} y(t-1+n_c|t-n_c) \end{aligned} \tag{4.28}$$

Then Equation (4.28) can be written in the following form:

$$y(t+1|t) = \phi^T(t)\theta(t) \tag{4.29}$$

Define the measurement vector $\phi(t)$, and the unknown parameter vector $\theta(t)$ as follows;

$$\phi^T(t) = \left[y(t),..y(t-n_g)\ u(t),...,u(t-n_b)\ y(t|t-1),...,y(t-n_c+1|t-n_c)\right]$$

$$\theta^T(t) = \left[g_0\ g_1,\ldots,g_{n_g}\ b_0\ b_1,\ldots,b_{n_b}\ c_1,\ldots,c_{n_c}\right]$$

The unknown controller parameter in $\theta^T(t)$ can be identified recursively by on-line calculation using extended RLS method.

4.3.2 Implicit Self-Tuning Adaptive Control Algorithm

Step 1: Plan trajectory and set initial values.
Step 2: Read $y_{ref}(t)$ and $y(t)$
Step 3: Update $\hat{G}(z^{-1})$, $\hat{B}(z^{-1})$ and $\hat{C}(z^{-1})$ by using Extended RLS method.
Step 4: Compute the optimal control input $u_0(t)$ by using Equation (4.26)
Step 5: Apply the computed values $u_0(t)$ to the robotics system. Obtain the current measurements from the system.
Step 6: Replace t by $t+1$ go to step 2.

4.4 Application of Self-Tuning Adaptive Control to Hydraulic Robotic System

Experimental result on a hydraulic manipulator, with two degrees of freedom, which has one revolute and one prismatic joint working in the vertical plane on which gravity is directly acting, are presented to demonstrate the effectiveness of self-tuning adaptive control schemes on performance of the system.

The dynamics of robotic manipulator are highly non-linear and strongly coupled. In particular, the effective inertia, coupling inertia, and the gravitational torque varying according to the angular positions, while the coriolis and centrifugal torque depend upon the velocities. These factors complicate the problems of the control of the robot manipulator arm.

By using self-tuning adaptive control method, the manipulator arm can be regarded as a time-varying linear system whose parameters will be estimated on-line, such that the controller can tune itself to generate corrective action to compensate for the non linearity of the manipulator dynamics. In this scheme, joint dynamics of manipulator are independent, difference equation is chosen as a single-input single-output model (SISO). Hence each joint is assumed to have a separate time-varying linear model (such as ARMAX or ARX) and will be controlled independently.

In the application of SISO self-tuning control, the manipulator joints are assumed to be loosely coupled, so that they may be treated as two separate working joints. Four different motion trajectories for each joint of the manipulator are planed in the joint space with MOTDES. These trajectories are very smooth for tracking, because a smooth motion of manipulator is achieved when the trajectory has the property that the positions, the velocities, and even the accelerations are continuous on the desired trajectory.

On the basis of the experimental implementation, the system model order in explicit form is selected such as second and third order with free noise parameters and second order with noise parameter. The implicit method is only selected as second order with noise parameter. Hence, there are four cases to be tested experimentally. The same motion trajectories are used for each case in order to compare the performance of tracking accuracy for the conventional and self-tuning adaptive control methods.

Case 1 : Second Order Explicit Self-tuning Adaptive Control

In this study, a second order ($n^i = n_a{}^i = n_b{}^i = 2$ and $n_c{}^i = 0$) auto regressive exogenous (ARX) discrete SISO model is chosen for designing explicit self-tuning adaptive control. In order to generate this model it is assumed that the manipulator has independent joints dynamic for each joint. Therefore, the model is established for each joint (i, $i = 1,2$ and 3) independently;

$$y_i(t) = -a_1{}^i y_i(t-1) - a_2{}^i y_i(t-2) + b_0{}^i u_i(t-1) + b_1{}^i u_i(t-2) + b_2{}^i u_i(t-3) + e_i(t) \quad (4.30)$$

Where integer i refers to each joint of manipulator. The input of each joint $u_i(t)$ is voltage and outputs of each joint $y_i(t)$ are position. The position of the second joint is angular (*rad*) and the third is linear displacement (*cm*).

In order to calculate the optimum control signal the controller is designed as given in Equation (4.11). To solve the minimisation problem subject to the constraints, the unknown parameters in Equation (4.30) are first estimated for this purpose.

The parameter vector θ_i^T in the ARX-model of joint i can be calculated recursively using U-D factorisation method. To obtain the parameter $\hat{\theta}_i(t)$ at t sample instant. The initial parameters of model for each joint are set to the following

$\hat{a}_1^i(0) = 0$, $\hat{a}_2^i(0) = 0$, $\hat{b}_0^i(0) = 1, \hat{b}_1^i(0) = 0$ and $\hat{b}_2^i(0) = 0$.

The following settings are used for two joints and initial values of system input and output are equal to zero i.e. $y_i(-1) = y_i(0) = u_i(-2) = u_i(-1) = u_i(0) = 0$ for data vector in parameter estimation method. The forgetting factor γ_i is set equal to 0.98 and weighting factor λ_i is 0.025 for each joint. It should be noted that the forgetting and weighting factor is selected experimentally. The initial parameters of diagonal and upper triangle matrix are set respectively such as;

$$D_g^i(0) = 1000 I_m \text{ and } U_p^i(0) = I_m \tag{4.31}$$

Where m represent the number of unknown parameter in the system model equation and I is diagonal matrix with dimension is shown by number of unknown parameters ($m = 5$).

Case 2 : Third Order Explicit Self-Tuning Adaptive Control

For this case, the explicit self-tuning adaptive control algorithm based on a third order ($n^i = n_a^i = n_b^i = 3$ and $n_c^i = 0$) ARX discrete SISO model is applied to the manipulator for each joint. Independent joint dynamics are assumed for the manipulator motion.Equation (4.2) can be written as for third order,

$$\begin{aligned} y_i(t) = &-a_1^i y_i(t-1) - a_2^i y_i(t-2) - a_3^i y_i(t-3) + b_0^i u_i(t-1) \\ &+ b_1^i u_i(t-2) + b_2^i u_i(t-3) + b_3^i u_i(t-4) + e_i(t) \end{aligned} \tag{4.32}$$

The unknown parameter of this model for each joint are estimated by on-line calculation using U-D factorisation method. The controller is constructed as an LQG-controller by minimising a well defined single-stage criterion. Similar to in Case 1, the optimal control signal for each joint is obtained.

For implementation, the initial approximations for the parameter to be estimates are $\hat{a}_1{}^i(0) = 0$, $\hat{a}_2{}^i(0) = 0$, $\hat{a}_3{}^i(0) = 0$, $\hat{b}_0{}^i(0) = 1$, $\hat{b}_1{}^i(0) = 0$, $\hat{b}_2{}^i(0) = 0$ and $\hat{b}_3{}^i(0) = 0$. The following settings are used for two joints and initial values of system input and output are equal to zero i.e. $y_i(-2) = y_i(-1) = y_i(0) = 0$ and $u_i(-3) = u_i(-2) = u_i(-1) = u_i(0) = 0$ for data vector in parameter estimation method. The forgetting factor γ_i is set equal to 0.98 and the weighting factor λ_i is 0.025 for each joint. The initial diagonal and upper triangle matrix are set respectively such as;

$$D_g{}^i(0) = 1000 I_m \text{ and } U_p{}^i(0) = I_m \quad (m = 7) \tag{4.33}$$

Case 3: Second Order Explicit Self-Tuning Adaptive Control Including Noise

For this case the system model is contaminated with noise parameter. Hence, the polynomial $C(z^{-1})$ in Equation (4.1) can not be assumed to be unity. So, the model order of each joint is selected as second order and order of noise polynom is one.

$$\begin{aligned} y_i(t) = & -\hat{a}_1{}^i y_i(t-1) - \hat{a}_2{}^i y_i(t-2) + \hat{b}_0{}^i u_i(t-1) + \hat{b}_1{}^i u_i(t-2) \\ & + \hat{b}_2{}^i u_i(t-3) + e_i(t) + \hat{c}_1{}^i e_i(t-1) \end{aligned} \tag{4.34}$$

Where $e(t)$ represents a white Gaussion zero-mean noise process. The parameter vector θ_i^T in the ARMA-model of joint i is estimated recursively by on-line calculation using the recursive extended least-square as U-D factorisation method. To obtain the parameter $\hat{\theta}_i(t)$ at t sample instant.

In order to estimate $C(z^{-1})$ in the general case, knowledge of $e_i(t-1),\ldots,e_i(t-n_c)$ is required. It can be seen that in the data vector, knowledge of $e_i(t-1)$ is necessary. In this case, $e_i(t)$ is replaced by an estimate, usually taken to be the prediction error $\varepsilon(t)$ in the parameter estimation method. Equation (4.34) is used recursively to express $y_i(t+1)$ in terms of $y_i(t)$ and $u_i(t)$. The result can be written as,

$$y_i(t+1) = -\hat{a}_1^i y_i(t) - \hat{a}_2^i y_i(t-1) + \hat{b}_0^i u_i(t) + \hat{b}_1^i u_i(t-1) + \hat{b}_2^i u_i(t-2) + e_i(t+1) + \hat{c}_1^i e_i(t) \tag{4.35}$$

$e_i(t+1)$ is future disturbance which can not be predicted since the noise sequence is uncorrelated. Therefore, it is not considered in the minimisation process of J. Hence, if $y_i(t+1|t)$ is a "one-step-ahead predictor" of $y_i(t)$, given the input/output history up to time t, the following expression is valid.

$$y_i(t+1|t) = -\hat{a}_1^i y_i(t) - \hat{a}_2^i y_i(t-1) + \hat{b}_0^i u_i(t) + \hat{b}_1^i u_i(t-1) + \hat{b}_2^i u_i(t-2) + \hat{c}_1^i e_i(t) \tag{4.36}$$

Nothing that, whatever the choice of controller, $u_i(t)$ cannot physically be a function of $y_i(t+1)$, then $y_i(t+1|t)$ is functionally independent of $e_i(t+1)$. Further, from Equation (4.34) and (4.36), we see that $y_i(t+1|t)$ is a linear or non-linear function of $e_i(t)$, $e_i(t-1),\ldots$, according to whether $u_i(t)$ is a linear or non-linear controller. The significance of these properties emerges when we form the cost function J [47]. Equation (4.36) may be rewritten as

$$y_i(t+1) = y_i(t+1|t) + e_i(t+1) \tag{4.37}$$

$$J = E\{[y_i(t+1)]^2\} = E[y_i(t+1|t) + e_i(t+1)]^2 \tag{4.38}$$

$$J = E[y_i(t+1|t)]^2 + E[e_i(t+1)]^2 + 2E[y_i(t+1|t)e_i(t+1)] \tag{4.39}$$

The last term on the right-hand side vanishes for

a) Any linear controller.
b) Any non-linear controller, provide $\{e_i(t)\}$ is an independent sequence (not just uncorrelated).

Condition (b) is satisfied if we make the common assumption of gaussian white noise, then Equation (4.39) becomes.

$$J = E[y_i(t+1|t)]^2 + \sigma^2 \tag{4.40}$$

Under these condition, the substitution of Equation (4.37) into Performances index and performing the conditional expectation result in

$$J = E\left[y_i(t+1|t) - y_{ref_i}(t+1)\right]^2 + \lambda_i u_i(t)^2 + \sigma^2 \tag{4.41}$$

Where the constant σ^2 signifies the variance of the error term. If $e_i(t)$ can be expressed as a function of available data. This is achieved by inverting the process model Equation (4.34) such as transfer function terms.

$$e_i(t)=\frac{1}{1+c_1}\left[(1+a_1{}^i z^{-1}+a_2{}^i z^{-2})y_i(t)-(b_0{}^i z^{-1}+b_1{}^i z^{-2}+b_2{}^i z^{-3})u_i(t)\right] \qquad (4.42)$$

Substituting the value of $e_i(t)$ into Equation (4.36) and setting $y_i(t+1|t)$ to Equation (4.41), the performance criterion can be minimised with respect to $u_i(t)$. The minimising value of the input is obtained as

$$\begin{aligned} u_i(t)=&\frac{\hat{b}_0{}^i}{(\hat{b}_0{}^i)^2+\lambda_i}\left[y_{ref_i}(t+1)+\hat{c}_1{}^i y_{ref_i}(t)+(\hat{a}_1{}^i-\hat{c}_1{}^i)y_i(t)+\hat{a}_2{}^i y_i(t-1)\right] \\ &+\frac{\hat{b}_0{}^i}{(\hat{b}_0{}^i)^2+\lambda_i}\left[-\hat{b}_1{}^i u_i(t-1)-\hat{b}_2{}^i u_i(t-2)\right]-\frac{\lambda_i}{(\hat{b}_0{}^i)+\lambda_i}\left[\hat{c}_1{}^i u_i(t-1)\right] \end{aligned} \qquad (4.43)$$

The control law in Equation (4.43) is in the feedback form. The control input $u_i(t)$ can be calculated because the values of $y_i(t)$, $y_i(t-1)$, $u_i(t-1)$, $u_i(t-2)$, $y_{ref_i}(t+1)$ and $y_{ref_i}(t)$ appearing in Equation (4.43) are known. The parameters are first estimated at the sampling instant t according to parameter estimation method. Then the control input is computed by Equation (4.43). These calculations are repeated at each sampling instant. For the initial approximations the parameters to be estimate are taken as

$\hat{a}_1{}^i(0)=0$, $\hat{a}_2{}^i(0)=0$, $\hat{b}_0{}^i(0)=1$,$\hat{b}_1{}^i(0)=0$, $\hat{b}_2{}^i(0)=0$ and $\hat{c}_1{}^i(0)=0$.

The following settings are used for two joints and initial values of system input and output are equal to zero i.e. $y_i(-1)=y_i(0)=0$, $u_i(-2)=u_i(-1)=u_i(0)=0$ and $e_i(0)=0$ for data vector in parameter estimation method. The forgetting factor γ_i is set equal to 0.98 and weighting factor λ_i in Equation (4.43) is 0.025 for each joint. The initial diagonal and upper triangle matrix are set respectively such as;

$$D_g{}^i(0)=1000I_m \text{ and } U_p{}^i(0)=I_m \quad (m=6) \qquad (4.44)$$

Case 4: Second Order Implicit Self-Tuning Adaptive Control Including Noise

The implicit self-tuning adaptive controller for the gross motion of manipulator is designed for each joint position on the basis of an ARMA-model.

Firstly the parameters of controller are estimated directly. This is done by the reparameterization of the process model. Model order is chosen as second order with noise parameter. Consider this model that is described as;

$$A^i(z^{-1})y_i(t) = z^{-1}B^i(z^{-1})u_i(t) + C^i(z^{-1})e_i(t) \tag{4.45}$$

By using Equation (4.26) the optimal control signal $u_{0,i}(t)$ is rewritten as

$$u_{0,i}(t) = \frac{C^i(z^{-1})y_{ref_i}(t+1) - G^i(z^{-1})y_i(t)}{\left[B^i(z^{-1}) + \alpha C^i(z^{-1})\right]} \tag{4.46}$$

This controller polynomial of $F^i(z^{-1})$ and $G^i(z^{-1})$ are obtained by solving the modified identity (known as a Diphontine equation) is introduced ;

$$\frac{C^i(z^{-1})}{A^i(z^{-1})} = F^i(z^{-1}) + z^{-1}\frac{G^i(z^{-1})}{A^i(z^{-1})} \tag{4.47}$$

According to the given time delay and system model order, the orders of polynomial $F^i(z^{-1})$ and $G^i(z^{-1})$ are zero and one respectively. These polynomial are represented by

$$F^i(z^{-1}) = f^i_0 = 1 \tag{4.48}$$

$$G^i(z^{-1}) = g_0^{\,i} + g_1^{\,i}z^{-1} = (c_1^{\,i} - a_1^{\,i}) + (c_2^{\,i} - a_2^{\,i})z^{-1} \tag{4.49}$$

Hence, the control input will be

$$\begin{aligned} u_{o,i}(t) = {} & \frac{1}{\hat{b}_0^{\,i} + \alpha}\left[y_{ref_i}(t+1) + \hat{c}_1 y_{ref_i}(t) - \hat{g}_0^{\,i} y_i(t) - \hat{g}_1 y_i(t-1)\right] \\ & + \frac{1}{\hat{b}_0^{\,i} + \alpha}\left[-\hat{b}_1^{\,i}u_i(t-1) - \hat{b}_2^{\,i}u_i(t-2) - \alpha\hat{c}_1 u_i(t-1)\right] \end{aligned} \tag{4.50}$$

The function of the recursive estimation process is to identify the unknown parameters of polynomials $B^i(z^{-1}), C^i(z^{-1})$ and $G^i(z^{-1})$ by using on-line measurements of the input and output of the robotics system. The system model is rearranged by using Equation (4.23) for parameter identification.

$$C^i(z^{-1})y_i(t+1|t) = G^i(z^{-1})y_i(t) + B^i(z^{-1})u_i(t) \tag{4.51}$$

Hence,

$$y_i(t+1|t) = g_0{}^i y_i(t) + g_1{}^i y_i(t-1) + b_0{}^i u_i(t) + b_1{}^i u_i(t-1) + b_2{}^i u_i(t-1) - c_1{}^i y_i(t|t-1) \tag{4.52}$$

Then Equation (4.52) can be written for sampling instant t:

$$\begin{aligned} y_i(t|t-1) = {} & g_0{}^i y(t-1) + g_1{}^i y_i(t-2) + b_0{}^i u_i(t-1) + b_1{}^i u_i(t-2) \\ & + b_2{}^i u_i(t-3) - c_1{}^i y_i(t-1|t-2) \end{aligned} \tag{4.53}$$

For implementation, the initial approximations for the parameter to be estimates are $\hat{g}_0{}^i(0) = 0,\ \hat{g}_1{}^i(0) = 0,\ \hat{b}_0{}^i(0) = 1, \hat{b}_1{}^i(0) = 0,\ \hat{b}_2{}^i(0) = 0$ and $\hat{c}_1{}^i(0) = 0$. The following settings are used for two joints and initial values of system input and output are equal to zero i.e. $y_i(-1) = y_i(0) = 0$, $u_i(-2) = u_i(-1) = u_i(0) = 0$ and $y_i(0|-1) = 0$ for data vector in parameter estimation method. $\gamma_i = 0.98$ and $\lambda_i = 0.025$ for each joint. The initial diagonal and upper triangle matrix are set respectively such as;

$$D_g{}^i(t) = 1000 I_m \text{ and } U_p{}^i(t) = I_m \quad (m = 6) \tag{4.54}$$

4.4.1 Experimental Results

Four different examples for each case given above are presented on the hydraulic robot. The experimental results of each case for revolute joint (elevation angle) and prismatic joints (reach) are shown the first and second column of Figures 4.4 - 4.16 respectively. In these figures, the actual measurement output of each joint is represented by dashed lines and the desired trajectory is represented by solid lines.

Example 1

In this example, sample time is selected as *0.1* second for each case. For case one, experimental results are given in Figure 4.4 *a* and *d* which show that the tracking errors are large when the desired reference signals are dispatched to system under conventional control method. An acceptable execution of the job process may not be expected from the system with this amount of error. However, when the explicit self-tuning adaptive control algorithm is applied to the robot the tracking errors of both joints are dramatically reduced (see Figure 4.4 *b* and *e*). However, a small amount of oscillation may be seen at the beginning of motions that is the characteristic future of adaptive control. The initial parameters have to be estimated by the user for the estimation process. If these initial parameters are not estimated properly then they will exhibit very sharp oscillation at the beginning of motions (see Figure 4.4 *c* and *f*) and as they approach to the actual values oscillations settle down. Figures 4.4 *c* and *f* show the model parameters for revolute and prismatic joint that change with time.

The experimental results for case 2 are shown in Figure 4.5 *a-f*. In this case, the model order is selected as third order, therefore, the number of unknown parameters is increased by two. Figure 4.5 *a* and *d* is obtained with conventional control method. As it is seen that both joint follow the desired trajectory with a steady state error, however when the explicit self-tuning adaptive algorithm is applied to the robot, the responses of joints follow the desired trajectory very well (see Figure 4.5 *b* and *e*). Figure 4.5 *c* and *f* show the parameter of the system versus time for revolute and prismatic joint respectively.

For case 3, experimental results are given in Figure 4.6 *a-f*. In this case, model contains noise parameters. The responses of joints and the desired reference trajectories are given in Figure 4.6 *b* and *e* for explicit self-tuning adaptive control with noise parameter. Variation of model parameters of joints are given in Figure 4.6 *c* and *f*).

For case 4, the control of joint motions are performed by implicit self-tuning adaptive control. In this case the parameters of controller are estimated directly. Experimental results are shown in Figure 4.7 *a-f*. Figures 4.7 *b* and *e* show the tracking trace between the response of joint and reference trajectory. Initially the output of joint's variable oscillates while trying to follow the desired reference trajectory and they settle down.

Example 2

It is seen that these joint trajectories are more difficult then the trajectories of previous example. For each case, experimental results are given in Figure 4.8, Figure 4.9, Figure 4.10 and Figure 4.11 respectively. The sampling time is chosen as 0.2 second. The speed of the system is twice of the previous example. It is seen that the responses of both joints follow the reference trajectory with a steady state error when conventional control method is used see Figure 4.8 - 4.11 *a* and *d*.

For all cases the responses of joints are very close to the desired trajectories when the system is under adaptive control (see Figure 4.8-4.11) But, initially some amount of oscillation may be seen at the beginning of motion due to the roughly estimation of model parameters. Figure 4.8-4.11 *c* and *f* show the variations of parameters of models.

Example 3

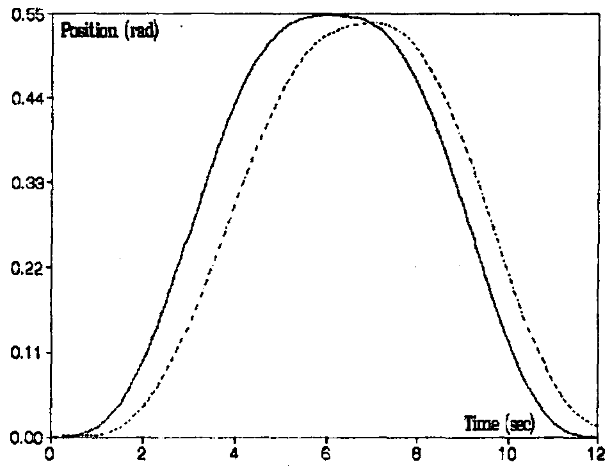
This example is given as an example of tracing of difficult joint trajectories. However, they are is successfully planned and implemented to the robot. The sample time is chosen 0.2 seconds. The experimental results of each case for this implementation are shown in Figure 4.12, 4.13, 4.14 and 4.15 respectively. The first column shows experimental results for revolute joint and the second column is obtained from prismatic joint in these figures.

Example 4

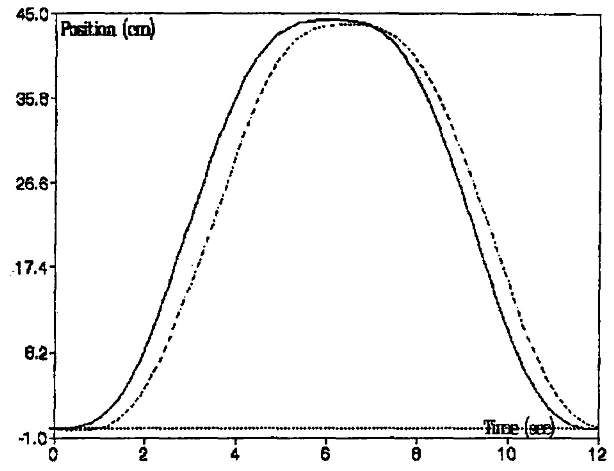
Explicit self-tuning adaptive control method for the gross motion of hydraulic robot is designed to operate directly on the position of the end-effector expressed in the Cartesian base co-ordinate system in this example. In order to control the motion of the end-effector of the robot, it is necessary to describe the desired trajectories in the joint co-ordinate system. The desired position of the end-effector expressed in the Cartesian co-ordinate system is converted to the desired positions of the joints by using the inverse kinematic equations.

In implementation, two different desired trajectories are planned for end-effector in the Cartesian co-ordinate system. These are a circle and an equilateral quadrangle. These two desired trajectories are combined in the implementation to investigate how the system can adapts itself to the trajectory changes. The end-effector of robot is follow the circle first and then immediately follow on the equilateral quadrangle in 72 seconds.

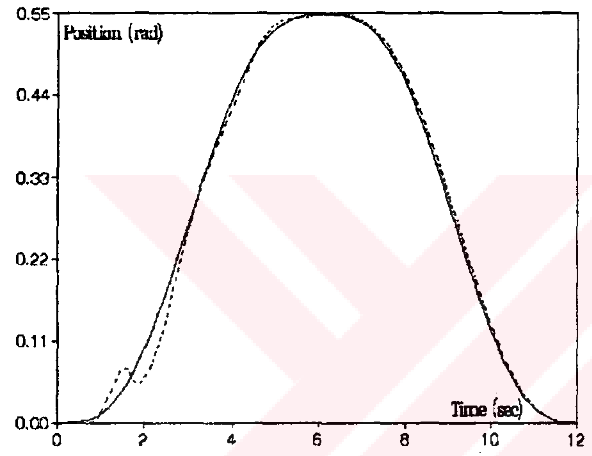
The experimental results are shown in Figure 4.16 in joint co-ordinates, and the trajectory for end-effector in the space co-ordinates in Figure 4.17. In these figures, the actual measurement outputs are represented by dashed lines and the desired trajectories are represented by solid lines. Sample time is selected as 0.2 sec. Figure 4.16 *a* and *c* for revolute and prismatic joints versus time. Figure 4.17 *a* show the motion of end-effector of robot when conventional control method is applied. It is seen from these figures that, the end-effector tracks the desired trajectory with steady state error. However, when explicit self-tuning algorithm is applied to each joint of hydraulic robot, the tracking errors between responses and desired trajectories are almost zero. At the end of the task of circle, the performance of the system is improved for the second task which shows ' self-learning ' ability. As a result of this implementation, one can say that the system can easily adapt itself to trajectory changing.



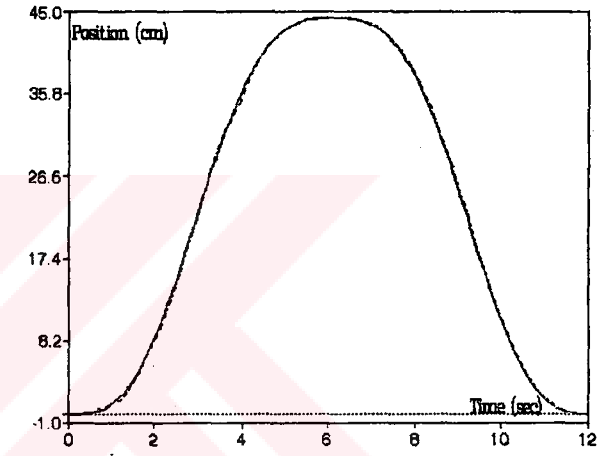
a) Response of revolute joint and desired reference trajectory when conventional control is used.



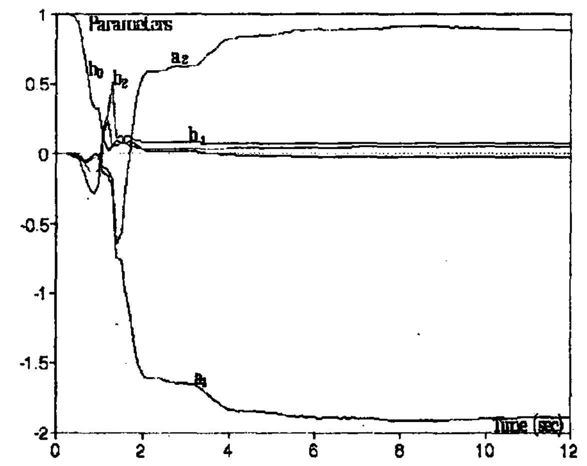
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



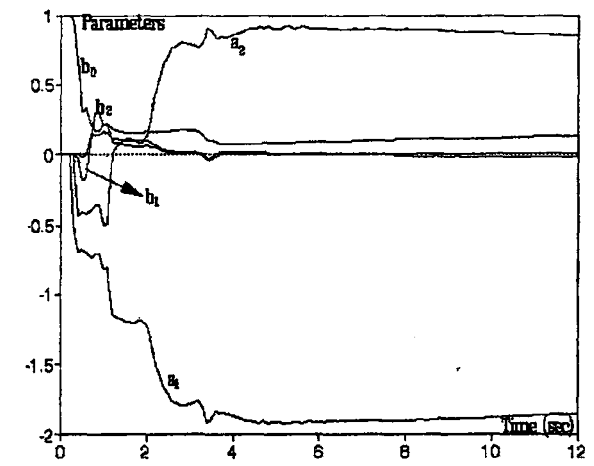
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



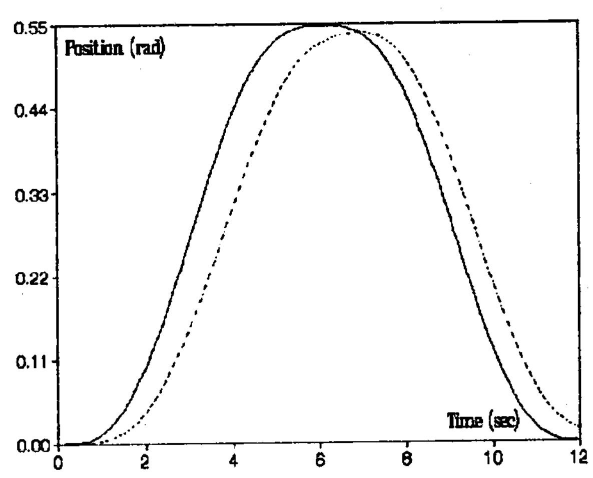
c) Changing of system parameters versus time.



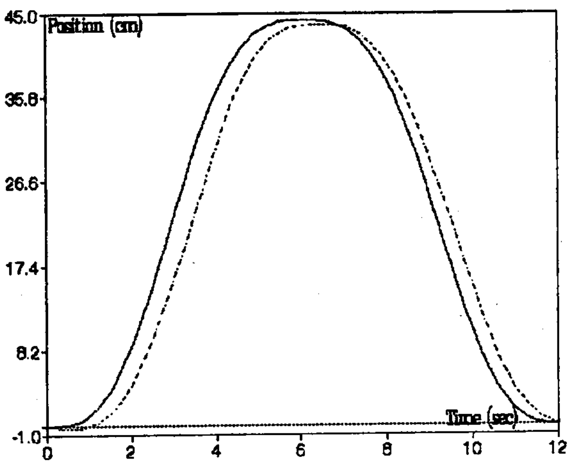
f) Changing of system parameters versus time.

__________ Desired reference trajectory .... Response of the system

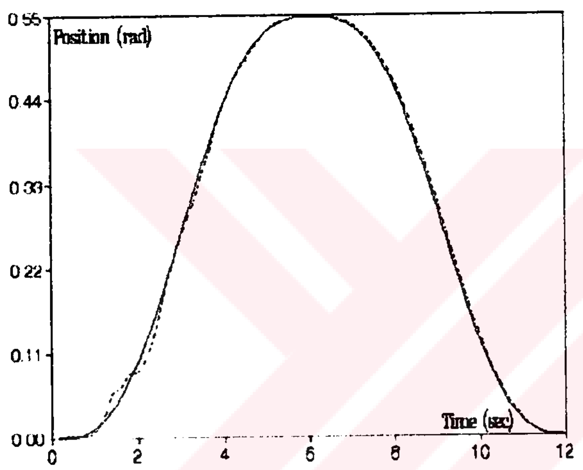
Figure 4.4 Experimental results on Hydraulic robot manipulator for case 1



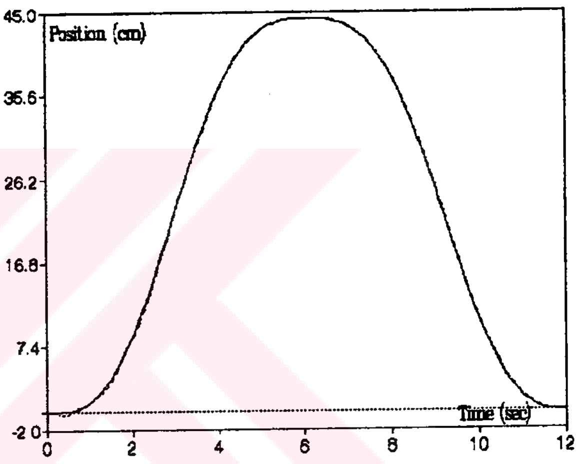
a) Response of revolute joint and desired reference trajectory when conventional control is used.



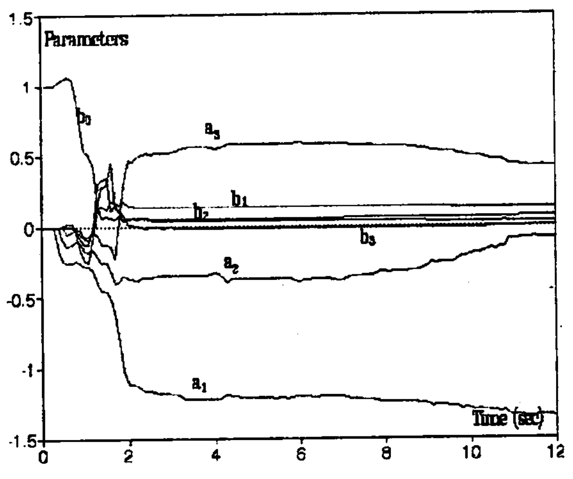
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



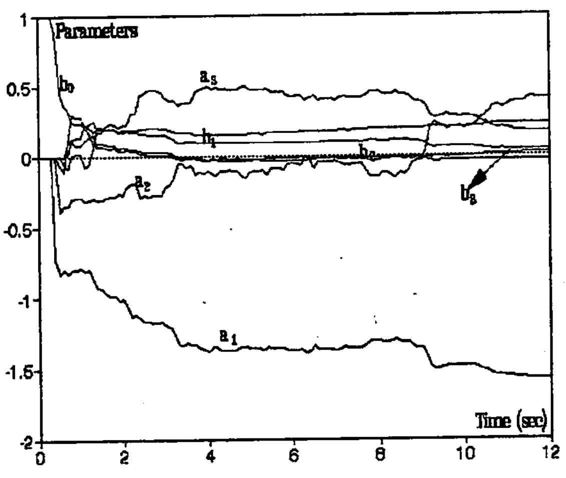
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



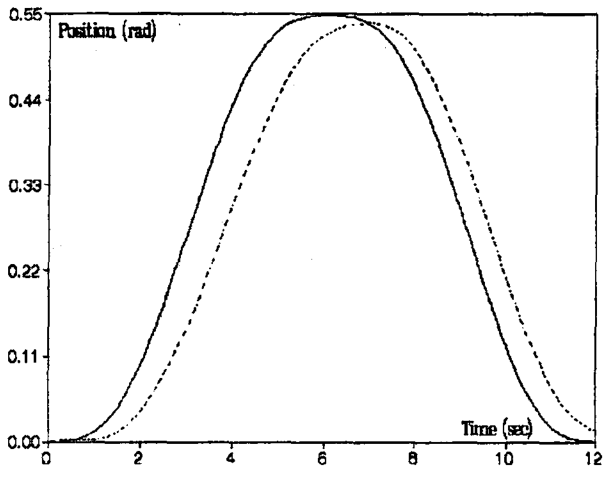
c) Changing of system parameters versus time.



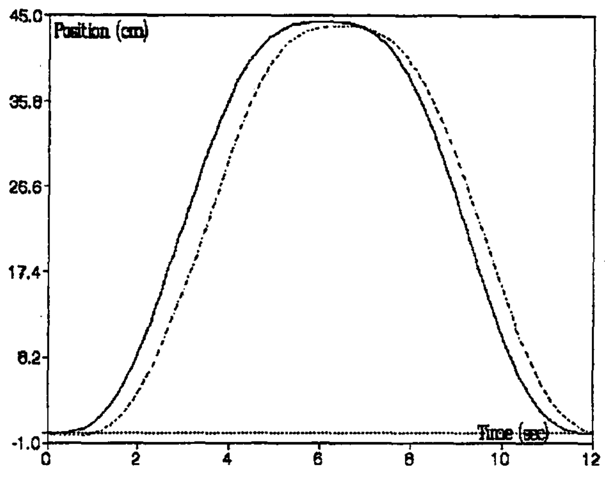
f) Changing of system parameters versus time.

__________ Desired reference trajectory .... Response of the system

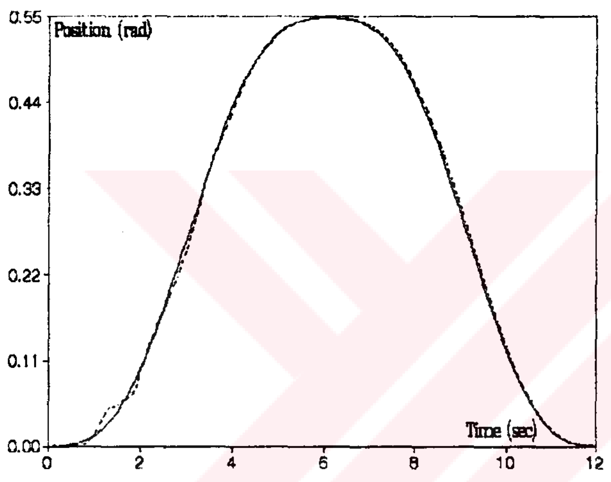
Figure 4.5 Experimental results on Hydraulic robot manipulator for case 2



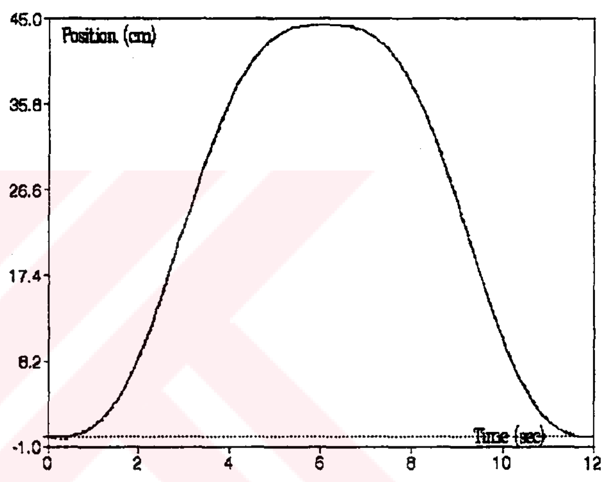
a) Response of revolute joint and desired reference trajectory when conventional control is used.



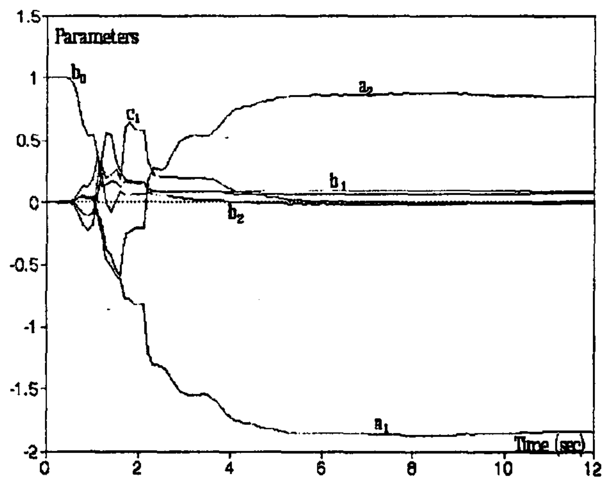
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



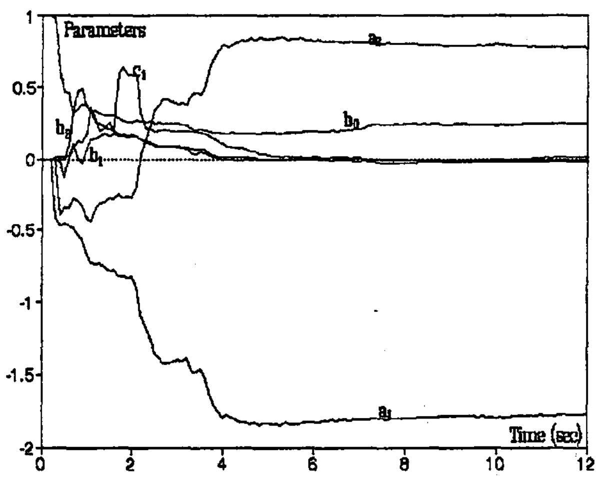
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



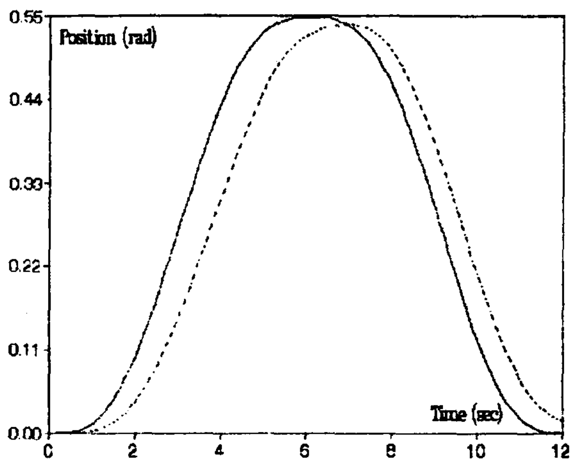
c) Changing of system parameters versus time.



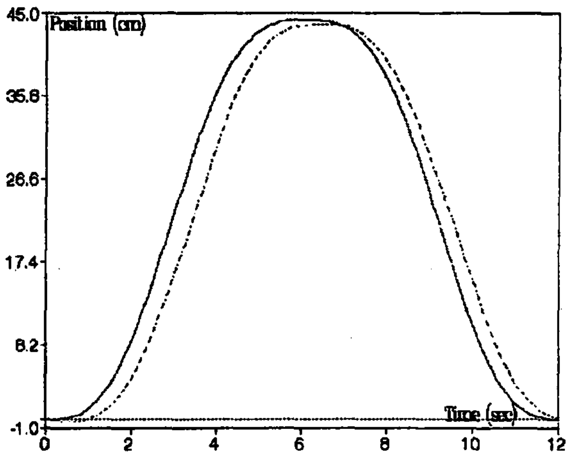
f) Changing of system parameters versus time.

___________ Desired reference trajectory .... Response of the system

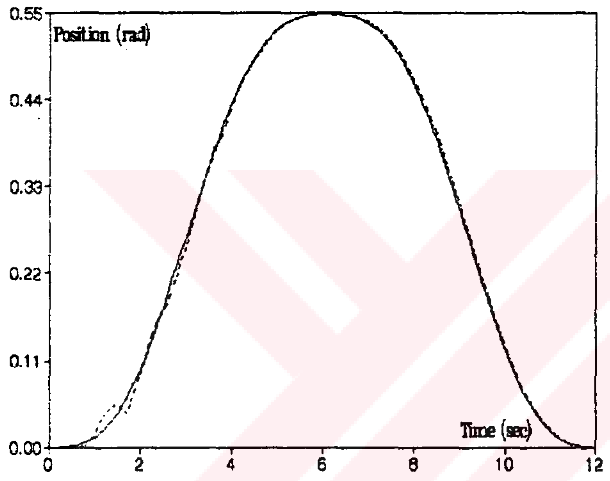
Figure 4.6 Experimental results on Hydraulic robot manipulator for case 3



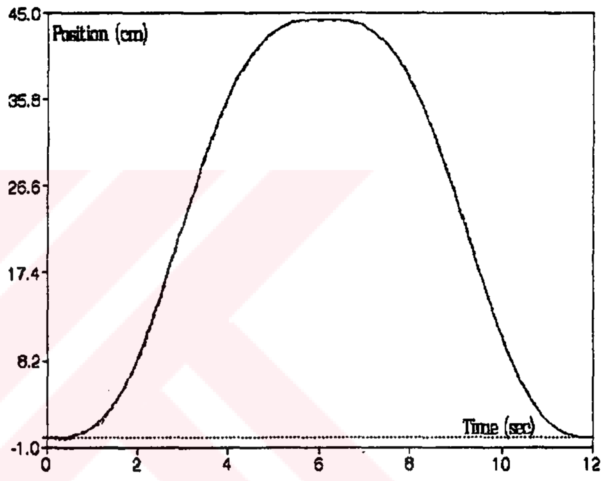
a) Response of revolute joint and desired reference trajectory when conventional control is used.



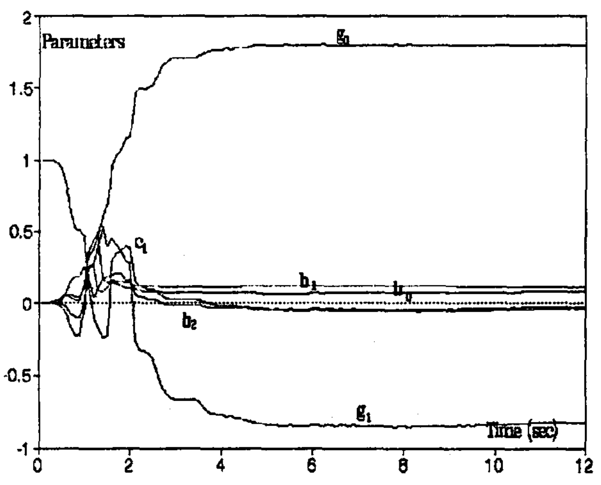
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



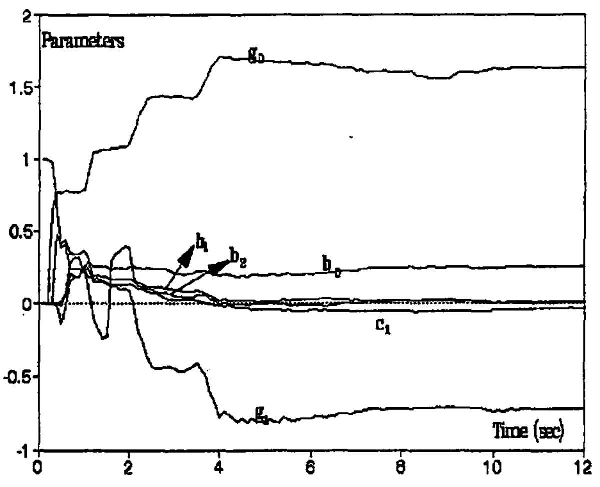
b) Response of revolute joint and desired reference trajectory when implicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when implicit STAC algorithm is used.



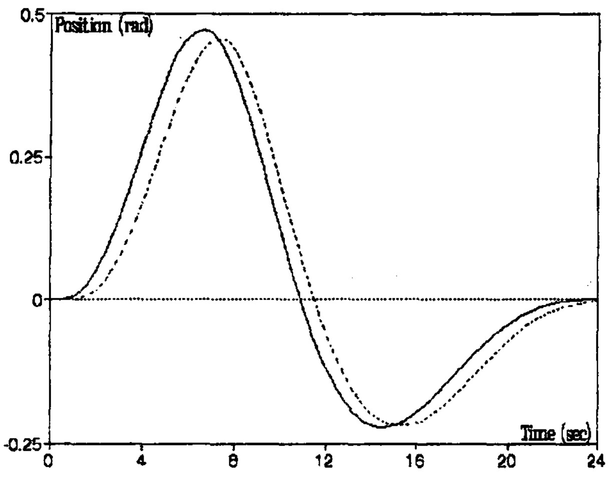
c) Changing of controller parameters versus time.

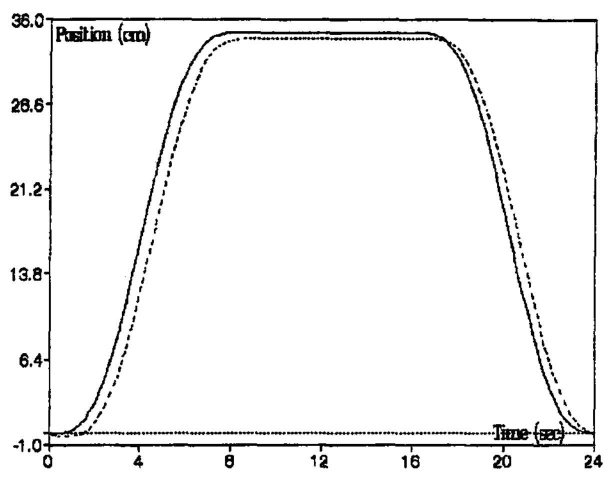


f) Changing of controller parameters versus time.

__________ Desired reference trajectory .... Response of the system

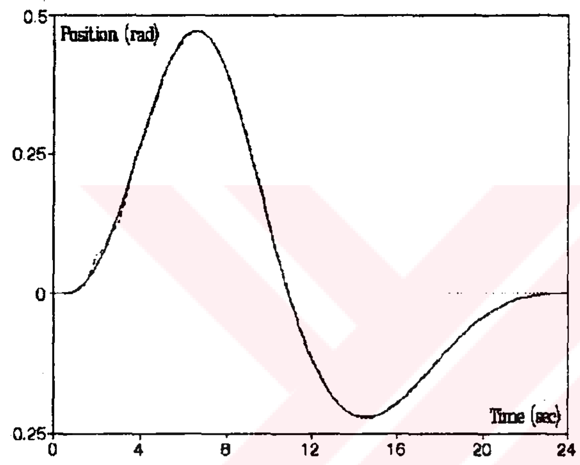
Figure 4.7 Experimental results on Hydraulic robot manipulator for case 4

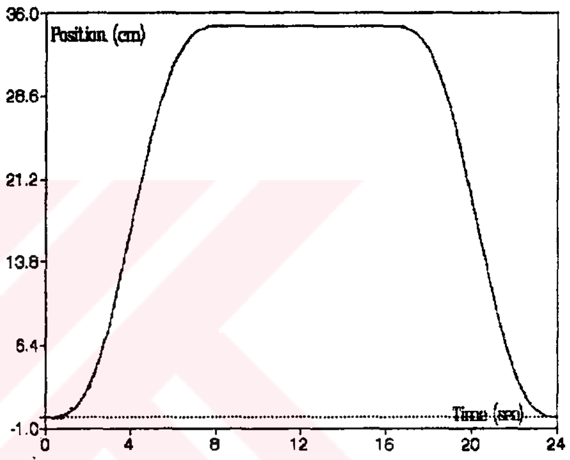




a) Response of revolute joint and desired reference trajectory when conventional control is used.

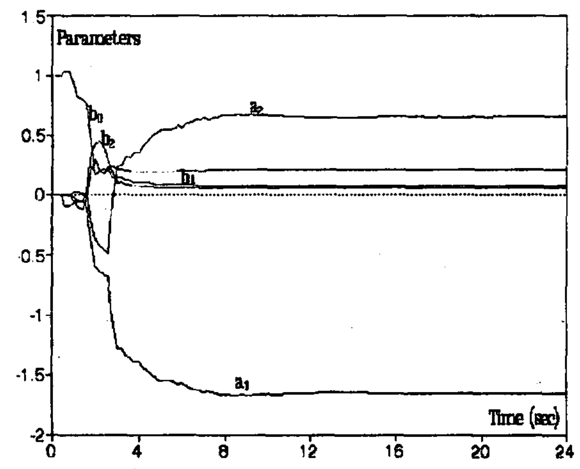
d) Response of prismatic joint and desired reference trajectory when conventional control is used.

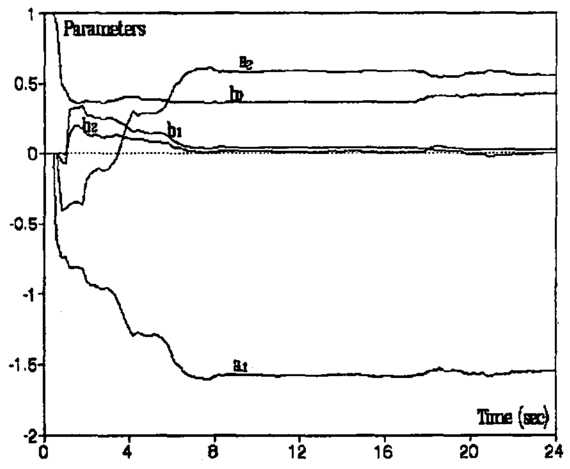




b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.

e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



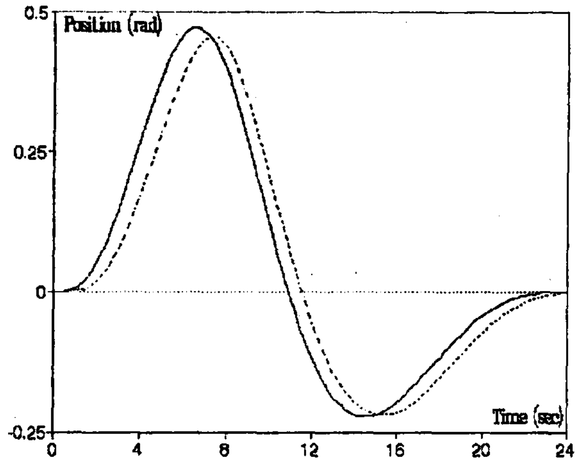


c) Changing of system parameters versus time.

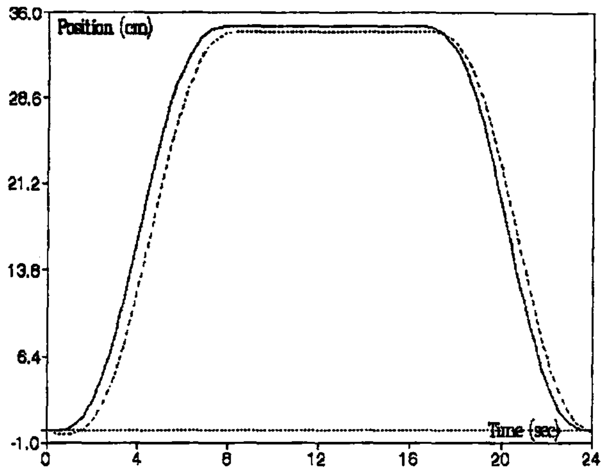
f) Changing of system parameters versus time.

___________ Desired reference trajectory .... Response of the system

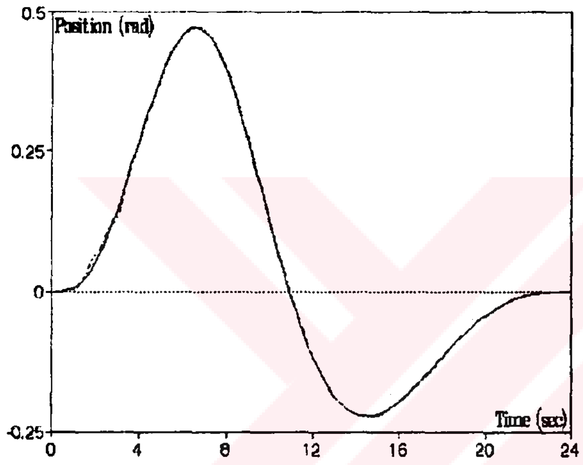
Figure 4.8 Experimental results on Hydraulic robot manipulator for case 1



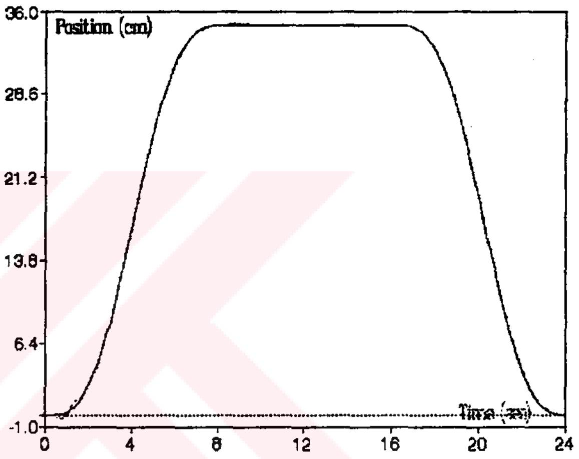
a) Response of revolute joint and desired reference trajectory when conventional control is used.



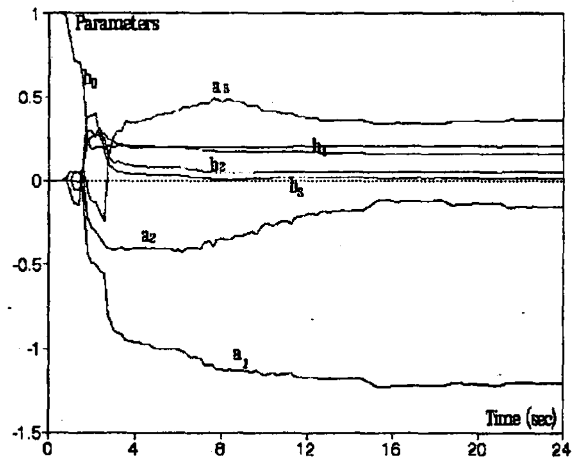
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



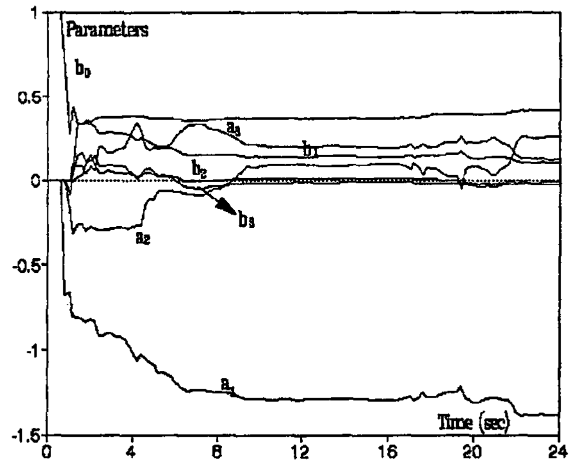
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



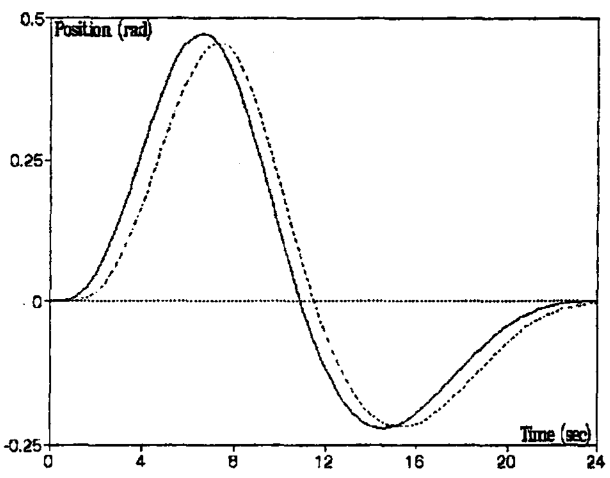
c) Changing of system parameters versus time.



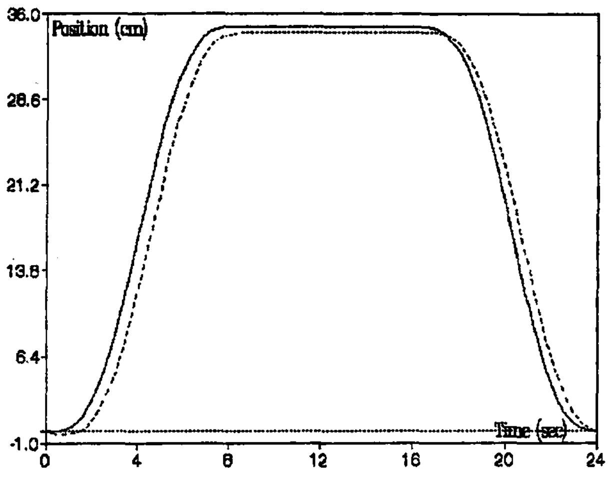
f) Changing of system parameters versus time.

__________ Desired reference trajectory .... Response of the system

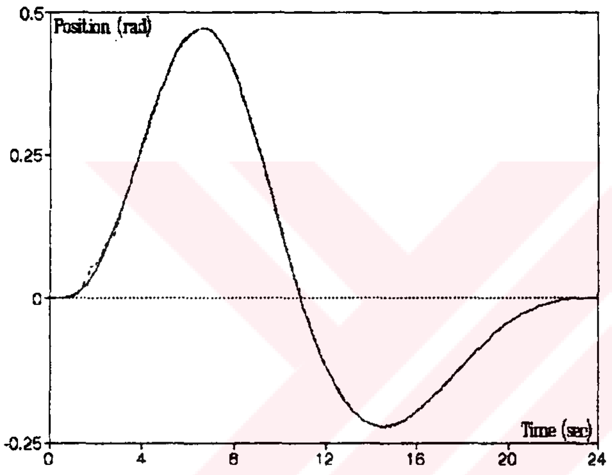
Figure 4.9 Experimental results on Hydraulic robot manipulator for case 2.



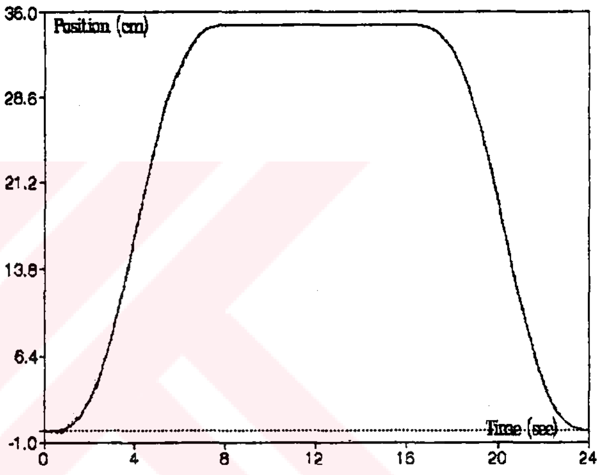
a) Response of revolute joint and desired reference trajectory when conventional control is used.



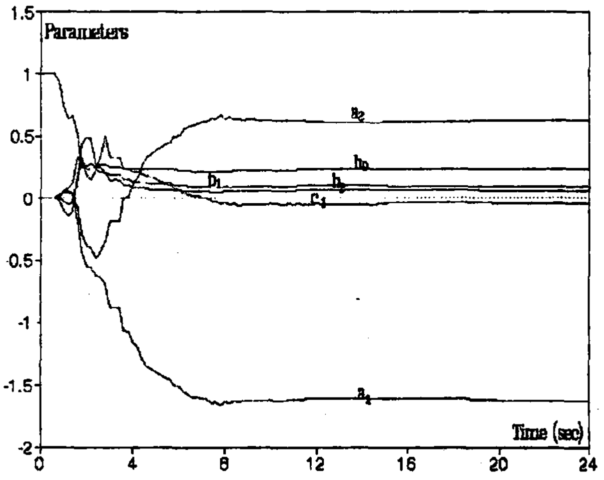
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



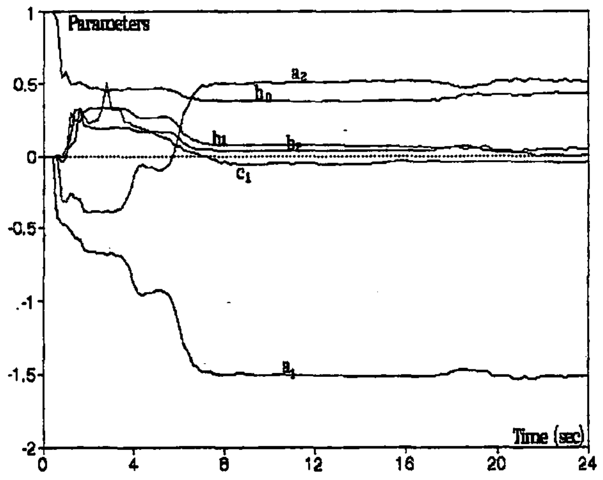
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



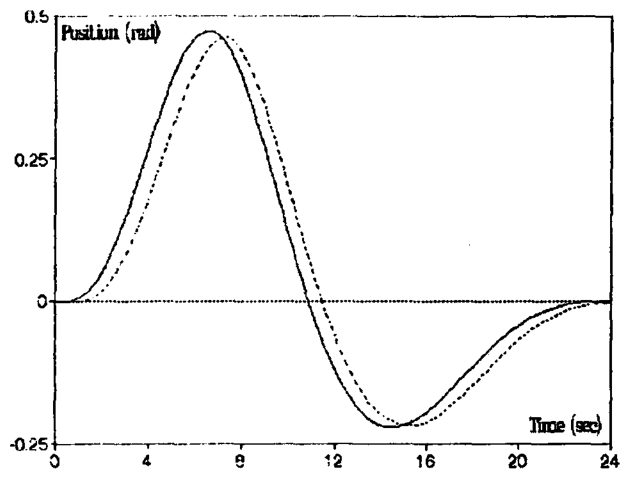
c) Changing of system parameters versus time.



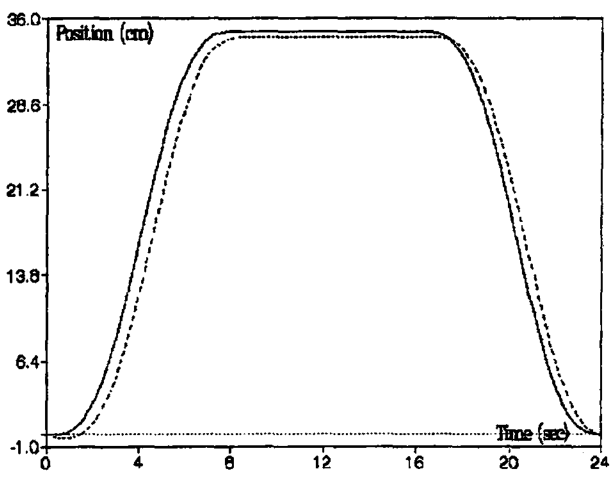
f) Changing of system parameters versus time.

__________ Desired reference trajectory .... Response of the system

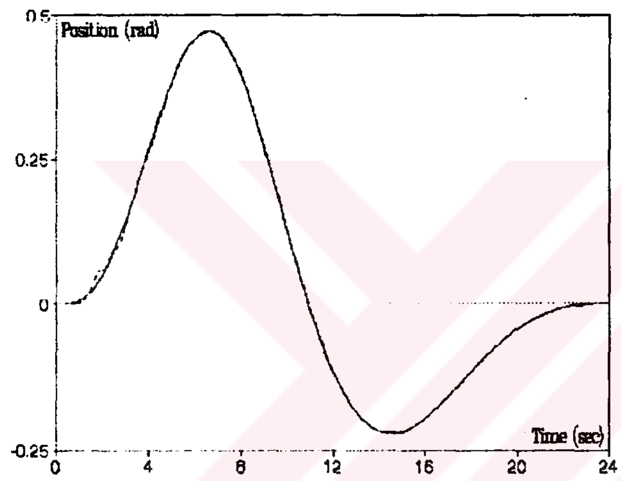
Figure 4.10 Experimental results on Hydraulic robot manipulator for case 3



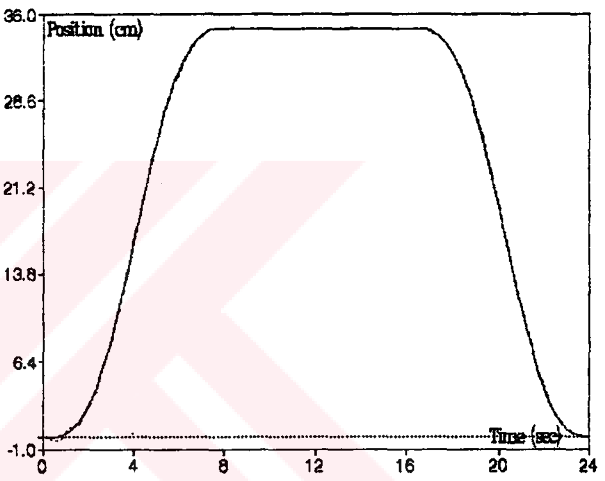
a) Response of revolute joint and desired reference trajectory when conventional control is used.



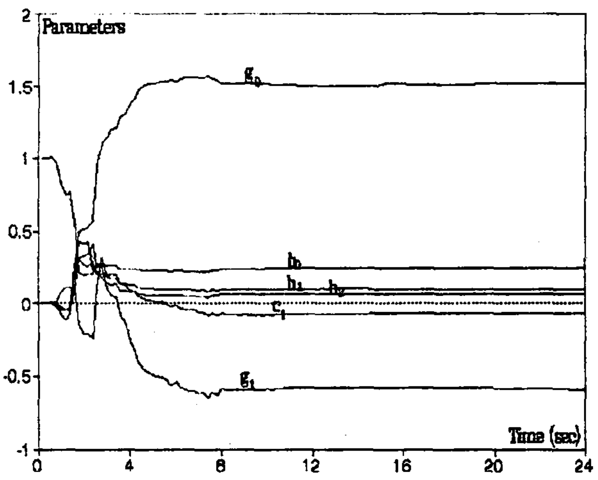
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



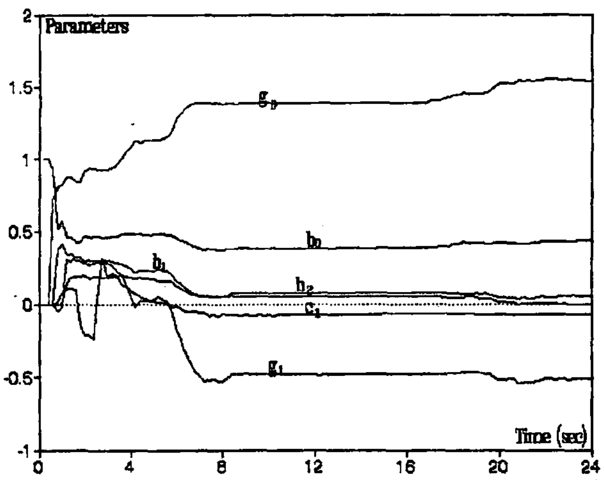
b) Response of revolute joint and desired reference trajectory when implicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when implicit STAC algorithm is used.



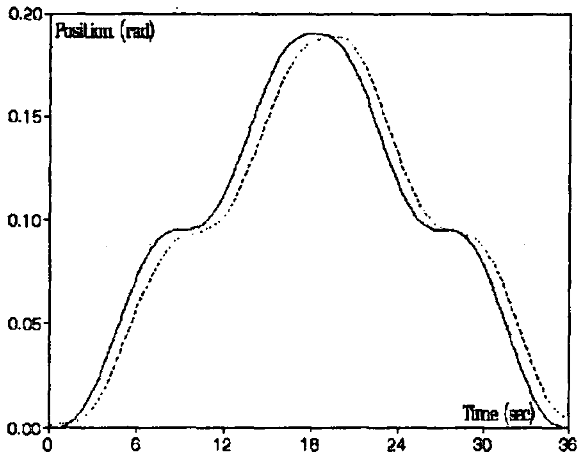
c) Changing of controller parameters versus time.



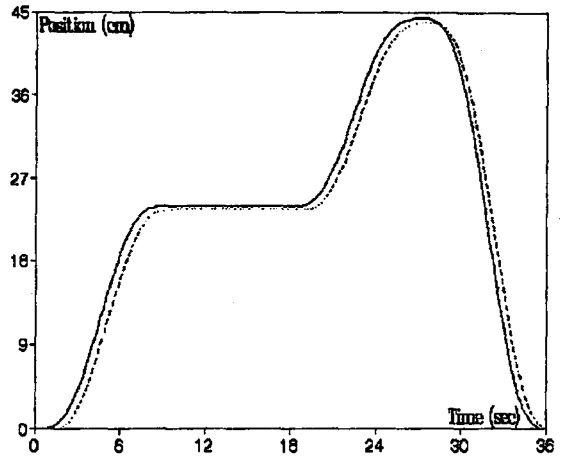
f) Changing of controller parameters versus time.

__________ Desired reference trajectory .... Response of the system

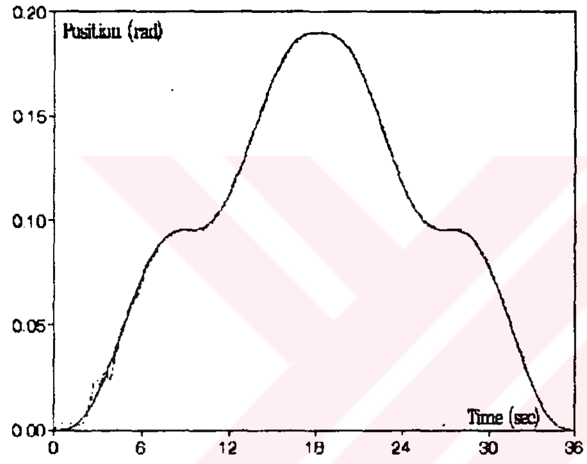
Figure 4.11 Experimental results on Hydraulic robot manipulator for case 4



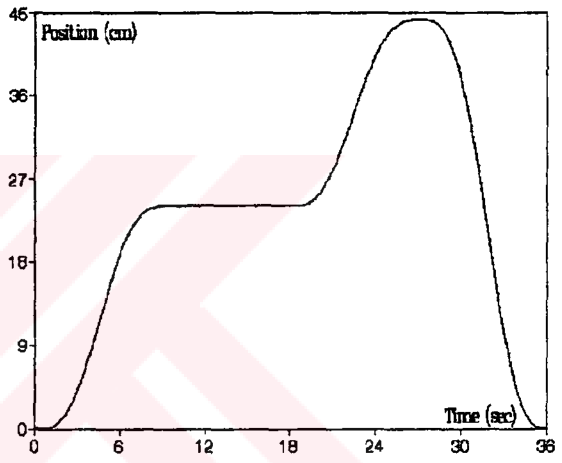
a) Response of revolute joint and desired reference trajectory when conventional control is used.



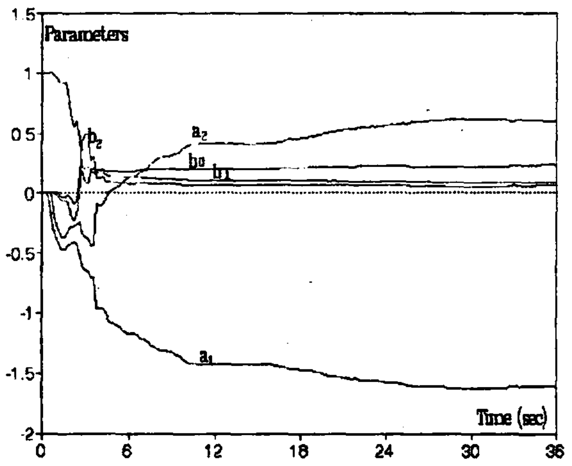
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



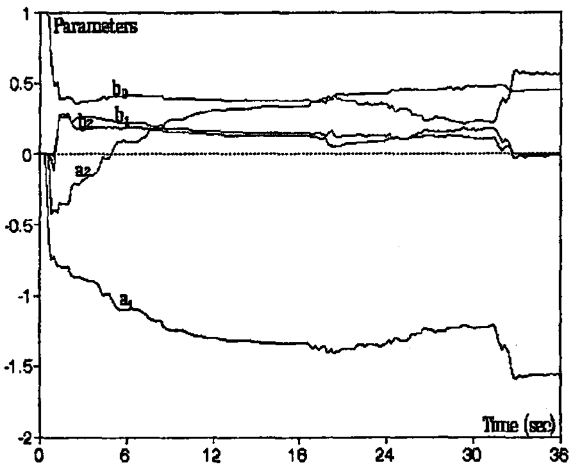
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



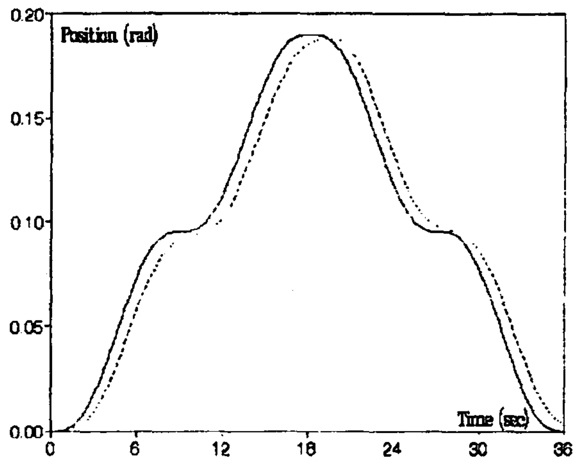
c) Changing of system parameters versus time.



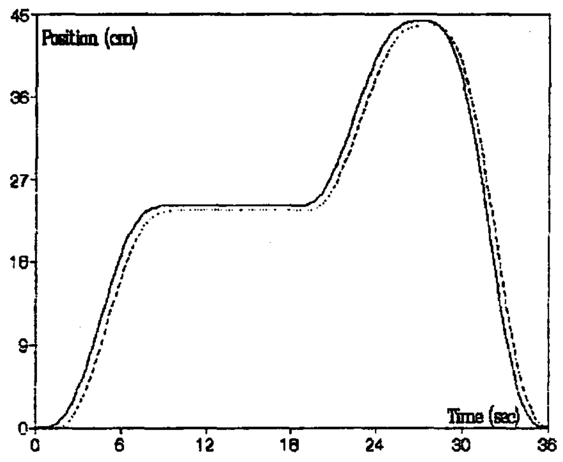
f) Changing of system parameters versus time.

__________ Desired reference trajectory .... Response of the system

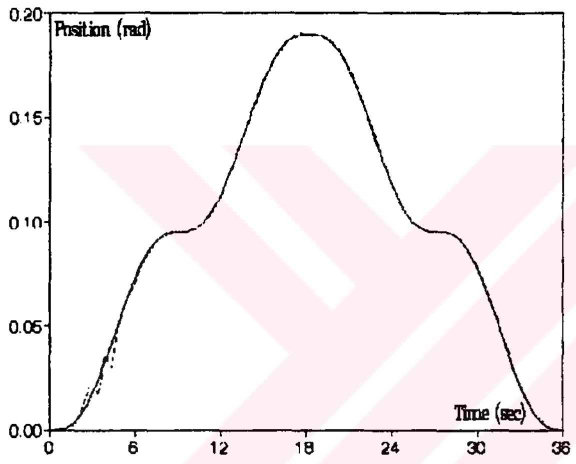
Figure 4.12 Experimental results on Hydraulic robot manipulator for case 1



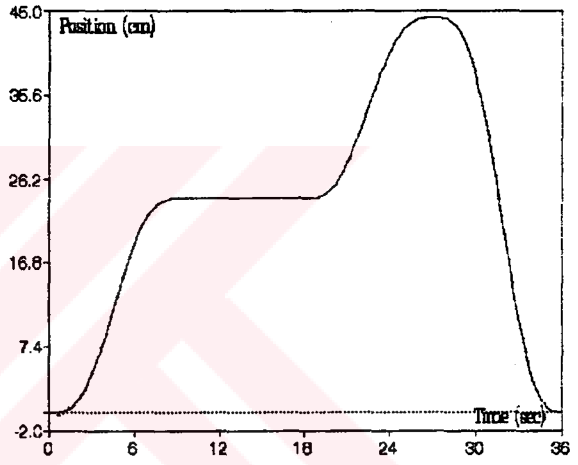
a) Response of revolute joint and desired reference trajectory when conventional control is used.



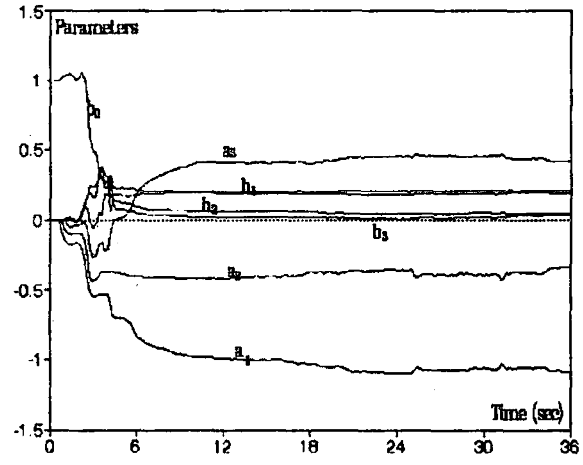
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



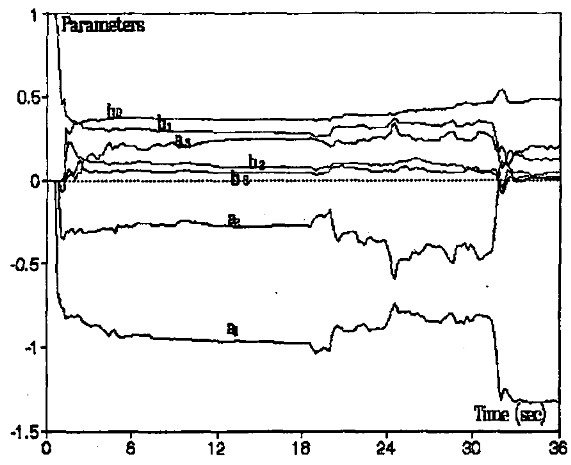
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



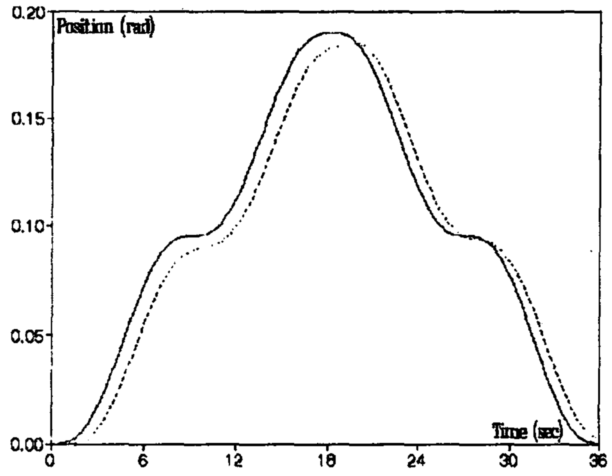
c) Changing of system parameters versus time.



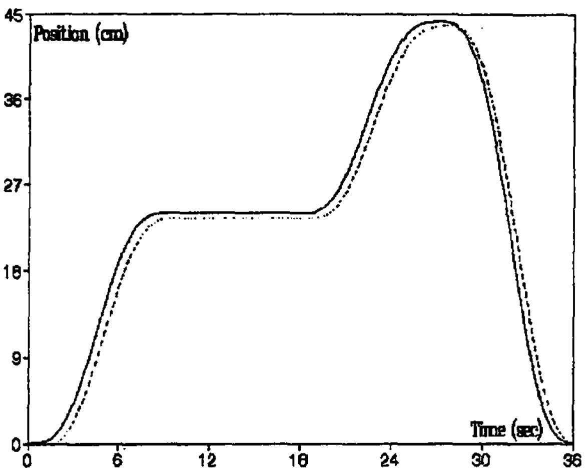
f) Changing of system parameters versus time.

__________ Desired reference trajectory .... Response of the system

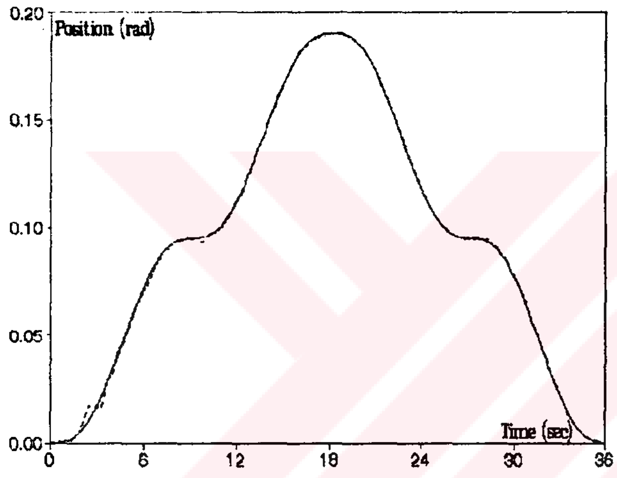
Figure 4.13 Experimental results on Hydraulic robot manipulator for case 2.



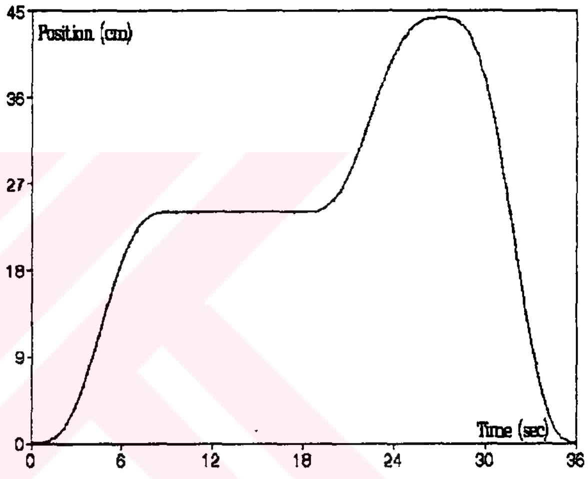
a) Response of revolute joint and desired reference trajectory when conventional control is used.



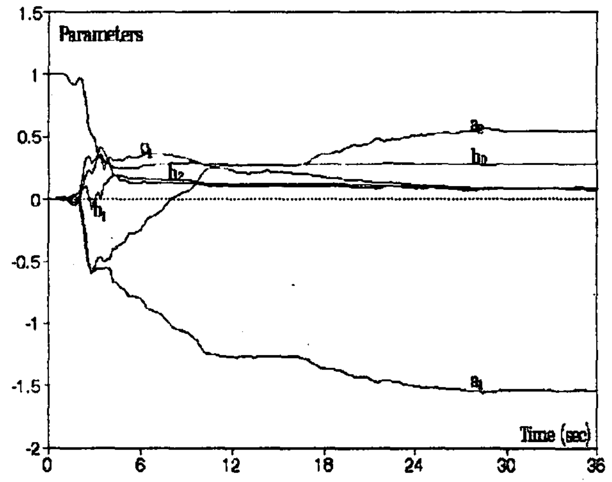
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



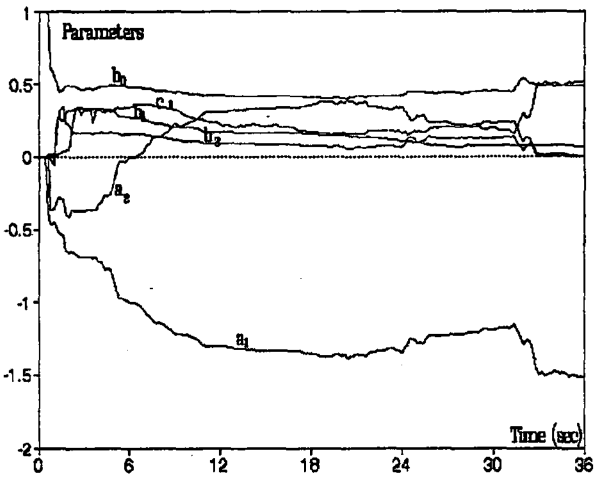
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.



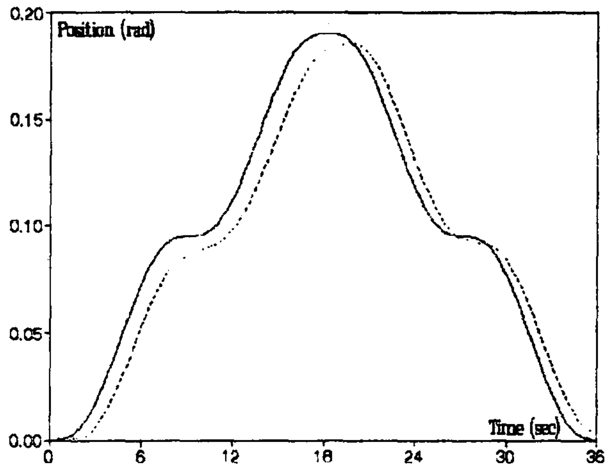
c) Changing of system parameters versus time.



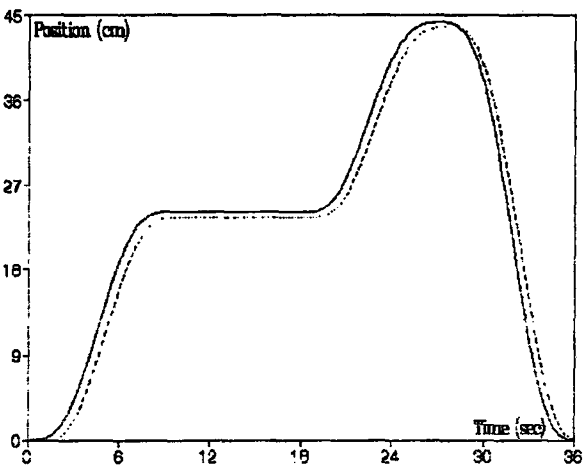
f) Changing of system parameters versus time.

___________ Desired reference trajectory .... Response of the system

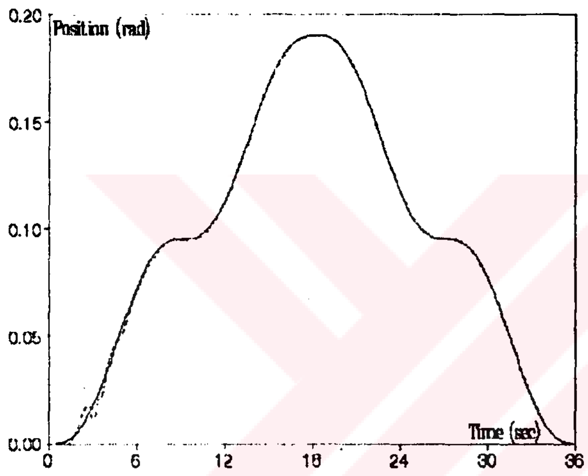
Figure 4.14 Experimental results on Hydraulic robot manipulator for case 3.



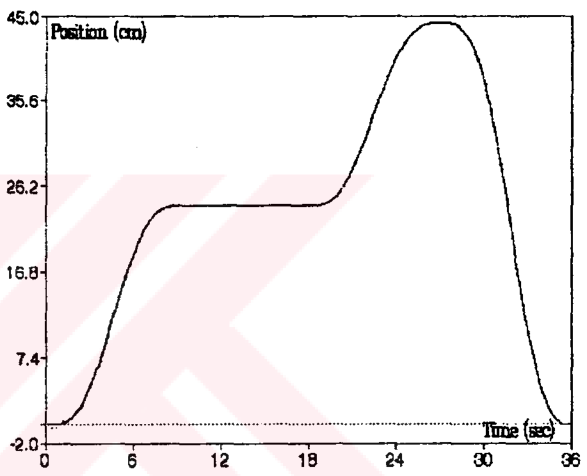
a) Response of revolute joint and desired reference trajectory when conventional control is used.



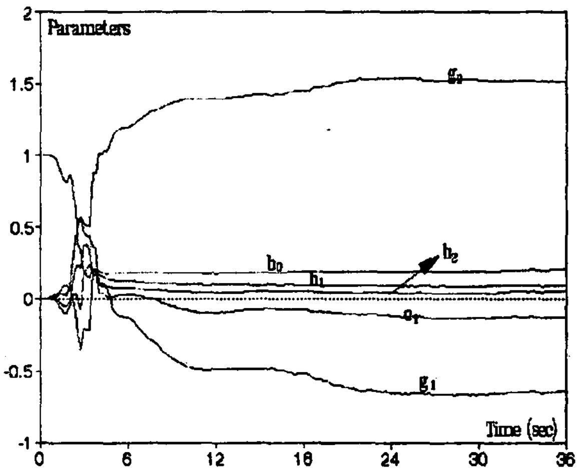
d) Response of prismatic joint and desired reference trajectory when conventional control is used.



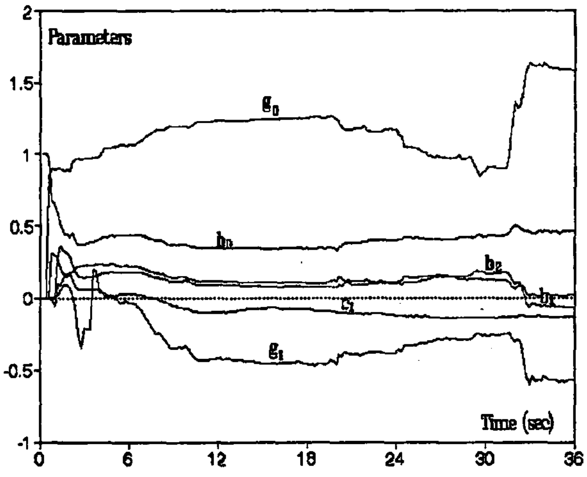
b) Response of revolute joint and desired reference trajectory when implicit STAC algorithm is used.



e) Response of prismatic joint and desired reference trajectory when implicit STAC algorithm is used.



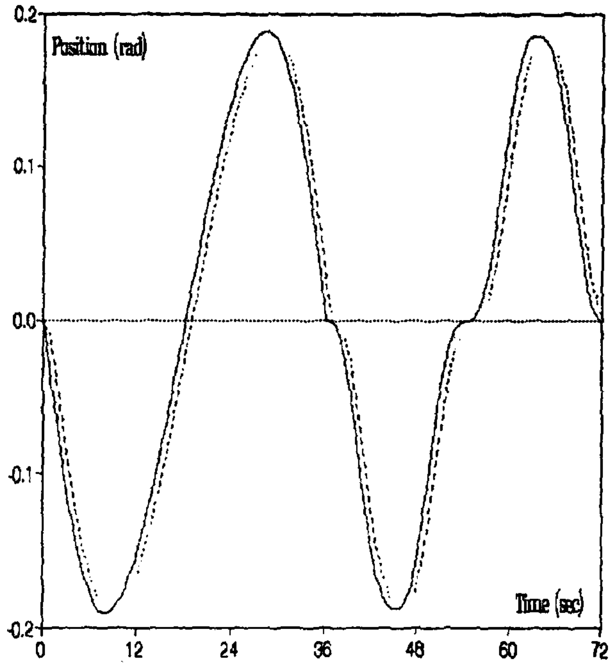
c) Changing of controller parameters versus time.



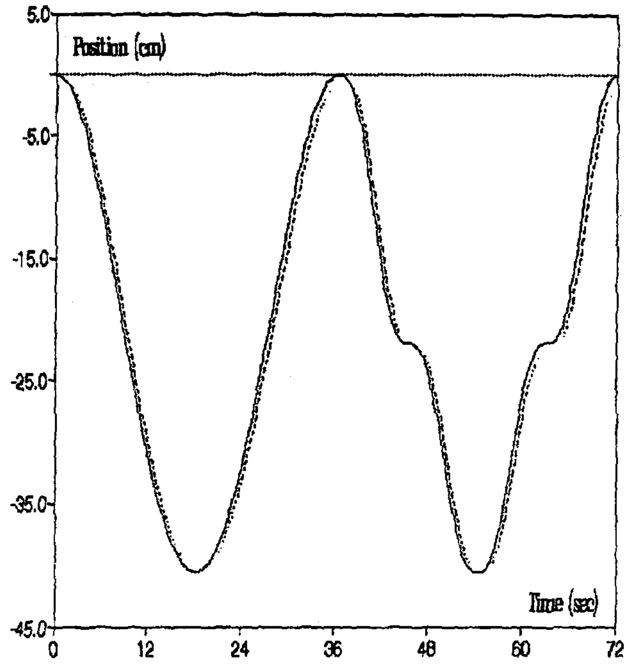
f) Changing of controller parameters versus time.

__________ Desired reference trajectory .... Response of the system

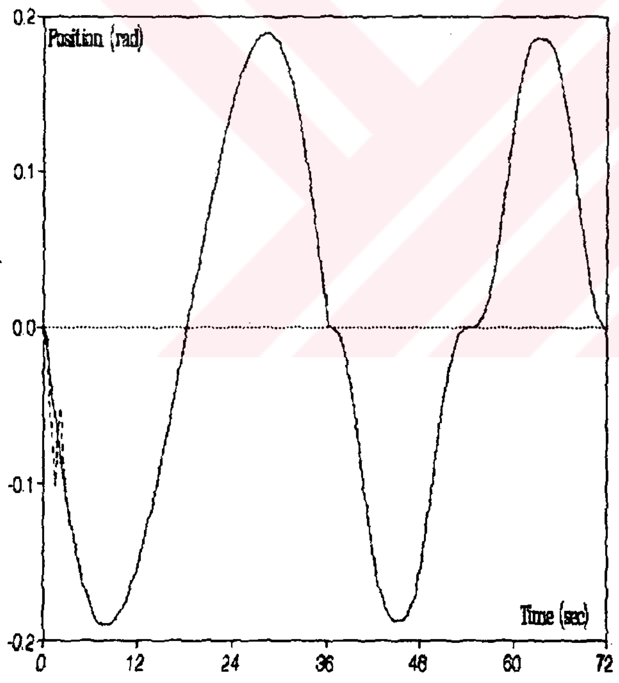
Figure 4.15 Experimental results on Hydraulic robot manipulator for case 4.



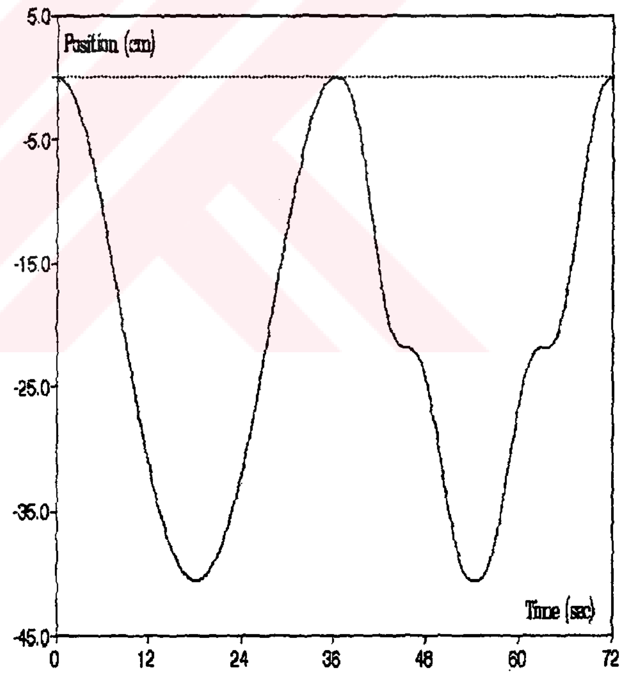
a) Response of revolute joint and desired reference trajectory when conventional control is used.



c) Response of prismatic joint and desired reference trajectory when conventional control is used.



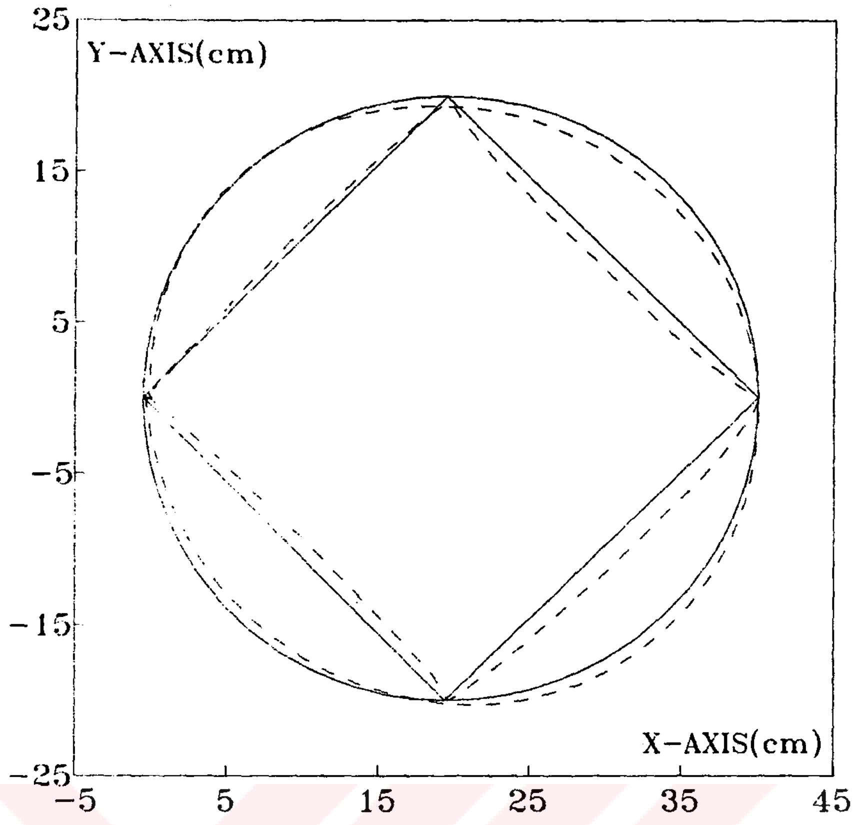
b) Response of revolute joint and desired reference trajectory when explicit STAC algorithm is used.



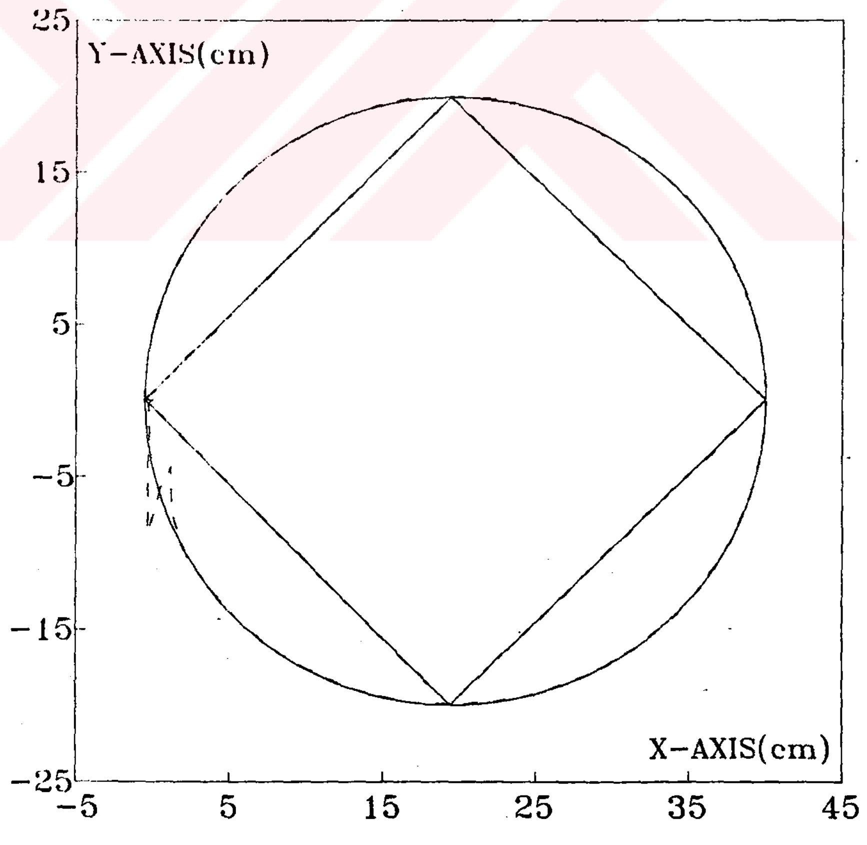
d) Response of prismatic joint and desired reference trajectory when explicit STAC algorithm is used.

__________ Desired reference trajectory .... Response of the system

Figure 4.16 Experimental results on Hydraulic robot manipulator in joint co-ordinate for example 4.



a) End-effector position in Cartesian co-ordinates when conventional control method is used



b) End-effector position in Cartesian co-ordinates when the explicit STAC is used.

__________ Desired reference trajectory .... Response of the system

Figure 4.17 Experimental results on Hydraulic robot manipulator in Cartesian co-ordinates for example 4.

4.5 Application of Self-Tuning Adaptive Control to Dc Servo System

This part investigates experimentally the application of explicit and implicit self-tuning adaptive control algorithms to a DC servo drive system. DC motors are extremely versatile drives, capable of reversible operation over a wide range of speeds, with accurate control of the speed at all times. They can be controlled smoothly from zero speed to full speed in both directions. DC motors have high torque to inertia ratios that give them quick response to control signal.

DC motors are used as actuators in many industrial robots for many reasons. They are easily controlled by a microprocessor. The experimental system is a direct drive system that means the motors are coupled directly to the load without the use of belts or gears. This enables high speeds to be achieved and avoids the problems of backlash.

The same adaptive control methods (second order, third order and second order with noise explicit self-tuning adaptive control methods and second order with noise implicit self-tuning adaptive control method) are implemented to the servo motor-driven system in order to examine these adaptive control methods on different systems.

4.5.1 Experimental Results

Four examples are presented to demonstrate the effects of self-tuning adaptive control algorithms. The sample time is selected as 0.083 second for all implementation.

Example 1

The experimental results of Case 1, Case 2, Case 3 and Case 4 are shown in Figure 4.18 - 4.21. As seen from these figures the desired trajectory of servo system are smooth and simple. The actual measurement of servo motor represented by dashed lines and the desired trajectory represented by solid lines.

Figure 4.18 *a* - 4.21 *a* are obtained for each case respectively when conventional control method is used. As seen from these figures that there are steady state tracing errors between motor positions and desired trajectories. However, when explicit self-tuning algorithm for Case 1, 2 and 3 and implicit self-tuning algorithm for case 4 are applied to servo motor, the tracking errors between motor outputs and desired trajectories are much better (see Figure 4.18 *b* - 4.21 *b*). Figure 4.18 *c* - 4.21 *c* shows the variation of parameters of the system.

Example 2

The experimental results for all cases are given in Figure 4.22 - 4.25 respectively. This example is given as an example of tracking of a difficult trajectory because this trajectory has an initial velocity and acceleration values at the both ends. This situation required a step change, at the velocity curve, therefore the response of motor exhibits a large tracking error at these points. However, a dramatic improvement is obtained in the response of the system when explicit and implicit algorithm are applied as shown in Figure 4.22 *b* - 4.25 *b*.

Example 3

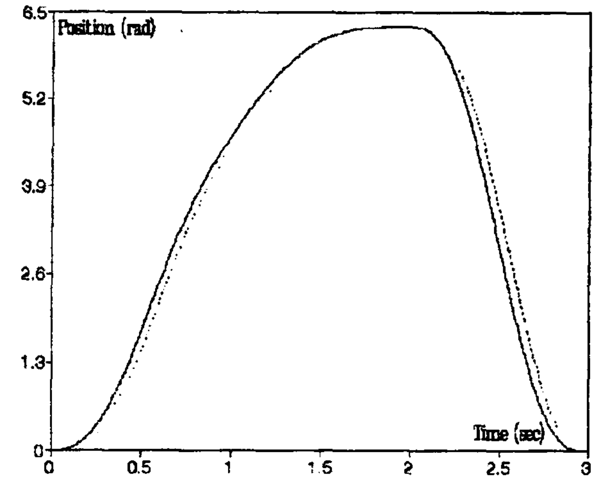
The trajectory is formed by 5 segments in this example. It starts with cycloidal rise segment and then a dwell segment takes place. In the third and fourth segments dwells follow again cycloidal rises. Finally, trajectory ends with a quick cycloidal return segment.

The experimental results for case 1, 2, 3 and 4 are shown in Figure 4.26 - 4.29 respectively. System responses fit to the desired trajectories when self-tuning adaptive control methods are used.

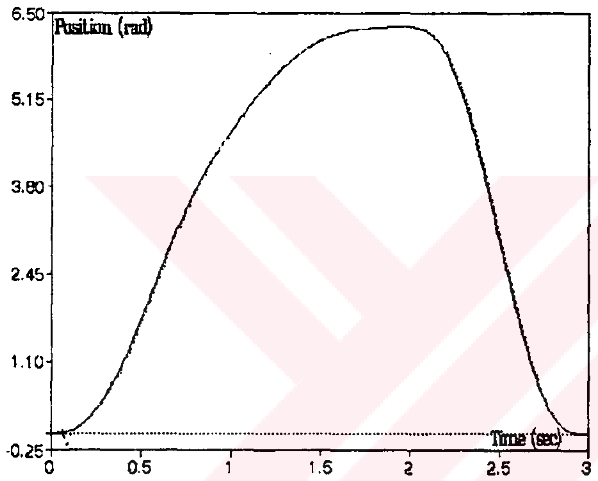
Example 4

Two different trajectories are planned for this example as shown in Figure 4.30 and 4.31. The trajectory, given in Figure 4.30 is formed from the combination of two same motion. The first trajectory is used to examine the response of the system at the starting point of second cycle when second order self-tuning adaptive control is applied. The result shows ones again that the reason of the oscillations is poor initial estimation of the parameter of the system.

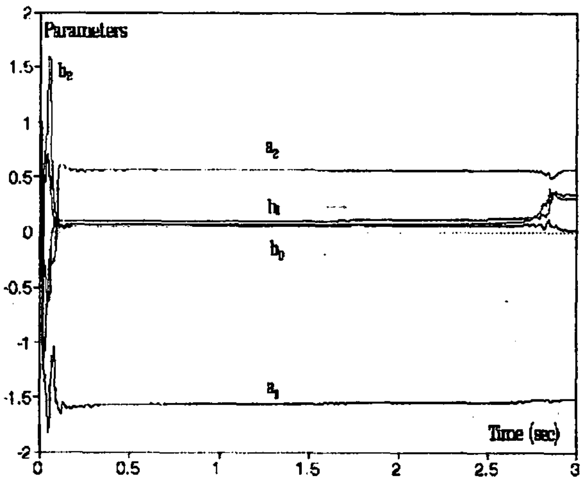
The second trajectory has several abrupt changes in the position curve as seen in Figure 4.31. In spite of large tracking errors of the response of the conventional control method, second order explicit self-tuning adaptive control algorithm provides very good tracking trace (see Figure 4.31 *b*).



a) Response of servo system and desired reference trajectory when conventional control is used.

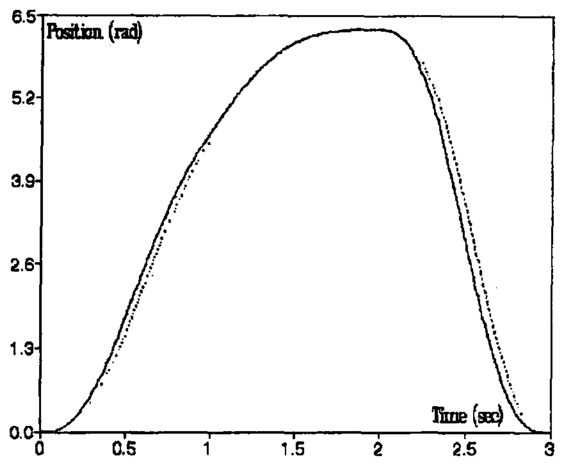


b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.

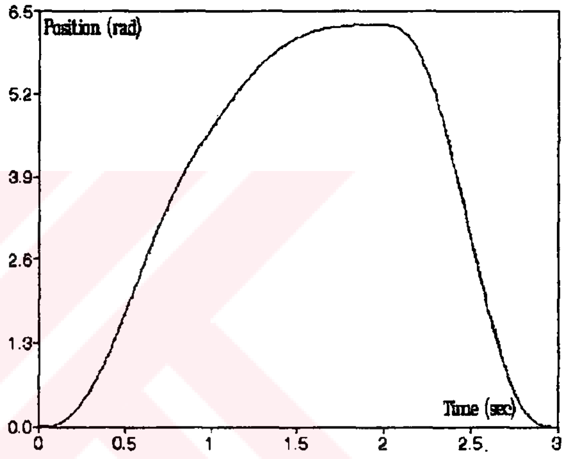


c) Changing of system parameters versus time.
____________ Desired reference trajectory

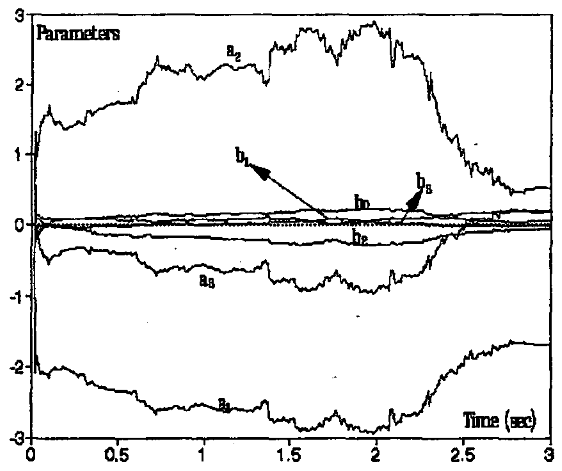
Figure 4.18 Experimental results on servo motor for case 1.



a) Response of servo system and desired reference trajectory when conventional control is used.

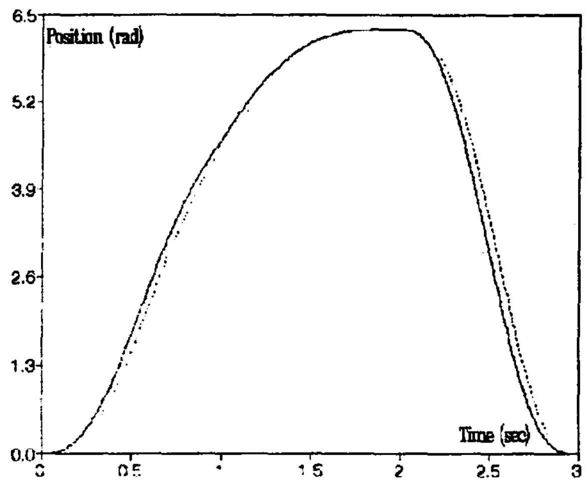


b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.

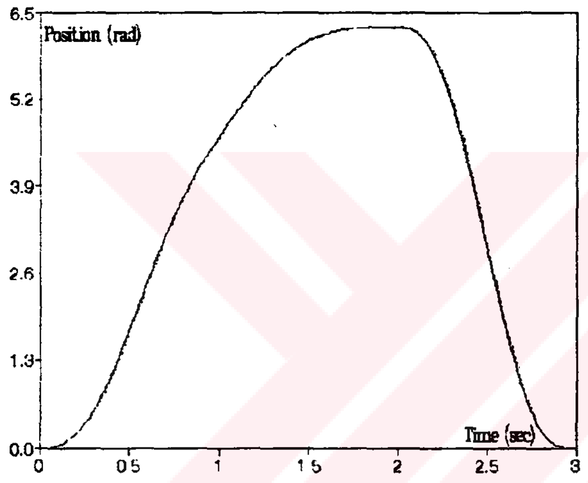


c) Changing of system parameters versus time.
.... Response of the system

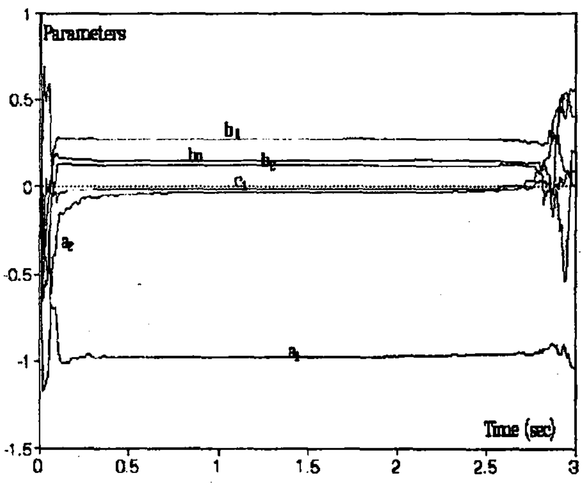
Figure 4.19 Experimental results on servo motor for case 2.



a) Response of servo system and desired reference trajectory when conventional control is used.

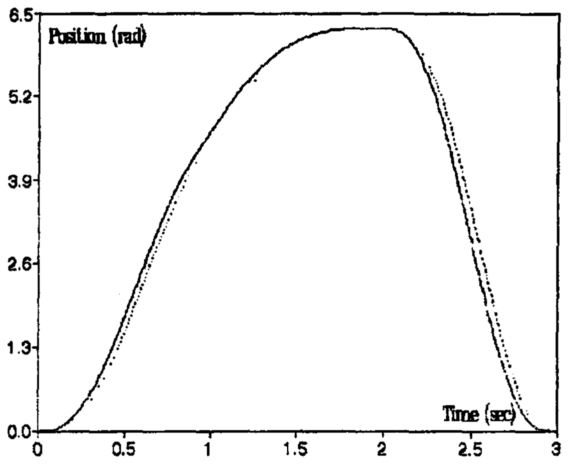


b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.

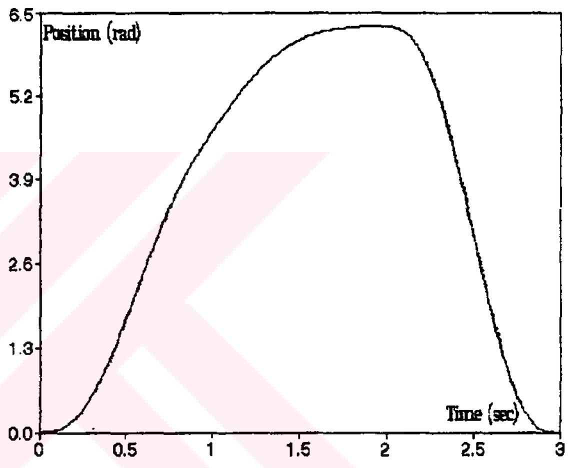


c) Changing of system parameters versus time.
__________ Desired reference trajectory

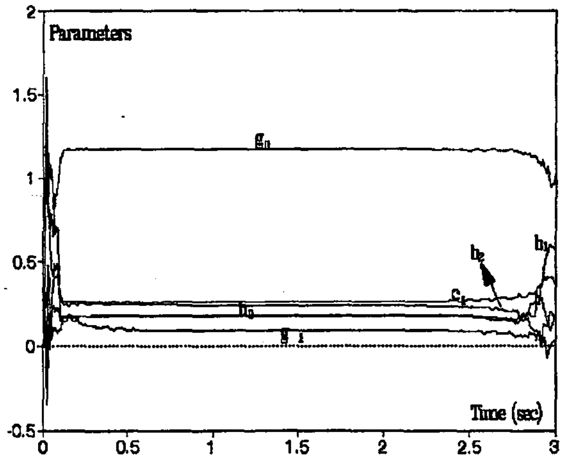
Figure 4.20 Experimental results on servo motor for case 3.



a) Response of servo system and desired reference trajectory when conventional control is used.

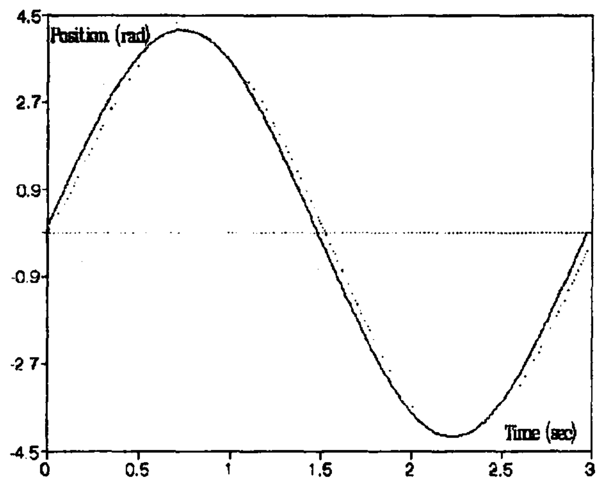


b) Response of servo system and desired reference trajectory when implicit STAC algorithm is used.

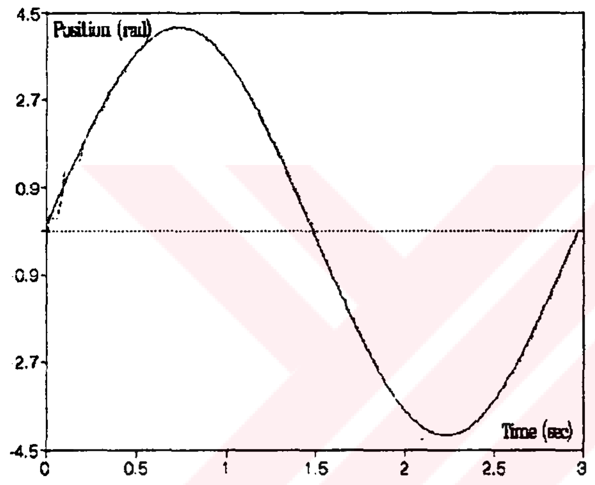


c) Changing of controller parameters versus time.
.... Response of the system

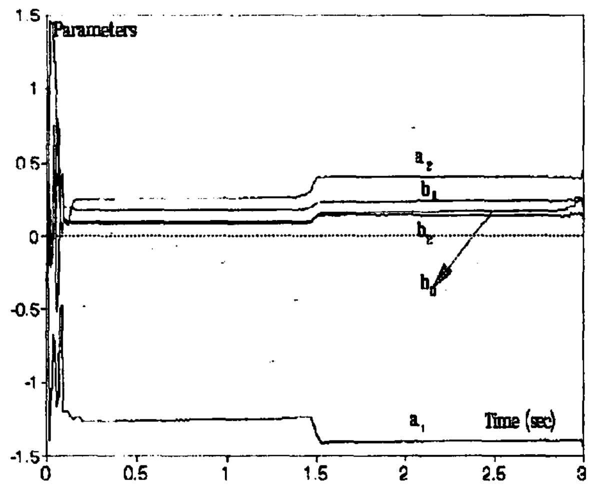
Figure 4.21 Experimental results on servo motor for case 4.



a) Response of servo system and desired reference trajectory when conventional control is used.

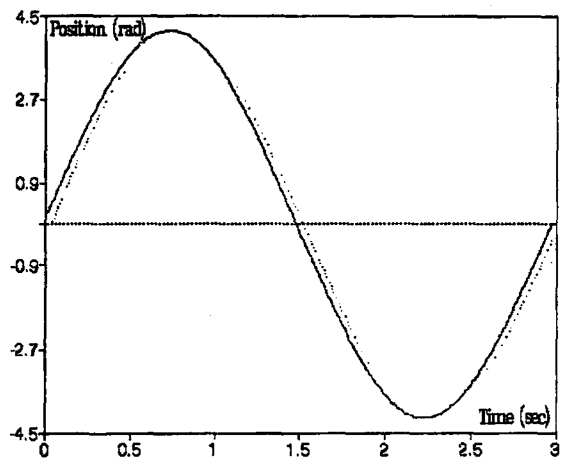


b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.

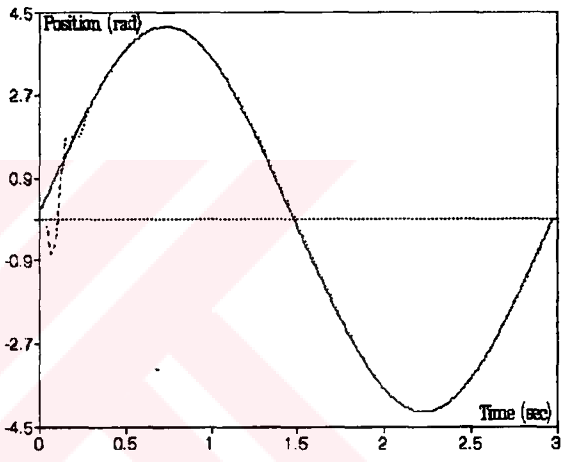


c) Changing of system parameters versus time.
__________ Desired reference trajectory

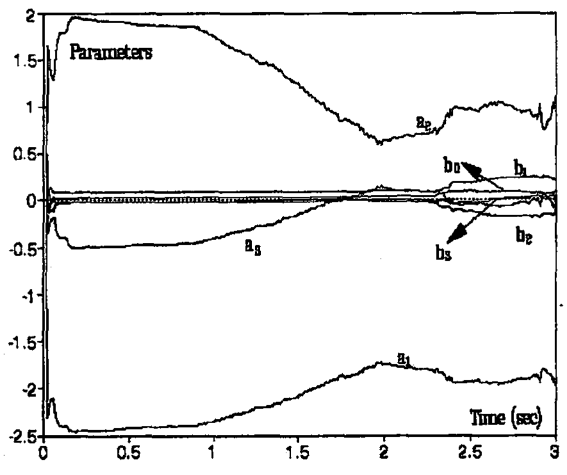
Figure 4.22 Experimental results on servo motor for case 1.



a) Response of servo system and desired reference trajectory when conventional control is used.

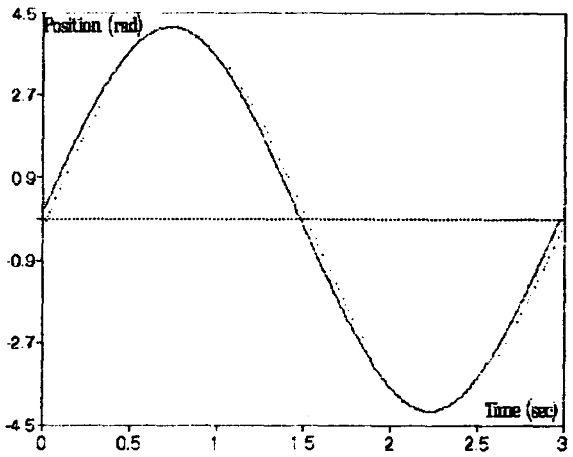


b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.

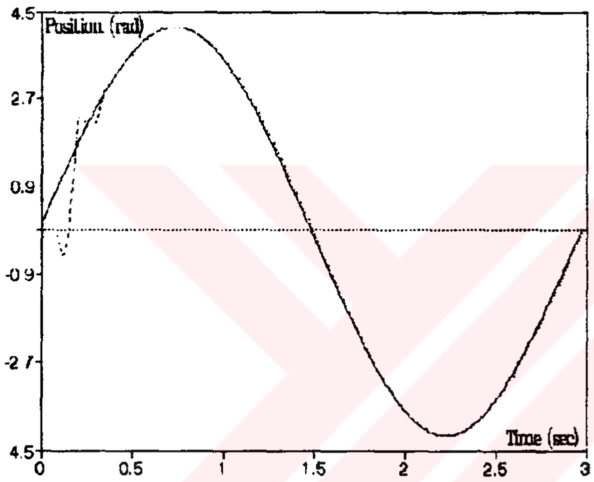


c) Changing of controller parameters versus time.
.... Response of the system

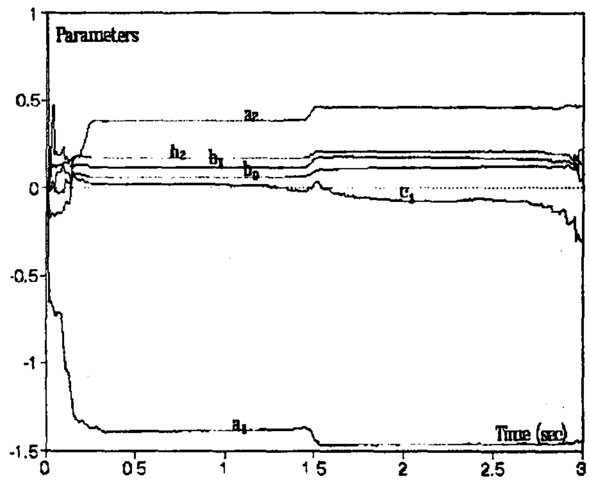
Figure 4.23 Experimental results on servo motor for case 2.



a) Response of servo system and desired reference trajectory when conventional control is used.



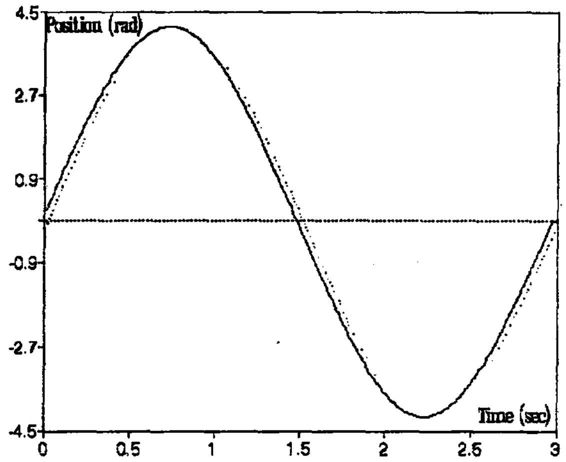
b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.



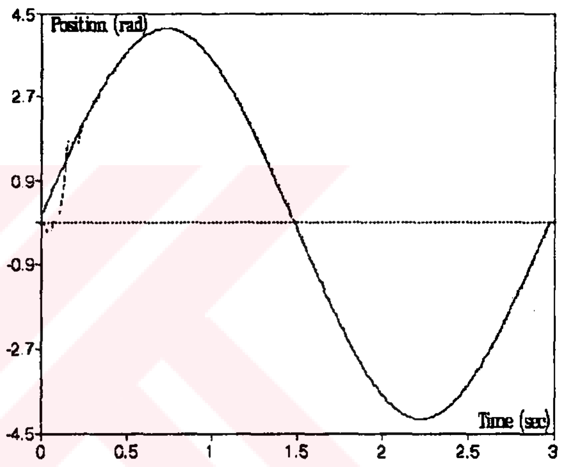
c) Changing of system parameters versus time.

__________ Desired reference trajectory

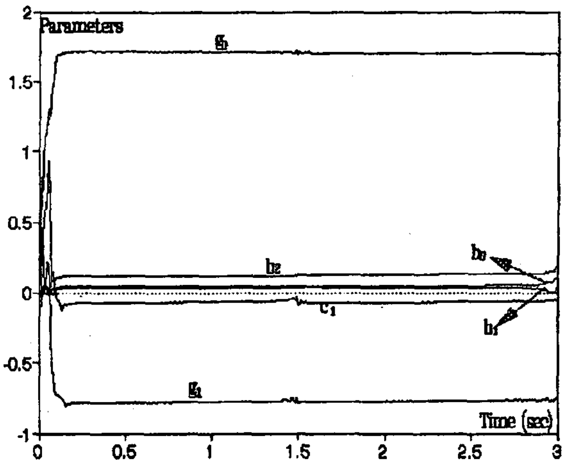
Figure 4.24 Experimental results on servo motor for case 3.



a) Response of servo system and desired reference trajectory when conventional control is used.



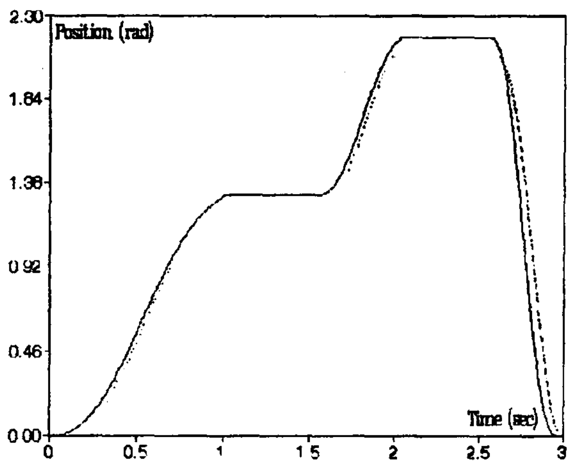
b) Response of servo system and desired reference trajectory when implicit STAC algorithm is used.



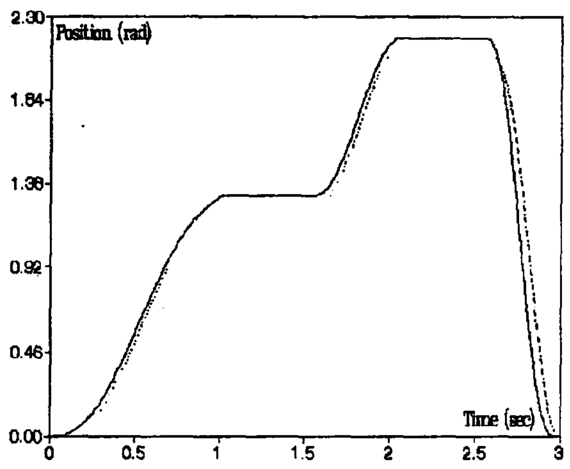
c) Changing of controller parameters versus time.

.... Response of the system

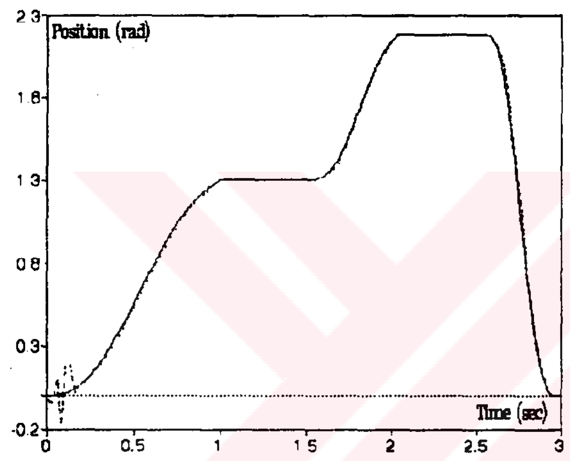
Figure 4.25 Experimental results on servo motor for case 4.



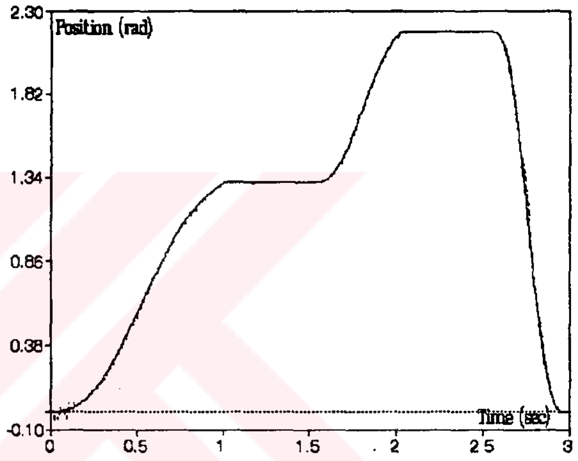
a) Response of servo system and desired reference trajectory when conventional control is used.



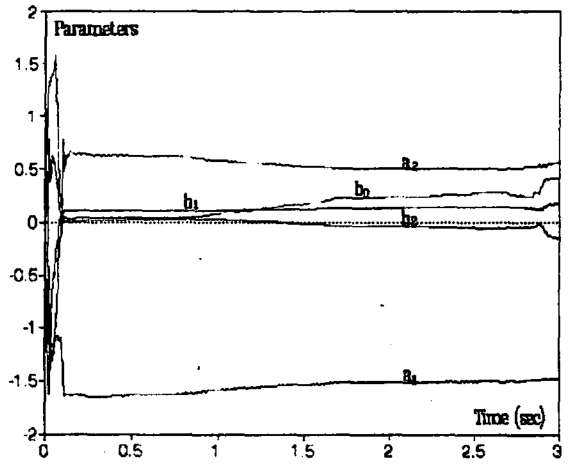
a) Response of servo system and desired reference trajectory when conventional control is used.



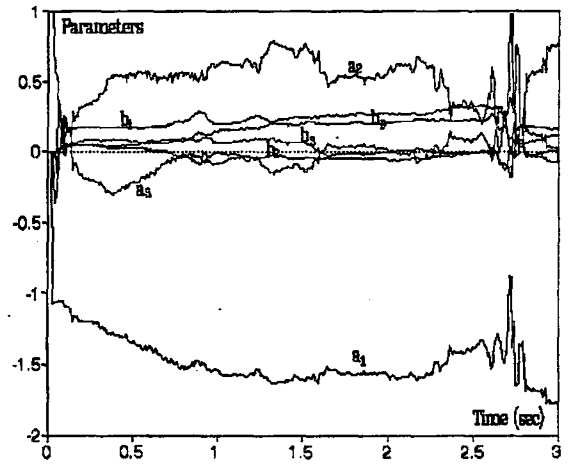
b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.



b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.



c) Changing of system parameters versus time.



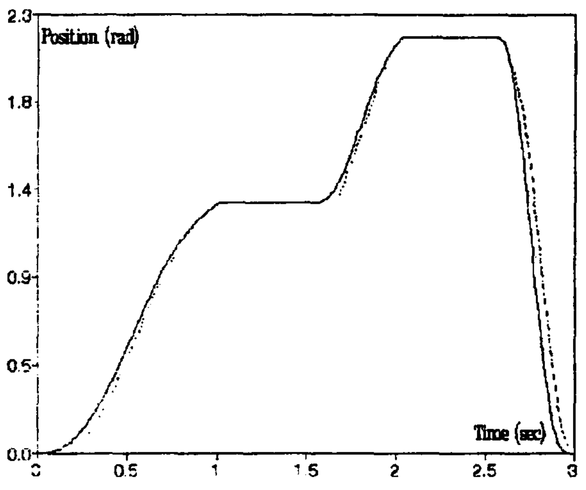
c) Changing of system parameters versus time.

__________ Desired reference trajectory

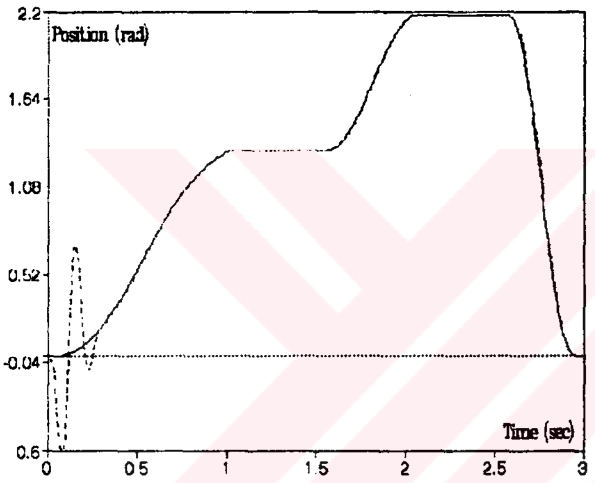
.... Response of the system

Figure 4.26 Experimental results on servo motor for case 1.

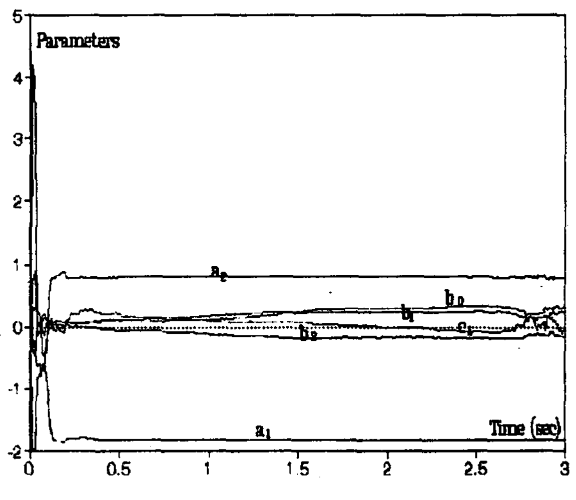
Figure 4.27 Experimental results on servo motor for case 2.



a) Response of servo system and desired reference trajectory when conventional control is used.

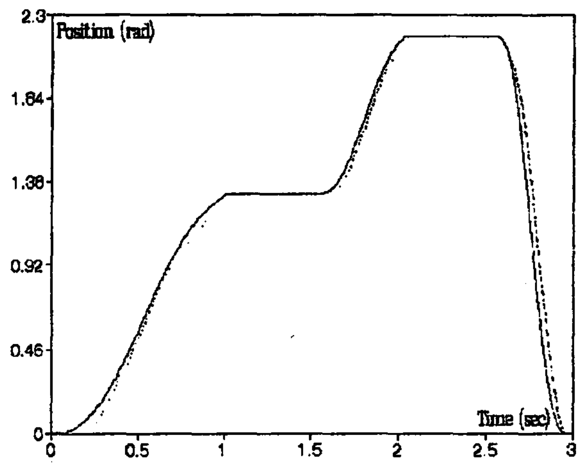


b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.

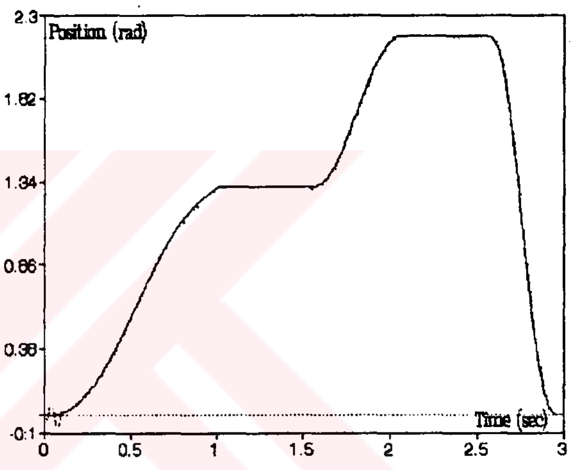


c) Changing of system parameters versus time.
_________ Desired reference trajectory

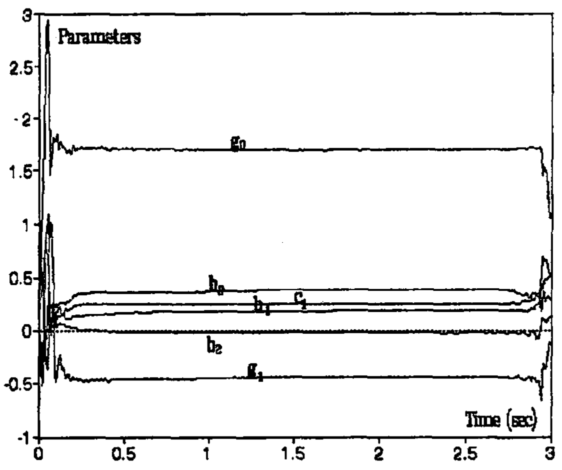
Figure 4.28 Experimental results on servo motor for case 3.



a) Response of servo system and desired reference trajectory when conventional control is used.

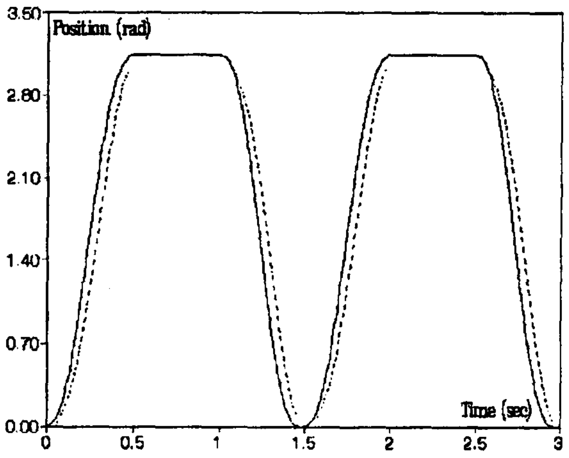


b) Response of servo system and desired reference trajectory when implicit STAC algorithm is used.

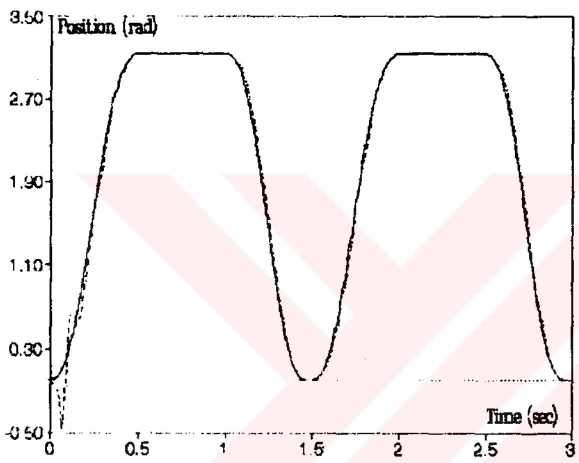


c) Changing of controller parameters versus time.
.... Response of the system

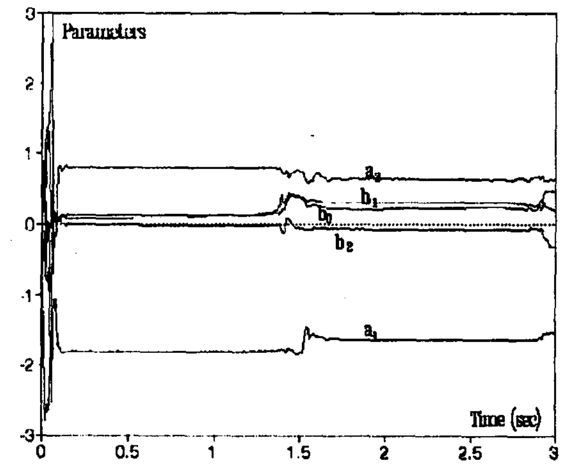
Figure 4.29 Experimental results on servo motor for case 4.



a) Response of servo system and desired reference trajectory when conventional control is used.

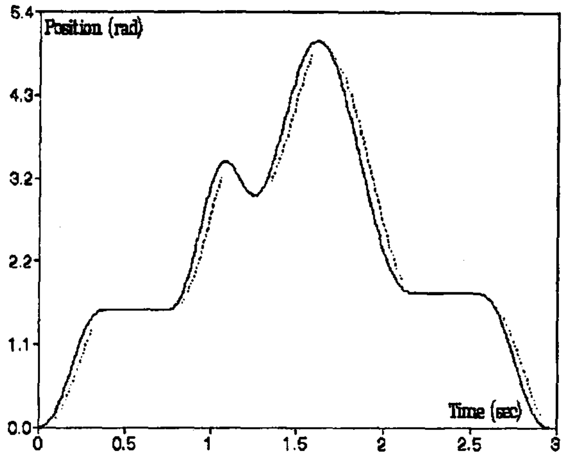


b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.

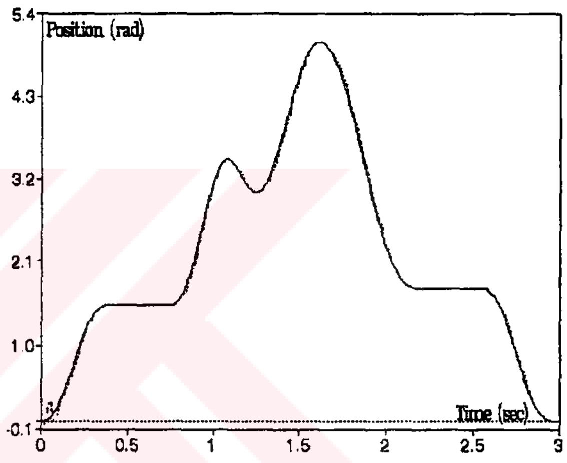


c) Changing of system parameters versus time.
__________ Desired reference trajectory

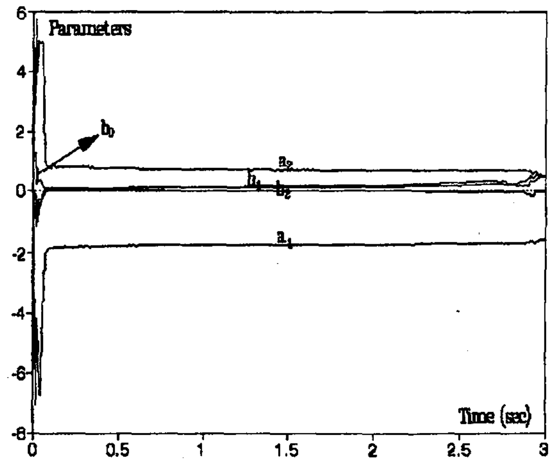
Figure 4.30 Experimental results on servo motor for case 1.



a) Response of servo system and desired reference trajectory when conventional control is used.



b) Response of servo system and desired reference trajectory when explicit STAC algorithm is used.



c) Changing of system parameters versus time.
.... Response of the system

Figure 4.31 Experimental results on servo motor for case 1.

CHAPTER 5

MODEL REFERENCE ADAPTIVE CONTROL

5.1 Introduction

Model reference adaptive control method (MRAC), an explicit adaptive technique, has been very attractive from the very beginning of the adaptive control area. In this control method the desirable dynamic characteristics of the system are specified in a reference model. This reference model is constructed in such a way as to track the reference input optimally as is the feedback controller. Its output is the reference signal of the name "model-reference".

The structure of model reference adaptive control is illustrated in Figure 5.1. As seen from this figure, the actual output of system is compared with the reference model output and the parameters of the controller are modified in such a way that the response of the system follows that of the reference model. This is done by an adaptation mechanism. Since this adaptation takes the form of adjustment of controller parameters so as to force the response of the resulting closed-loop control system towards that of desired model output. Therefore the adaptation mechanism must be designed so as to reduce to zero measured output error $(y_m(t) - y(t))$. This mechanism must be able on the basis of the observed output error, to determine which may adjust the controller coefficients and must also remain stable under all operating conditions. As a result, the concept of MRAC has depended on selecting an appropriate reference model and adaptation algorithm.

This control method is a useful approach to the control of the dynamic system with unknown parameters. Such a scheme generally requires complicated computations, therefore the adaptive controller is best implemented with a digital computer. From this point of view, the adaptive control system must be designed in the discrete form.

Some practical applications of MRAC have been reported [26, 27, 28, 29]. It is important that those algorithms used in practice are all relatively simple, and only a very limited set of theoretical results is used. This is mainly due to the fact that MRAC theory is based on assumptions that are generally not met complex algorithms may achieve improvements in theory, but in practise they make the system more sensitive to violations of the assumptions made, for example, in practice it is observed than less number of adjustable parameters is favoured, because convergence is generally slower if more parameters are used. Besides violation of the mentioned linearity requirement, for example may prohibit convergence altogether, especially if the number of adjustable parameters is large.

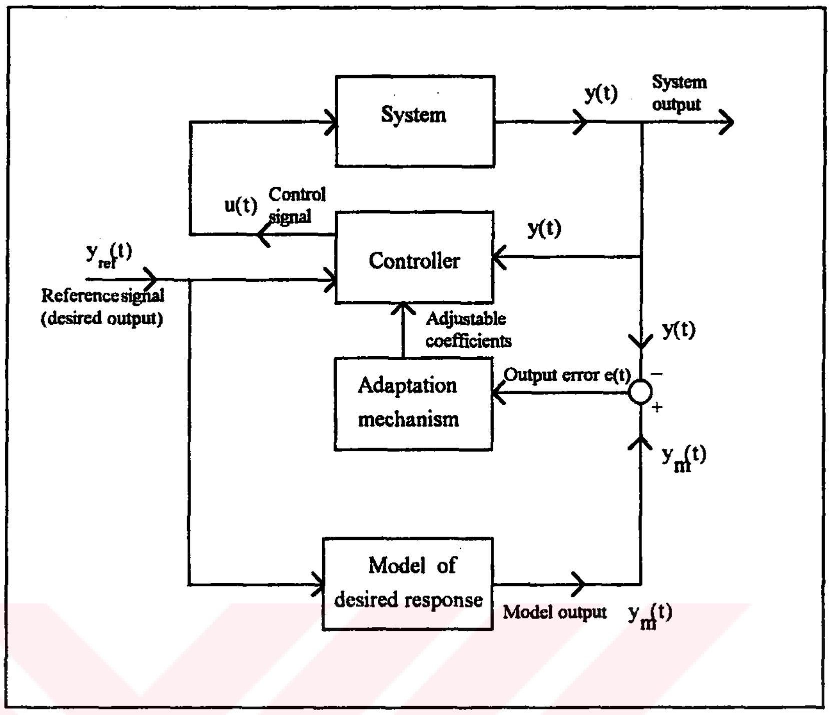


Figure 5.1 Structure of a model reference adaptive control

5.2 The Problem of Model Reference Adaptive Systems

For the robotic system with adjustable parameters the model reference adaptive method gives a general approach for adjusting the parameters so that the closed-loop transfer function will be close to a prescribed model. This is called the model-following problem. Therefore model following is an important part of the MRAC. This method can concern a number of very general control system design techniques by giving some desired properties or restrictions to the reference model. These are the convergence to zero of the output error, the convergence rate given by the poles of the reference model and the restriction of the bandwidth of the controlled system by a suitable choice of reference model. Due to these properties, the model reference adaptive control system is often one of the most feasible and effective approaches for an adaptive control system.

The problem in MRAC to be considered is to design an adaptive control system that will cause the error between the model and the system output to approach zero, without using any anticipate values of the system output.

Some author has treated this control problem in MRAC. In these studies, the adaptive law designs are based on the use of sensitivity models, the stability theory of Lyapunov and Popov's hyberstability theory, served as standard design methods, yielding a guaranteed stable adaptive system. That is, to guarantee that the control input and the system output remain bounded for all time, they have assumed the system to be linear, its transfer function to be minimum phase and completely disturbance free. That is, the MRAC is designed on the basis of the linearized model that may exhibit a non minimum phase or minimum phase type behaviour. However, it is a rule more than an exception to have unstable system zeros when the system output is sampled irrespective of whether the continuous system transfer function has them or not.

Two methods are suggested for solving this control problem in MRAC. One of this method is the designing of MRAC for non minimum phase discrete-time system using approximate inverse systems obtained from least-squares approximation [54]. The other method is the designing of indirect MRAC scheme using parameter estimation methods for system identification that explicitly minimises a H^2 criterion in LQG sense [31]. Both methods are suitable for applying to robotic systems. But, this study is only comprised the second method.

In the second method, H^2 criterion is a norm-space in advance mathematics in which this criterion is expressed generally gives the name of the method. The H^2 optimisation method has evolved in this way. In this method, the H^2-norm of the output error and the weight control input is minimised in LQG sense and that the stability of the closed loop system is guaranteed, regardless of whether or not the controller system is unstable or non minimum phase. Also, without the assumption that the unstable system zeros must be the zeros of the reference model. There are two reasons for minimising this criterion.

One of these is minimisation of output error under the characteristic polynomial of the closed-loop system contain stable system zeros. Thus, the stable system zeros are cancelled by the closed-loop system poles. Chih-Lyang Hwang and Bor-Sen Chen [31] have shown that the minimisation output error in this criterion. In order to determine a stable controller, the poorly damped modes should not be cancelled. If one of this stable system zeros is in the poorly damped region, the cancellation of stable system zeros will cause large ripple in control input. This large ripple control input will damage the actuator or motor in a mechanical system in practical application [55], and is not easy to implement because very large change rate of control input is required.

The other method is determined the suitable weighting function to control input. Tsai, Wang, Chih-Lyang Hwang and Bor-Sen Chen [31, 56] have shown that the determination weighting function to control input in this criterion based on the knowledge of the desired tolerable bounds of system frequency-response change. In this manner the suitable choice of weighting function to control input can reduce the effect of system parameter uncertainty.

5.2.1 System Model

Consider a single-input/single-output (SISO) linear discrete time-series of an Auto Regressive Exogenous (ARX) model can be used to represent the system with unknown parameters and assuming this system model does not have disturbances.

$$A(z^{-1})y(t) = z^{-d}B(z^{-1})u(t) \tag{5.1}$$

where

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} +, \ldots, + a_{n_a} z^{-n_a}$$

$$B(z^{-1}) = b_0 + b_1 z^{-1} +, \ldots, + b_{n_b} z^{-n_b} \qquad b_0 \neq 0$$

Where t sampling instant $t = n, n+1, \ldots \text{integer}\, n > 0$ n specifies the order of the model, $u(t)$ is control input for the system and $y(t)$ is system output. The positive integer d represents the time delay between the input and output of the system. Time delay is primarily the delay between the change in $u(t)$ and its effect on $y(t)$ and z^{-1} delay operator, that is $z^{-1}y(t) = y(t-1)$. n_a and n_b are degrees of $A(z^{-1})$ and $B(z^{-1})$ polynomials respectively. These polynomials are relatively prime (having unknown parameters) and it is assumed that d $(d \geq 1)$, n_a and n_b are known.

5.2.2 Reference Model for System

The input/output relationship between $y_{ref}(t)$ and $y_m(t)$ for a reference model can be written as

$$A_m(z^{-1})y_m(t) = z^{-d}B_m(z^{-1})y_{ref}(t) \tag{5.2}$$

The transfer function of the reference model can be written as

$$M(z^{-1}) = \frac{B_m(z^{-1})}{A_m(z^{-1})} \tag{5.3}$$

Hence,

$$y_m(t) = z^{-d}M(z^{-1})y_{ref}(t) \tag{5.4}$$

5.2.3 Controller Design

The goal of the controller is to make the output of the system track the desired reference model output. From Figure 5.2, the input and output relationship for a general controller can be written as

$$R(z^{-1})u(t) = T(z^{-1})y_{ref}(t) - S(z^{-1})y(t) \tag{5.5}$$

Where $R(z^{-1})$, $S(z^{-1})$ and $T(z^{-1})$ are polynomials of the controller that can be found, and $y_{ref}(t)$ is reference input.

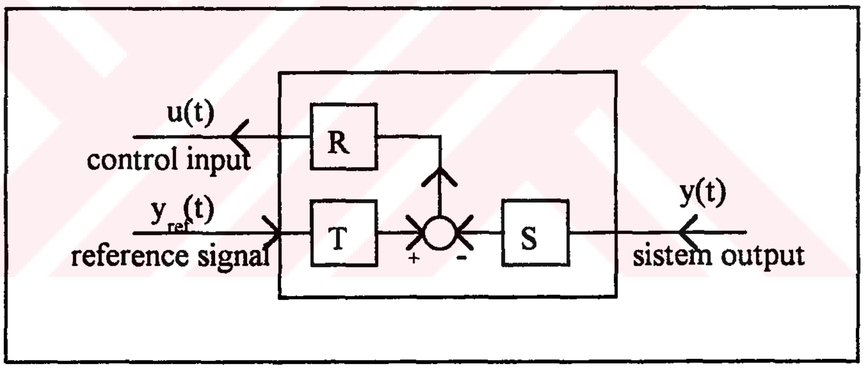


Figure 5.2 The inner structure of controller

The closed-loop system relationship between $y_{ref}(t)$ and $y(t)$ is obtained from Equation (5.1) and (5.5).

$$y(t) = \frac{z^{-d}B(z^{-1})T(z^{-1})y_{ref}(t)}{\left[R(z^{-1})A(z^{-1}) + z^{-d}B(z^{-1})S(z^{-1})\right]} \tag{5.6}$$

The characteristic polynomial of the closed-loop system, $L(z^{-1})$ can be written as

$$L(z^{-1}) = \frac{B_+(z^{-1})T(z^{-1})}{\left[R(z^{-1})A(z^{-1}) + z^{-d}B(z^{-1})S(z^{-1})\right]} \tag{5.7}$$

Hence,

$$y(t) = z^{-d}B_-(z^{-1})L(z^{-1})y_{ref}(t) \tag{5.8}$$

Where the notation of $B_+(z^{-1})$ represents the stable part of polynomial $B(z^{-1})$. On the other hand, $B_-(z^{-1})$ denotes the unstable part of polynomial $B(z^{-1})$i.e. $B(z^{-1}) = B_+(z^{-1})B_-(z^{-1})$.

The next step is to design an optimal controller that consists a stable transfer function of the system. This transfer function will be used to calculate the unknown parameters of the controller ($R(z^{-1})$, $S(z^{-1})$ and $T(z^{-1})$). Therefore, the following cost function in H^2-norm space is minimised [30, 31].

$$J = \left\| E(z^{-1}) + z^{-d}G(z^{-1})U(z^{-1}) \right\|_2 \tag{5.9}$$

Where $G(z^{-1})$ is an appropriately chosen rational weighting function, $E(z^{-1})$ and $U(z^{-1})$ are transfer function of output error ($e(t) = y_m(t) - y(t)$) and control input ($u(t)$) respectively.

$$E(z^{-1}) = z^{-d}\left[M(z^{-1}) - B_-(z^{-1})L(z^{-1})\right]\frac{N(z^{-1})}{D(z^{-1})} \tag{5.10}$$

$$U(z^{-1}) = \frac{A(z^{-1})L(z^{-1})N(z^{-1})}{B_+(z^{-1})D(z^{-1})} \tag{5.11}$$

Where $N(z^{-1})/D(z^{-1})$ is the transfer function of reference signal. $N(z^{-1})$ is a polynomial of degree n_n and $D(z^{-1})$ is a polynomial of degree n_d having all zeros on $|z^{-1}| \leq 1$.

Substituting Equation (5.10) and (5.11) into Equation (5.9) gives,

$$J = \left\| z^{-d}\left[M(z^{-1}) - B_-(z^{-1})L(z^{-1}) + \frac{G(z^{-1})A(z^{-1})L(z^{-1})}{B_+(z^{-1})}\right]\frac{N(z^{-1})}{D(z^{-1})} \right\|_2 \tag{5.12}$$

A rational function $Y(z^{-1})$ in H^2 is inner. If $Y(e^{-j\theta}) = 1$ for all $\theta \in [0, 2\pi]$, and is outer if it has no zeros in $|z^{-1}| \geq 1$ it is easy to check that multiplication by an inner function preserves the value of norm [56]. If $Y(z^{-1})$ is an inner function and $\|Y(z^{-1})Z(z^{-1})\|_2$ is well defined, then $\|Y(z^{-1})Z(z^{-1})\|_2 = \|Z(z^{-1})\|_2$. Based on this normpreserving of the inner function, it is immediate from Equation (5.12) that J is reduced to

$$J = \left\| \left[M(z^{-1}) - B_-(z^{-1})L(z^{-1}) + \frac{G(z^{-1})A(z^{-1})L(z^{-1})}{B_+(z^{-1})} \right] \frac{N(z^{-1})}{D(z^{-1})} \right\|_2 \tag{5.13}$$

Then problem being treated is equivalent to finding a stable rational function of the closed-loop system $L(z^{-1})$ such that Equation (5.13) is minimised. Once an optimal $L(z^{-1})$ is available, the polynomials $R(z^{-1})$, $S(z^{-1})$ and $T(z^{-1})$ can be obtained.

Let

$$G(z^{-1}) = \frac{G_1(z^{-1})}{G_2(z^{-1})} \tag{5.14}$$

$$C(z^{-1}) = G_2(z^{-1})B(z^{-1}) - G_1(z^{-1})A(z^{-1}) \tag{5.15}$$

Where the zeros of the monic polynomial $G_2(z^{-1})$ are on $|z^{-1}| > 1$, the polynomial $C(z^{-1})$ does not have zeros on the unit circle. Note that $G_1(z^{-1})$ and $G_2(z^{-1})$ should be chosen so that the zeros of the polynomial $C(z^{-1})$ are inside the unit circle. If $C(z^{-1})$ has a zero on $|z^{-1}| = 1$ we can modify this situation according to the definition of $A_-(z^{-1})$ that denotes the unstable part of polynomial $A(z^{-1})$.

Consider the system in Figure 5.2 and assume that the cost function Equation (5.13) is to be minimised, with weighting function Equation (5.14) and the definition of $C(z^{-1})$ given in Equation (5.15). Then the optimal controller Equation (5.5) is accomplished by solving the following Diophantine Equation to obtain $R(z^{-1})$ and $S(z^{-1})$ [31].

$$R(z^{-1})A(z^{-1}) + z^{-d}S(z^{-1})B(z^{-1}) = A_m(z^{-1})C_+(z^{-1})\overline{C}_-(z^{-1})N_+(z^{-1})\overline{N}_-(z^{-1})X(z^{-1}) \tag{5.16}$$

Where the notation of $\overline{C}(z^{-1})$means $z^{-n_c}C(z)$, deg $[C(z^{-1})] = n_c$ and $\overline{N}(z^{-1})$ means $z^{-n_n}N(z)$. $X(z^{-1})$ is an arbitrary, strictly Hurwitz polynomial, and by calculating the next equation to yield $T(z^{-1})$:

$$T(z^{-1}) = F_0(z^{-1})G_2(z^{-1})X(z^{-1}) \tag{5.17}$$

Where $F_0(z^{-1})$ is a polynomial with degree $h-1$ $(h = n_{m_a} + n_d)$ and its coefficients ($f_{0,i}$ $0 \le i \le h-1$) are obtained from the following equation.

$$\begin{bmatrix} 1 & z_1 & . & z_1^{h-1} \\ 1 & z_2 & . & z_2^{h-1} \\ . & . & . & . \\ 1 & z_h & . & z_h^{h-1} \end{bmatrix} \begin{bmatrix} f_{0,0} \\ f_{0,1} \\ . \\ f_{0,h-1} \end{bmatrix} = \begin{bmatrix} W(z_1) \\ W(z_2) \\ . \\ W(z_h) \end{bmatrix} \tag{5.18}$$

Where $z_i (1 \le i \le h)$ are distinct zeros of $A_m(z^{-1})D(z^{-1})$ and $W(z^{-1}) = B_m(z^{-1})N_+(z^{-1})\overline{N}_-(z^{-1})\overline{C}_-(z^{-1}) / C_-(z^{-1})$. The minimum cost function $J_{\min}$ is described as follows:

$$J_{\min} = \left\| \frac{F_c(z^{-1})}{C_-(z^{-1})} \right\|_2 \tag{5.19}$$

Where $F_c(z^{-1})$ is a polynomial of degree $q-1$ $(q = \deg[C_-(z^{-1})])$, and it is accomplished from Equation (5.20) after obtaining $F_0(z^{-1})$ from Equation (5.18) by using the method of undetermined coefficients:

$$F_c(z^{-1})A_m(z^{-1})D(z^{-1}) + F_0(z^{-1})C_-(z^{-1}) = B_m(z^{-1})N_+(z^{-1})\overline{C}_-(z^{-1})\overline{N}_-(z^{-1}) \tag{5.20}$$

If $C(z^{-1})$ is free of zeros in $|z^{-1}| \ge 1$ $(C_-(z^{-1}) = 1)$, using [31], $F_c(z^{-1}) = 0$ $(J_{\min} = 0)$ and $F_0(z^{-1}) = B_m(z^{-1})N_+(z^{-1})\overline{N}_-(z^{-1})$. Hence Equation (5.16) and (5.17) are replaced by the following equations :

$$R(z^{-1})A(z^{-1}) + z^{-d}S(z^{-1})B(z^{-1}) = A_m(z^{-1})C(z^{-1})X(z^{-1}) \tag{5.21}$$

$$T(z^{-1}) = B_m(z^{-1})G_2(z^{-1})X(z^{-1}) \tag{5.22}$$

Where $R(z^{-1}), S(z^{-1})$ and $T(z^{-1})$ are the polynoms of the controllers of degrees $n_r = n_b + d - 1$, $n_s = n_a - 1$ and $n_t = n_{m_b} + n_{n_{g2}}$ respectively. Then the optimal controller is accomplished by solving Diophantine Equation (5.21) and Equation (5.22) to obtain $R(z^{-1})$, $S(z^{-1})$ and $T(z^{-1})$ respectively.

5.3 Recursive Parameter Estimation

Recursive parameter estimator is the most important part the adaptive control algorithms. The main idea is to update continuously (recursively) the unknown parameters in the digital system model. If the degree of the polynomials n_a and n_b are known then the unknown system model Equation (5.1) is rewritten as

$$y(t) = \phi(t-1)^T . \theta \tag{5.23}$$

Where

$$\phi(t-1)^T = \left[-y(t-1),\dots,-y(t-n_a);\ u(t-d),\dots,u(t-d-n_b)\right] \tag{5.24}$$

$$\theta^T = \left[a_1\ a_2 \dots\ a_{n_a}\ b_0\ b_1 \dots\ b_{n_b}\ \right] \tag{5.25}$$

Where θ^T is a vector consisting of the parameters to be estimated, $\phi(t-1)^T$ is a vector of delayed control inputs and system outputs. The unknown parameters of model in Equation (5.23) can be estimated recursively by on-line calculations using U-D factorisation method. Let the vector of parameter estimates obtained at time instant t be denoted by $\hat{\theta}$.

5.4 Model Reference Adaptive Control Algorithm;

Step 1: Use recursive parameter estimation method such as Biermen U-D factorisation method to update $\hat{A}(z^{-1},t)$ and $\hat{B}(z^{-1},t)$

Step 2: Select the polynomial of $G_1(z^{-1})$ and $G_2(z^{-1})$ to obtain $\hat{C}(z^{-1},t)$ in Equation (5.15).

Step 3: Solve the Diophantine Equation (5.21) to obtain $\hat{R}(z^{-1},t)$ and $\hat{S}(z^{-1},t)$ and calculate Equation (5.22) to yield $T(z^{-1},t)$.

Step 4: The adaptive controller Equation (5.5) is achieved.

Step 1-4 are repeated at every adaptive step.

5.5 Application of MRAC Algorithm to the Hydraulic Robot

MRAC has been applied to a hydraulic robot to demonstrate the effectiveness of MRAC on tracing error. The robot has three axes, but, axis 2 and axis 3 (see Figure 2.1), which are working in a vertical plane, are used for the example trajectories.

In order to implement MRAC to the system, first of all, it is required to design a reference model which represent the desired performance of the system. Some of the parameters, such as the system model and the reference model orders and time delay have to be defined in advance. All the experimental studies show that second order system model and the reference models give good tracking. Therefore they are selected as second orders ($n_a = n_b = n_{m_a} = n_{m_b} = 2$) for the following examples, and the time delay is taken as one ($d = 1$).

Consider the following single input-single output linear discrete time varying model with system unknown parameters and it is selected for each joint ith under the assumption of independent joint dynamics.

$$y_i(t) = -a_1^i y_i(t-1) - a_2^i y_i(t-2) + b_0^i u_i(t-1) + b_1^i u_i(t-2) + b_2^i u_i(t-3) \tag{5.26}$$

Where integer i refers to the ith joint, ($i = 1,2$ and 3). For the system, the input of joint $u_i(t)$ that is voltage applied to the joint actuator, and outputs are position of the joints. The position of the second joint is angular (*rad*) and third joint is linear displacement (*cm*). Parameters of $A^i(z^{-1})$ and $B^i(z^{-1})$ polynomial are be determined by using the U-D factorisation method. The reference model for each joint is chosen as a linear, single input-single output model and discrete time model such as

$$y_{m_i}(t) = -a_{m_1}{}^i y_{m_i}(t-1) - a_{m_2}{}^i y_{m_i}(t-2) + b_{m_0}{}^i y_{ref_i}(t-1) + b_{m_1}{}^i y_{ref_i}(t-2) + b_{m_2}{}^i y_{ref_i}(t-3) \tag{5.27}$$

Where $y_{m_i}(t)$ is reference model output. The parameters of this reference model $a_{m_1}{}^i, a_{m_2}{}^i, b_{m_0}{}^i, b_{m_1}{}^i$ and $b_{m_2}{}^i$ for each joint are the same but they have fixed values. This model is obtained by using system identification method and the model matches with all the desired trajectories. The polynomial of reference model for each joints determined as;

$$A_m^i(z^{-1}) = 1 + 0.195z^{-1} + 0.005z^{-2} \text{ and } B_m^i(z^{-1}) = 1 + 0.8z^{-1} - 0.6z^{-2} \tag{5.28}$$

The initial values of estimated system parameter are set to the following.

$\hat{a}_1^i(0) = -1$, $\hat{a}_2^i(0) = 0$, $\hat{b}_0^i(0) = -0.5, \hat{b}_1^i(0) = 0.5$ and $\hat{b}_2^i(0) = 0.5$

The following settings are used for two joints and initial values of system input and output are equal to zero i.e. $y_i(-1) = y_i(0) = u_i(-2) = u_i(-1) = u_i(0) = 0$ for data vector in parameter estimation method. The forgetting factor γ_i is set equal to 0.98 for each joint. The initial diagonal and upper triangle matrix are set respectively such as; $D^i{}_g(0) = 1000I_m$ and $U^i{}_p(0) = I_m$ $(m = 5)$.

In this application, the weighting polynomial is chosen as $G^i(z^{-1}) = 0.3$ ($G_2{}^i(z^{-1}) = 1$ and $G_1{}^i(z^{-1}) = 0.3$) The effect of the values of weighting polynomials on the performance of system will be discussed later. Hurwitz polynomial in Equation (5.21) and (5.22) is arbitrary selected as $X'(z^{-1}) = 1$.

The reference model output is described by the discrete points in the joint space. The control for the motion of manipulator is discussed by a designing a controller with MRAC property for each joint position on the basis of a SISO. In this stochastic SISO-model, the system work under the position control mode. The inputs for the system are voltages that are directly proportional with the angular rotation and linear displacement of the potentiometers of the system.

5.5.1 Experimental Results

Several examples are tested on the hydraulic robot to demonstrate the capability of the MRAC algorithm. All the trajectories used in this study has been designed by the motion design software called ". MOTDES".

Example 1

The experimental results for revolute and prismatic joints are shown in Figure 5.3 and 5.4 respectively. As seen from the figures, the trajectories of both joints are smooth and similar. Sample time is chosen as *0.2* seconds. In Figure 5.3 *a-b* and Figure 5.4 *a-b* the dashed lines correspond to actual measurement output of each joint variable while solid lines are the reference model outputs. Figure 5.3 *a* and 5.4 *a* are obtained by using conventional control method. It is seen that, there is a steady state tracing error between the response of joints (dashed lines) and the reference model output (solid lines). The graphs in Figure 5.3 *b* and 5.4 *b* demonstrate that good trackings are achieved for each joint using MRAC method. Initially, the responses of joints oscillates while trying to follow the reference model output and then settles down. Each joint of the manipulator has a controller in the same form, naturally the parameter values are different.

Figure 5.3 *c* and 5.4 *c* show the model parameters for revolute and prismatic joint respectively. All the variables change with time. The controller parameters for each joint are shown in Figures 5.3 *d* and 5.4 *d*.

Example 2

The trajectories of this example are similar to those given in Example 1, however, the stroke of the prismatic joint is increased to 45 cm and the sample time is selected as *0.1* seconds to examine the manipulator at higher speeds. It is seen that the tracing error of Figure 5.5 *a* and 5.6 *a* are larger than the tracking error of Figure 5.3 *a* and 5.4 *a* when conventional control is used. This situation shows that the tracing error is directly proportional with the speed of system. However, the tracking error tends to zero when MRAC algorithm is used (see Figure 5.5 *b* and 5.6 *b*), but, some small amount of oscillations may be seen at the beginning of motion due to the roughly estimation of model parameters.

The time evolution of the each joint model parameters is shown in Figure 5.5 *c* and 5.6 *c* . Figure 5.5 *d* and 5.6 *d* show the controller parameter variation with respect to time for each joint variable.

Example 3

The reference models and responses of the robot for revolute and prismatic joints are shown in Figure 5.7 and 5.8 respectively. It is seen that these joint trajectories are more difficult than the trajectories of previous examples. The sampling time is taken as 0.1 seconds.

Figure 5.7 *a* and 5.8 *a* are obtained with conventional control method. As it is seen that, when the MRAC algorithm is applied to the robot the tracking error of both joints is dramatically reduced (see Figure 5.7 *b* and 5.8 *b*). The changing of the system model parameters and the controller parameters for each joint with time are given in Figure 5.7 *c-d* and 5.8 *c-d* respectively.

Example 4

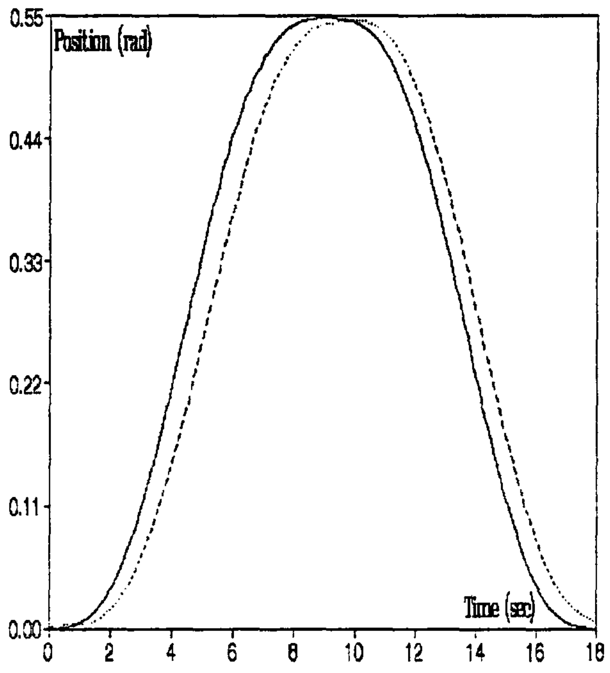
This is an example of the quite difficult trajectory, however, it is successfully planned and implemented to the robot. The sample time is chosen 0.2 seconds. The experimental result for this implementation is given in Figure 5.9 and 5.10 for each joint respectively. Figure 5.9 *a* and 5.10 *a* show the desired reference model output and the response of the revolute and prismatic joints when conventional control method is applied. Figure 5.9 *b* and 5.10 *b* show the improvement in the tracking error due to MRAC. Figure 5.9 *c-d* and 5.10 *c-d* show the changing of system model and controller parameters for each joint respectively during the execution of motions.

Example 5

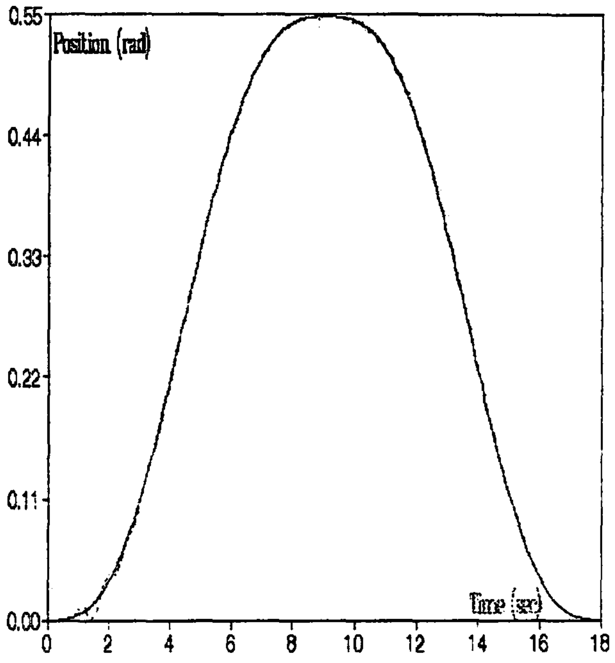
The same trajectory, that is previously designed to test self-tuning adaptive control on robot, is used for this example. The trajectory, is initially designed at end-effector level, however, the joint motions are obtained by inverse kinematic solution to control the system. For implementation, selecting of joint model and reference model order, initial values of parameter vector, data vector, diagonal and upper triangle matrix, forgetting factor, weighting function and Hurwitz polynomial are the same as the previous application.

The experimental results for revolute and prismatic joints are shown in the first and second column of Figure 5.11 respectively. Sample time is chosen as *0.2* seconds. Figure 5.11 *a* and *c* is obtained by using conventional control method. It is clear that there is offset between the response and the reference model. An acceptable execution of the job process may not be expected from the system with this amount of error. However, the curves in Figure 5.11 *b* and *d* show that very good results are achieved by using MRAC method.

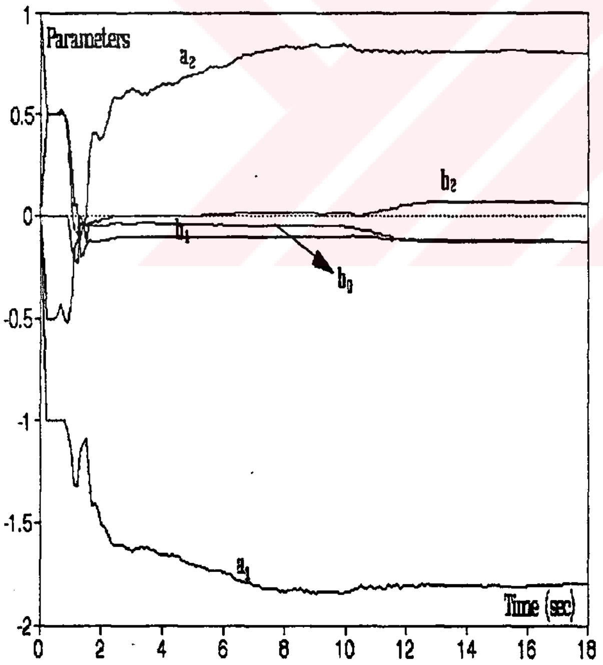
Figure 5.12 *a* and *b* shows the responses of the end effector of the robot when conventional and MRAC are applied respectively. A small oscillation is seen at the beginning of the motion when MRAC is applied, however, after a few seconds it settles down and then the robot produces an excellent tracking for the rest of the motion.

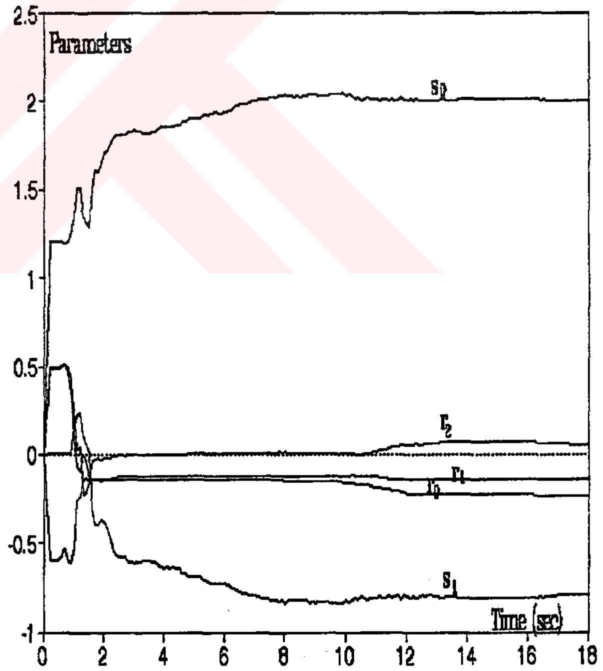


a) Response of revolute joint and reference model output when conventional control is used.



b) Response of revolute joint and reference model output when MRAC algorithm is used.





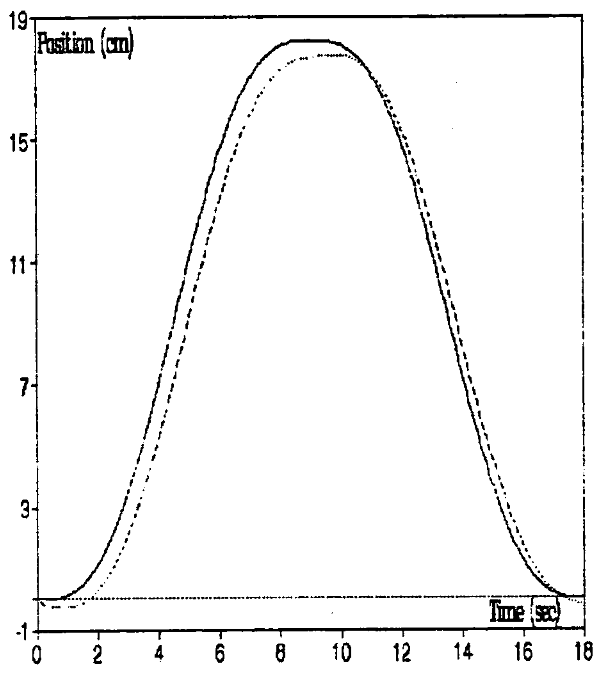
c) Changing of system parameters versus time.

d) Changing of controller parameters versus time.

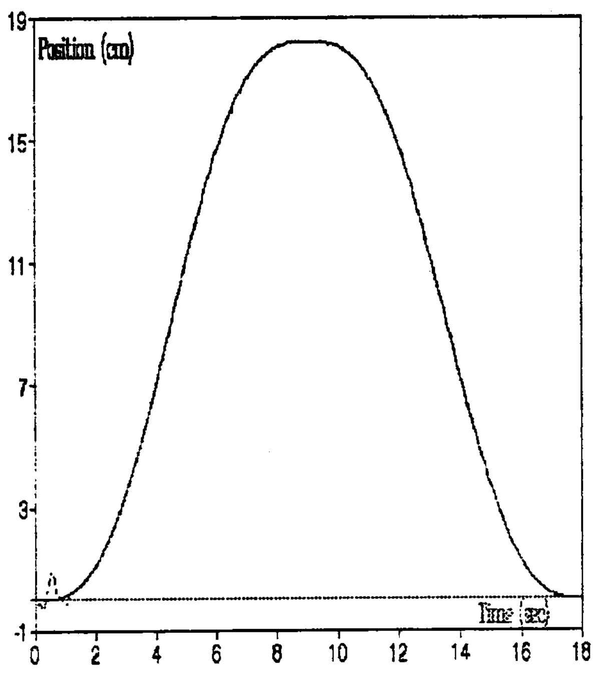
____________ Reference model output

.... Response of the system

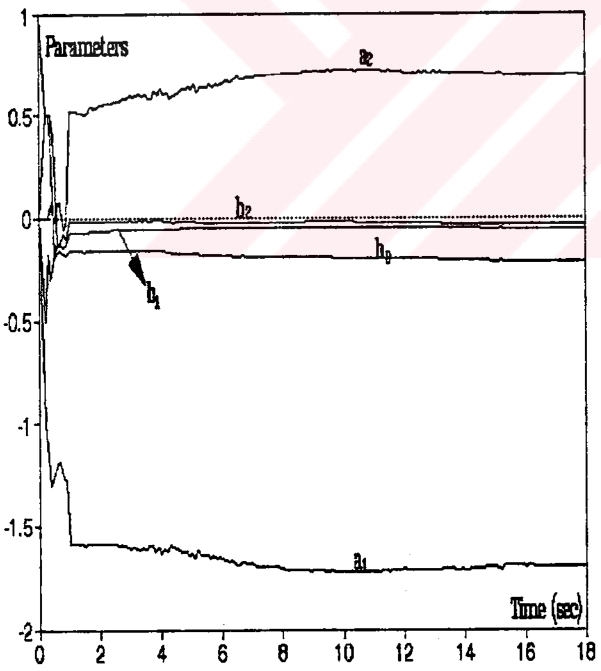
Figure 5.3 Experimental results on hydraulic robot manipulator for axis 2 (revolute joint).



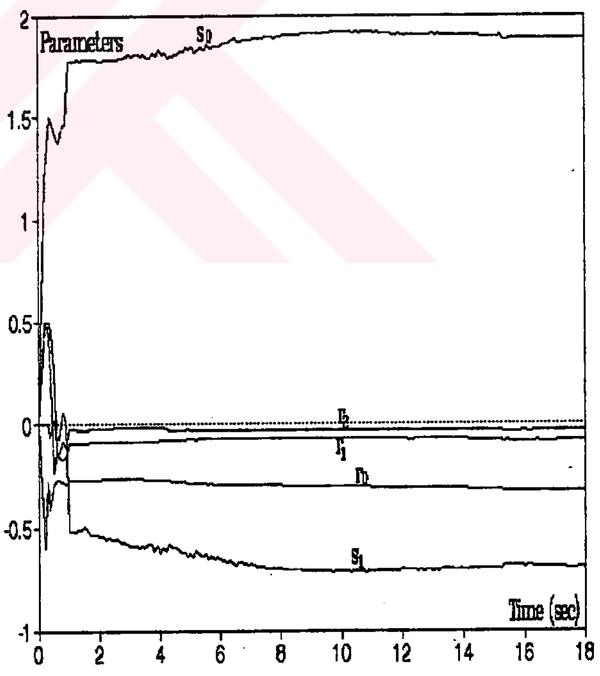
a) Response of prismatic joint and reference model output when conventional control is used.



b) Response of prismatic joint and reference model output when MRAC algorithm is used.



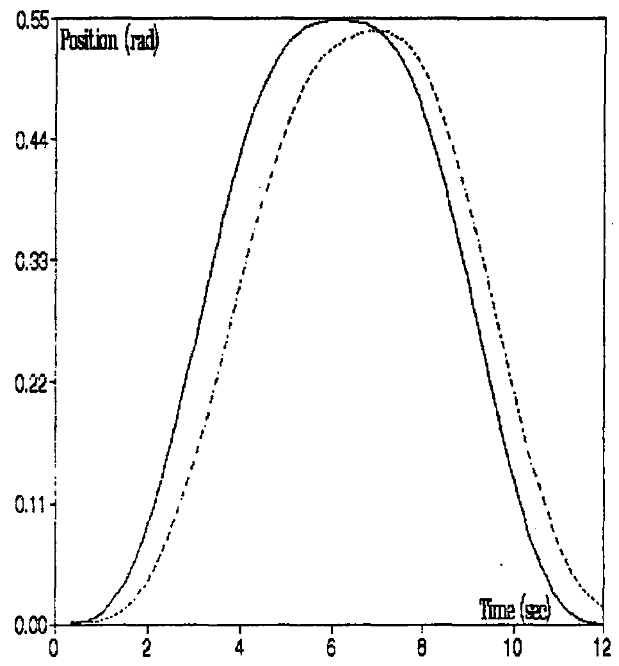
c) Changing of system parameters versus time.



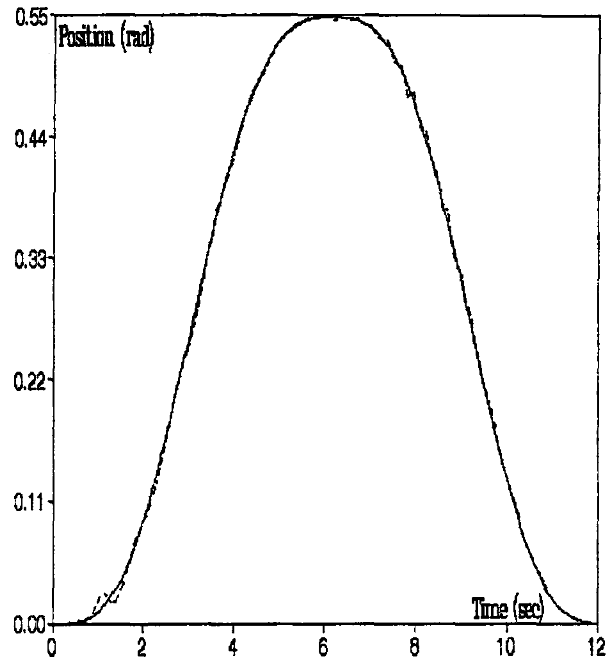
d) Changing of controller parameters versus time.

____________ Reference model output .... Response of the system

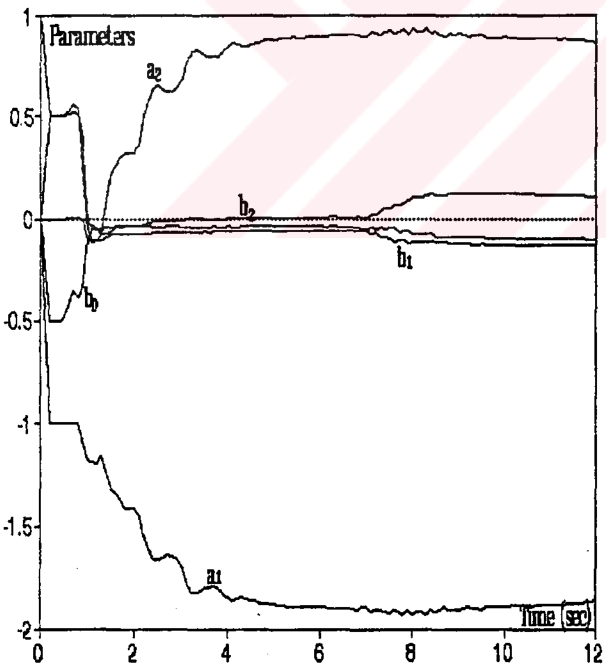
Figure 5.4 Experimental results on hydraulic robot manipulator for axis 3 (prismatic joint).



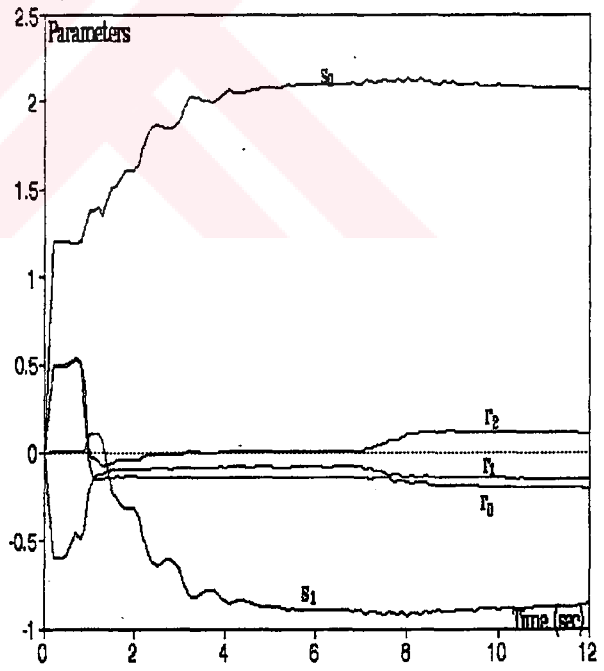
a) Response of revolute joint and reference model output when conventional control is used.



b) Response of revolute joint and reference model output when MRAC algorithm is used.



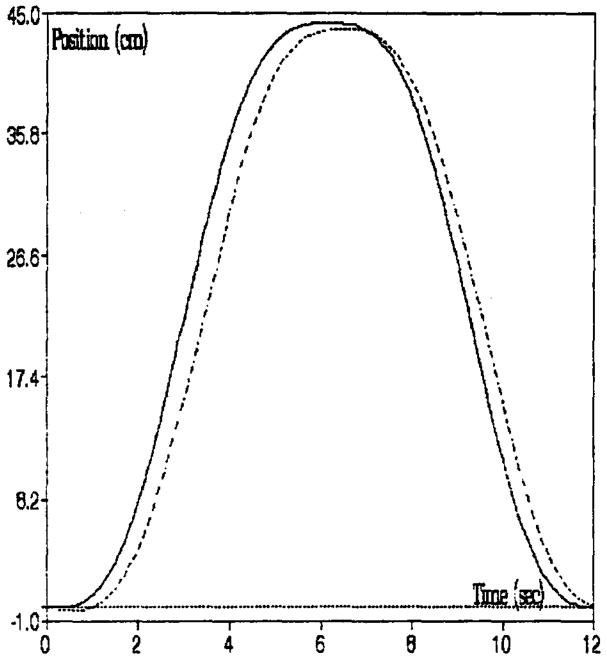
c) Changing of system parameters versus time.



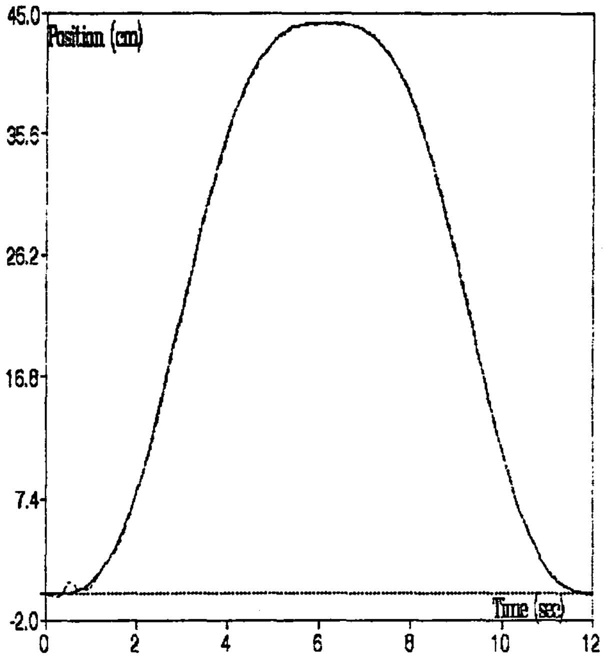
d) Changing of controller parameters versus time.

____________ Reference model output .... Response of the system

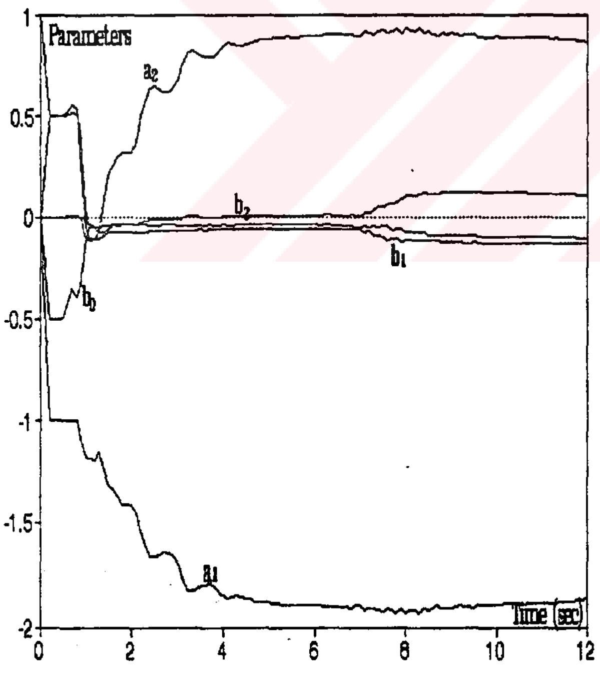
Figure 5.5 Experimental results on hydraulic robot manipulator for axis 2 (revolute joint).



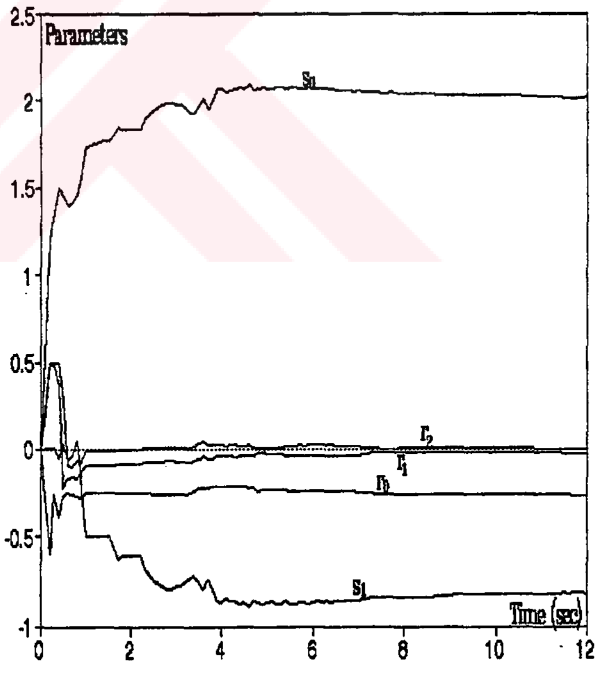
a) Response of prismatic joint and reference model output when conventional control is used.



b) Response of prismatic joint and reference model output when MRAC algorithm is used.



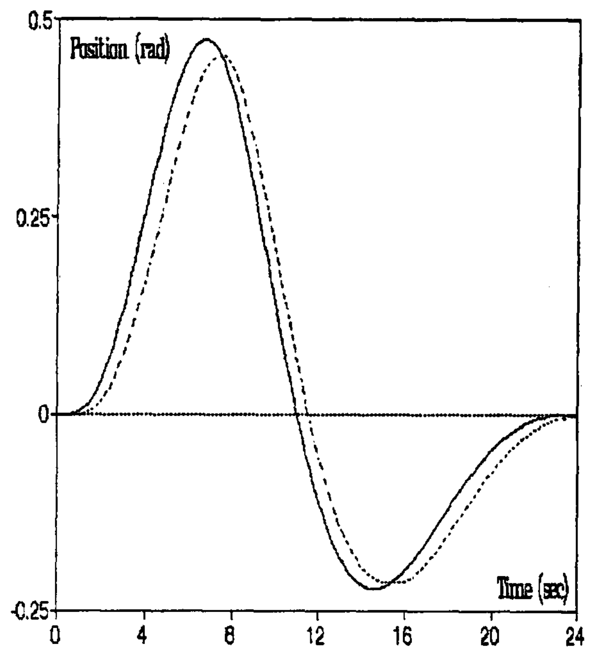
c) Changing of system parameters versus time.



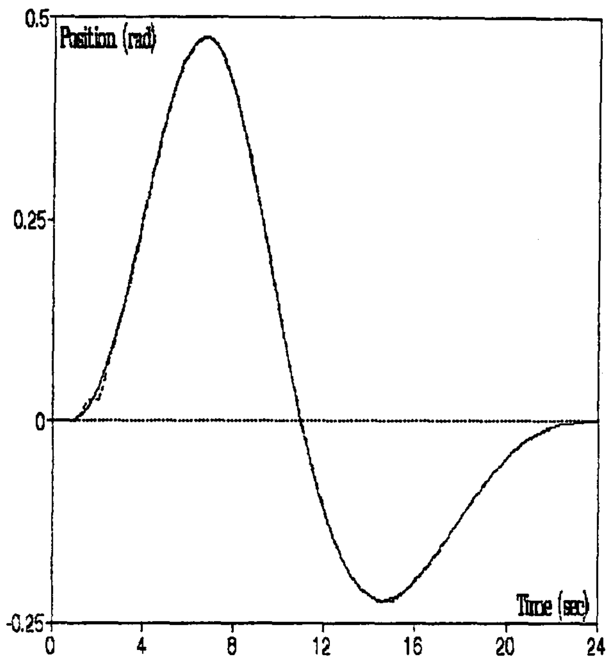
d) Changing of controller parameters versus time.

___________ Reference model output .... Response of the system

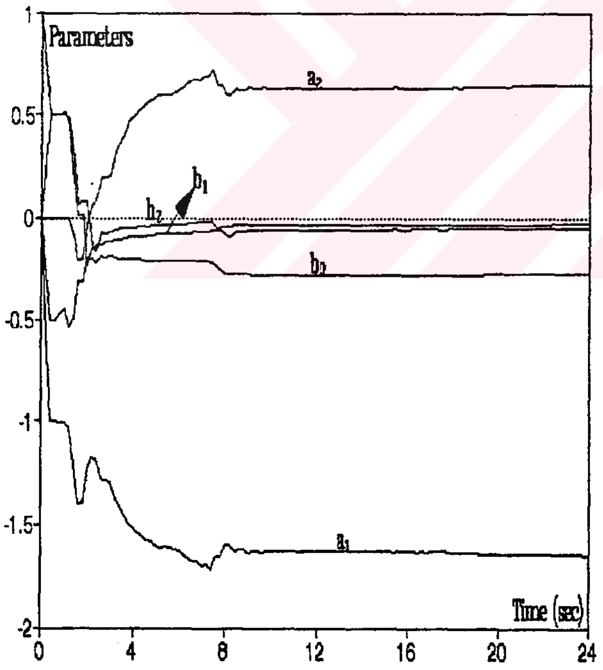
Figure 5.6 Experimental results on hydraulic robot manipulator for axis 3 (prismatic joint).

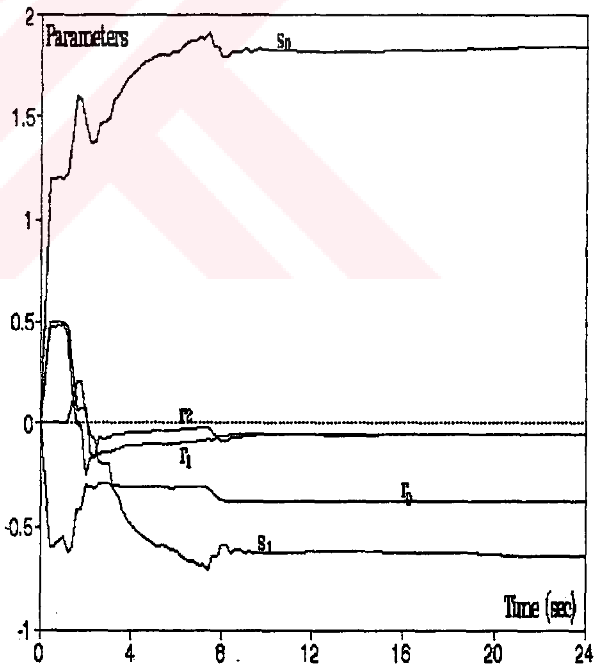


a) Response of revolute joint and reference model output when conventional control is used.



b) Response of revolute joint and reference model output when MRAC algorithm is used.

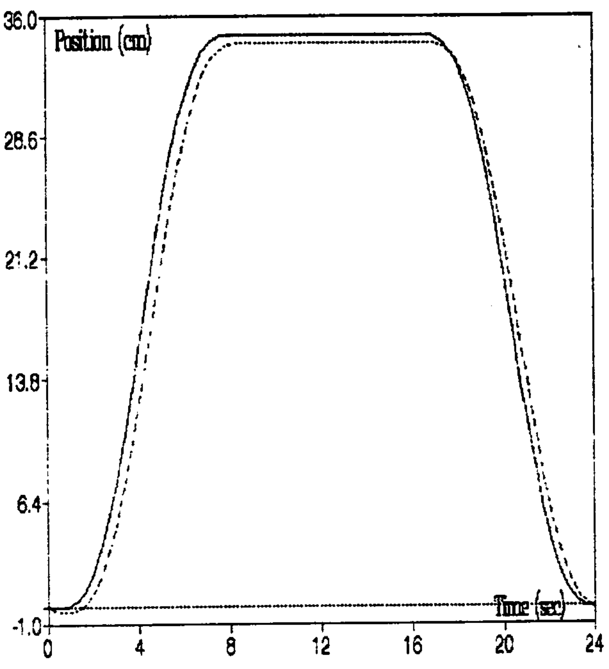




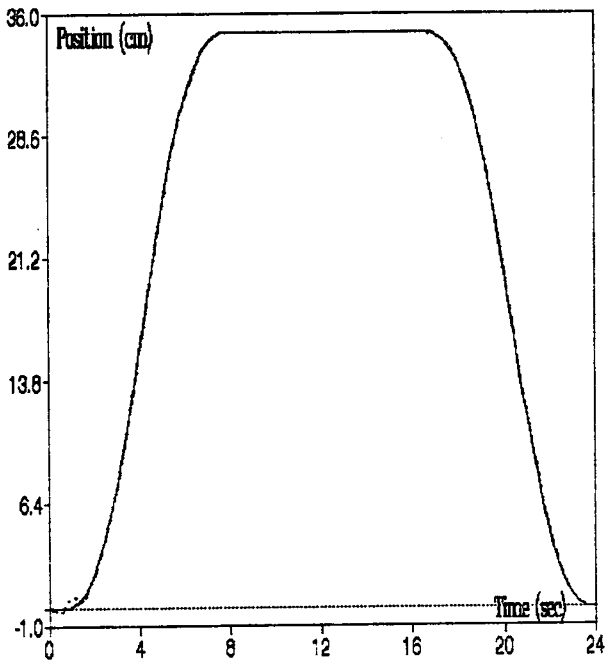
c) Changing of system parameters versus time. d) Changing of controller parameters versus time.

____________ Reference model output .... Response of the system

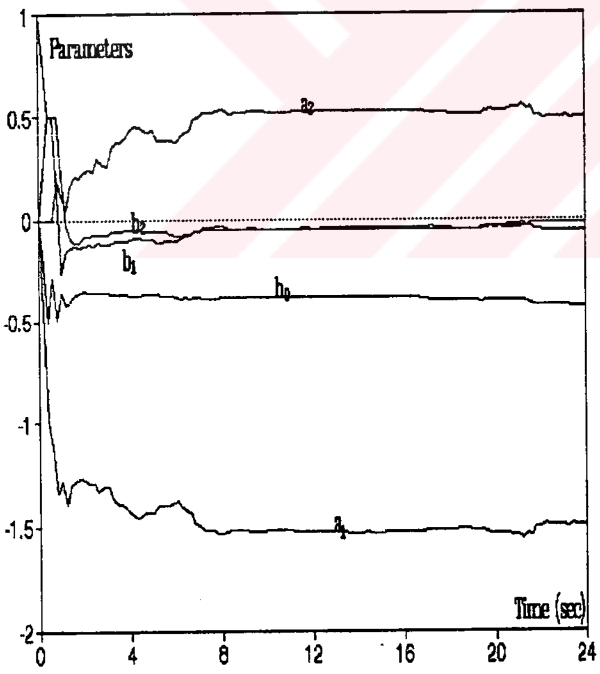
Figure 5.7 Experimental results on hydraulic robot manipulator for axis 2 (revolute joint).



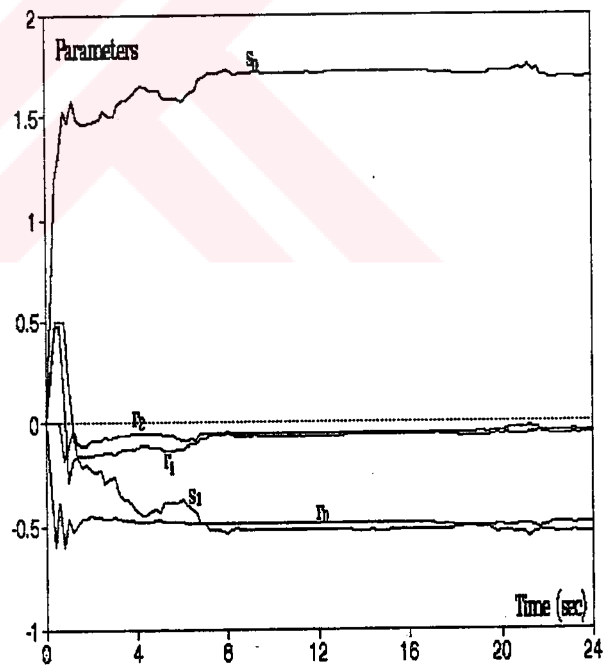
a) Response of prismatic joint and reference model output when conventional control is used.



b) Response of prismatic joint and reference model output when MRAC algorithm is used.



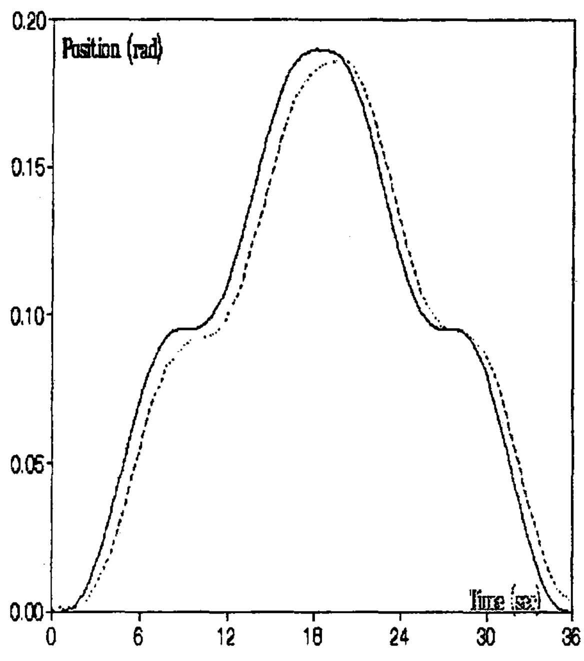
c) Changing of system parameters versus time.



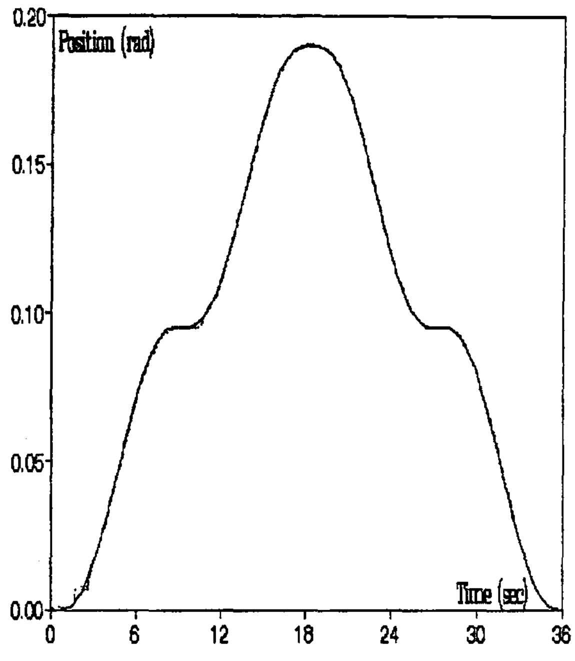
d) Changing of controller parameters versus time.

____________ Reference model output .... Response of the system

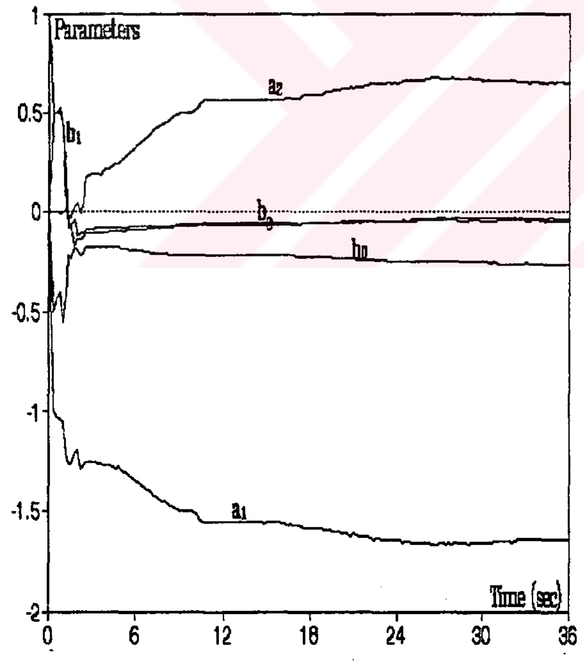
Figure 5.8 Experimental results on hydraulic robot manipulator for axis 3 (prismatic joint).



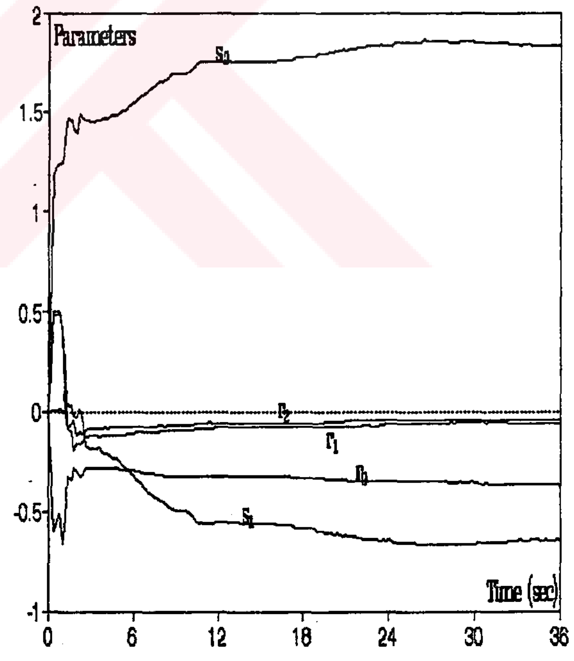
a) Response of revolute joint and reference model output when conventional control is used.



b) Response of revolute joint and reference model output when MRAC algorithm is used.



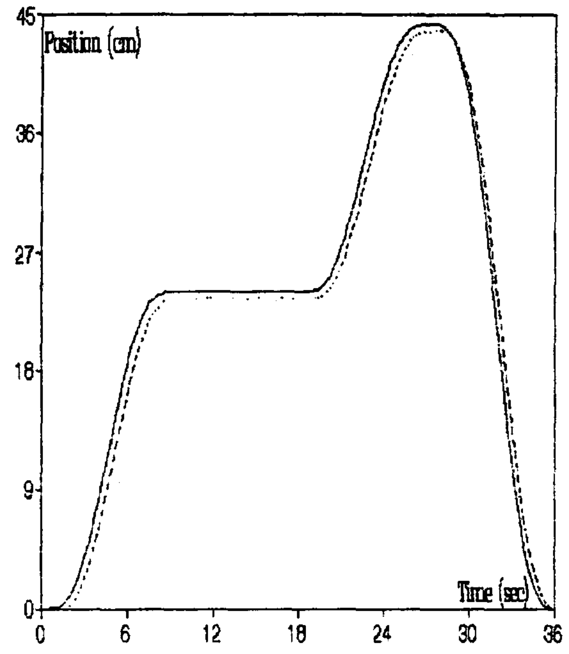
c) Changing of system parameters versus time.



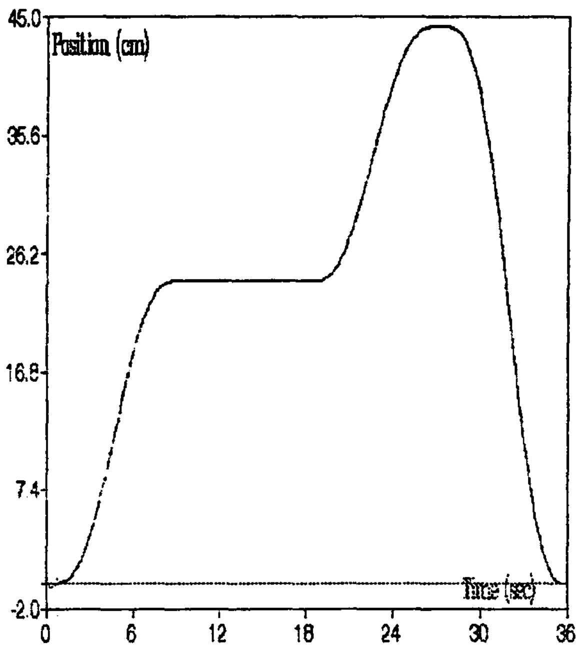
d) Changing of controller parameters versus time.

____________ Reference model output .... Response of the system

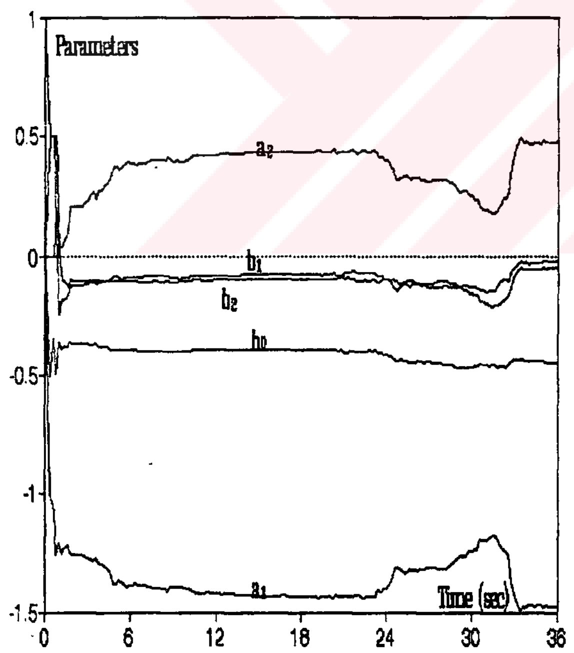
Figure 5.9 Experimental results on hydraulic robot manipulator for axis 2 (revolute joint).



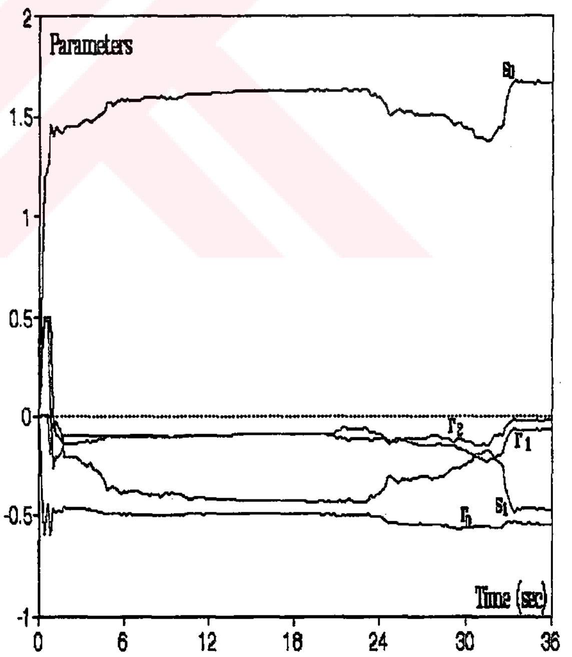
a) Response of prismatic joint and reference model output when conventional control is used.



b) Response of prismatic joint and reference model output when MRAC algorithm is used.



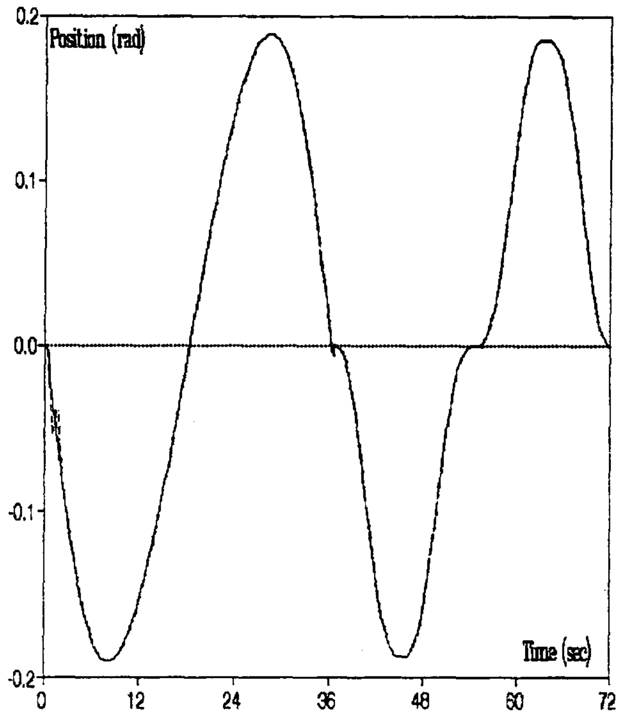
c) Changing of system parameters versus time.



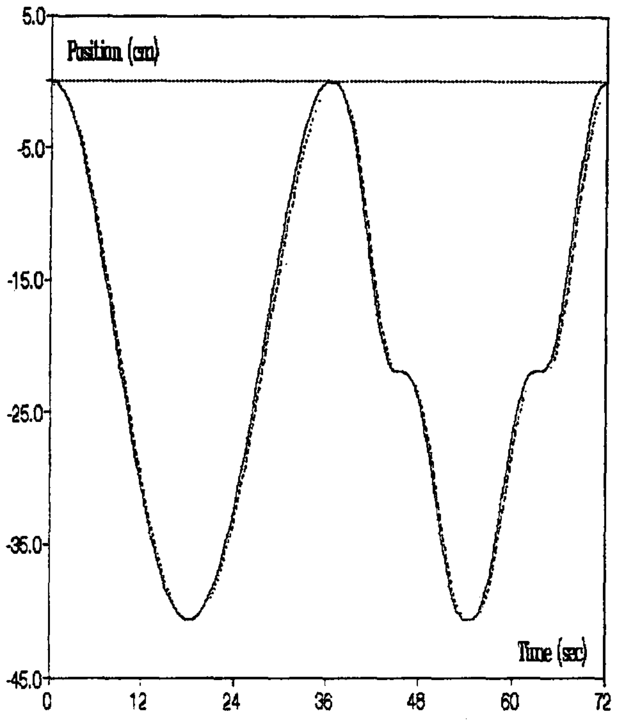
d) Changing of controller parameters versus time.

____________ Reference model output .... Response of the system

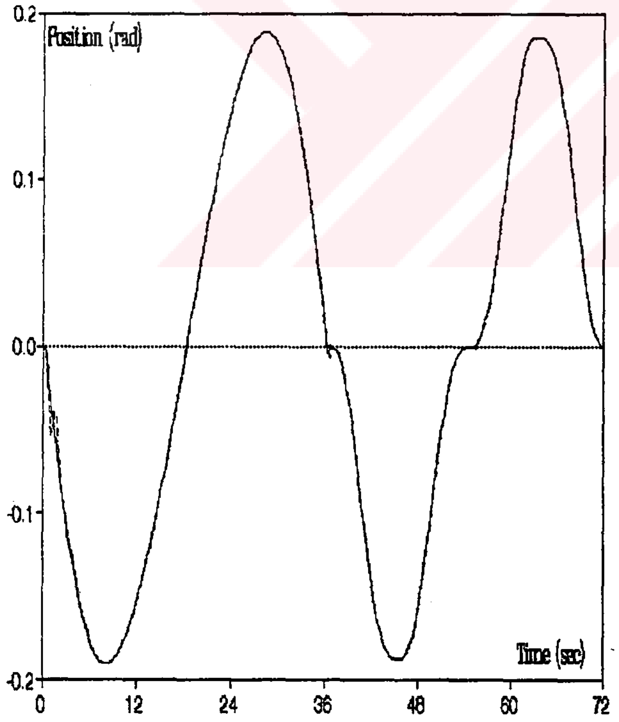
Figure 5.10 Experimental results on hydraulic robot manipulator for axis 3 (prismatic joint).



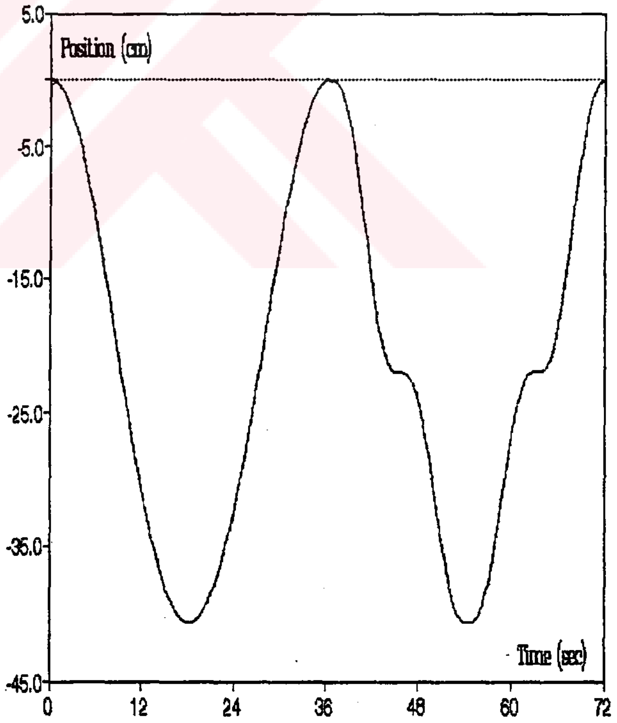
a) Response of revolute joint and reference model output when conventional control is used.



c) Response of prismatic joint and reference model output when conventional control is used.



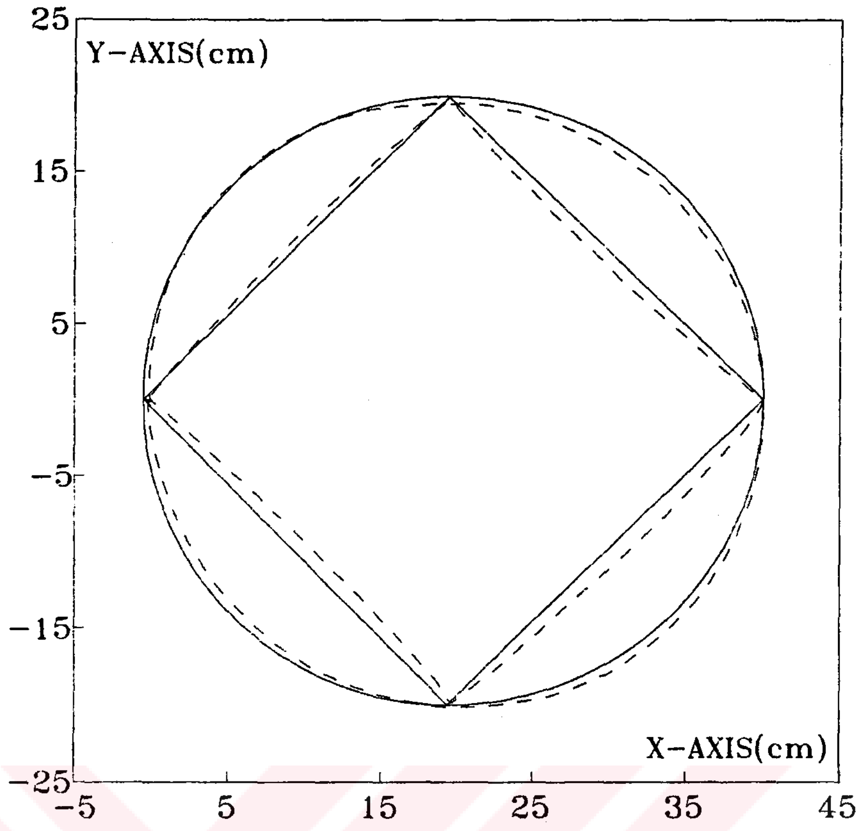
b) Response of revolute joint and reference model output when MRAC algorithm is used.



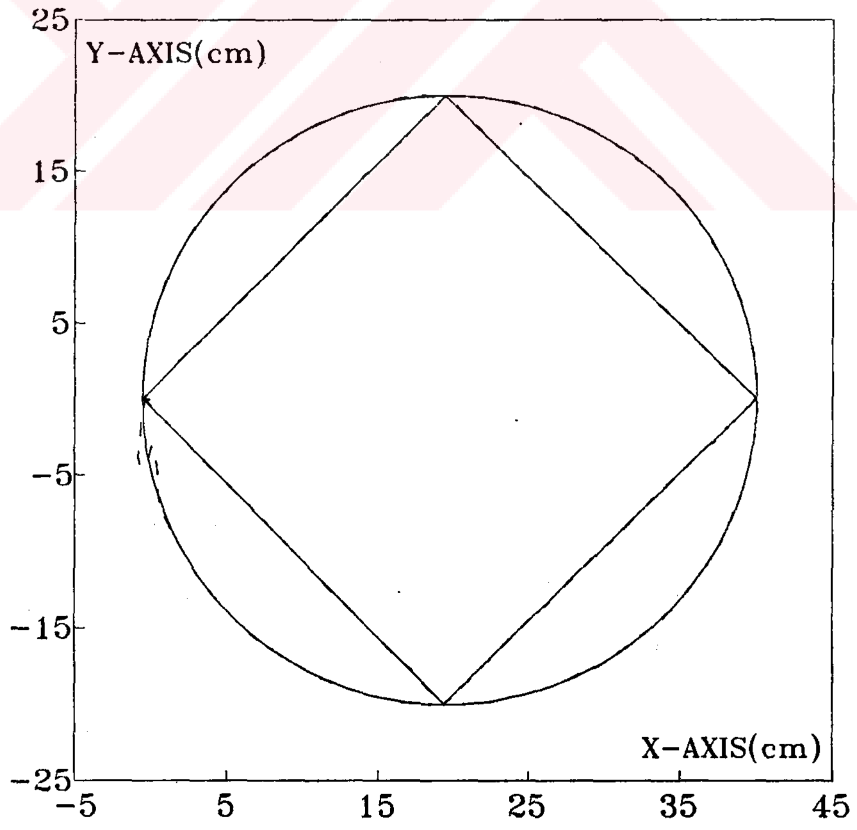
d) Response of prismatic joint and reference model output when MRAC algorithm is used.

____________ Reference model output .... Response of the system

Figure 5.11 Experimental results on hydraulic robot manipulator in joint space for example 5.



a) End-effector position in Cartesian co-ordinates when the conventional method is used.



b) End-effector position in Cartesian co-ordinates when the MRAC is used.
___________ Reference model output trajectory .... Response of the system

Figure 5.12 Experimental results on Hydraulic robot manipulator in Cartesian co-ordinates for example 5.

5.6 Application of MRAC Algorithm to DC Servo System

This part investigates the implementation of MRAC algorithm to a DC servo drive system experimentally. The system is a direct drive system that means the motor is coupled directly to the load without the use of belts or gears.

The orders for system model and reference model are selected as second order $(n_a = n_b = n_{m_a} = n_{m_b} = 2)$ and time delay is one $(d = 1)$. The parameters of $A(z^{-1})$ and $B(z^{-1})$ polynomials are determined by using the RLS parameter estimation method. The reference model parameters $a_{m_1}, a_{m_2}, b_{m_0}, b_{m_1}$ and b_{m_2} have fixed values. This model is obtained by using system identification method so that the reference model output has the same properties of the reference signal. The reference model is determined as;

$$A_m(z^{-1}) = 1 + 0.195z^{-1} + 0.005z^{-2} \text{ and } B_m(z^{-1}) = 1 + 0.8z^{-1} - 0.6z^{-2} \tag{5.29}$$

The initial values of estimated system parameter are set to the followings: $\hat{a}_1(0) = -1$, $\hat{a}_2(0) = 0$, $\hat{b}_0(0) = -0.5, \hat{b}_1(0) = 0.5$ and $\hat{b}_2(0) = 0.5$. Initial values of system input and output are set to zero for data vector in parameter estimation method. The forgetting factor γ is 0.98 and the weighting polynomial is chosen as $G(z^{-1}) = 0.5$ $(G_2(z^{-1}) = 1$ and $G_1(z^{-1}) = 0.5)$. Initial value of the covariance matrix is chosen as $P(0) = 1000I_m$ $(m = 5)$.

5.6.1 Experimental Results

Four examples are presented to demonstrate experimental results of DC servo system with and without MRAC. The sample time is chosen as 0.083 sec for all implementations.

Example 1

The results for Example 1 are shown in Figure 5.13. Figure 5.13 *a* is obtained by using conventional control method. Improvement in the tracking error, that is obtained by the application of MRAC algorithm, is given in Figure 5.13 *b*. The system response fit into the model output, however, the characteristic oscillations in the response at the beginning of the motion are seen. Figure 5.13 *c* and *d* show how change the system and the controller parameters with time. Initially the parameters change sharply to reach their actual values, then they change smoothly through the motion.

Example 2

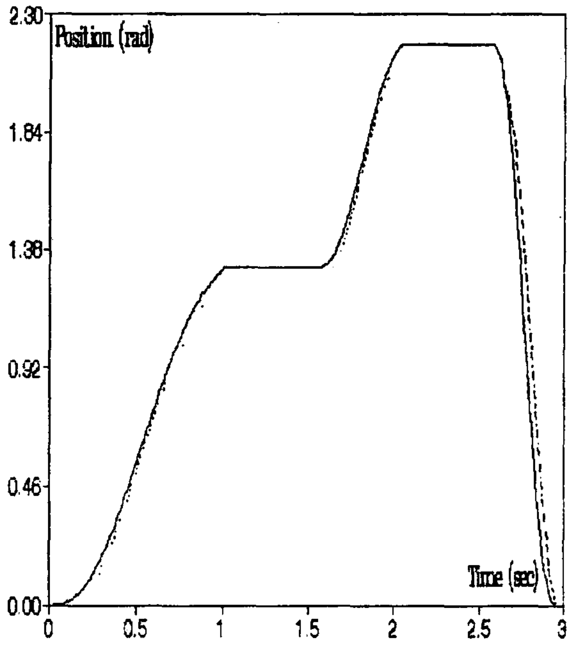
The experimental results for the second example are shown in Figure 5.14. This trajectory has different characteristics than the other trajectories, because it has an initial velocity and acceleration values of the both ends of the trajectory. This situation required a step change, at the both ends of velocity. An improvement is obtained in the response of the system when MRAC algorithm is applied as shown in Figure 5.14 *b*.

Example 3

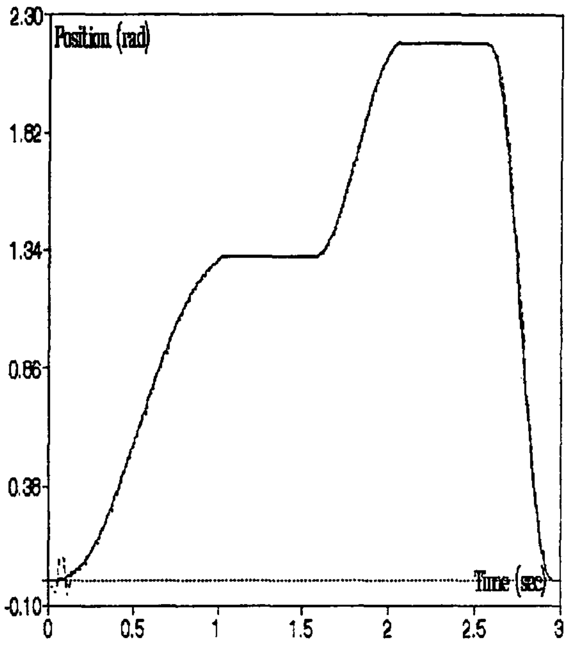
The results of this application are shown in Figure 5.15. Two same reference model signals are applied simultaneously to the system to examine the response of the system of the beginning of the second cycle. As seen from Figure 5.15 *b* that no oscillations is observed at this point and the response completely matches with the desired reference. This result shows ones again that the reason of oscillations at the beginning of motion is poor initial estimation of the parameters of the system. Figure 5.15 *c* and *d* show the variation of the parameters of system and controller versus time.

Example 4

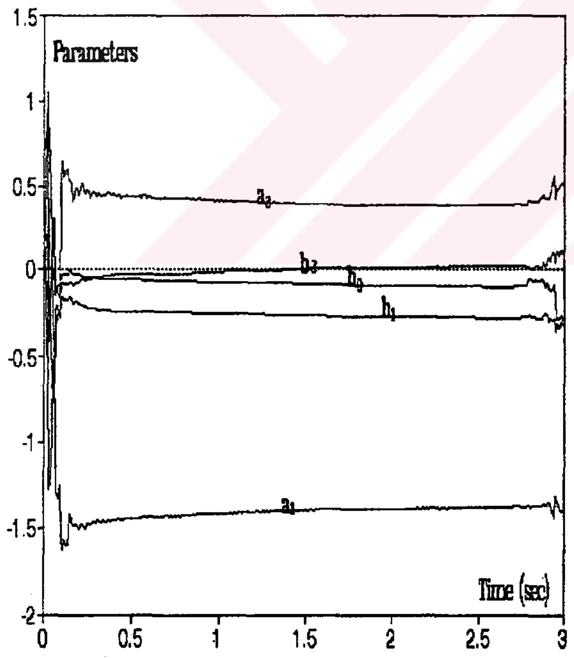
This example is given as an example of tracking of a difficult trajectory. Because, the trajectory has several abrupt changes in the position curve as seen in Figure 5.16. In spite of large tracking error in the response with the conventional control method, MRAC method provides very good tracking. As a result, all the experimental results show that MRAC method can be applied successfully for high performance tracking of DC-servo motor driven system. However, a significant effort is inevitably required in order to find out a prior information for the MRAC identifier. Such prior information is essential both to protect the DC hardware and to ensure reasonable tuning in performance.



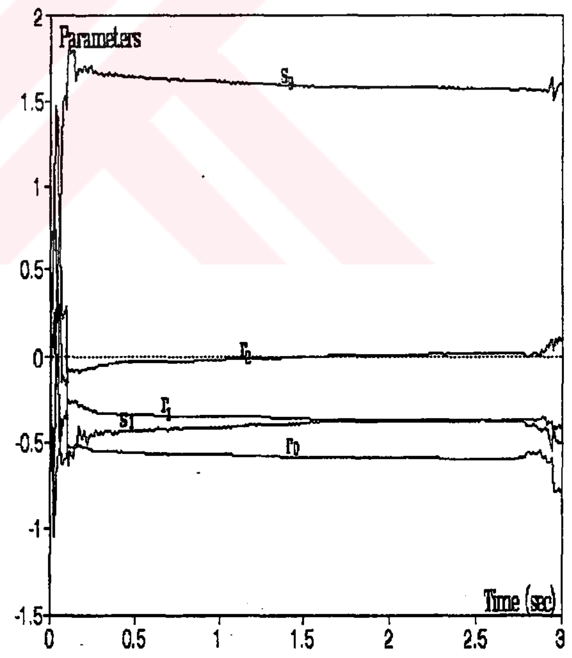
a) Response of servo system and reference model output when conventional control is used.



b) Response servo system and reference model output when MRAC algorithm is used.



c) Changing of system parameter versus time.

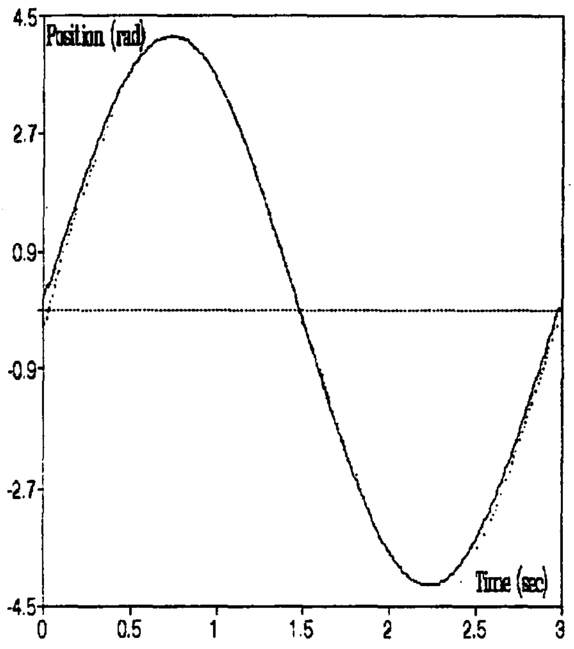


d) Changing of controller parameter versus time.

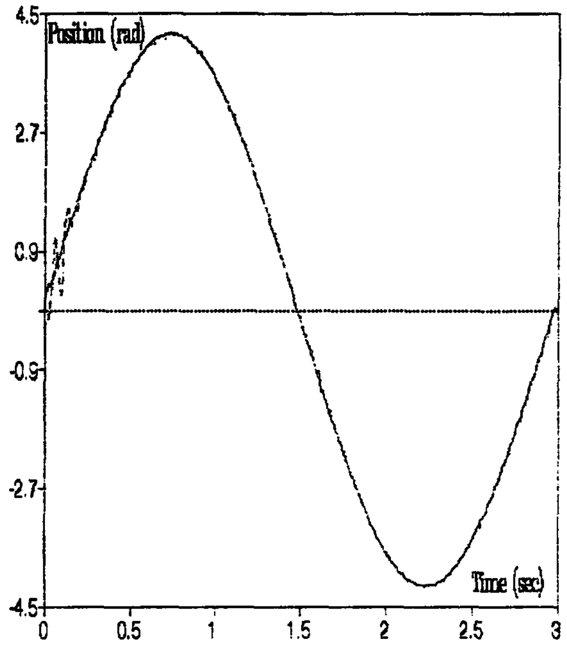
_____________ Reference model output

.... Response of the system

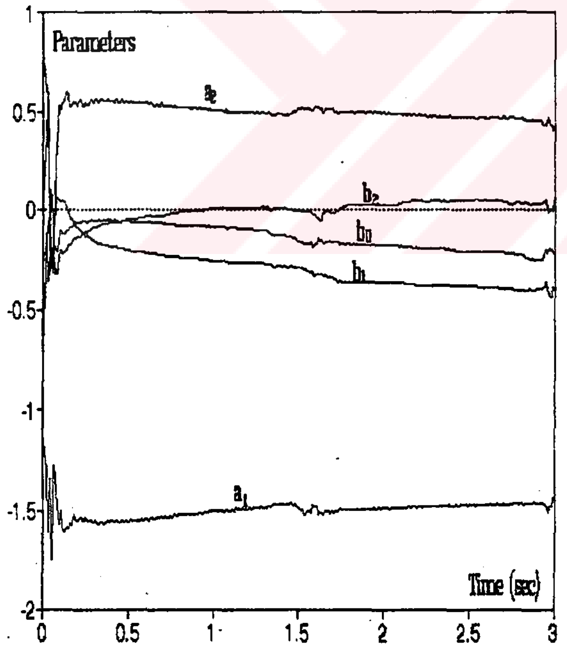
Figure 5.13 Experimental results on DC servo system for example 1



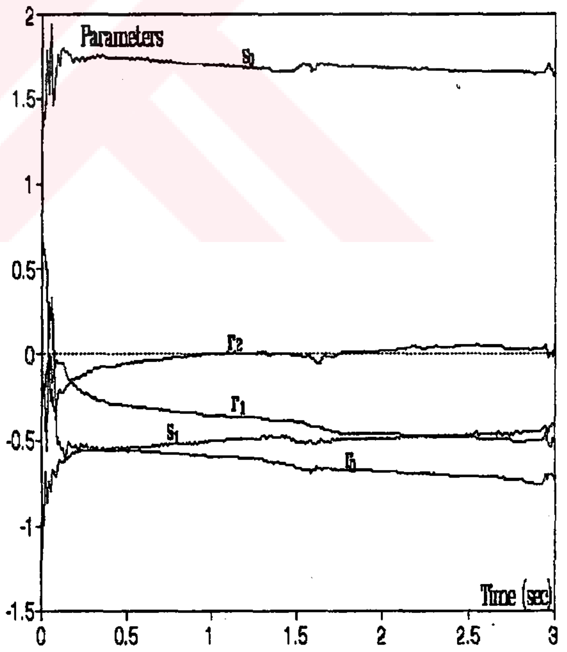
a) Response of servo system and reference model output when conventional control is used.



b) Response servo system and reference model output when MRAC algorithm is used.



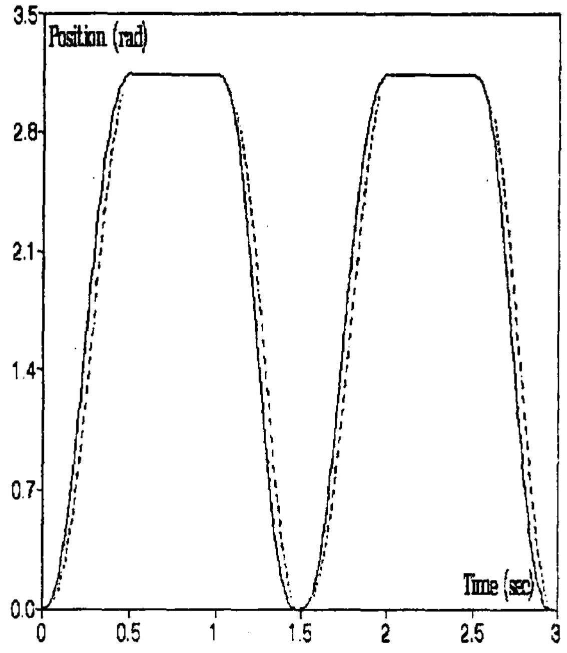
c) Changing of system parameter versus time.



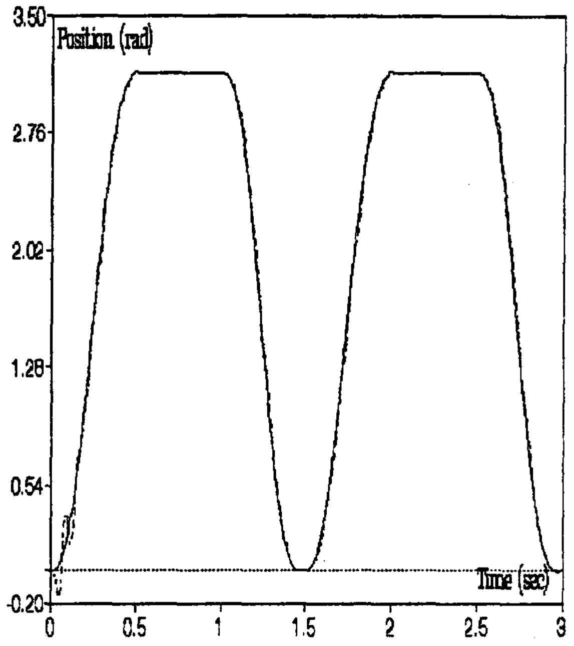
d) Changing of controller parameter versus time.

______________ Reference model output .... Response of the system

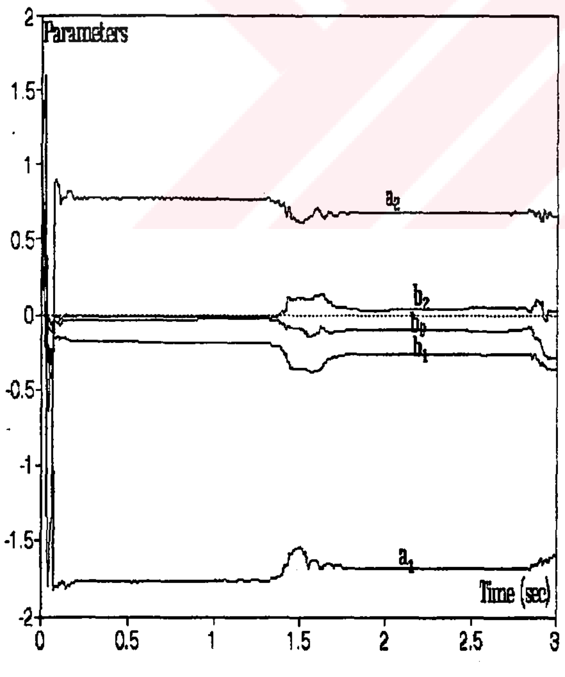
Figure 5.14 Experimental results on DC servo system for example 2

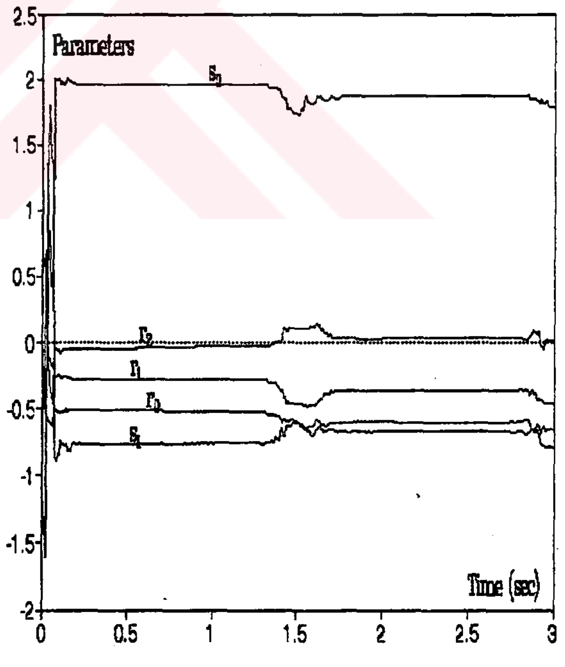


a) Response of servo system and reference model output when conventional control is used.



b) Response servo system and reference model output when MRAC algorithm is used.



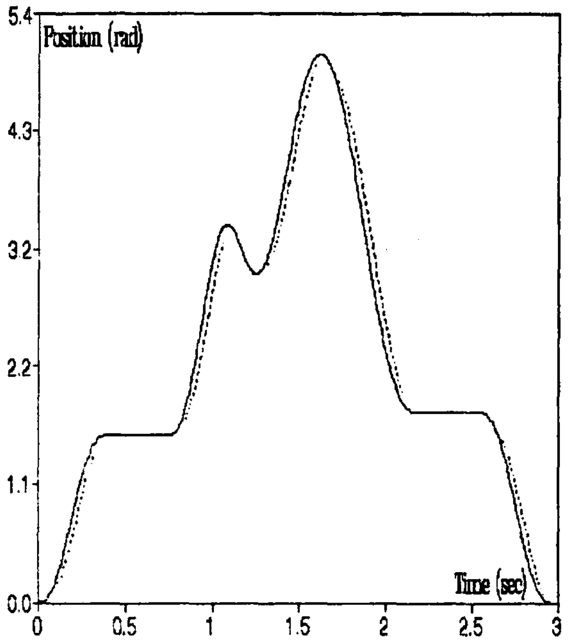


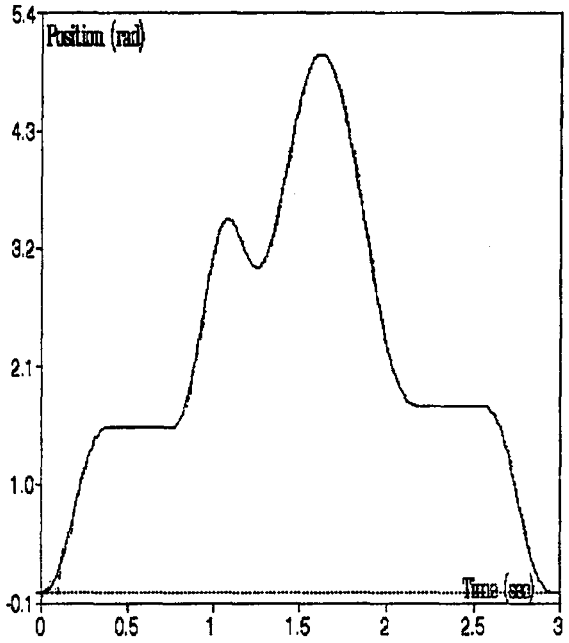
c) Changing of system parameter versus time.

d) Changing of controller parameter versus time.

______________ Reference model output .... Response of the system

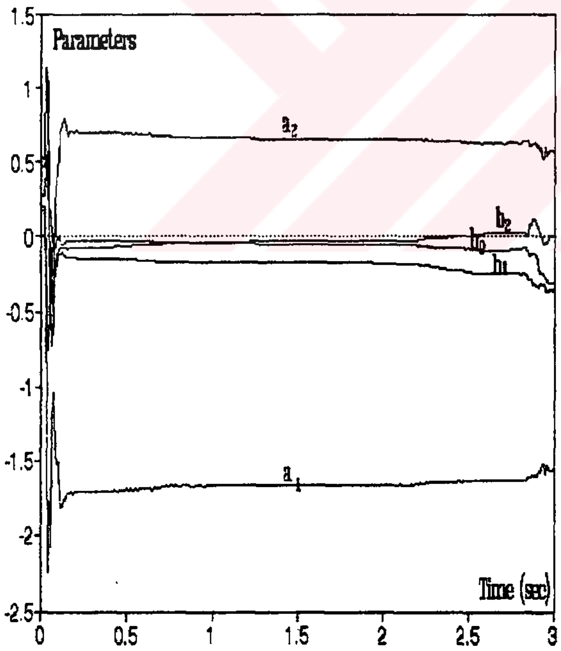
Figure 5.15 Experimental results on DC servo system for example 3

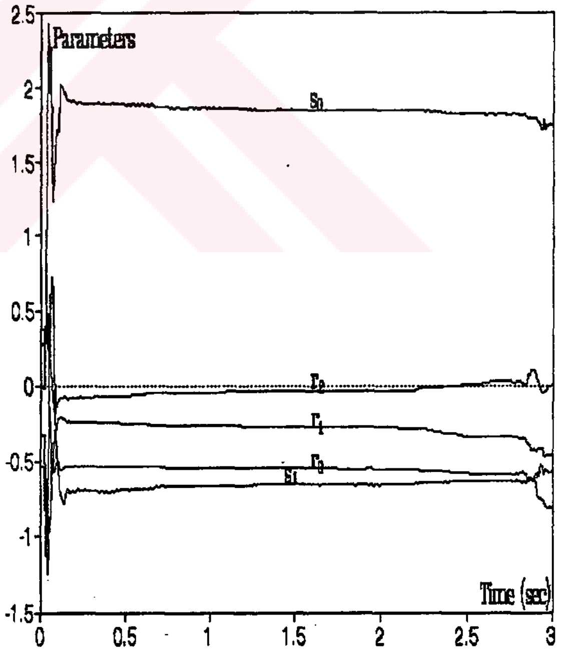




a) Response of servo system and reference model output when conventional control is used.

b) Response servo system and reference model output when MRAC algorithm is used.





c) Changing of system parameter versus time.

d) Changing of controller parameter versus time.

______________ Reference model output .... Response of the system

Figure 5.16 Experimental results on DC servo system for example 4

The choice of weighting polynomial proposed in MRAC algorithm is important to adjust the trade off between the system response and the desired reference model output. Therefore, the improper selection of weighting polynomial may cause the following results.

1) If the value of weighting polynomial is too large abrupt changes may happen in the control input that leads to saturation of actuator. The saturation may damage the actuators.

2) If the value of the weighting polynomial is smaller than the actual value then the control signal may be too weak, so the response follows the reference model with a steady state error.

The weighting polynomial is selected as $G(z^{-1}) = 1$ ($G_2(z^{-1}) = 1$ and $G_1(z^{-1}) = 1$) and the MRAC algorithm is applied both hydraulic robot and DC servo drive system. Experimental results are shown in Figure 5.17 and 5.18 and Figure 5.19 for hydraulic system and servo system respectively. As seen in the figures, both systems track the steady state errors.

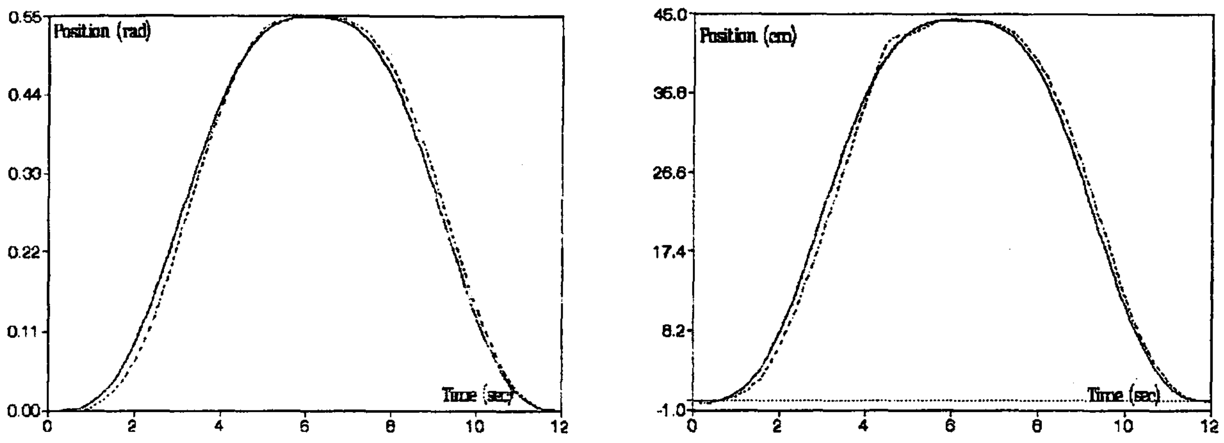


Figure 5.17 Experimental results with MRAC algorithm when the weighting polynomial are selected as $G(z^{-1}) = 1$ ($G_2(z^{-1}) = 1$ and $G_1(z^{-1}) = 1$) for each example in the hydraulic system.

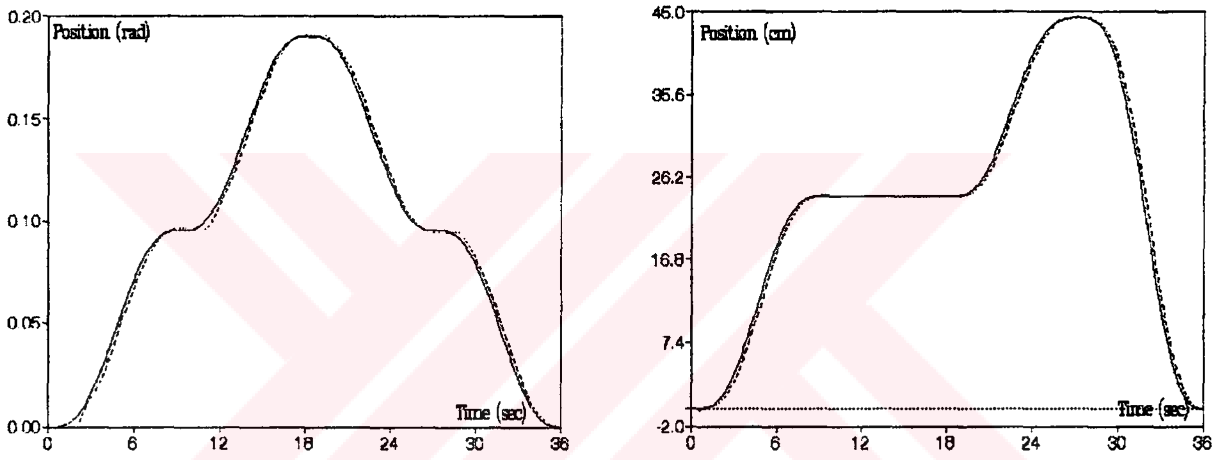


Figure 5.18 Experimental results with MRAC algorithm when the weighting polynomial are selected as $G(z^{-1}) = 1$ ($G_2(z^{-1}) = 1$ and $G_1(z^{-1}) = 1$) for each example in the hydraulic system.

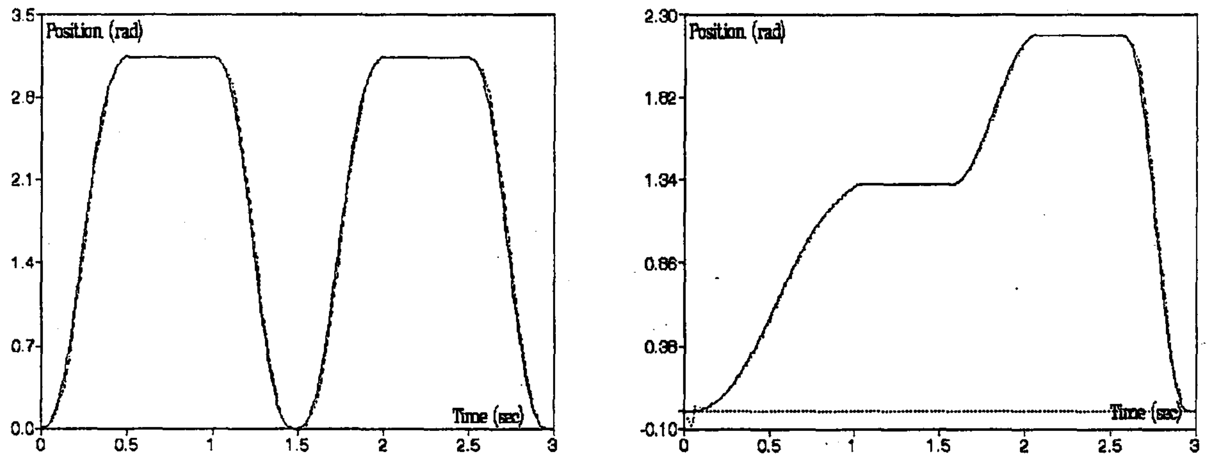


Figure 5.19 Experimental results with MRAC algorithm when the weighting polynomial are selected as $G(z^{-1}) = 1$ ($G_2(z^{-1}) = 1$ and $G_1(z^{-1}) = 1$) for each Example in DC servo drive system.

CHAPTER 6

LEARNING CONTROL

6.1 Introduction

In practical, the use of robotic systems, the task cycles are often routine operations, the task cycle has to perform same kind of function repeatedly and maintain acceptable specifications at the same time. For instance, the tasks of picking and placing, painting and pallezing in manufacturing assembly lines. In such kind of tasks, the system is required to move along a desired trajectory repetitively. Therefore the learning control method is useful for robotic system under the using it in the sequence of repetitive operation in those tasks.

The basic idea of learning control is to endow the system the ability to improve its performance using previous experience about the task in hand. So, the aim of this control method is to generate appropriate input series that force the motions of system follow the desired trajectory. In the algorithm, each time the system operates the input to the system control signal is stored, along with the resulting system output. The learning controller then evaluates the performance error as compared to the desired signal and computes a new control signal, which is stored for the use the next time the system operates. The new control signal is chosen in such a way as to guarantee that the performance error will be reduced on the next trial. The task in learning control is to speciy the algorithm for generating the next control input, given the current control input and system output, so that convergence to the desired output is attained. The proposed algorithm is tested on the DC servo driven system and some experiments was performed for position control. The control algorithm and results of implementation of learning control on DC servo driven systems are presented in this chapter.

The objective of this chapter is to realise a robotic system having a learning capability. The dynamics of this system have uncertainties and highly non-linear characteristics. These dynamic characteristics make it difficult to design a controller for an accurate tracking performance. To overcome these difficulties adaptive control method such as self-tuning can be considered. However these methods requires a series of calculations to be made, for example, parameter estimation for the system and the determination of the controller parameters. The complexity of the adaptive control algorithm limits the real time use of the adaptive control approach in a trajectory tracking control task especially for systems that are required to follow trajectories at high-speeds.

Learning control is an iterative approach to the problem of improving transient behaviour for processes that are repetitive in nature and operate over a fixed time interval [57]. Therefore, iterative learning is one of the advanced control technique that has been applied in recent years. The method can be easily applied to systems which are non-linear and time variant for which even the mathematical model is unknown. The block diagram for the learning control is illustrated in Figure 6.1. In this scheme, the inputs for the system are tuned systematically until the desired outputs from the system are achieved. Since this approach is used for cyclic processes, the inputs for the present cycle are modified using the experience of the previous cycle, therefore the method is called "learning". The learning operation continues until the response approaches the desired output within an acceptable tolerance band. Learning control can be applied more easily in real time for high-speed systems, because its algorithm is much more simple that the algorithm of the adaptive control, therefore, it requires less computational time. Another advantage of the learning control is that the system works as a closed loop system until the error is drawn into reasonable limits, then the system works as a open loop system. As it is known that open loop systems are much more stable than closed loop systems.

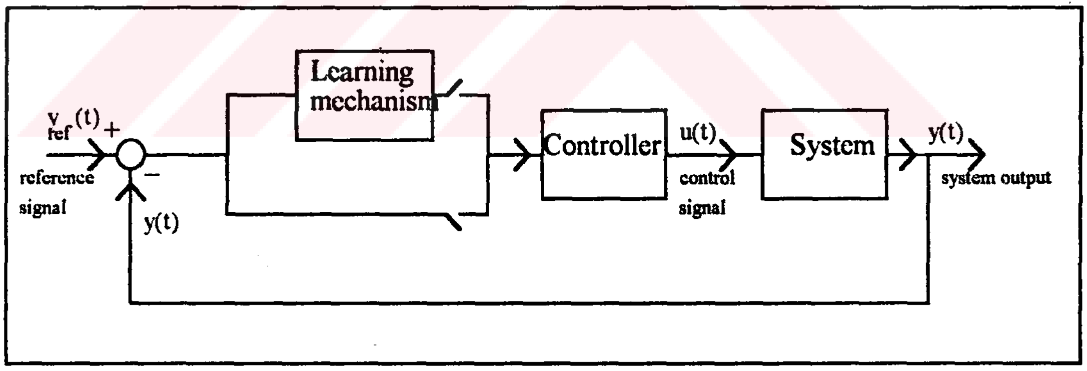


Figure 6.1 Structure of the Learning control method

6.2 Learning Control Algorithm

The dynamic equation of the system is not necessary when using learning control and control input for learning control can be directly calculated from the desired reference trajectory. The learning algorithm for position control can be simplified in the following way;

$$u_n(t) = u_{n-1}(t) + e_{n-1}(t) \tag{6.1}$$

Where t is the sampling instant, $u(t)$ is the control signal for system, $e(t)$ is error is based on the experience of the previous cycle and n is used to indicate index for cycle number. The error $e(t)$ is expressed as [32]:

$$e_{n-1}(t) = \lambda(y_{ref}(t) - y_{n-1}(t)) \tag{6.2}$$

Where $y_{ref}(t)$ is desired reference input, $y(t)$ is the system output and λ is learning gain and its value changes between (0-1). At the first cycle, the control input is equal to desired reference input in this learning control method such as;

$$u_1(t) = y_{ref}(t) \tag{6.3}$$

After the first cycle, the control input for the learning algorithm is obtained as the Equation (6.2) is substituted into Equation (6.1):

$$u_n(t) = u_{n-1}(t) + \lambda(y_{ref}(t) - y_{n-1}(t)) \tag{6.4}$$

The effectiveness of the above learning algorithm, depends on the selection of the learning gain. The servo motors of the system may achieve a saturation point after a few learning cycles if the learning gain is improper [32, 33, 58]. However, it is observed experimentally that whatever the gain values, the servo motors achieve saturation after 7-10 cycles. The main reason for the saturation of actuators is abrupt changes in the shape of trajectory caused by the learning algorithms. As an example, the motion curves of Figure 6.2 are produced from a polynomial function and the curves are smooth and continuous up to the acceleration. The position curve of this trajectory is used in order to control the system and the learning algorithm is applied (Equation 6.3) to tune the position curve in order to obtain the desired response from the system. In this case, the learning gain value is selected as (λ=0.3) and the speed of the system is 150 cycles/min. After a few learning cycle the tuned command curve is obtained as shown in the figure.

In order to prevent saturation, input is filtered after each application of the learning algorithm using a digital filter. A digital filter can be described as a process that generates a cleaned output from a contaminated input. Filtering can produce various outputs when different cut-off frequencies are applied.

Non-recursive and recursive filters are commonly used digital filter types. A non-recursive filter is a function of the given inputs producing its output by simply weighting the inputs by constants and then summing the weighted inputs.

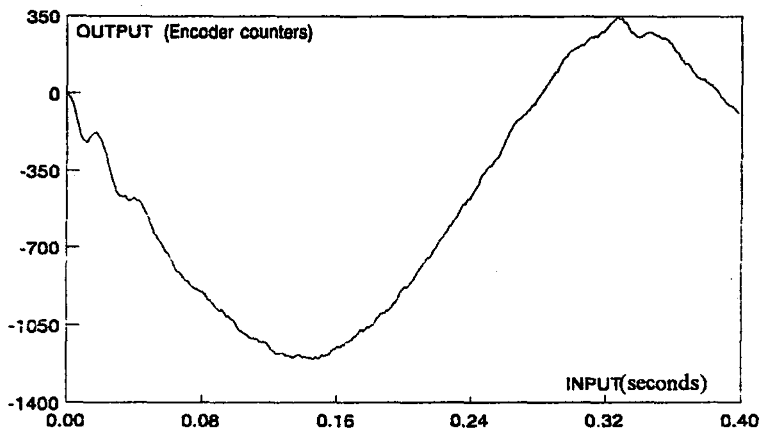


Figure 6.2. Oscillations in the control input due to the learning algorithm.

The constants are called the coefficients and these determine the filter. However, a recursive filter is not only a function of inputs, but also depends on the past outputs. In this study a non-recursive filter has been applied for data smoothing. A non-recursive filter for control input can be applied as [59, 60]:

$$u_n(t) = \sum_{k=-m}^{m} (u_n(t+k)) * C_k \tag{6.5}$$

Where, C_k represent the coefficients of the filter, m is the half length of the filter and k is the index for the filter coefficients. The calculation of the coefficients of a non-recursive filter is given below. However, since the operation is required in real time the number of the coefficients must be restricted. The coefficients of the filter are determined as [61]:

$$C_k = \frac{1}{2\pi} \int_{-w_c}^{w_c} e^{jwk} * dw = \frac{1}{\pi k} \sin(kw_c) \tag{6.6}$$

Where ω_c = cut-off frequency for the filter (rad/sec). The zeroth coefficient is determined as:

$$C_0 = \lim_{k=0} C_k = \lim_{k=0} \frac{1}{\pi k} \sin(kw_c) = \frac{0}{0} \qquad \text{Applying L'Hospital Rule's,}$$

$$C_0 = \frac{w_c}{\pi} \tag{6.7}$$

The summations of all filter coefficients must be equal to one. Therefore, the summation is continued until the its value is greater or equal to one. However, an error result if the summation is not equal to one. Consequently, a remainder term is used to compensate error if the summation is not equal to one. The remainder is calculated as:

$$R=\left(\sum_{k=-m}^{m}C_k\right)-1 \tag{6.8}$$

The learning algorithm is combined with the above filtering algorithm and the following equation is obtained:

$$u_n(t)=\frac{1}{R}\left\{\sum_{k=-m}^{m}\left[u_{n-1}(t+k)+\lambda\left(y_{ref}(t+k)-y_{n-1}(t+k)\right)\right]*C_k\right\} \tag{6.9}$$

An application of filtering to the noisy data set used in Figure 6.2 is illustrated in Figure 6.3.

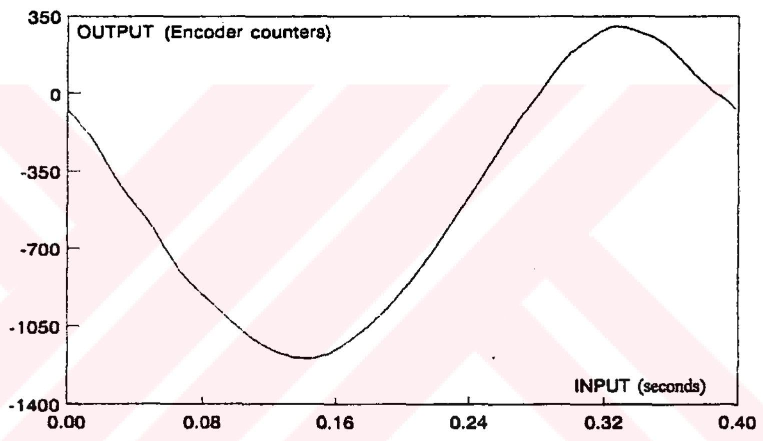


Figure 6.3. Filtering of a noisy control input

6.3 Application of Learning Control to a Servo System

In this section, the learning control algorithm is implemented for the position control of the DC servo system. The performance of this algorithm was compared to that of conventional control method in terms of position tracking error. All the numerical results show that the learning control can produce satisfactory results after a few trials.

Four example trajectories have been tested on the system. The first three examples are given to compare the response of the system for smooth and difficult motion where the speed of the system is 60 rpm. Final example is given to demonstrate the improvement of the tracing error cycle by cycle for the system where the speed of the system is 107 rpm. In all application, the learning gain value is selected as $\lambda = 0.3$ and the parameter of digital filter $D(z)$ of the system controller is used as,

Digital filter gain, $K = 60$
Digital filter zero, $A = 236$
Digital filter pole, $B = 252$

In the first three and final examples, the sampling time T is chosen to be 2.77 milliseconds and 1.556 milliseconds respectively.

Example 1

The implementation result for this example is given in Figure 6.4 *a-b*. As seen from these figures the desired trajectory of the servo system is smooth. The responds of the system to desired trajectory with a noticeable tracking error shown in Figure 6.4 *a* when conventional control method is used. However, when the learning control algorithm is applied to the system for 10 cycles, the response of system is obtained as shown in Figure 6.4 *b*.

Example 2

The experimental results for second example are shown in Figure 6.5 *a-b*. This trajectory has different characteristics than the previous example trajectory. As seen from figure, the desired trajectory are planned as a rise-dwell-rise and dwell motion. Figure 6.5 *a* is obtained by using conventional control method that show the tracking errors between the motor position and the desired trajectory. However, after ten cycles of learning, the system follows the desired trajectory perfectly (see Figure 6.5 *b*).

Example 3

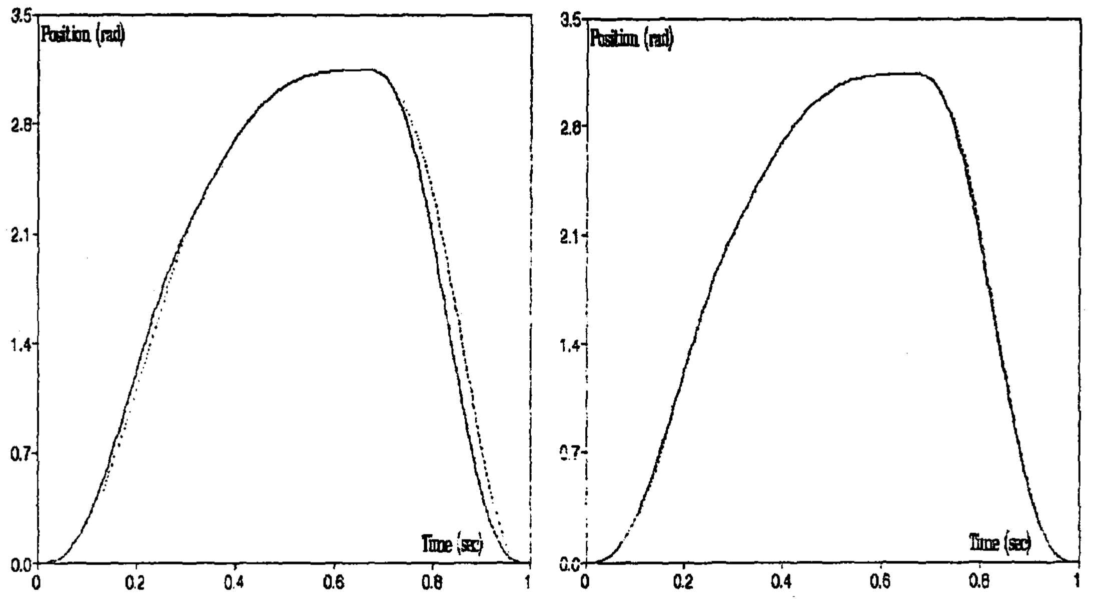
The results of this example are shown in Figure 6.6 *a-b*, is given as the example of difficult trajectory. Since the trajectory has several abrupt changes in the position it. Such trajectories can not be easily implemented to the servo system, however, the responses that are obtained without and with learning are displayed in Figure 6.6 *a* and *b* respectively.

As seen, the response is far away from the desired output, but the tuning of the command using the learning control technique improves the response as shown in Figure 6.6 *b*.

Example 4

This example is given to show the improvement of the tracking error cycle by cycle for the system. Sample time is chosen as 1.556 milliseconds. Experimental results for this case are shown in Figure 6.7 *a-b* where the speed of the system is 107 rpm.

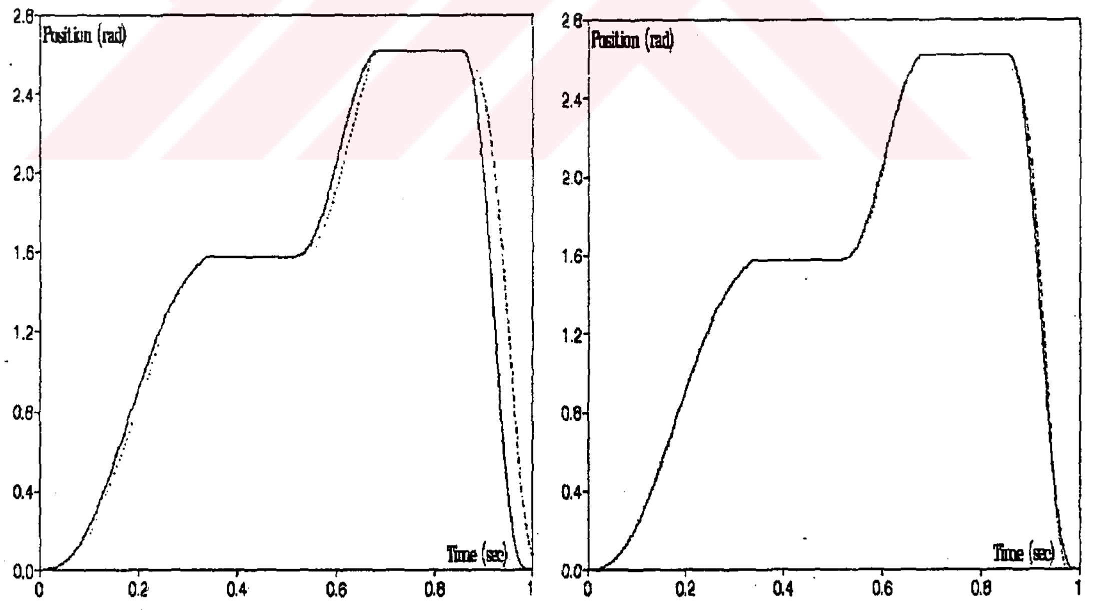
When this motion is implemented, the system follows the desired trajectory with large error at high speeds using conventional control method (see Figure 6.7 *a*). Activating learning for 23 cycles improves the response of the system and it matches with the desired trajectory as shown in Figure 6.7 *b*.



a) Response of servo system and desired reference trajectory when conventional control is used. b) Response of servo system and desired reference trajectory when learning control algorithm is used.

__________ Desired reference trajectory .... Response of the system

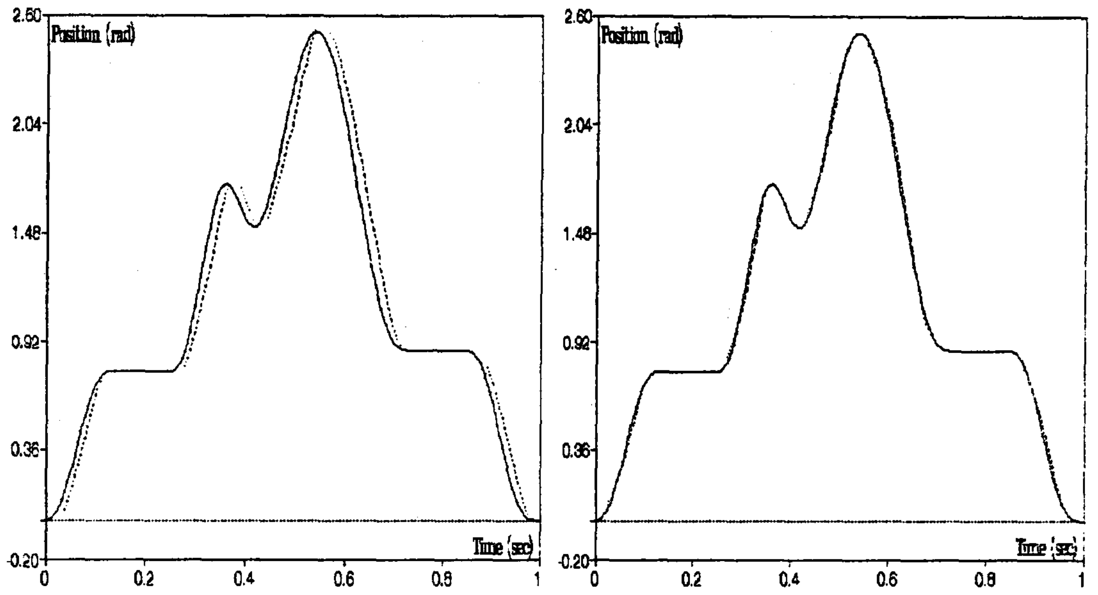
Figure 6.4 Experimental results on DC servo system for example 1.



a) Response of servo system and desired reference trajectory when conventional control is used. b) Response of servo system and desired reference trajectory when learning control algorithm is used.

__________ Desired reference trajectory .... Response of the system

Figure 6.5 Experimental results on DC servo system for example 2

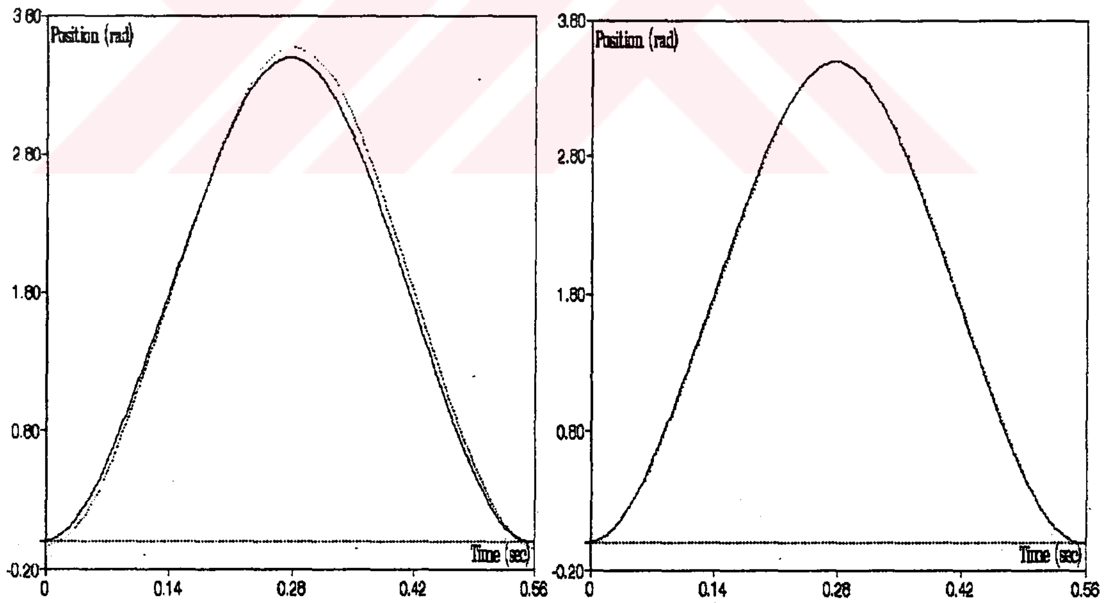


a) Response of servo system and desired reference trajectory when conventional control is used.

b) Response of servo system and desired reference trajectory when learning control algorithm is used.

__________ Desired reference trajectory .... Response of the system

Figure 6.6 Experimental results on DC servo system for example 3.



a) Response of servo system and desired reference trajectory when conventional control is used.

b) Response of servo system and desired reference trajectory when learning control algorithm is used.

__________ Desired reference trajectory .... Response of the system

Figure 6.7 Experimental results on DC servo system for example 4

Also, in order to clarify the process of learning control and indicate the effect of filtering operation to prevent saturation of actuators a three-dimensional graph is given in Figure 6.8. The graph shows the improvement of the tracking errors cycle by cycle for DC servo system. However, the error of the axis has been reduced dramatically by learning control. At the beginning, the tracking error was about (0.15) radians, but after 23 learning cycles the maximum value of the error is dropped to (0.007) radians. Despite the high speed of the manipulator (107 cycles/min) and the large number of learning cycles, saturation is not observed.

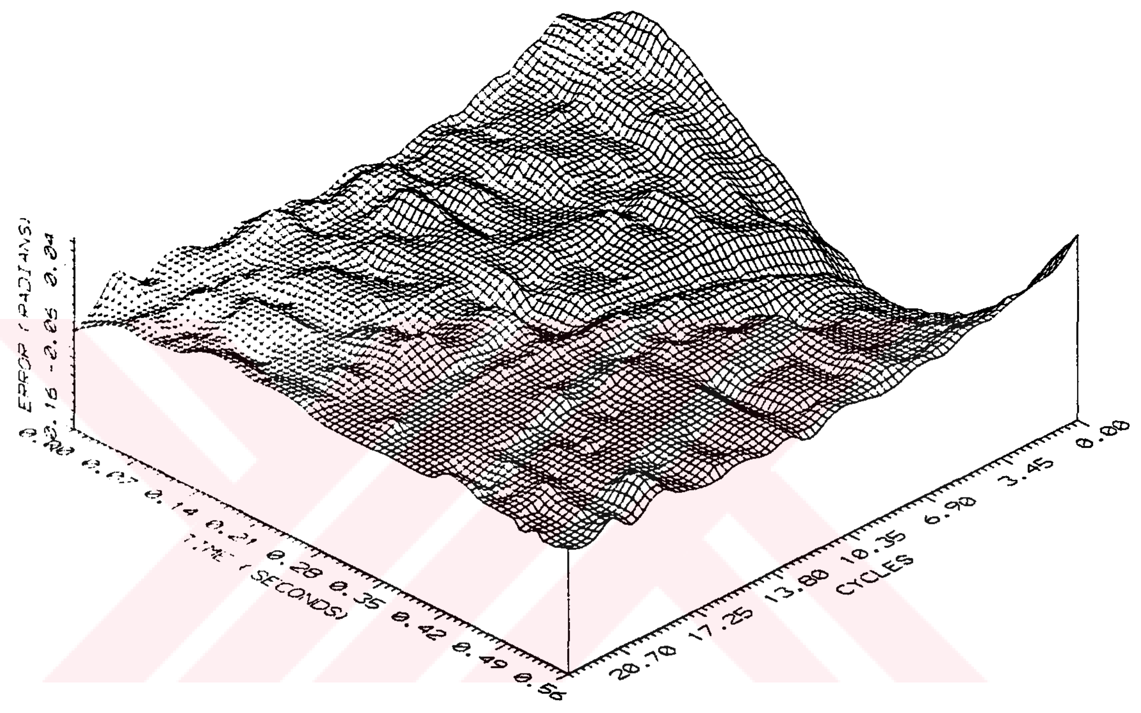


Figure 6.8 Improvement of tracking errors steps by steps for example 4.

CHAPTER 7

DISCUSSION AND CONCLUSION

Robots are being used for many operations such as assembly, welding, cutting and painting in industry that are usually based on high-precision servo technology. These system dynamic's characteristics are complex and highly non-linear, because of the non-linear mechanical coupling between the links, the variation in moment of inertia and compliance with geometry. For this reason, the control of such systems for many researchers has excited both theoretical and practical interests. Thus, it is very difficult for robotic systems to follow a desired trajectory perfectly. If the dynamics of the system and characteristics of disturbances affecting are known, then the controller of the system will yield the desired performance can be designed.

In general the dynamics of such systems can be characterised by a set of highly coupled non-linear differential equations which is derived from the Newtonian mechanics. On the basis of such models, the control of systems has been extensively studied in recent years and two control design approaches have been proposed, non-adaptive and adaptive control. Non-adaptive control methods, such as computed torque or non-linear feedforward usually require a more accurate modelling of system dynamics. However, in real case, the accurate of the dynamic model of robotic systems are affected by many factors such as friction, backlash and uncertainty of the parameters like inertia, mass centre and air resistance etc. Some of these are uncertain and change with time and therefore cannot be obtained exactly in advance. Hence, the controlled systems are significant enough to render the above control methods ineffective due to the changes of load in the task cycle. As a result, using these control methods limits the precision and speed of the end-effector.

Adaptive control is one which continuously and automatically measures the dynamic characteristics of the system and compares them the desired dynamic characteristics. The controller parameters of the system are automatically adjusted since system parameters and disturbances are changed with time. The optimal performance can be maintained regardless of the environmental changes.

The most difficult part of adaptive control is to design the system model. Because, to be able to use the adaptive control theory, the system must be linear, the order of its transfer function numerator and denominator polynomials must be known, the system transfer function must be minimum phase and the system must be completely disturbance free. However, as mentioned, the system can never satisfy all these requirements. The exact order is never known, no system is linear or disturbance free. The facts that the requirements on the system are never met imply that the adaptive controller was designed is same with was used. It is therefore important to know how the adaptive controller operates under non perfect condition such as on real system.

For this reason, the complicated manipulator systems are modelled by a single-input/single-output linear discrete time-series of ARMAX model for each joint in designing of adaptive control methods. Under this model, it is assumed that interactions between the joints are small, and thus, coupling terms in the ARMAX model of the manipulator system are omitted, and each joint will be controlled independently of the other. The inputs in the model are the actuator voltages, and the outputs are selected as the positions of the manipulator joints.

Adaptive control can be classified basically into two groups, namely, self-tuning adaptive control and model reference adaptive control. Self-tuning adaptive control is further divided into two groups as implicit and explicit self-tuning. In this study self-tuning and model reference adaptive controls have been designed and experimentally tested on hydraulic robot and DC servo drive system to examine their effect on trajectory tracking error.

In the explicit self-tuning adaptive control, the parameters of the system model are directly identified. This gives the indirect adaptive algorithm. The controller parameters are not updated directly, but rather indirectly, which are computed based on minimising the expected of the auto regressive difference model and available measurements. Thus this controller makes the output of each manipulator joint follow the desired trajectory specified at discrete points.

In the implicit self-tuning adaptive control methods, the control law is determined such that the variance of generalised cost function is minimise, and the controller parameters are directly identified. This can be made by reparameterization of the system model. This gives a direct adaptive algorithm.

In MRAC, the desirable dynamic characteristics of the system are specified in a reference model. This reference model is constructed in such a way as to track the reference input optimally as is the feedback controller. In this control method, the actual output of the system is compared with the reference model output and the parameters of the controller are modified in such way that the response of the system follows that the reference model. This is done by an adaptation mechanism. Therefore the adaptation mechanism must be designed so as to reduce to zero measurement output error between model output and system output. On-line estimation of the parameters that determine the maximum obtainable system performance, and use of these parameters in the reference model, makes it possible to improve the overall performance. Therefore, MRAC law is developed for maintaining uniformly good performance over a wide range of motion and payloads.

In self-tuning and MRAC methods, real-time system identification and controller are required high precision arithmetic. The precision requirement is related to numerically sensitive algorithm the RLS algorithm is an example of these. However recursive least square parameter estimation method is inadequate due to either restricted computational precision and restricted computational time. Hence, it is best to use a numerically stable updating method such as the U-D factorisation in order to achieve a working algorithm for adaptive control methods.

Some series of experiments were performed on hydraulic robot and DC servo system for position tracking control of the system. Implementation studies on this system have been presented to demonstrate the applicability of the adaptive control schemes. These experimental results show that both systems perform very well under the explicit and the implicit adaptive control methods. On the other hand, for conventional control it is verified that the system tracks the desired trajectory with an offset. When experimental results of Explicit STAC second order model with free noise parameter and Explicit STAC second order model with noise parameter are compared with each other in trafectory tracking manner. Explicit STAC second order model with noise parameters provide good trajectory tracking then the other method. In addition, when experimental results of Explicit STAC second order model with free noise parameter and Implicit STAC second order model with noise parameter are compared with each other for trafectory tracking. Implicit STAC second order model with noise parameters provides better trajectory tracking. However, Explicit STAC algorithm require more much computational time then Implicit STAC algorithm.

Also, compared application result of MRAC method is more effective than conventional control method for tracing trace desired performance. All implementation studies demonstrate that the controller designed performs well in the described performance. When experimental results of MRAC and STAC methods are compared with each other. Both control methods satisfy good trajectory tracking with the same accuracy. But, STAC methods require less computational time as compared to MRAC method.

One of the problem of adaptive control methods is the initialisation of the system model parameters. The model and the numerical values of the parameters are not known at a sufficient degree of accuracy. Initially, the large tracing error is come into existence between the response of the system and desired performance. However these parameters may be set correctly after a few experimental study. The system output with self-tuning adaptive control and MRAC can track the desired trajectory very well, because the controller can adapt the parameter variation. System model parameters are adaptively contained in the characteristic of dynamic system, therefore, any change in the load, disturbance and the change in the trajectory are included in the system model parameters.

All implementation results demonstrate that the self-tuning adaptive control and MRAC algorithm can successfully be applied to any programmable system which has feedback controller when independent system dynamics are assumed for the motion of robotic systems. Therefore self-tuning adaptive control and MRAC algorithms require neither complex mathematical models of the system dynamics nor a prior knowledge of the environment.

In learning control method, a discrete iterative learning control algorithm is proposed to realise an accurate DC servo system. In the algorithm, the control input sequence for next operation is determined by utilising the tracking error as well as information of the dynamic characteristics obtained from the past operations such that the output trajectory tracks the given desired trajectory as closely as possible. To investigate gradual improvement of tracking performance in consecutive operations, the proposed algorithm is implemented to DC servo system. A series of experiments was performed for position tracking control of the system. The experimental results show that regardless of inherent nonlinearities and uncertainties associated with DC servo system, an accurate tracking control performance is obtained using the proposed learning control method.

Learning control can be applied more easily in real time for high speed systems since its algorithm is much more simple than the algorithm of the adaptive control methods. However, learning control is valid for repetitive tasks and it need some cycles to go to reach an acceptable tracking. On the other hand, adaptive control can adapt itself to any changes in the system or in the environment at the current cycle. Experimental implementation is the most important of this study. Because, it is not found any realistic experimental application of adaptive control to robotic systems in the literature survey. All the studies have been done as computer simulations.

RECOMMENDATION FOR FURTHER STUDY

The demands to the robotic systems are further increased with the development of robotic technologies and the application area being widened. Current industrial robots are now being used for material transfer, welding or spray painting, all of which operation can generally be handled by position control. But it is clearly not enough for some task, like assembly, inspection and test...etc. to use only position control. They need more complicated control function with several kinds of external sensor, such as tactile, force and vision. Since, the controlled systems are significant enough to render the conventional control methods ineffective due to the changes of load, trajectories and system speed in the task cycle. As a result of using the conventional control methods is reduced servo response speed which limits the precision and speed of the end-effector.

In recent years, three different concepts has been studied on robotic systems, separately. These are vision, trajectory planning and trajectory tracking. These three concepts may be combined in order to achieve an on-line trajectory planning and control. On-line trajectory planning and control refer to the determination of the time history of a motion by means of on-board sensory equipment and then generation and execution of the motion in real time. That means, for a specific task, the constraints for the trajectory including for obstacles along the path and constraints for co-ordination with other machine will be determined by means of sensory equipments. Then, this motion will be down loaded to the system to track the desired trajectory without human intervention.

For example, five robots are now used for cylinder head assembly in automobile industry. In this assembly, two robots are used to insert the inlet and exhaust valves which equipped with sensitive grippers that detect correct assembly and reject bent or oversized parts. The other two robots are used to assemble the spring seat and oil seal over the valve stem guide. Finally, a single robot is used for loading the assembly consisting of the spring, the top washer and the retaining collets over each off the eight valve stems. Only one robot can be used instead of five robots when the above concept is applied for the assembly operation. Results, it reduces of the cost of production due to the low cost for initial investment.

Also some of specific physical parameters of robotic system such as masses, moments of inertia and friction are determined by using an appropriate model and measurements in adaptive control methods.

REFERENCES

[1] Q. Wang and D. R. Broome, "A new simulation scheme for self-tuning adaptive control of robot manipulator.", Robotica vol. 9, pp 335-339, 1991.

[2] Hans Buter, "Model reference adaptive control from theory to practice.", Prentice Hall, 1992.

[3] Karl Johan Astrom Bjorm Wittenmark, "Adaptive control.", Addison-Wesley Publishing Company, 1989.

[4] B. Wittenmark, " Stochastic adaptive control method: a survey.", Int. J. Control, Vol. 21, No. 5, pp 705-730, 1975.

[5] Mei-Hua Liu W. Lin, "Multivariable self-tuning control with decoupling for robotic manipulators.", IEE Proceedings, Vol. 135, No. 1, pp 43-48, January 1988.

[6] R. Isermann, K. -H. Lachmann, D. Matko, "Adaptive control systems.", Prentice Hall, 1992.

[7] Bar-Shalom Y. and Tse E., "Dual effect, certainty equivalence and separation in stochastic control.", IEEE Trans. Auto. Control, vol. AC-19, No. 5, pp 494, 1974.

[8] Patchell J. W. and Jacobs O.L.R., "Separability, neutrality and certainty equivalence.", Int. J. Control, Vol. 13, pp 337, 1971.

[9] Isermann R., "Parameter-adaptive control algorithms-a tutorial.", Automatica, Vol. 18, pp 513-528, 1982.

[10] Astrom K. J., "Theory and application of adaptive control - a survey.", Automatica, Vol. 9, No. 5, pp 471-486, 1983.

[11] Kalman R. E., "Design of self-optimizing control system.", Transactions ASME, Vol. 13, pp 59-75, 1958.

[12] Astrom K.J and Wittenmark B., "On self-tuning regulators.", Automatica, Vol. 9, pp 189-199, 1973.

[13] Cegrell T. and Hedquist T., "Successful adaptive control of paper machines.",

Automatica, Vol. 11, pp 53-59, 1975.

[14] Borrison U. and R. Syding, "Self-tuning control of an ore crusher.", Automatica, Vol. 12, pp 1-7, 1976.

[15] Clarke D. W. and P. J. Gawthrope, "Self-tuning controller.", Proc. IEE., Vol. 122, pp 929-935, 1975.

[16] Saridis G. N. and R. N. Lobbia, "Parameter identification of linear discrete-time systems.", IEEE Trans. Auto. Control., AC-17, pp 52, 1972.

[17] C.J.Harris and S.A.Billings,"Self-tuning and adaptive control: theory and applications.", Peter Peregrinus Ltd., 1981.

[18] A. J. Koivo, "Self-tuning manipulator control in cartesian base coordinate system.", Journal of Dynamic Systems, measurement, and control., Vol. 107, pp 316-323, December 1985.

[19] Koivo and Guo, "Adaptive linear controller for robotic manipulators.", IEEE Transaction on Automatic Control, Vol. AC-28, No. 2, February 1983.

[20] M. B. Zarrop, "Generalized pole-placement self-tuning control of a robot manipulator.", Control Systems Centre Report 658, UMIST, Manchester M60 1QD, UK.

[21] Elliot, H.," Non-linear adaptive control of mechanical linkage systems with application to robotics.", Proceedings of Joint Automatic Control Conference, pp 1050-1055, 1984.

[22] W. Wahab and P.E. Wellstead, "Self-tuning control of a robot manipulator arm.", INT. J. PROD. RES., Vol. 27, No. 3, pp 423-449, 1989.

[23] A. J. Koivo and K. S. Lee., "Self-tuning control of planar two-link manipulator with non-rigid arm." CH2750-8/89/0000/1030$01.00, 1989 IEEE

[24] M. A. El-Sharkawi and Siri Weerasooriya., "Development and implementation of self-tuning tracking controller for Dc motors.", IEEE Transactions on Energy Conversion, Vol. 5, No. 1, March 1990.

[25] J. M. Finney, M. S. Bloor and G. S. Gill, "A pole-assignment controller for an electrohydraulic cylinder drive.", Journal of Dynamic Systems, measurement, and control ASME, Vol. 107, pp 145-150, June 1985.

[26] S. Dudowsky and D. T. DesForges, "The application of model reference adaptive control to robotic manipulators.", Journal of Dynamic System, measurement, and Control ASME, vol. 101, pp 193-200, September 1979.

[27] S. Nicosia and P. Tomei, " Model reference adaptive control algorithms for industrial robots.", Automatica, Vol. 20, No. 5, pp 635-644, 1984.

[28] A. Cerda, C. Abdallah, and R. Jordan, "Experimental implementation of MRAC for a Dc Servo Motor.", Proceedings of the 31st Conference on Decision and Control Tucson, Arizona, December 1992.

[29] A. Balestrino, G. De Maria and L. Sciavicco "An adaptive model following control for robotic manipulators.", Journal of Dynamic System, measurement, and Control ASME, Vol. 105, pp 143-151, September 1983.

[30] Chih-Lyang Hwang and Bor-Sen Chen, "Adaptive control optimal model matching in H^{∞}-norm space.", IEE Proc. D, Vol. 135, No. 4, pp 295-301, 1988.

[31] Chih-Lyang Hwang and Bor-Sen Chen, "Model reference adaptive control via the minimisation of output error and weighting control input.", IEE Proceedings., Vol. 136, No. 5, September 1989.

[32] Kirecci Ali, "Motion design for high-speed machines.", PhD. Thesis, Liverpool Polytechnic-UK., 1993.

[33] S. Arimoto, S. Kawamura and F. Miyazaki, "Bettering Operation of Robots by Learning.", J. Robotic Systems, Vol. 1, pp. 123-140, 1984.

[34] S. Kawamura, F. Miyazaki, and S. Kawamura, "Realisation of Robot Motion Based on a Learning Method.", IEEE Transactions on Systems Man and Cybernetics, Vol. 18, No. 1, pp. 126-134, 1988.

[35] Y. Gu and N. Loh, "Learning control in robotic systems.", Proc. IEEE. Int. Symp. Intelligent Control, PA, pp 360-364, 1987.

[36] P. Bondi, G. Casalina and L. Gambardella, "On the iterative learning control theory of robotic manipulators.", IEEE J. Robotics and Automation, Vol. 4, No. 1, pp 14, 1988.

[37] Z. Geng, J. D. Lee, R. L. Carroll and J. Xie, " Learning control system design for a paralel link robot manipulator.", Proc. IEEE. Int. Symp. Intelligent Control, 1988.

[38] Gultekin H. Murat, "Design and testing of a control software for a RRP robot manipulator.", M.S Thesis, University of Gaziantep, 1994.

[39] GW Instruments MacAdios II Hardware Series Instructions Manual, 1990, GWI, Somerville, Massachusetts 02143, USA.

[40] Bosh-Hydraulik, Katalog 2 section 13, closed loop proportional valves, Robert Bosh GmbH, Geschaftsbereich K6 Hydraulic, 7000 Stuttgart, Postfach 300240, Germany.

[41] Electro-craft Ltd Fourth Avenve crewe CW1 1XL, England.

[42] K.Warwick and D.Rees, "Industrial digital control systems.", Peter Peregrinus Ltd., 1988.

[43] R.Isermann, "Practical aspects of process identification.", Automatica, vol. 16, pp 575-587, 1980.

[44] Torsten Soderstrom and Peter Stoica., "System identification.", 1988.

[45] Peterka. V., "Adaptive recursive least squares identification algorithm.", IEEE Symposium on Adaptive processes, decision and control, 1970.

[46] M. Vidyasagar, " Controller system synthesis - a factorization approach." , MIT Press, 1985.

[47] P. E. Wellstead and M. B. Zarrop, "Self-tuning systems control and signal processing.", John Wiley and Sons, 1991

[48] H. Kurz, R.Isermann and R. Schumann., "Experimental comparison and application of various parameter adaptive control algorithm.", Automatica, Vol. 16, pp 117-133, 1980.

[49] Antti J. Koivo., "Fundamentals for control of robotic manipulators.", John Wiley and Sons, Inc., 1989.

[50] M. J. Grimble., "Implicit and explicit LQG self-tuning controllers.", Automatica, Vol. 20, No. 5, pp 661-669, 1984.

[51] J. K. Pieper, B. W. Surgenor and J. Z. Liu., "On self-tuning control of processes with time varying dead time.", FA2 10:15.

[52] H. T. Toivonen, "Variance constrained self-tuning control.", Automatica, Vol 19, No. 4, pp 415-418, 1983.

[53] D. W. Clark and P. J. Gawthrop, "Implementation and application of microprocessor-based self-tuners.", Automatica, Vol.17, No. 1, pp 233-244, 1981.

[54] J. Lu and T. Yahagi, "New design method for model reference adaptive control for nonminimum phase discrete systems with disturbances.", IEE Proceedings D, Vol. 140, No. 1, January 1993.

[55] A. G. Ulsory, Y. Koren and F. Rasmussen, "Principal developments in the adaptive control of machine tools.", ASME J. Dynamic Systems, Measurement and Control, Vol. 105, pp 107-112, 1983.

[56] T. P. Tsai and T.S. Wang, "Optimal design of control system large plant uncertainty.", Int. J. Control, Vol. 43, No. 3, pp.1015-1028, 1986.

[57] K. L. Moore, M. Dahled and S. P. Bhattacharyya, "Iterative learning for trajectory control.", Proceedings of the 28th Conference on Decision and Control Tempa, Florida, December 1989.

[58] H.J. Park, H.S. Cho and S.R. Oh., "A discreate iterative learning control method with application to electric servo motor control.", Proceedings 1989 American Control Conference (IEEE) Vol.3, pp 2640-2645, 1989.

[59] C.S. Williams, "Designing Digital Filters.", Prentice-Hall, Inc., 1986.

[60] A. Antoniou, Digital Filters Analysis and Design, McGraw-Hill, Inc. 1979.

[61] S.M. Bozic, Digital and Kalman Filtering, Edward Arnold Ltd., 1979.