A STUDY ON MANİPULATOR CONTROL

by

NEVZAT KILINÇ

B.S. in E.E., Boğaziçi University, 1982

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering

Boğaziçi University

1985

ACKNOWLEDGEMENTS

I would like to express my gratitudes to Doç.Dr. Yorgo Istefanopulos for his valuable advice, guidance, encouragement, and for all the help he has given to me throughout this study.

I also wish to thank Doç.Dr. Okyay Kaynak and Y.Doç.Dr. Ahmet Denker for their interest and helpful suggestions.

Nevzat Kılınç

A STUDY ON MANIPULATOR CONTROL

ABSTRACT

In this study, a brief information about the kinematics and dynamics of a mechanical manipulator is presented and a control method is examined. The method used to obtain the dynamic equations of motion is well suited for computer usage.

In the control of a manipulator the Model Reference Adaptive System theory is used. The ideal formulation of the manipulator is chosen as the Reference Model and the manipulator system outputs are forced to follow the reference model.

Three different control methods are also examined to make a comparison with the presented control scheme.

To evaluate the performance of the Manipulator control system a simulation study is performed and it is observed that for different motions and varying loads the system shows a good performance.

ROBOT KOLU KONTROLU ÜZERİNE BİR ÇALIŞMA

KISA ÖZET

Bu çalışmada mekanik bir robot kolunun kinematik ve dinamiği hakkında özet bir bilgi verilmekte ve bir kontrol yöntemi incelenmektedir. Robot kolunun dinamik denklemlerini elde etmekte kullanılan yöntem, bilgisayar kullanımı için oldukça uygun bir yapıdadır.

Robot kolu kontrolunda Model Referans Adaptif Sistem teorisi kullanılmaktadır. Bu yöntemde, robot kolunun ideal denklem biçimi referans model olarak seçilmiş ve robot kolu kontrol sistemi çıktılarının, bu modelin çıktılarını izlemeleri istenmiştir.

Ek olarak, sunulan kontrol yöntemi ile bir karşılaştırma yapabilmek amacıyla üç ayrı kontrol yöntemi incelenmektedir.

Robot kolu kontrolunda kullanılan yöntemin performansını değerlendirmek amacıyla bir bilgisayar benzeşim çalışması yapılmış ve farklı hareket ve yükler için sistemin iyi bir performans gösterdiği gözlenmiştir.

TABLE OF CONTENTS

Page No.

LIST OF FIGURES

Page No.

LIST OF TABLES

LIST OF SYMBOLS

a	Unit approach vector of the end effector
$A_{i,i-1}$	Transient matrix between link i and link i-1
e_i	Unit vector of the ith joint axis
	Angular acceleration
F	System matrix
G	Input matrix
G_i	Gibbs-Appel function for the ith degree of freedom
$H(s)$	Transfer function of the system
$H(x,u,\nabla V,t)$	A prehamiltonian function
I_i	Inertia dyadics of the ith link
J	Performance index of the system
$J(\theta)$	Jacobian matrix of the system
K_p	Position feedback gain matrix
K_v	Velocity feedback gain matrix
m_i	Mass of the ith element
$M(q)$	Inertia matrix
n	Unit normal vector of the end-effector
P_i	Position vector of the ith joint
q_i	Generalized coordinate corresponding to the ith degree of freedom
r_i	Input voltages of the adaptive linear unit
R	Output matrix of the system
$R_{x,\theta}$	Rotation matrix corresponding to the angular rotation about the X axis
$R_{y,\theta}$	Rotation matrix corresponding to the angular rotation about the Y axis

$R_{z,\theta}$	Rotation matrix corresponding to the angular rotation about the Z axis
s	Unit slide vector of the end-effector
S_i	Input vector of a perceptron
T_i	Vector of center of mass of the ith link
T	Homogeneous transformation matrix
U	Generalized force vector
V	Value function
∇V	Gradient of the value function
V(q,q)	Column matrix representing the gravitational, centrifugal and coriolis's forces
W	Angular velocity
W_{ij}	Adjustable weights
x	State vector of the system
y	Output vector of the system
η	Output signal of the adaptive linear unit
θ	Vector of joint angles
μ	Input conductances of the adaptive linear unit

I. INTRODUCTION

Manipulators have been used extensively in the fields of undersea and space explorations and maintenance operations, industrial auotomation and nuclear industry. The main reasons of manipulator applications in these fields are the dangerous working conditions for human beings and for the reduction of human labor and time consumption. In most of these applications, the control is accomplished by a remote human operator which is called tele-operator configuration. However, the manipulators used in industrial automation are generally automatically controlled devices.(1)

In order to achieve advanced automation, a mechanical manipulator with human-like versatility is required. Although manual tasks may be achieved by a conventional system with various types of machines, the inflexibility of these machines makes the human-like manipulators more attractive. In reality, it is difficult to design a system with the versatility that a human has in carrying out various tasks. It is reasonable to attempt the structural and motional capabilities of a human arm in an artificial manipulator, but it may not be possible to design a manipulators that behaves equivalently to that of a human arm. Since the human arm and brain is a highly developed manipulation system.

The developments of the low-cost, high performance, small size computers have contributed to improvement of the control methods of manipulators. It is desired that a manipulator should fulfil complicated tasks with high speeds and over a wide range of applications. Since, the dynamics of a manipulator is characterized by the inherent nonlinearities and complexity, the difficulties in realization makes the classical control techniques inadequate to obtain high performance control functions.

The kinematic considerations are important for the proper selection of manipulator configuration. The first stage in improving a control technique is to obtain a mathematical formulation of the dynamic model of a manipulator. Many methods, e.g. Newton-Euler formulation, have been developed for obtaining dynamic equations of manipulators.[2,3]

The control concept of a manipulator can be expressed as the process of finding what each joint actuator does at every time interval and under various external conditions. That is, it may be affected from external effects as well as inherent limitations and nonlinearities so that closed form analytic solutions may not be available.

A number of control techniques have been proposed for dynamic control of manipulators. In the computed torque drive method [4,5], a prescribed trajectory is given and the corresponding torque signals are obtained numerically. The main drawback of this method which makes it difficult to use in on-line control, is the significant computational effort to solve the dynamic equations of the manipulator.

The resolved motion rate control [6,7], is a method in which the motions of the various joint actuators are combined and resolved into separately controllable end-effector movements along the world-coordinates which specify the position and orientation of the manipulator.

In some other control methods, the dynamic equations are linearized along the desired trajectory and the control is obtained analytically.[8] In some cases, nonlinear feedback compensation is used.[9]

The optimal control theory can also be used to[10,11] obtain suboptimal or intentionally nonlinear controllers.

An adaptive control algorithm can also be utilized for handling the control problem such that the system makes the necessary internal modifications to obtain the desired response.[12,13]

There are some other control techniques that the dynamic model of the manipulator is not required. A table-look up method is used to produce output signals that are joint actuator drive signals. [14,15]

Another method makes use of the variable structure systems theory.[16]

In Chapter II, a brief information is given about the kinematics and dynamics of mechanical manipulators. In Chapter III, an adaptive control algorithm called Model Reference Adaptive system is examined to be used in a simulation study for a manipulator.

In Chapter IV, three control methods of the above techniques are studied to compare with the one used in simulation study.

In Chapter V, the simulation results are presented which show that the selected algorithm achieves the objective of controlling the manipulator.

II. KINEMATICS AND DYNAMICS OF A MECHANICAL MANIPULATOR

2.1 KINEMATIC CONSIDERATIONS

In order to specify the position and orientation of an object in space, it is necessary to define six independent coordinates.

Manipulators consist of links which are rigid bodies connected by spherical, cylindrical, cartesian, and screw type joints. One of the most popular type of joint is the cylindrical which allows a rotation about the cylinder axis and a translation along the axis.

A revolute joint is a special version of cylindrical joint which allows rotation about the axis and a prismatic or translational joint allows a translation along the axis. In other words, when the lead of the cylindrical joint is zero, it becomes a revolute joint, and when the rotational action is not allowed, it is called a prismatic joint.[17]

Typical representations of these joints are shown in Figure 2.1.

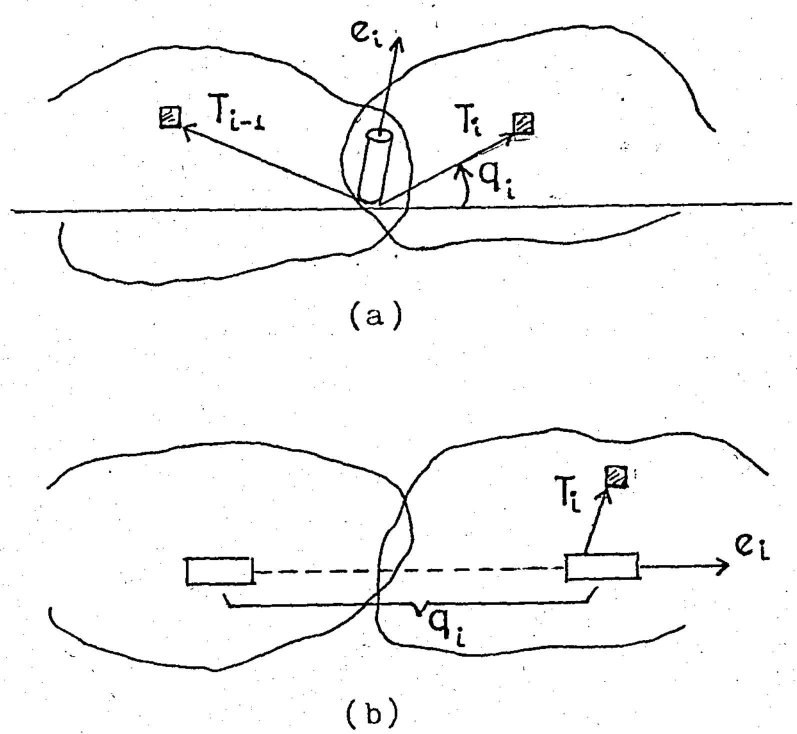


Figure 2.1 Typical representations of revolute and prismatic joints. a) Scheme of revolute joint, b) Scheme of prismatic joint.

In this figure, e_i is unit vector of the i th joint axis, T_i , the vector of center of mass of the i th link, and q_i is the generalized coordinate corresponding to the i th degree of freedom. As shown in the Figure, a generalized coordinate can be defined as the angle of joint rotation or a sliding along the unit vector.

After the addition of any link to the kinematic chain, and considering a revolute joint, a transient matrix A_i can be defined. Assigning one coordinate system to each joint, the transient matrix columns represent the unit vectors of the connected coordinate system expressed in terms of the fixed base system. In order to form the transient matrix which corresponds to some defined angle q_i, it is necessary to rotate each unit vector by an angle q_i about axis e_i, and then apply Rodrig's formula,

$$Q_{ij} = q_{ij} \operatorname{Cos} q_i + (1 - \operatorname{Cos} q_i)(e_i q_{ij}) e_i + e_i x q_{ij} \operatorname{Sin} q_i \quad (2.1)$$

where q_{ij} is the j'th column of transient matrix which is unit vector before rotation and Q_{ij} is the j'th column of the transient matrix after rotation. Then the transient matrix is obtained as follows

$$\underline{\underline{A}}_i = \begin{bmatrix} Q_{i1} & Q_{i2} & Q_{i3} \end{bmatrix} \quad (2.2)$$

Additionally the relationship

$$\underline{\underline{A}}_{i-1,i} = \underline{\underline{A}}^{-1}_{i,i-1} \quad (2.3)$$

exists between the coordinate systems.

Using the information given above, the transient or transformation matrices which correspond to the angular rotations about the X, Y, and Z axes by the angles θ, ϕ, ψ respectively, can be found as in the following form

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \operatorname{Cos}\theta & -\operatorname{Sin}\theta \\ 0 & \operatorname{Sin}\theta & \operatorname{Cos}\theta \end{bmatrix} \quad (2.4)$$

$$R_{y,\Phi}=\begin{bmatrix} \cos\Phi & 0 & \sin\Phi \\ 0 & 1 & 0 \\ -\sin\Phi & 0 & \cos\Phi \end{bmatrix} \quad (2.5)$$

$$R_{z,\Psi}=\begin{bmatrix} \cos\Psi & -\sin\Psi & 0 \\ \sin\Psi & \cos\Psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.6)$$

These rotation matrices do not give any information about translational joints.Therefore considering a vector which gives the position of the origin of connected coordinate system relative to the fixed base system, a 4x4 matrix called " Homogeneous trasformation matrix" is obtained in the following form[18]

$$\underline{\underline{T}} = \left[\begin{array}{c|c} \text{Rotation Matrix} & \text{Position Vector} \\ \hline \text{Perspective Transformation} & \text{Scaling Factor} \end{array}\right] = \left[\begin{array}{c|c} \underline{\underline{R}}_{3x3} & \underline{p}_{3x1} \\ \hline \underline{f}_{1x3} & 1x1 \end{array}\right] \quad (2.7)$$

In this form of transformation,the lower left 1x3 matrix,i.e. the perspective transformation is used for calibration the camera model and computer vision.When no camera model and computer vision is involved,the elements of this matrix are set to zero.The lower right submatrix with dimension 1x1 is scaling factor which is generally equal to 1 for robotics applications.

In summary, a homogeneous transformation matrix is a mapping between the reference coordinate system and a connected coordinate system, i.e. a vector defined in any connected coordinate system can be expressed in terms of fixed base system by using corresponding homogeneous transformation matrix.

The homogeneous transformation matrix can also be written in the form,

$$\underline{\underline{T}} = \begin{bmatrix} \underline{n} & \underline{s} & \underline{a} & \underline{p} \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad (2.8)$$

In this representation, $\underline{p}$ is the position vector of the end-effector, $\underline{n}$, $\underline{s}$, and $\underline{a}$ are the unit normal, unit slide and unit approach vectors of the end-effector, respectively. These three orthonormal vectors specify the orientation of the end-effector as shown in Figure.2.2 and by a translation corresponding to $\underline{p}$ the origins of X_o, Y_o, Z_o and $\underline{n}, \underline{s}, \underline{a}$ coincide with each other.

The inverse of the homogeneous transformation can be written as follows

$$\underline{\underline{T}}^{-1} = \left[\begin{array}{ccc|c} & & & -\underline{n}^T\underline{p} \\ & \underline{\underline{R}}^T_{3x3} & & -\underline{s}^T\underline{p} \\ & & & -\underline{a}^T\underline{p} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \qquad (2.9)$$

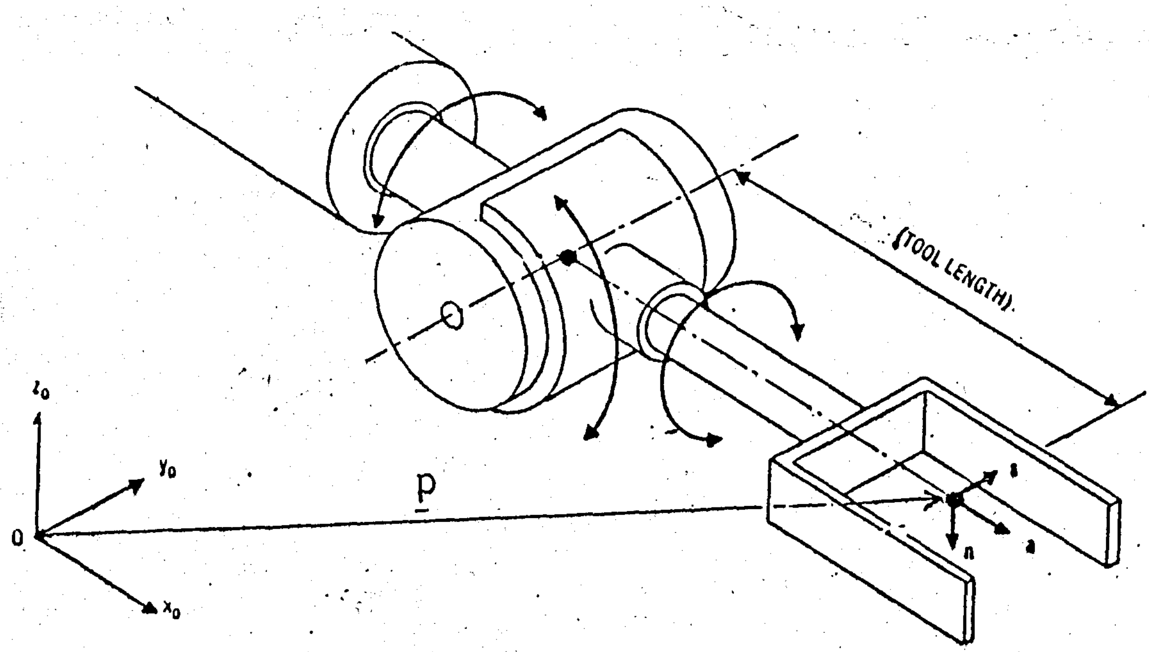


Figure 2.2 Position and orientation vectors of the end-effector.

The inverse homogeneous transformation is used to express the X_0,Y_0,Z_0 coordinate system with respect to n,s,a coordinate system,which is associated with the end-effector.

Since a manipulator is an open active chain consisting of rigid links connected in a serial manner with rotational or translational joints,one coordinate system can be assigned to each joint and a homogeneous transformation matrix can be defined for each pair of them.Defining the coordinate transformation matrix between link i-1 and link i as A_{i-1}^{i} ,then the transformation between the base and end-effector of the manipulator can be expressed as the following

$$\underline{\underline{T}} = \underline{\underline{A}}_o^1\, \underline{\underline{A}}_1^2 \,\ldots\ldots \underline{\underline{A}}_{i-1}^i = \prod_{i=1}^{n} \underline{\underline{A}}_{i-1}^i \qquad (2.10)$$

where n is the number of degrees of freedom of the system. For a six degrees of freedom manipulator;

$$\underline{\underline{T}} = \underline{\underline{A}}_o^1\, \underline{\underline{A}}_1^2\, \underline{\underline{A}}_2^3\, \underline{\underline{A}}_3^4\, \underline{\underline{A}}_4^5\, \underline{\underline{A}}_5^6 = \begin{bmatrix} \underline{n} & \underline{s} & \underline{a} & \underline{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.11)$$

then the resulting matrix specifies the position and orientation of the end-effector with respect to the reference frame.

2.2 MANIPULATOR DYNAMICS

The major problem encountered in the applications of industrial robotic devices is the low operation speed in their motions.To increase the power of actuator driving motors and demanding higher speeds from the servos is not a remedy to handle the low operation speed problem. This due to the fact that,it is not possible to see a manipulator as a purely kinematic device at high speeds. As the operation speed of a manipulator increases ,inertial effects of the links originating from the velocities and accelerations,leads to some complex problems in control of robot arms.It is required that the motors should provide forces and torques to compensate for these internal effects. However,since it is very difficult to compute these forces, an exact dynamical model of a manipulator is not possible to obtain, instead, under some simplifying assumptions an approximate model is tried to be established.

The dynamic model of a manipulator can be represented by a set of n- nonlinear, second order differential equations describing the motion of the system in the space of internal (joint) coordinates. By neglecting friction forces in joints the equations of motion will be of the following form;

$$\underline{\underline{M}}(q)\ \ddot{\underline{q}} + \underline{V}(q,\dot{q}) = \underline{U} \qquad (2.12)$$

or

$$\ddot{\underline{q}} = \underline{\underline{M}}^{-1}(q)\left[\underline{U} - \underline{V}(q,\dot{q})\right] \qquad (2.13)$$

where $\underline{q} = \left[q_1\ q_2 \ldots q_n\right]^T$ is the vector of generalized coordinates, U is the vector of driving torques and forces. M is the inertia matrix whose elements are functions of generalized coordinates, V is a column matrix which is also dependent on joint positions and velocities. It represents gravitational, centrifugal and Coriolis's forces. In these equations n is the number of degrees of freedom.

Although, there other equivalent forms representing the dynamic model of a manipulator, the above mentioned form is especially suitable for synthesis of cotrol algorithms and for simulation of particular control laws, since it is convenient to solve the " inverse problem" of the mechanics, that is, to determine the motions for given driving forces and torques.(2)

In principle, the methods for forming dynamic model of a mechanical manipulator can be classified into three basic groups;

a. The models based on the Newton-Euler equations and the principal theorems of the mechanics of the system.

b. The method of Lagrange's equations.

c. The models formed on the basis of acceleration, "energy" or Gibbs function.

All these have some common properties based on the information about the kinematic scheme of the manipulator and the recursive nature for calculation of velocities and accelerations in an outward manner from the fixed-base coordinate frame.It is possible to solve both direct and inverse problems of mechanics by using all these methods.However they differ from each other in terms of computational efforts.When on-line applications are considered both Lagrangian and Newton-Euler formulations are almost equivalent in this respect.The method based on the Gibbs-Appel equations makes use of the recursiveness in internal coordinate systems rather than computing the velocities and accelerations of centers of masses in terms of fixed-base coordinate system.Therefore,it is the most efficient method to obtain the dynamic equations of a mechanical manipulator.

2.2.1 The Method of Gibbs-Appel Equations

The Gibbs-Appel function is defined for a mechanical system with n degrees of freedom in cartesian coordinates as in the foolowing form

$$G = \sum_{i=1}^{n} G_i = \sum_{i=1}^{n} \frac{1}{2} m_i(\ddot{x}_i^2 + \ddot{y}_i^2 + \ddot{z}_i^2) \qquad (2.14)$$

where m is the mass of the element, $\ddot{x}$, $\ddot{y}$, and $\ddot{z}$ are accelerations in the direction of x,y,z coordinates respectively. The function G_i can also be written in the form

$$G_i = \frac{1}{2} m_i a_i + \frac{1}{2} \epsilon_i^T J_i \epsilon_i - \left[2(J_i w_i) X \; w_i\right] \epsilon_i \qquad (2.15)$$

In equation (2.15) m is the mass of the element, a is the acceleration of the center of the mass of the element, ϵ is the angular acceleration, w is the angular velocity, and J is the moment of inertia about the mass center.

Introducing the generalized coordinates $q=[q_1 \cdots q_n]^T$, the motion of a system can be expressed by Appel's equations as

$$\frac{\partial G_i}{\partial \ddot{q}_i} = P_i \quad , \quad i=1,2,3,\ldots.n \qquad (2.16)$$

or in matrix form

$$\frac{\partial G}{\partial \ddot{q}} = \underline{P} \quad , \quad \left[\frac{\partial G}{\partial \ddot{q}}\right] = \left[\frac{\partial G}{\partial \ddot{q}_1} \frac{\partial G}{\partial \ddot{q}_2} \cdots \frac{\partial G}{\partial \ddot{q}_n}\right]$$

and

$$\underline{P} = \left[P_1 P_2 \ldots\ldots P_n\right]^T \qquad (2.17)$$

P_i represents the generalized force corresponding to the generalized coordinate q_i .

Using the equation (2.16) the dynamic equations of a mechanical manipulator is obtained as shown in detail in[19] ,and of the form

$$\underline{\underline{M}}(q) \; \underline{\ddot{q}} + \underline{V}(q,\dot{q}) = \underline{U} \qquad (2.18)$$

in this equation M is the symmetric and positive definite inertia matrix, V is the nonlinear function of generalized coordinates which represents gravitational, centrifugal and Coriolis's forces. U is the generalized force vector which includes torques and sliding forces at joints, potential forces, friction forces, etc. In this form of modelling the vector V has the following form

$$\underline{V}(q,\dot{q}) = \left[\underline{\dot{q}}^T \underline{\underline{C}}^1 \underline{\dot{q}}, \underline{\dot{q}}^T \underline{\underline{C}}^2 \underline{\dot{q}}, \ldots \underline{\dot{q}}^T \underline{\underline{C}}^n \underline{\dot{q}}\right]^T \qquad (2.19)$$

and the elements of C^k matrices satisfy the following conditions

$$c^k_{ij} = c^k_{ji} \qquad , i,j,k=1,2,\ldots n$$

and (2.20)

$$c^k_{ik} = 0 \quad , c^k_{ij} = -c^j_{ik} \quad , \quad i \leqslant j,k$$

This method is well suited for realizing a computer oriented algorithm in the on-line control applications of a mechanical manipulator.

2.3 MANIPULATOR USED IN SIMULATION STUDY

Figure 2.3 shows the six-degrees of freedom manipulator considered in this study. The system elements are assumed to be rigid and some effects such as connection clearances and motor backlash are neglected.[20]

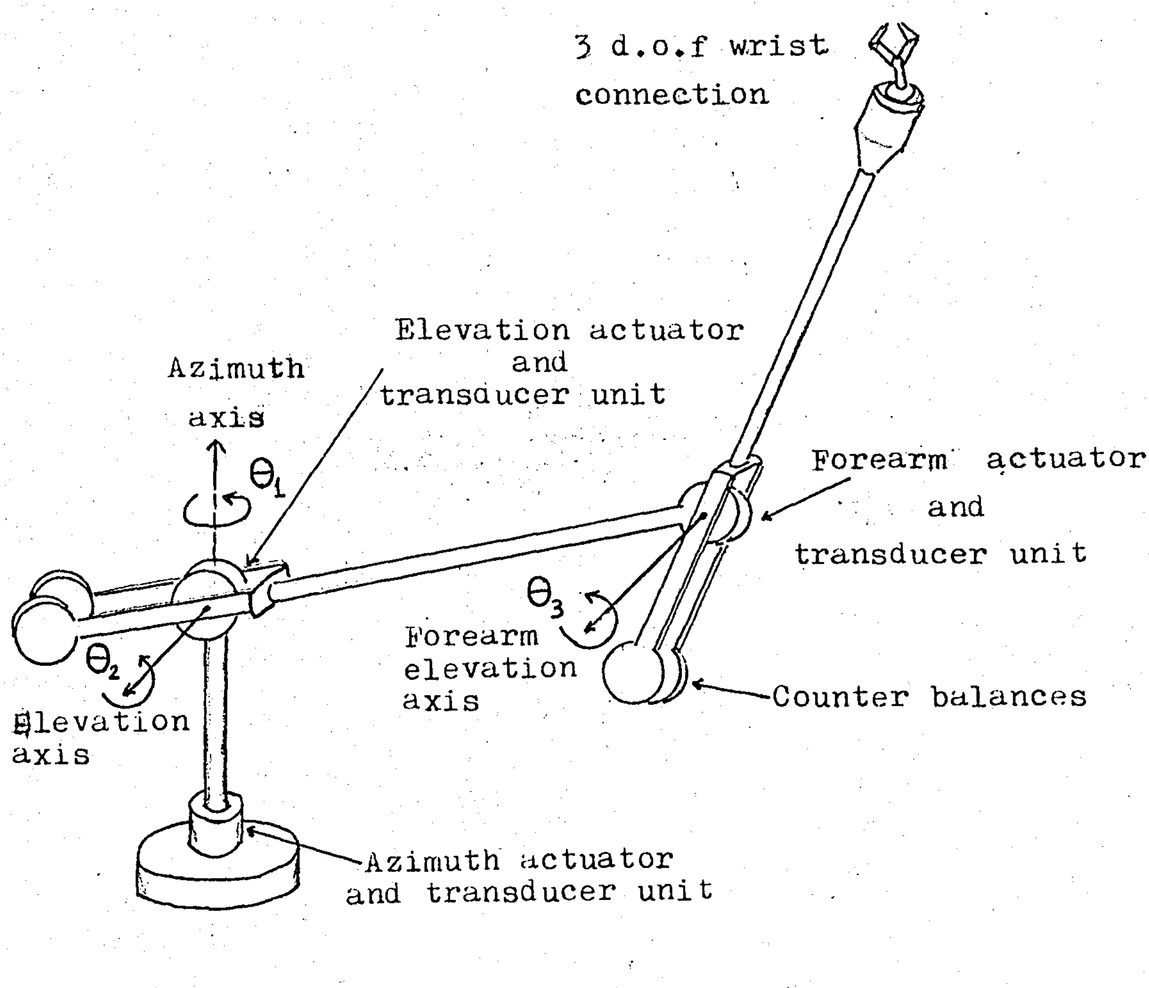


Figure 2.3 Six Degrees of Freedom Manipulator

In this arm configuration, the counter balances are used to minimize the effects of gravity. For the dynamic analysis and control of this manipulator, it is assumed that the payload is grasped by end-effector of the manipulator and so the mass properties of the payload and wrist can be combined. In addition, the dimensions of the wrist and payload are very small with respect to the other system elements.

Under the above assumptions, in the dynamics of the system only the generalized coordinates $q = [\theta_1 \theta_2 \theta_3]^T$ will be effective.

The schematic representation of the manipulator is shown in Figure 2.4.

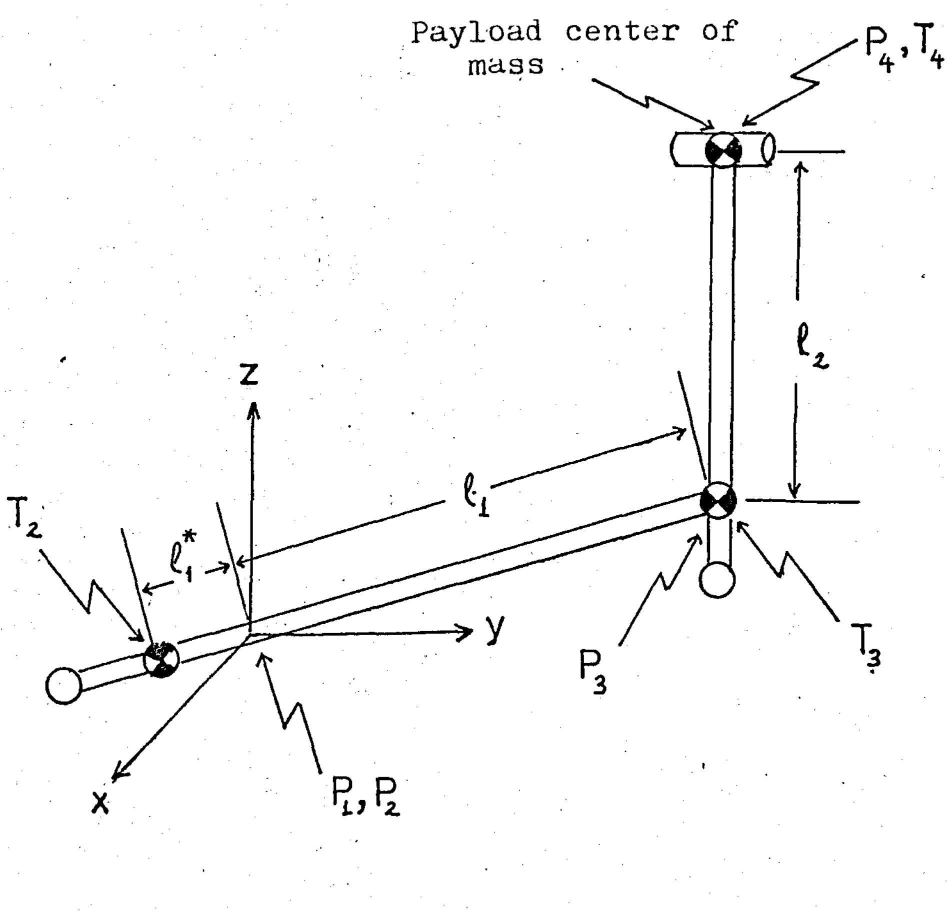


Figure 2.4 Schematic representation of the manipulator.

In Figure 2.4 P_i is three dimensional position vector of the joint between link i-1 and link i . T_i is three dimensional vector of center of mass of link i . e_i is unit vector of rotation, and θ_i is the rotation angle about axis e_i .

Using the right-hand coordinate systems, one coordinate frame is assigned to each joint of the manipulator as shown in Figure 2.5. (21)

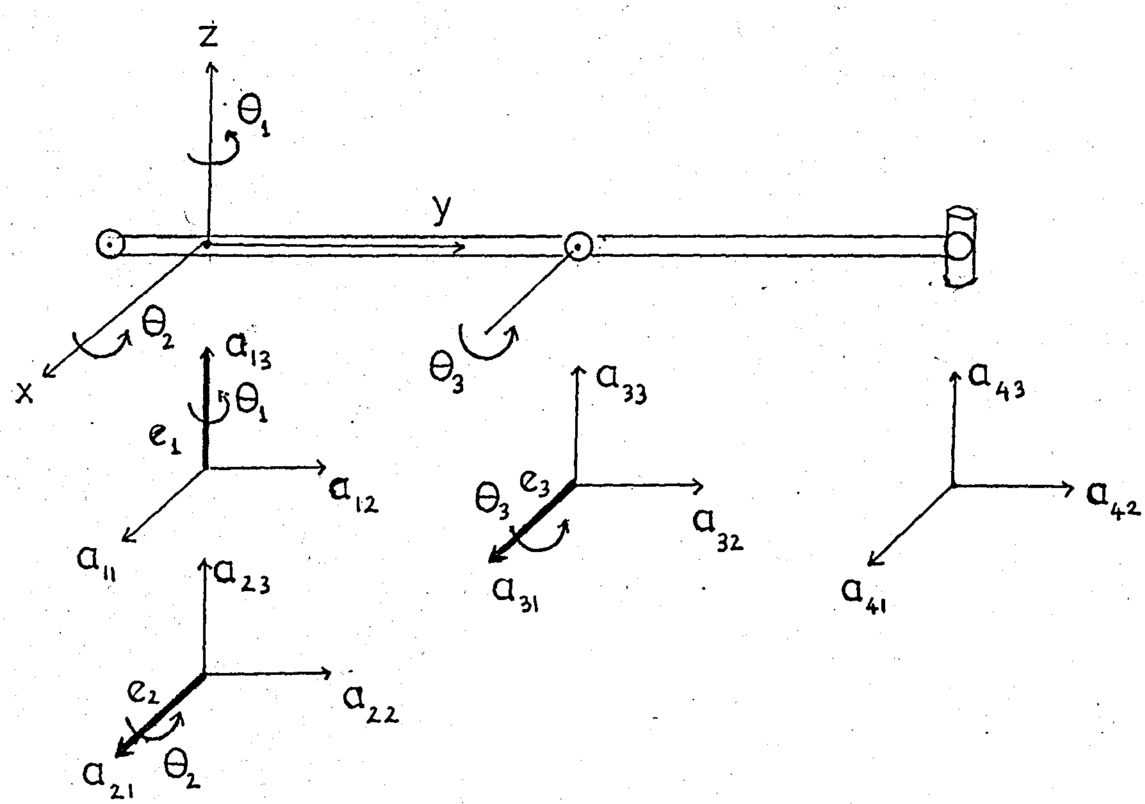


Figure 2.5 Manipulator coordinate frames.

In Figure 2.5, the joint axes $\underline{e}_1$, $\underline{e}_2$, $\underline{e}_3$ are defined as;

$$\underline{e}_1 = (0,0,1)^T$$
$$\underline{e}_2 = (1,0,0)^T$$
$$\underline{e}_3 = (1,0,0)^T$$

where $\underline{e}_2$ and $\underline{e}_3$ are always parallel to each other during any motion. Then, the rotation matrices for each joint can be found as follows

$$R_1 = \begin{bmatrix} \text{Cos}\theta_1 & -\text{Sin}\theta_1 & 0 \\ \text{Sin}\theta_1 & \text{Cos}\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \text{Cos}\theta_2 & -\text{Sin}\theta_2 \\ 0 & \text{Sin}\theta_2 & \text{Cos}\theta_2 \end{bmatrix}$$

$$R_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \text{Cos}\theta_3 & -\text{Sin}\theta_3 \\ 0 & \text{Sin}\theta_3 & \text{Cos}\theta_3 \end{bmatrix}$$

III. ADAPTIVE CONTROL SYSTEMS

3.1 INTRODUCTION

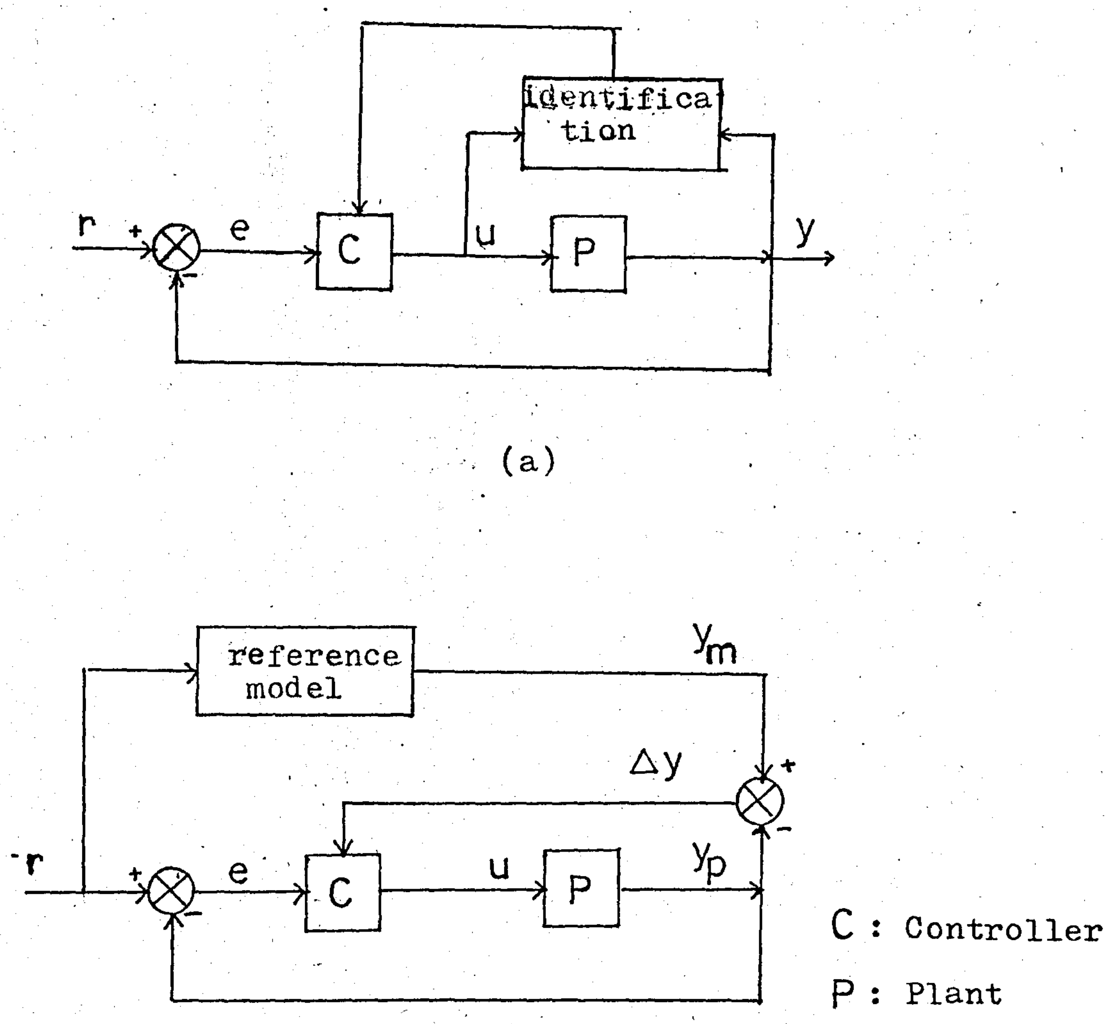
The developments in the field of digital computers, have increased the popularity of adaptive (self-organizing) control algorithms, due to the contributions to the facilities in realization.

Adaptive control systems adapt their internal behaviour to the environmental changes and the controlled system, as well. Although, different classifications are possible, mainly, two types of adaptive control methods may be considered; open-loop (feedforward) or closed loop (feedback) adaptation schemes. The selection of these schemes is based upon the information about the system to be controlled.

The feedback adaptation schemes can be examined under two main groups. Self optimizing adaptation aims to obtain an optimal control performance depending on the system information and the controller type.

Model reference adaptation, however, try to get an input signal such that the system behaviour will be similar to a given reference model.

The basic schemes of feedback adaptive control systems are shown in Figure 3.1



(a)

(b)

Figure 3.1 Basic configurations of feedback adaptive controllers; a) Self optimazing adaptive system b) Model Reference Adaptive system.

3.2 MODEL REFERENCE ADAPTIVE SYSTEMS (MRAS)

The basic scheme of a Model Reference Adaptive system is shown in Figure 3.2.

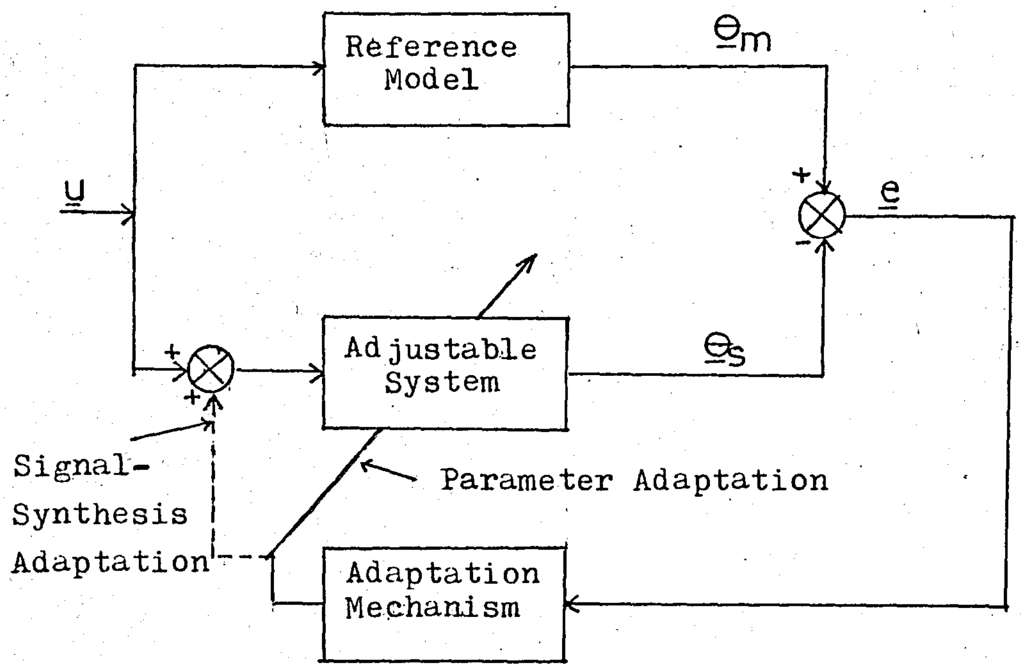


Figure 3.2 Basic scheme of Model Reference Adaptive System (MRAS).

In Figure 3.2 the adjustable system is a system which adjusts its behaviour by either changing its internal structure or modifying its input signals.(12,13,22)

The output of the Reference Model is the desired response of the adjustable system. The aim of adaptation is to minimize the difference between the responses of the adjustable system and the reference model. This task can be accomplished by making necessary changes in adjustable system using adaptation mechanism whose input is the generalized error vector.

3.2.1 Classifications of MRAS

When the classifications of MRAS are concerned, various criteria should be taken into account, since it is difficult to express all types of MRAS considering one criterion. Some criteria for categorization of MRAS are as follows;

a. Structure
b. Principle of adaptation
c. Conditions of operation
d. Index of performance
e. Application fields

The classifications according to these criteria are given in the following mánner;

a. Structure

i) Parallel MRAS
ii) Series MRAS
iii) Series-parallel MRAS

The configurations of these MRAS are shown in Figure 3.3 In the figure, when the parallel MRAS is used for parameter identification, it is called " output error method ". The series and series-parallel MRAS schemes are also called "input error method" and "equation error method" respectively.

b. Principle of Adaptation

i) Adjusting the parameters of the adjustable system.

ii) Signal synthesis adaptation.

iii) Combination of these adaptations.

c. Conditions of Operation

i) Using test signals applied to the input of the system or to the adjustable system.

ii) Without test signals.

d. Performance Index

i) Minimization of generalized error which is the vector resulting from the differences of responses of adjustable system and reference model.

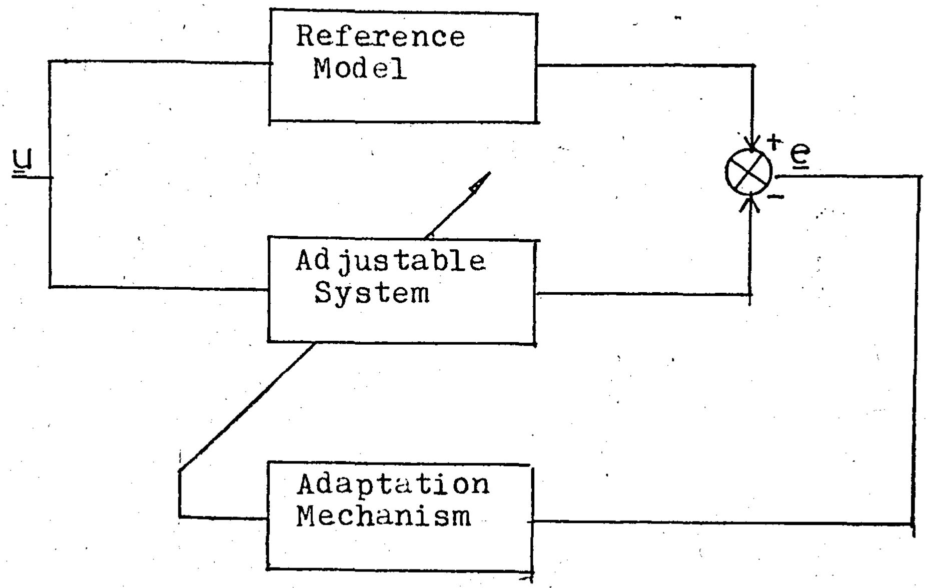
ii) Minimization of state distance, the vector obtained by the difference of states of reference and adjustable model.

iii) Minimization of structure distance, vector of difference of parameter vectors of adjustable and reference models.

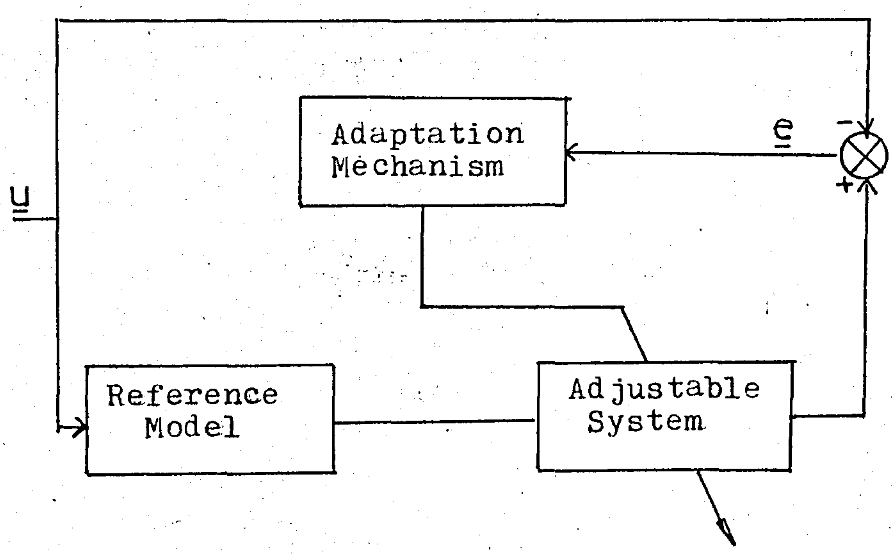
These performance indices are different from the desired performance index of the whole system, they are only used to obtain necessary adaptation rules for MRAS.

e. Application Fields

i) Model Following systems

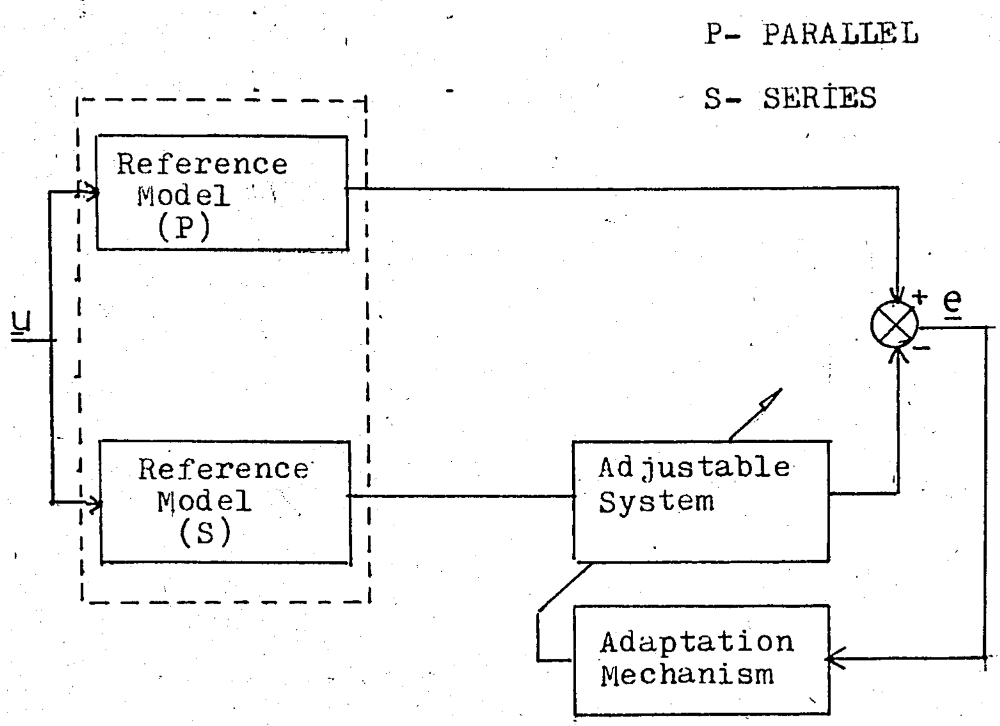


(a)

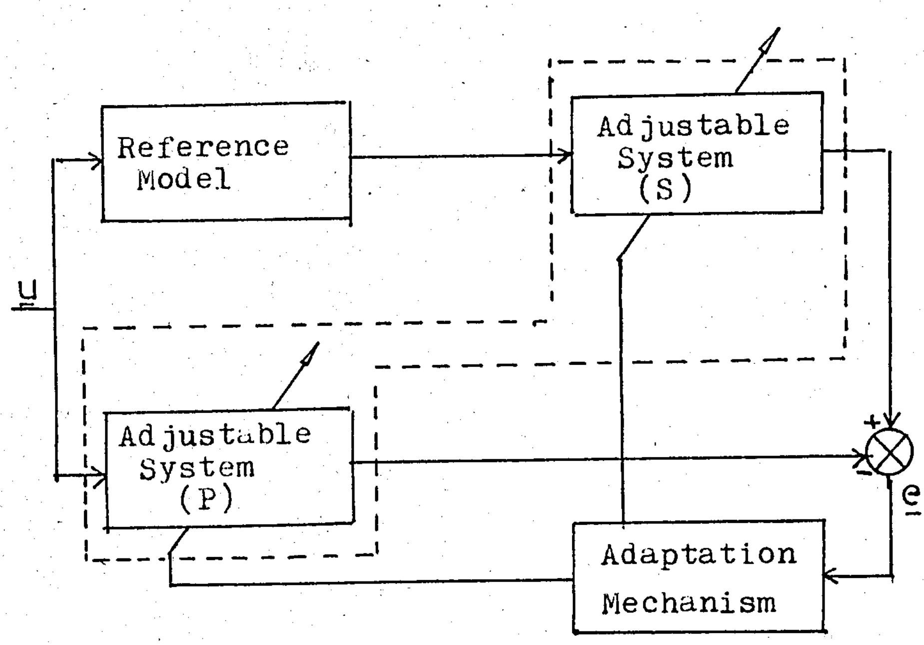


(b)

Figure 3.3 The configurations of MRAS according to the classification based on the structure. a) parallel, b) series, c,d) series-parallel.



(c)



(d)

Figure 3.3 (CONTINUED)

ii) Identification of unknown parameters of systems

iii) State observation

iv) Self-adaptive regulation.

There are many methods for the design of MRAS adaptation mechanisms. Some of the fundamental design methods are as follows;

a. Design methods based on the Estimation Theory

b. Design methods based on the Stability Theory

c. Design methods based on the Local Parameter Optimization Theory

d. Design methods based on the Optimal Control Theory

Although, there exists other design methods developed recently, the second method above, based on the stability theory will be examined in this chapter.

3.2.2 Design Method Based on the Stability Theory

Although, the second method of Lyapunov can be used for the design of MRAS, more satisfactory results can be obtained using the hyperstability theory of Popov, since this method directly gives the structure of the adaptation mechanism.[23]

Consider the multivariable standard system shown in Figure 3.4. The state equations of the system is of the form;

$$\dot{\underline{x}} = \underline{\underline{F}}\underline{x} + \underline{\underline{G}}\underline{u}_1 \qquad (3.1)$$

$$\underline{y} = \underline{\underline{R}}\underline{x} \qquad (3.2)$$

$$\underline{u} = -\underline{u}_1 \qquad (3.3)$$

$$\underline{u} = g(\underline{y}(\tau),t) \quad , \tau \leqslant t \qquad (3.4)$$

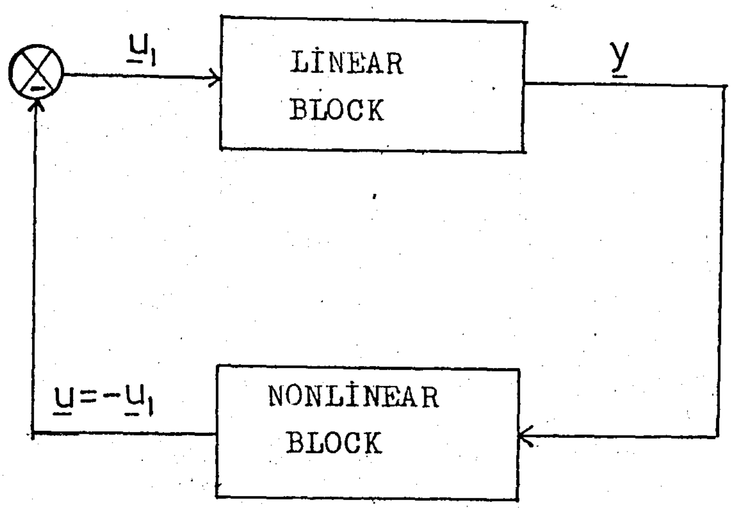


Figure 3.4 Standard multivariable nonlinear control system.

It is assumed that, the matrix pairs $[\underline{F},\underline{G}]$ and $[\underline{R},\underline{F}]$ comprises completely controllable and observable sets, respectively. Additionally, it is assumed that the vectors $\underline{u}$ and $\underline{y}$ have the same dimension, the function g is a nonlinear function of $\underline{y}$. Taking into account the subset of $\underline{u}$ such that it satisfies the inequality;

$$\int_0^t \underline{u}^T(\tau).\underline{y}(\tau)\, d\tau \geqslant -\gamma_o^2 \quad , \forall t \geqslant 0 \qquad (3.5)$$

where γ_o^2 does not depend on time but it depends on the initial conditions of the system.

The theorem for asymptotic stability is the following;

Theorem: The necessary and sufficient conditions for the above system being asymptotically hyperstable are;

a) The transfer function from $\underline{u}_1$ to $\underline{y}$;

$$\underline{\underline{H}}(s) = \underline{\underline{R}}(s\underline{\underline{I}}-\underline{\underline{F}})^{-1}\underline{\underline{G}} \tag{3.6}$$

is strictly positive real.

b) All poles of $\underline{\underline{H}}(s)$ being in the left half plane,

Re $s < 0$

c) $\underline{\underline{H}}(jw) + \underline{\underline{H}}^{T*}(jw)$ is positive definite Hermitian for all real w .

Consider the parallel MRAS described by the following equations;

$$\dot{\underline{x}}_m = \underline{\underline{A}}_m\underline{x}_m + \underline{\underline{B}}_m\underline{u} \tag{3.7}$$

$$\underline{\theta}_m = \underline{\underline{C}}\underline{x}_m \tag{3.8}$$

$$\dot{\underline{x}}_s = \underline{\underline{A}}_s(t)\underline{x}_s + \underline{\underline{B}}_s(t)\underline{u} \tag{3.9}$$

$$\underline{\theta}_s = \underline{\underline{C}}\underline{x}_s \tag{3.10}$$

$$\underline{e} = \underline{\theta}_m - \underline{\theta}_s \tag{3.11}$$

$$\underline{y} = \underline{\underline{D}}\underline{e} \tag{3.12}$$

$$\dot{\underline{\underline{A}}}_s(t) = \underline{\underline{\Phi}}(\underline{y}(\tau),t) \quad , \quad \tau \leq t \tag{3.13}$$

$$\dot{\underline{\underline{B}}}_s(t) = \underline{\underline{\Psi}}(\underline{y}(\tau),t) \quad , \quad \tau \leq t \tag{3.14}$$

where, $\underline{\underline{A}}_m$, $\underline{\underline{B}}_m$ are nxn , and nxm dimensional model state and input matrices respectively. $\underline{\underline{A}}_s$, $\underline{\underline{B}}_s$ are corresponding state and input matrices of the real system with the dimensions nxn, and nxm respectively. $\underline{\underline{C}}$ is rxn output matrix, $\underline{e}$ is the generalized error, and $\underline{\underline{D}}$ is constant matrix defining a linear transformation.

The matrices $\underline{\underline{\Phi}}$ and $\underline{\underline{\Psi}}$ represents the nonlinear dependence on $\underline{y}$ in the interval $\tau \leqslant t$. $\underline{\theta}_m$, $\underline{\theta}_s$, are model and system output vectors, respectively.

The necessary and sufficient conditions for the MRAS described by equations (3.7) -- (3.14) being a hyperstable system are;

a. The transfer matrix

$$\underline{\underline{H}}(s) = \underline{\underline{D}}\,\underline{\underline{C}}(s\underline{\underline{I}}-\underline{\underline{A}}_m)^{-1}$$

must be strictly positive real,

b. The vectors $(\underline{\underline{A}}_m - \underline{\underline{A}}_s)\,\underline{x}_s$, $(\underline{\underline{B}}_m - \underline{\underline{B}}_s)\,\underline{u}$, and $\underline{y}$ have the same dimension.

c. The solutions of the equations (3.13) and (3.14) satisfying the equation (3.5) , have the following form.

$$\underline{\underline{\Phi}}(t) = \left[\alpha_{ij}\,\underline{y}_i^T\,\underline{x}_{sj}\right] = \underline{\underline{J}}\,\underline{y}\,\underline{x}_s \qquad (3.15)$$

$$\underline{\underline{\Psi}}(t) = \left[\beta_{ij}\,\underline{y}_i^T\,\underline{u}_i\right] = \underline{\underline{K}}\,\underline{y}\,\underline{u} \qquad (3.16)$$

where $\underline{\underline{J}}$, $\underline{\underline{K}}$ are positive definite matrices.

For obtaining the asymptotic hyperstability, it is necessary to select the matrix $\underline{\underline{D}}$ properly such that the transfer matrix from $\underline{y}$ to $\underline{e}$ is strictly positive real.

3.2.3 Application to the Mechanical Manipulator

The dynamic equations of a mechanical manipulator is of the form;

$$\underline{\underline{M}}(q)\ddot{\underline{q}} + \underline{V}(q,\dot{q}) = \underline{U} \tag{3.17}$$

The joint displacements (rotational or translational) can be defined as generalized coordinates of the system. If all the joints are revolute, the joint angles are generalized coordinates of the manipulator as mentioned in the second chapter. Then the dynamic equation will be of the form;

$$\underline{\underline{M}}(\theta)\ \ddot{\underline{\theta}} + \underline{V}(\theta,\dot{\theta}) = \underline{U} \tag{3.18}$$

This equation is characterized by a highly nonlinear and coupled structure of the parameters of the manipulator In the ideal case, the manipulator parameters are decoupled and no nonlinearity is concerned. Then, all joint angles, velocities and accelerations are related with each other by ideal integrators. This form of relation can be considered as the reference model for MRAS configuration in the control of manipulator.

Introducing the necessary terms to cancel out the system nonlinearity and achieving the decoupling of the parameters, the equation (3.18) can be expressed in the following form.[25]

$$\underline{\underline{M}}\ddot{\underline{\theta}} = \hat{\underline{\underline{M}}}(t)\underline{z} - \underline{V}(\theta,\dot{\theta}) + \hat{\underline{V}}(\theta,\dot{\theta}) - \underline{\underline{P}}(\theta - \theta_m) -$$

$$\underline{\underline{R}}(\dot{\theta} - \dot{\theta}_m) \tag{3.19}$$

In equation (3.19) the generalized force vector $\underline{U}$ is expressed as;

$$\underline{U} = \hat{\underline{\underline{M}}}(t)z + \hat{\underline{V}}(\theta,\dot{\theta}) - \underline{\underline{P}}(\theta - \theta_m) - \underline{\underline{R}}(\dot{\theta} - \dot{\theta}_m) \qquad (3.20)$$

where $\hat{\underline{\underline{M}}}(t)$, the matrix used to decouple the manipulator parameters,

$\hat{\underline{V}}(\theta,\dot{\theta})$, the vector for cancelling out the nonlinearity, $\underline{V}(\theta,\dot{\theta})$, of the system,

$- \underline{\underline{P}}(\theta - \theta_m) - \underline{\underline{R}}(\dot{\theta} - \dot{\theta}_m)$; the terms for obtain a stable MRAS.

$\underline{\theta}_m$, $\dot{\underline{\theta}}_m$ are the position (displacement) , and velocity vectors of the reference model which is represented by a double integrator .

$\underline{z}$; is the common input vector for both reference model and manipulator which is adjustable model in this case.

The open-loop transfer function between $\underline{z}$ and $\underline{\theta}_m$ is;

$$\frac{\underline{\theta}_m}{\underline{z}} = \frac{1}{S^2} \qquad (3.21)$$

a double integrator.

A vector denoted by $\underline{w}$ is defined as;

$$\underline{w}(t) = (\hat{\underline{\underline{M}}}(t) - \underline{\underline{M}})\underline{z} + (\hat{\underline{V}}(\theta,\dot{\theta}) - \underline{V}(\theta,\dot{\theta})) \qquad (3.22)$$

Then, subtracting the equation (3.19) from (3.21) ,

the following equation is obtained.

$$\frac{d}{dt}\begin{bmatrix} \underline{e} \\ --- \\ \underline{\dot{e}} \end{bmatrix} = \left[\begin{array}{c:c} \underline{\underline{O}} & \underline{\underline{I}} \\ \hdashline -\underline{\underline{M}}^{-1}\underline{\underline{P}} & -\underline{\underline{M}}^{-1}\underline{\underline{R}} \end{array}\right]\begin{bmatrix} \underline{e} \\ --- \\ \underline{\dot{e}} \end{bmatrix} + \begin{bmatrix} \underline{\underline{O}} \\ --- \\ \underline{\underline{M}}^{-1} \end{bmatrix}\underline{W}(t) \qquad (3.23)$$

where $\underline{e}$ and $\underline{\dot{e}}$ are error vectors, such that

$$\underline{e} = \underline{\theta} - \underline{\theta}_m \qquad (3.24)$$

and

$$\underline{\dot{e}} = \underline{\dot{\theta}} - \underline{\dot{\theta}}_m \qquad (3.25)$$

selecting a vector $\underline{y}$ so that;

$$\underline{y}(t) = \underline{\underline{D}}\begin{bmatrix} \underline{e} \\ \underline{\dot{e}} \end{bmatrix} = \begin{bmatrix} \underline{\underline{D}}_1 & \underline{\underline{D}}_2 \end{bmatrix}\begin{bmatrix} \underline{e} \\ \underline{\dot{e}} \end{bmatrix} \qquad (3.26)$$

The transfer function matrix from $\underline{w}(t)$ to $\underline{y}(t)$ is of the form,

$$\underline{\underline{H}}(s) = \begin{bmatrix} \underline{\underline{D}}_1 & s\underline{\underline{D}}_2 \end{bmatrix}\begin{bmatrix} \underline{\underline{M}}s^2 + \underline{\underline{R}}s + \underline{\underline{P}} \end{bmatrix}^{-1} \qquad (3.27)$$

In order that the MRAS satisfies the hyperstability conditions, this transfer function must be positive real. This transfer function represents the linear block of the Popov loop shown in Figure 3.4.

The selection of the matrices $\underline{\underline{P}}$, $\underline{\underline{R}}$, $\underline{\underline{D}}_1$, and $\underline{\underline{D}}_2$ can be accomplished under the above condition such that;

$$\underline{\underline{P}} = \mu_1 \underline{\underline{I}} \tag{3.28}$$

$$\underline{\underline{R}} = \mu_2 \underline{\underline{I}} \tag{3.29}$$

$$\underline{\underline{D}}_1 = \lambda_1 \underline{\underline{I}} \tag{3.30}$$

$$\underline{\underline{D}}_2 = \lambda_2 \underline{\underline{I}} \tag{3.31}$$

where $\underline{\underline{I}}$ is the identity matrix and the scalars μ_1 , μ_2 , λ_1 , λ_2 are all positive constants satisfying the conditions obtained from Kalman-Yakubovitch-Popov Lemma.(23,24)

$$\lambda_2 > \lambda_1 \tag{3.32}$$

$$\lambda_2 \mu_2 \underline{\underline{I}} - \mu_1 \underline{\underline{M}} > 0 \tag{3.33}$$

$$(\lambda_2 \mu_1 + \lambda_1 \mu_2)\underline{\underline{I}} - \lambda_1 \underline{\underline{M}} > 0 \tag{3.34}$$

If the results in equations (3.15) and (3.16) are applied to the MRAS , considering also the structures of the matrices $\hat{\underline{\underline{M}}}$ and $\hat{\underline{\underline{C}}}^k$ s in the dynamics equations of the manipulator, the adaptation mechanism, for 3 degrees of freedom manipulator, is found to be

For $\hat{\underline{\underline{M}}}$ matrix ;

$$\dot{\hat{m}}_{ii}(t) = -\alpha_{ii}(z_i y_i) \quad , \qquad i=1,2,3$$

$$\dot{\hat{m}}_{ij}(t) = -\alpha_{ij}(z_i y_j + z_j y_i) \quad , \qquad i=1,2 \text{ and } j=i+1,3$$

For $\hat{C}^k$ matrices ;

$$\dot{\hat{C}}^1_{12}(t) = -\beta^1_{12}\,(2y_1\dot{\theta}_1\dot{\theta}_2 - y_2\dot{\theta}_1^2)$$

$$\dot{\hat{C}}^1_{13}(t) = -\beta^1_{13}\,(2y_1\dot{\theta}_1\dot{\theta}_3 - y_3\dot{\theta}_1^2)$$

$$\dot{\hat{C}}^1_{22}(t) = -\beta^1_{22}\,(y_1\,\dot{\theta}_2^2\,)$$
$$\dot{\hat{C}}^1_{33}(t) = -\beta^1_{33}\,(2y_1\dot{\theta}_2\dot{\theta}_3 + y_1\dot{\theta}_3^2) \qquad (3.35)$$
$$\dot{\hat{C}}^2_{33}(t) = -\beta^2_{33}\,(2y_2\dot{\theta}_2\dot{\theta}_3 + y_2\dot{\theta}_3^2 - y_3\dot{\theta}_2^2)$$

where the scalars α_{ij} and β_{ij} are positive constants.

This adaptation algorithm represents the nonlinear block of the Popov loop in Figure 3.4

Then the new Popov loop becomes as shown in Figure 3.5

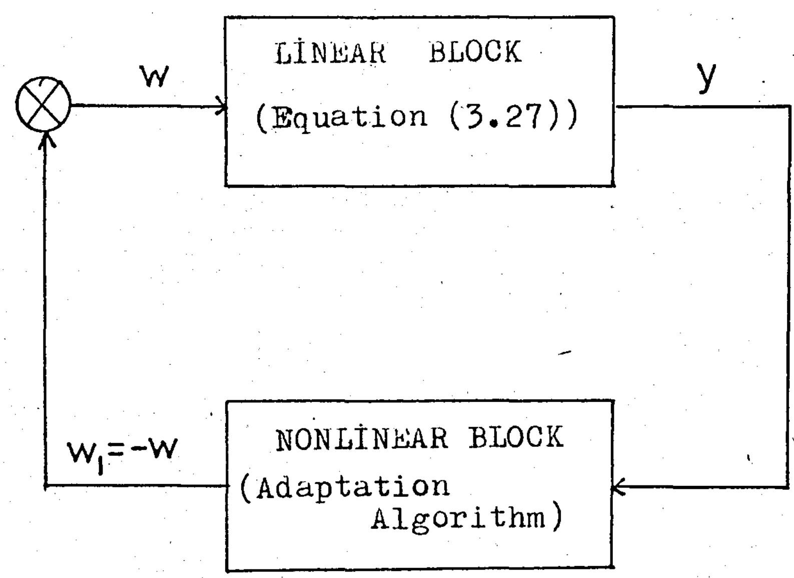


Figure 3.5 The Popov Loop Representing the Present MRAS.

This new Popov Loop satisfies the equation (3.5) for asymptotic hyperstability; such that;

$$\int_0^t \underline{w}_1^T(\tau)\ \underline{y}(\tau)\ d\tau \geqslant -\gamma_0^2 \quad , \quad \forall t \geqslant 0 \qquad (3.36)$$

Then the block diagram of the MRAS used in simulation study is shown in Figure 3.6

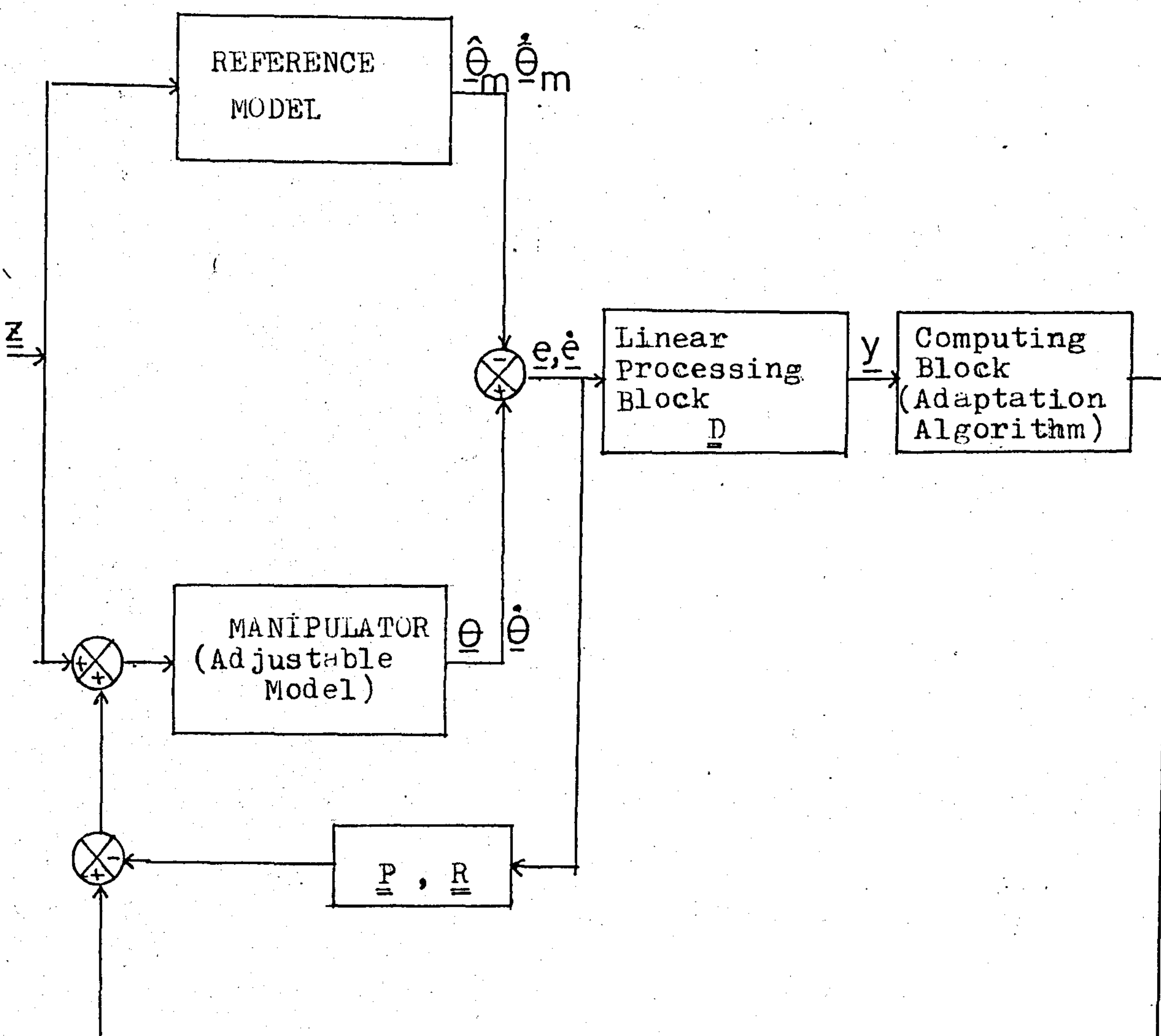


Figure 3.6 The Block diagram of the MRAS used in control of the manipulator.

IV. ALTERNATIVE CONTROL METHODS OF MANIPULATORS

4.1 INTRODUCTION

Due to the higher requirements on speed and accuracy, various manipulator control methods have been tried to be developed. As it is seen in the earlier Chapters, the basic problem that has to be overcome in manipulator control is the highly nonlinear dynamical equations and couplings that govern the motion of a manipulator. In this chapter, other than the presented Model Reference Adaptive System method, some control techniques are examined briefly.

Firstly , an approximately optimal control method is presented. In this method, an approximate optimal solution is tried to be found for manipulators with four or less degrees of freedom.

A table-look up control method is examined in which there is no need to solve the dynamic equations of manipulators, analytically.

In the Resolved Motion Rate Control (RMRC), joint velocities are tried to be controlled which are obtained from end-effector velocities, positions and/or forces and moments exerted by the end-effector.

4.2 AN APPROXIMATION THEORY OF OPTIMAL CONTROL FOR MECHANICAL MANIPULATORS

When the nonlinear systems are concerned the feedback solution of optimal control problem is of little application due to the difficulties in realization. The same problem is also considered for robotic manipulators because of the fact that their dynamic equations are in a highly nonlinear structure.

Recently, different ways of solutions to this problem have been proposed to obtain approximately optimal solutions or to get higher-order feedback terms for improving the performance of the system. [26,27]

In this section, another way of solution will be presented. In this method a recursive algorithm is applied for obtaining control laws converging to the optimal, that is, improving the performance of the system. [10,11]

An admissible control is selected and compared with other controls in terms of performance index. In the second stage another control is obtained which is used to reach a more improved performance. These steps are repeated until an allowable approximation to the optimal control is reached.

The theory of this approach is similar to the one used for obtaining the Hamilton-Jacobi-Bellman equation to construct the rules for the solution of the feedback control problem. The foundation of the method is based upon the definition of a value function which is used for updating the control law while keeping the system stable.

Consider a system in the form;

$$\dot{\underline{x}} = a(\underline{x}) + \underline{\underline{B}}\,\underline{u} \quad , \underline{x}(t_o) = \underline{x}_o \tag{4.1}$$

where $\underline{x}$ is nxl state vector, $a(\underline{x})$ a function of states which may consists of nonlinear terms, $\underline{\underline{B}}$ is nxm input matrix, and $\underline{u}$ is desired control law with the dimension of mxl.

The process is controlled to minimize (or maximize) the performance measure,

$$J(\underline{x}(t),t,\underline{u}(t)) = h(\underline{x}(t_f),t_f) + \int_{t_o}^{t_f}\left[g(x(\tau)) + \|u\|^2\right]d\tau \tag{4.2}$$

where $h(x(t_f),t_f) \geqslant 0$

and
$g(\underline{x}(t)) \geqslant 0$

A necessary condition for optimality is the following;

$$\min_{u} H(\underline{x},\underline{u},\nabla v,t) = 0 \tag{4.3}$$

where $V(x,t,u)$ is the associated value function, and

$V \triangleq \frac{\partial V}{\partial x}$, the gradient of $V(x)$.

A pre-Hamiltonian function is defined as;

$$H\left[\underline{x},\nabla V^*(\underline{x}),\underline{u},t\right] = \nabla V^{*T}(x)\,\dot{\underline{x}} + L(\underline{x},\underline{u}) \tag{4.4}$$

Where $L(x,u)$ is the integrand of the performance measure, and V^* is the optimal value function, then;

$$H\left[\underline{x},\ V(x),\underline{u},t\right] = \nabla V^{*T}(x)\ a(x) + \nabla V^{*T}(x)\ \underline{\underline{B}}\ \underline{u} + g(\underline{x}) + \|u\|^2 \tag{4.5}$$

The optimal control law is obtained by minimizing the pre-Hamiltonian function with respect to $\underline{u}$, and obtaining the H-J-B equation;

$$g(x) + \nabla V^{*T} a(x) - \frac{1}{4}(\nabla V^{*T} \underline{\underline{B}})^2 = 0 \qquad (4.6)$$

To get the optimal control law for this system, the partial differential equation (4.6) should be solved for $V^*(x)$, and used in (4.3) . In most cases, however, the solution of this type of equation is very difficult. Additionally, even the solution of this equation could be found, the realization of the optimal control law would be very difficult to implement.

For avoiding these difficulties, instead of trying to find an exact solution to the optimal control problem a formulation for obtaining an approximate solution will be presented hereafter.

Instead of minimizing the pre-Hamiltonian function, solving the necessity the H-J-B equation (4.6) for $V^*(x)$ to get the optimal control law, assume an arbitrary, positive definite and continuously differentiable value function $V(x)$. Substituting this value function, in (4.4) for a given control law, it is obtained that,

$$\nabla V^T(x)\underline{\underline{B}}\underline{u} + \nabla V^T(x)\, a(x) + g(x) + \|u\|^2 = 0 \qquad (4.7)$$

Additionally, the value function, for the given control law satisfies the properties;

$$V(x_o, t_o, u) = J(x_o, t_o, u) \qquad (4.8)$$

and

$$V(x(t_f),t_f,u) = h(x(t_f),t_f) \tag{4.9}$$

Using the necessary and sufficient conditions for optimality, it is found that for the optimal control $\underline{u}^*$ and V (x); (10)

$$V(x,t,u) \geqslant V^*(x,t,u\ ) > 0 \qquad , \ u \neq u^* \tag{4.10}$$

Assuming for a given control u_1, the value function $V_1(x,t,u)$ satisfying (4.7), (4.8), (4.9), system (4.1) has bounded trajectories in $[t_o,t_f]$ and as $t_f \rightarrow \infty$ $V_1(0,t,u) = 0$ for all t in the interval, $[t_o,t_f]$, so the origin, $x(\infty) = 0$, is a stable equilibrium point. The above property is also verified by the stability in the sense of second method of Lyapunov, such that;

as $t_f \rightarrow \infty$, if

$$V(\underline{x},t) > 0 \qquad , \quad \forall \underline{x} \neq 0$$
$$V(\underline{0},t) = 0 \qquad , \quad \forall t$$

and (4.11)

$$\dot{V}(\underline{x},t) < 0 \qquad , \quad \forall \underline{x} \neq 0$$

$$\dot{V}(0,t) = 0 \qquad , \quad \forall t$$

then the origin of the system of equation (4.1) is uniformly asymtotically stable.

Using the equation (4.5) for u_1 and another control law u_2, and minimizing with respect to u 's, where

$V_1(x)$ and $V_2(x)$ corresponding arbitrary value functions satisfying the properties mentioned before, for system (4.1) then;

$$H_{min1} = g(x) + \nabla V_1^T(x)\, a(x) - \frac{1}{4}(\nabla V_1^T \underline{\underline{B}})^2$$

(4.12)

$$H_{min2} = g(x) + \nabla V_2^T(x)\, a(x) - \frac{1}{4}(\nabla V_2^T \underline{\underline{B}})^2$$

The control law $u_2(x)$ may be a better sub-optimal control than $u_1(x)$ if the corresponding $H_{min}(x)$ satisfy the condition;

$$\frac{\partial V_2}{\partial t} + H_{min2} \geqslant \frac{\partial V_1}{\partial t} + H_{min1} \qquad (4.13)$$

then the inequality

$$V_1 \geqslant V_2 \geqslant V^* \qquad (4.14)$$

should be valid for all real x.[11] In this inequality $V^*(x)$ is the value function for the optimal control $u^*(x)$.

If the value functions $V_1(x)$ and $V_2(x)$ are not explicitely dependent on time, $h(x(t_f),t_f)=0$ and the integrand of the performance index $L(x,u) \neq 0$, then the equation (4.13) reduces to the form;

$$H_{min2} \geqslant H_{min1} \qquad (4.15)$$

and the inequality $V_1 \geqslant V_2$ is still valid.[11] From the equations (4.14) and (4.13)[10] it is concluded that;

$$(u_1 + \frac{1}{2} \underline{\underline{B}}^T \nabla V_1) \geqslant (u_2 + \frac{1}{2} \underline{\underline{B}}^T \nabla V_2) \qquad (4.16)$$

Under the above properties, this result can be generalized and, $V^*(x)$ constitutes a lower bound for value functions of approximate controls, i.e.;

$$(u_1 + \frac{1}{2} \underline{\underline{B}}^T \nabla V_1) \geqslant (u_2 + \frac{1}{2} \underline{\underline{B}}^T \nabla V_2) \geqslant \ldots . (u^* + \frac{1}{2} \underline{\underline{B}}^T \nabla V^*) = 0$$

since

$$u^* = -\frac{1}{2} \underline{\underline{B}}^T \nabla V^* \qquad (4.17)$$

<u>Theorem 1:</u> If control laws and associated value functions are obtained according to above rules, from the minimization of the Hamiltonian containing $V_i(x)$, value function for $u_{i-1}(x)$, in other words if the controls are selected according to this rule, the controls converges to optimal monotonically, i.e;

$$u_i(x) = -\frac{1}{2} \underline{\underline{B}}^T \nabla V_{i-1} \qquad (4.18)$$

then

$$V_{i-1} \geqslant V_i$$

The proof of this theorem is given in [10].

<u>Theorem 2:</u> For an arbitrary control law u_a , and $f(x,t)$, $|f(x,t)| < \infty$, if a value function V_a satisfying the properties given above, from equation (4.7) it is written that;

$$\nabla V_a^T \underline{\underline{B}}\, \underline{u}_a + \nabla V_a^T a(x) + g(x) + \|u_a\|^2 + \frac{\partial V_a}{\partial t} = \Delta V_a$$

then if

$$\Delta V_a \leqslant f(x) < 0 \qquad (4.19)$$

and

$$V_a(x(t_f), t_f, u_a) \geqslant h(x(t_f), t_f)$$

$V_a(x_o, t_o, u_o)$ is upper bound of $J(u_o, x_o, t_o)$
and if,

$$\Delta V_a \geqslant f(x,t) > 0 \quad \text{and} \quad V_a(x(t_f), t_f, u_a) \leqslant h(x(t_f), t_f) \qquad (4.20)$$

then, $V_a(x_o, t_o, u_a)$ is lower bound of $J(u_a, x_o, t_o)$.

The above theorem is proved in (10).
The difference $V_i - V_a = \Delta V_a$ can be used as a measure whether u_i is an acceptable approximation to the optimal control.

4.2.1 <u>Application of the Theory to the Linear Systems</u>

Consider the linear system;

$$\dot{\underline{x}} = \underline{\underline{A}}\, \underline{x} + \underline{\underline{B}}\, \underline{u} \qquad (4.21)$$

$$\underline{y} = \underline{\underline{C}}\, \underline{x}$$

subject to the performance index;

$$J(u,x,t) = \int_{t_o}^{t_f} (\underline{x}^T \underline{\underline{Q}}\, \underline{x} + \|\underline{u}\|^2)\, dt \qquad (4.22)$$

where $\underline{\underline{Q}}$ is a symmetric and positive definite matrix.

The application of the above method to the linear systems is essentially based upon the selection of the value function.

Selecting a value function in the following form;

$$V_i(x) = \underline{x}^T \underline{\underline{P}}_i\, \underline{x} \qquad (4.23)$$

where P_i is a symmetric and positive definite matrix.
The pre-Hamiltonian function;

$$H(x, V(x),u,t) = \nabla V^T(x)\underline{\underline{A}}\underline{x} + \nabla V^T(x)\underline{\underline{B}}\underline{u} + \underline{x}^T \underline{\underline{Q}}\, \underline{x} + \|u\|^2 \qquad (4.24)$$

and applying the equation;

$$u_i(x) = -\frac{1}{2} B^T \nabla V_{i-1}(x)$$

then, a linear matrix equation is obtained in the following form.

$$P_iA - \frac{1}{2} P_i BB^T P_{i-1} + \frac{1}{4} P_{i-1} BB^T P_{i-1} + Q = 0 \qquad (4.25)$$

$\underline{\underline{P}}_i$ converges to $\underline{\underline{P}}^*$ in time, so $V_i(x)$ converges $V^*(x)$, the optimal value function.

In some cases the selection of value functions satisfying equations (4.8) and (4.9) is very difficult .

In these cases, the upper and lower bound conditions as in Theorem 2 may be useful. For a given control u_1, a value function V_1 is found as a lower bound, and as the next step, for another control, u_2, a value function is tried to be found as an upper bound. This procedure is repeated until an acceptable approximation is reached.

4.2.2 Application to the Mechanical Manipulator

As in the most cases of nonlinear systems, the optimal solution is not possible to be found or realization of it is very difficult for mechanical manipulators.

The dynamic equations of motion of a manipulator was of the following form;

$$\underline{\underline{M}}\,\ddot{\underline{\theta}} + \underline{V}(\theta,\dot{\theta}) = \underline{U} \qquad (4.26)$$

where the matrices $\underline{\underline{M}}$ and $\underline{V}$ are nonlinear functions of joint angles $\underline{\theta} = (\theta_1,\theta_2,\theta_3,\ldots\theta_n)^T$, and their velocities, and $U = (U_1,U_2,U_3,\ldots U_n)^T$ is the input torque, and vector of other forces.

From equation (4.26);

$$\ddot{\underline{\theta}} = \underline{\underline{M}}^{-1}(\underline{U} - \underline{V}) = \underline{\underline{M}}^{-1}\underline{U} - \underline{\underline{M}}^{-1}\underline{V} \qquad (4.27)$$

Defining the matrices,

$$\underline{P} = \underline{\underline{M}}^{-1}\underline{U} \quad \text{and} \quad \underline{G} = -\underline{\underline{M}}^{-1}\underline{V}$$

then

$$\ddot{\underline{\underline{\theta}}} = \underline{G} + \underline{P} \qquad (4.28)$$

This model of the arm is linear in P, which is an input vector obtained from the actual input u by a nonlinear transformation.

Rewriting the state equation for a 3 degrees of freedom manipulator.

$$\dot{\underline{x}}(t) = \underline{\underline{A}}\ \underline{x}(t) + \underline{\underline{B}}(\underline{G} + \underline{P}) \qquad (4.29)$$

where

$$\underline{\underline{A}} = \begin{bmatrix} \underline{\underline{0}} & \underline{\underline{I}} \\ \underline{\underline{0}} & \underline{\underline{0}} \end{bmatrix} \quad , \quad \underline{\underline{B}}^T = \begin{bmatrix} \underline{\underline{0}} \\ \underline{\underline{I}} \end{bmatrix}$$

and state vector

$$\underline{x}(t) = (\theta_1, \theta_2, \theta_3, \dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3)$$

The dynamic equations derived in the second chapter is approximate since it is based on several simplifying assumptions. Additionally, a performance index compatible with the desired response is difficult to construct, because of the fact that the system has coupling effects between each subsystems besides the inherent nonlinearities.

Therefore, assuming the system is in decoupled form, then a performance index for the proper kinematic configuration can be defined as;

$$J(u,x,t) = \int_0^\infty \left[\| x(t) - x_d(t) \|^2_{\underline{\underline{Q}}} + \| u \|^2 \right] dt \qquad (4.30)$$

where $\underline{x}_d = (\theta_{1d}, \theta_{2d}, \theta_{3d}, 0, 0, 0)$, desired final coordinates, and

$$\underline{\underline{Q}} = \begin{bmatrix} \underline{\underline{Q}}_1 & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{Q}}_2 \end{bmatrix}$$

is diagonal matrix related to the damping and the speed of the system.

The matrices Q_1 and Q_2 are adjusted for the purpose of improving the overall dynamic performance of the system, the values of which affect the relative speed of the joints and are selected in order to minimize the overall energy needed to perform a motion.

According to the method given in this chapter, a feedback control is selected at this stage. A control law compensating the nonlinearities of the system structure may be of the form;

$$\underline{U}(x) = -M\left[\underline{G}(x) + \underline{\underline{S}}\,\underline{x}\right] \tag{4.31}$$

where $\underline{\underline{S}} > 0$, and symmetric matrix,

Substituting this control law into system equation (4.29)

$$\begin{aligned} \dot{\underline{x}}(t) &= \underline{\underline{A}}\,\underline{x}(t) + \underline{\underline{B}}(\underline{G} + \underline{P}) \\ \dot{\underline{x}}(t) &= \underline{\underline{A}}\,\underline{x}(t) + \underline{\underline{B}}(-\underline{\underline{S}}\,\underline{x}) \end{aligned} \tag{4.32}$$

As can be seen from the equation (4.31), the control law consists of two parts, one is linear and the other nonlinear part used for compensating the nonlinear term of the system. Because of the fact that upon convergence of the values of G and S, to their nominal values, the compensating inputs $\underline{U}_i$'s cancel out the nonlinearities of the system. Then the system will be reduced to the form of ideally double integrators with quadratic performance index, (4.24) i.e. satisfying the controllability condition.

4.3 CEREBELLAR MODEL ARTICULATION CONTROLLER (CMAC)

The CMAC is a table lookup manipulator control method in which control functions for each degrees of freedom can be accessed from a table, but not solving the dynamic equations of a manipulator, analitically. In other words mathematical formulation of a manipulator is not necessary for this method of control. The main drawback of this method is the necessary size of memory in which the control functions are stored. The CMAC algorithm, however, introduces some remedies to reduce the memory size to a realizable amount.[14,15]

4.3.1 The Cerebellum

The cerebellum is an organ of the human body which takes care of the coordination of movements. A neurophysiological study on the cerebellum reveals that its operation is based upon the inputs obtained from sensory organs and feedback signals coming from the muscles, skin and joints of related moving part of the body. All the inputs form an address whose content is used for obtaining the output signals driving the joint actuators. The functional operation of cerebellum is the origin of CMAC algorithm which is essentially an adaptive control method.

4.3.2 An Adaptive Linear Unit and Perceptron

To understand the method, a simple adaptive system is introduced and shown schematically in Figure 4.1.

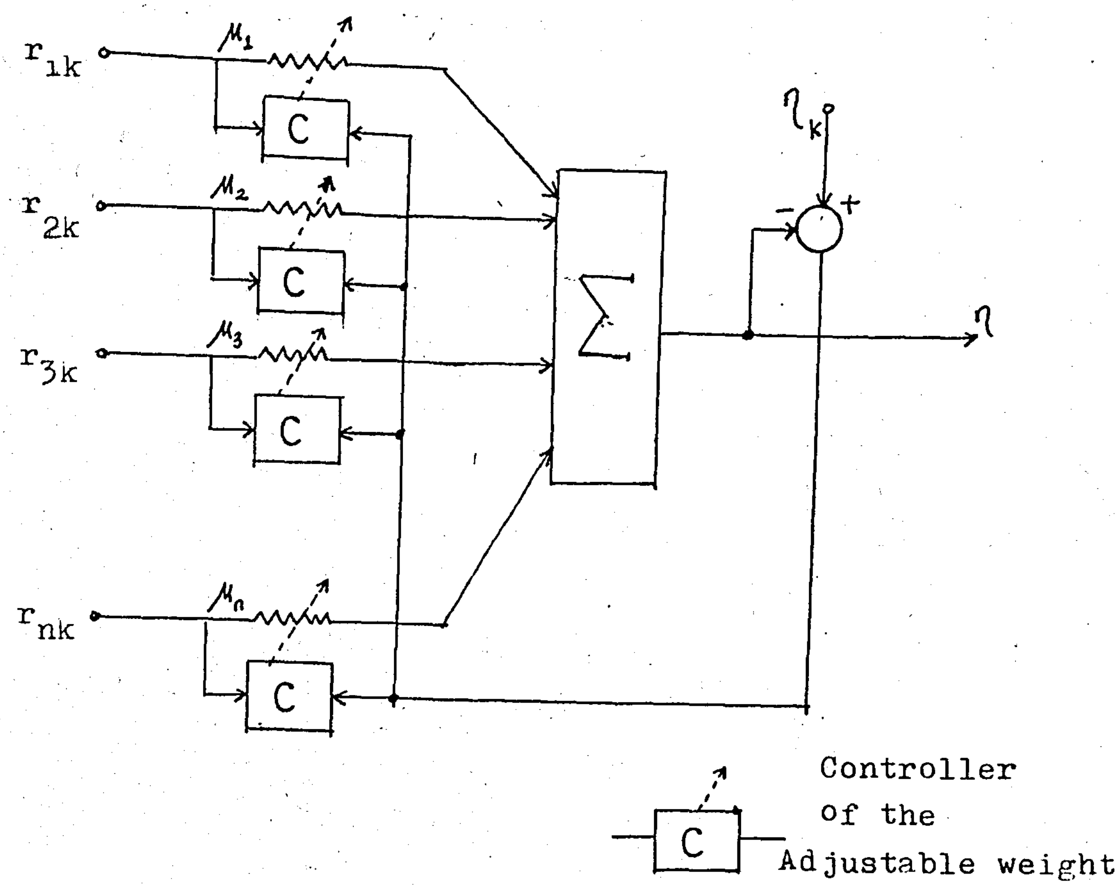


Figure 4.1 An Adaptive Linear Unit

The device in Figure 4. consists of a set of variable resistors connected to a summing circuit which sums up the currents caused by the applied input voltages. The number of parallel outputs can be increased so that the system canbe realizable by multiple units similar to the given configuration.(40)

The input conductances are represented by μ_i , the input voltages r_i , and the output signal by η , such that;

$$\eta = \sum_{i=1}^{n} \mu_i r_i \tag{4.33}$$

The aim of this unit is to obtain a given value η_k when the inputs r_i , are applied to the system. For this purpose, the system operates in such a manner that, it adjusts the weights to obtain the value η_k at the output.

This is an adaptive operation so that a mechanism used for adjusting μ_i s can be found to reach the desired value. This is also a problem of associative mapping such that if a proper mapping is not found only an approximate value of η_k is obtained at the output of the system.

Sometimes, this type of adaptive systems may include nonlinear operations in the form of tresnold-logic unit. The device called Perceptron combines several threshold-logic units into a single system which is used to realize an adaptive-pattern classifier.

A typical perceptron configuration is shown in Figure 4.2

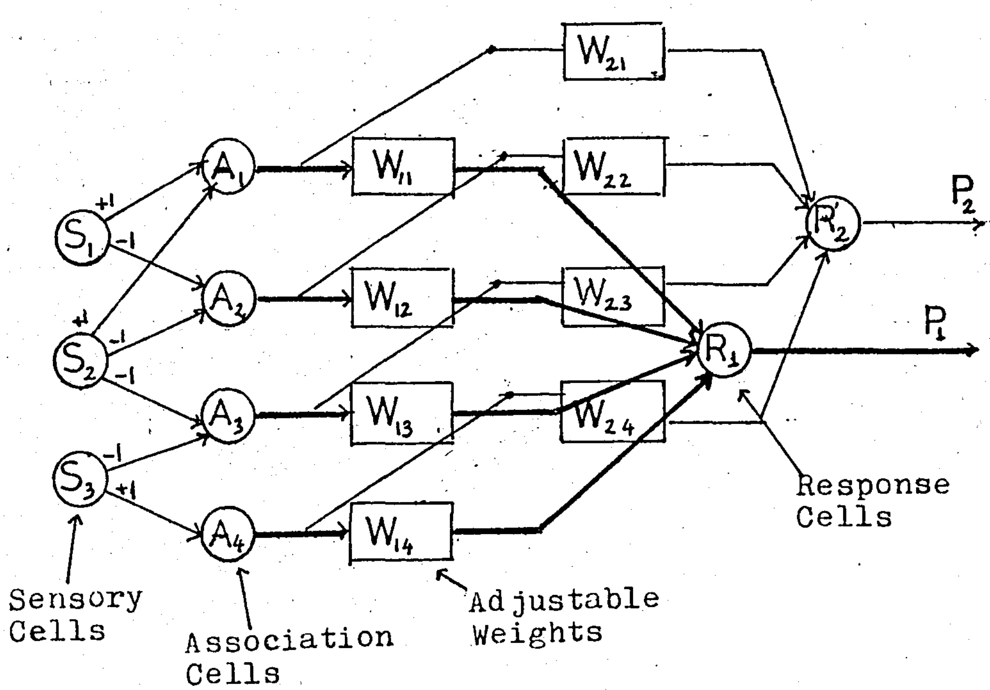


Figure 4.2 A typical Perceptron

In the figure, the sensory cells, S_i , receive the inputs either 1 or 0 . These inputs are transmitted to the association cells , A_i , as 1 or -1 , If the input to an association cell exceeds 0 , it is activated and, the signal is transferred to the response cells, R_i , through adjustable weights .

If the value of the input signal exceeds a threshold value, the response cell is activated and outputs a 1 , otherwise it outputs a 0. The operation of Perceptron is essentially a mapping of the form;

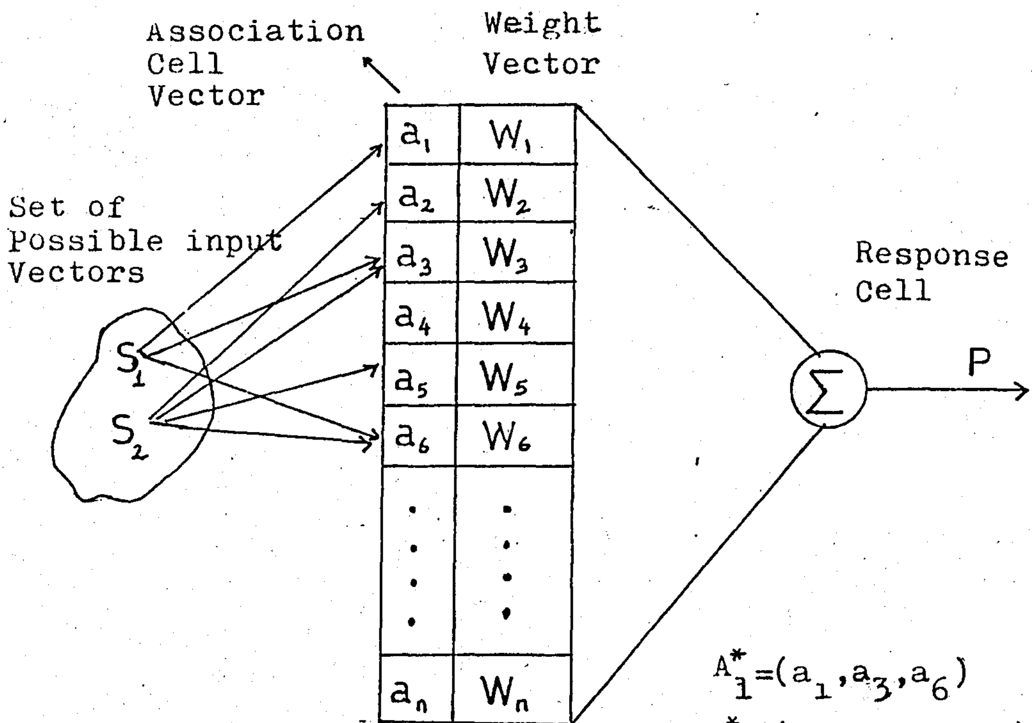
$$S \longrightarrow A$$

and

$$A \longrightarrow P$$

The only way to change A to P mapping is to adjust the weights. The input vectors S_i can be interpreted as addresses whose contents are corresponding output vectors P_i s. To produce independent outputs for each input vector, it is necessary to have a large number of association cells which may be in unrealizable size. Therefore, some elements of association cell vector can be used in common by some input vectors.

As shown in Figure 4.3 an intersection of active association cells which are nonzero elements of association cell vector and denoted by A^* , may be related with some input patterns. This intersection can be utilized when the similar outputs are desired for different inputs. However, it creates problems whenever dissimilar outputs are necessary.



$A_1^* = (a_1, a_3, a_6)$

$A_2^* = (a_2, a_3, a_5, a_6)$

$A_1^* \cap A_2^* = (a_3, a_6)$

Figure 4.3 The perceptron when the intersection of association cells is concerned.

Since the joint actuator signals of manipulators are in a continuous structure, the intersections between the inputs which are close to each other may be utilized for generalization, but for the inputs far apart from each other, the overlap should not be encountered to prevent the interference between outputs. Therefore, a properly designed mapping is necessary for satisfying the above conditions.

In CMAC algorithm, the mapping from S to A vector is accomplished in two stages; S to M and M to A, where M is a memory organization routine. This mapping is done for each input vector. To achieve this mapping using a physically realizable memory size , a memory storage technique called hash-coding (hash-addressing or scattering-addressing) algorithm is used. It is a purely programming technique using conventional computer memories in which the address of a storage location for data is a function of the data itself and it is determined by some mapping algorithm applied to the data contents. In other words, it takes the content of an address in a large memory and uses it as an argument in searching the data in a smaller memory.

A problem called collision may occur in hash-coding algorithm as in the case of all content-addressing methods. If S_1 and S_2 are two different input vectors and $f(S_1)$ and $f(S_2)$ are corresponding computed addresses, it is desired that $f(S_1) \neq f(S_2)$. This situation is of high probability so that if the function f scatters computed addresses randomly. However, there is a little probability that $f(S_1) = f(S_2)$, which is called collision. There are some ways for handling the problem of collision.(40)

4.3.3 Computation of Actuator Drive Signals

As mentioned above, the outputs of the response cells which sum the corresponding weights, are actuator drive signals of a manipulator. A typical relationship between input and output of a CMAC controller is that;

$$P = g(S) \tag{4.34}$$

There are different functions for each joint of the manipulator so that a coordination between joints is achieved during a movement. These functions are used in learning (data storage) stage of the method. At this stage, essentially, the weights are adjusted to obtain desired actuator drive signals. A schematic configuration of a CMAC system for a single joint is shown in Figure 4.4.

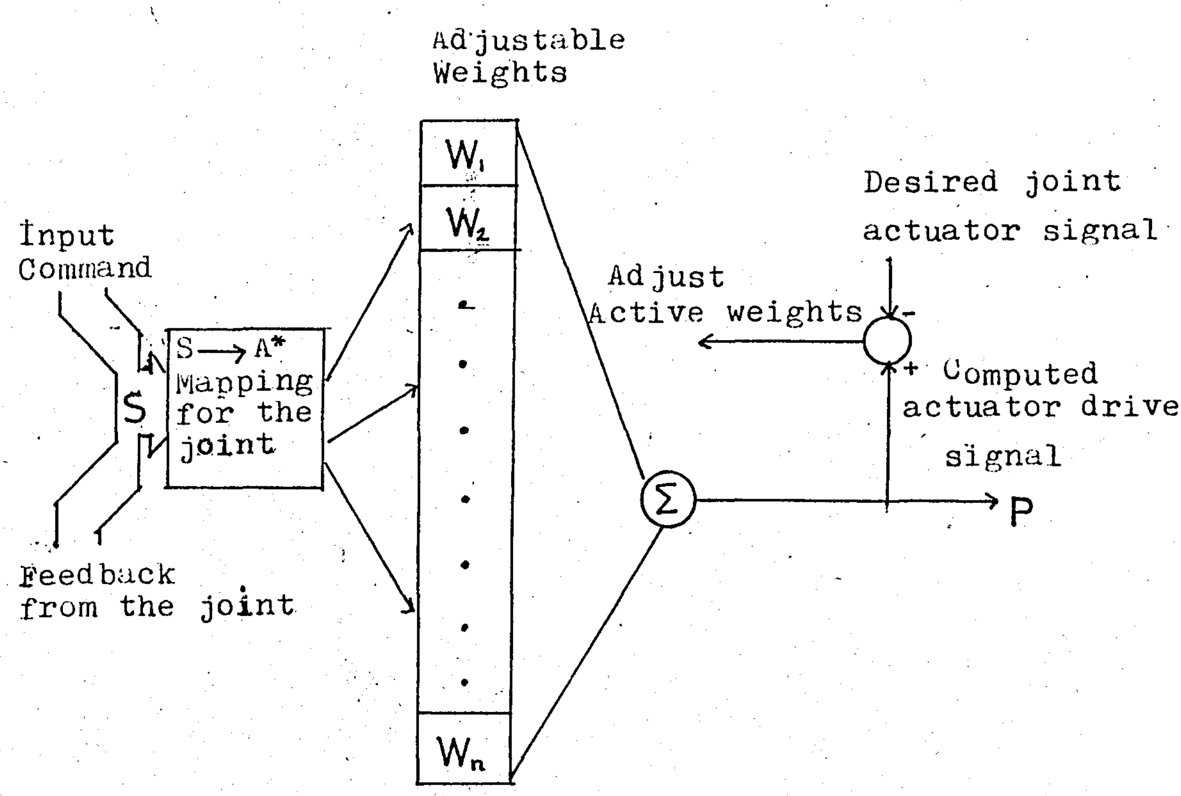


Figure 4.4 A basic configuration of CMAC system for a single joint.

This scheme is similar to the adaptive linear unit shown in Figure 4.1. The difference between the computed and desired drive signals, is used to adjust the weights such that an output is obtained equal to the desired output.

In summary, the CMAC algorithm produces continuous control signals for manipulator and the complicated dynamic equations need not be solved analytically. However, more effective memory storage and addressing techniques should be developed for solving memory size problem and improving the accuracy.

4.4 RESOLVED MOTION RATE CONTROL (RMRC)

Resolved Motion Rate control is one of the most common methods in the control of mechanical manipulators. This method deals directly with the position and orientation of the end-effector of the manipulator. The difference of this method from the others is that the velocities and displacements (sometimes accelerations) are specified and all the control is accomplished at the end-effector level.

Since, the six degrees of freedom are required to specify the position and orientation of an object in space a manipulator has also six degrees of freedom, three of them for position and the others are for orientation of the end-effector. Nevertheless, in some cases more degrees of freedom may be required to accomplish some tasks with smaller angle displacements and to reach more locations. This may increase the choice of various manipulator configurations.[6,7]

The joint displacements, rotational or translational, can be defined as $\underline{\theta}$ which is a mxl vector. The coordinates of position and orientation of the end-effector can be denoted as $\underline{x}$, a vector with dimension nxl. If, $n = m = 6$, the manipulator is called nonredundant, if $m > 6$, it is called a redundant manipulator. The relations between the driving motor positions and joint displacements are assumed to be linear.

The relationship between $\underline{x}$ and $\underline{\theta}$ is as follows;

$$\underline{x} = f(\underline{\theta}) \tag{4.35}$$

where sometimes $\underline{x}$ is called "world-coordinate vector", and $\underline{\theta}$ is "joint-coordinate vector".

The equation (4.35) is of a highly-nonlinear structure, depending on the task to be done, and the configuration of the manipulator.

Differentiating the equation (4.35) with respect to time it is obtained that;

$$\dot{\underline{x}} = \underline{\underline{J}}(\theta)\,\dot{\underline{\theta}} \tag{4.36}$$

where $J(\theta)$ is the Jacobian matrix of the system with the elements

$$J_{ij} = \frac{\partial f_i}{\partial \theta_j} \tag{4.37}$$

If $m = n$, and $J(\theta)$ is nonsingular, then the rates of $\underline{\theta}$ can be found as,

$$\dot{\underline{\theta}} = \underline{\underline{J}}^{-1}(\theta)\,\dot{\underline{x}} \tag{4.38}$$

Equation (4.38) means that, given the desired rate of the position and orientation of the end-effector, the rates of joint displacements can be obtained directly. The computations, however, should be in a coordinative level in terms of individual joint rates in order to produce a particular motion of the end-effector. These joint rates, are then converted to voltages and used to drive the the joint actuators.

4.4.1 Calculation of Matrix $J(\theta)$

Before beginning the calculation of $J(\theta)$, it is necessary to define the position and orientation vectors of the end-effector. Making use of the homogeneous transformations mentioned in Chapter II, the transformation matrix between the fixed-base frame and the end-effector can be expressed as ;

$$T = \left[\begin{array}{ccc|c} \underline{n} & \underline{s} & \underline{a} & \underline{p} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|c} & \underline{\underline{R}} & & \underline{p} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \tag{4.39}$$

Sometimes, this matrix is called "hand-matrix" and represents the actual position and orientation of the end-effector in terms of reference system.

Defining a linear velocity vector as ;

$$v_{i+1} = \sum_{j=1}^{i+1} c_j \dot{\theta}_j \tag{4.40}$$

and an angular velocity vector;

$$w_{i+1} = \sum_{j=1}^{i+1} d_j \dot{\theta}_j \tag{4.41}$$

where

$$c_j = \begin{cases} \underline{u}_{j-1} x (\underline{p}_{i+1} - \underline{p}_{j-1}) & \text{if joint j is rotational} \\ \underline{u}_{j-1} & \text{if joint j is translational} \end{cases} \quad (4.42)$$

and

$$d_j = \begin{cases} u_{j-1} & \text{if joint j is rotational} \\ 0 & \text{if joint j is translational} \end{cases} \quad (4.43)$$

In equation (4.41) the angular velocity in the direction of u_{j-1} with a magnitude of $\dot{\theta}_j$. For the six degrees of freedom manipulator, replacing the subscript i+1 by 6 in equations (4.40), (4.41), (4.42) and (4.43), the following equations are obtained ;

$$\underline{v} = \underline{v}_6 = \sum_{j=1}^{6} c_j \dot{\theta}_j \quad (4.44)$$

$$\underline{w} = \underline{w}_6 = \sum_{j=1}^{6} d_j \dot{\theta}_j \quad (4.45)$$

and

$$c_j = \begin{cases} \underline{u}_{j-1} x (\underline{p} - \underline{p}_{j-1}) & \text{if joint j is rotational} \\ \underline{u}_{j-1} & \text{if joint j is tranlational} \end{cases} \quad (4.46)$$

where $\underline{P} = \underline{P}_6$, the position vector of the end-effector relative to the reference system.

Combining the vectors $\underline{v}$ and $\underline{w}$ into a six dimensional vector $\dot{\underline{x}}$, the equation (4.36) can be written as

$$\dot{\underline{x}} = \begin{bmatrix} \underline{v} \\ \\ \underline{w} \end{bmatrix} = \underline{\underline{J}}(\theta)\ \dot{\underline{\theta}} \qquad (4.47)$$

and the i th column of the 6x6 J(θ) matrix will be in the following form.

$$J_i(\theta) = \begin{bmatrix} \underline{u}_{i-1} x (\underline{p} - \underline{p}_{i-1}) \\ \\ \underline{u}_{i-1} \end{bmatrix} \quad \text{if joint } i \text{ is rotational} \qquad (4.48)$$

and

$$J_i(\theta) = \begin{bmatrix} \underline{u}_{i-1} \\ \\ 0 \end{bmatrix} \quad \text{if joint } i \text{ is translational}$$

since u_i s depend on $\underline{\theta}$, $J(\underline{\theta})$ is also function of $\underline{\theta}$. To obtain motion in terms of hand coordinates, this J(θ) matrix should be premultiplied by the rotation part, $\underline{\underline{R}}$, of the transformation matrix .

4.4.2 Application with Other Criteria

In addition to the above procedure, it may be desired that the manipulator should satisfy some criterion while moving in space.

In the case of position control process, one may want the manipulator to reach a final position with a specified final time, t_f, while minimizing a performance index of the form;

$$I = \int_{0}^{t_f} \frac{1}{2} \dot{\underline{\theta}}^T \underline{\underline{M}} \, \dot{\underline{\theta}} \, dt \tag{4.49}$$

where M is a positive definite weighting matrix. The cost in equation (4.49) is the kinetic energy of the system. Using the relation

$$\dot{\underline{x}} = \underline{\underline{J}}(\theta) \, \dot{\underline{\theta}}$$

With the above criteria and utilizing Lagrange multipliers, for the optimal $\dot{\underline{\theta}}$ the following expression is obtained,

$$\dot{\underline{\theta}} = \underline{\underline{M}}^{-1} \underline{\underline{J}}^T(\theta) \left[\underline{\underline{J}}(\theta) \underline{\underline{M}}^{-1} \underline{\underline{J}}(\theta) \right]^{-1} \dot{\underline{x}} \tag{4.50}$$

If, however, the matrix $\underline{W} = \underline{\underline{J}}(\theta) \underline{\underline{M}}^{-1} \underline{\underline{J}}(\theta)^T$ is not invertible over the trajectory of some states then, it is not possible to obtain optimal feedback solution to satisfy the given conditions.

V. SIMULATION RESULTS

A computer simulation study is performed to investigate the performance of the MRAS manipulator control system. A state vector feedback with an integral action is added to complete the control system as shown in Figure 5.1. The feedback matrices are all in diagonal form.

The complete set of nonlinear, coupled second order differential equation obtained in Chapter II are used to model the manipulator dynamics. The numerical values in these equations are based on the UCLA experimental arm. (Table 5.1). Its mechanical configuration is given in Chapter II. The parameters used in the present MRAS are selected according to the analysis performed in Chapter III. In the simulation study the state-transition matrix method of integ ration is used, and the computer program written in Fortran IV language is given in appendix A . The flowchart of the algorithm is shown in Figure 5.2.

For different motions the joint angle trajectories are plotted using various payloads. It is observed that these trajectories are closely match to the outputs of the reference model.

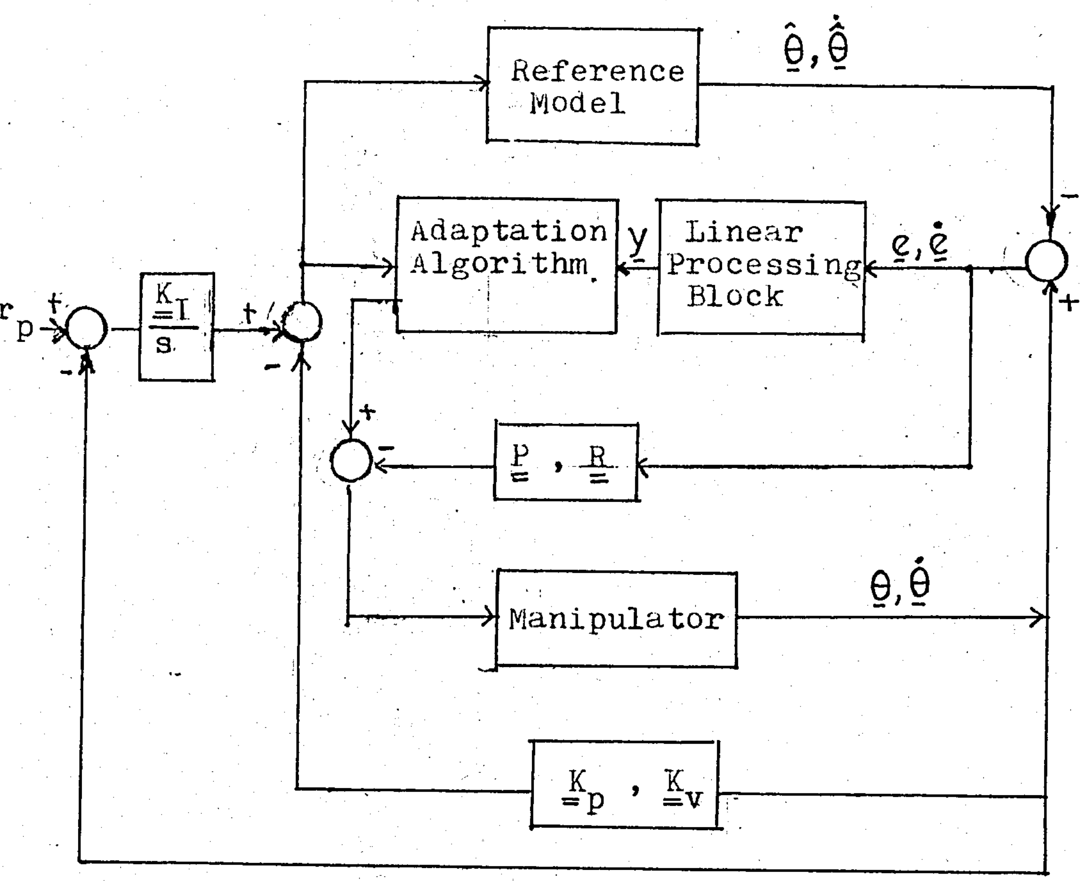


Figure 5.1 Block diagram of the simulated manipulator control system.

	Lenght(m)	Mass(kg.)	Inertia Dyadics(kg.m^2)
Forearm	$\ell_1^* = 0.36$ $\ell_1 = 0.89$	23.3	$\begin{bmatrix} 3.348 & 0.05 & 0.5 \\ 0.05 & 0.1 & 0.05 \\ 0.5 & 0.05 & 3.348 \end{bmatrix}$
Upper arm	$\ell_2 = 0.914$	6.71	$\begin{bmatrix} 0.924 & 0.02 & 0.1 \\ 0.02 & 0.1 & 0.02 \\ 0.1 & 0.02 & 0.924 \end{bmatrix}$

Table 5.1 Manipulator parameters used in simulation study

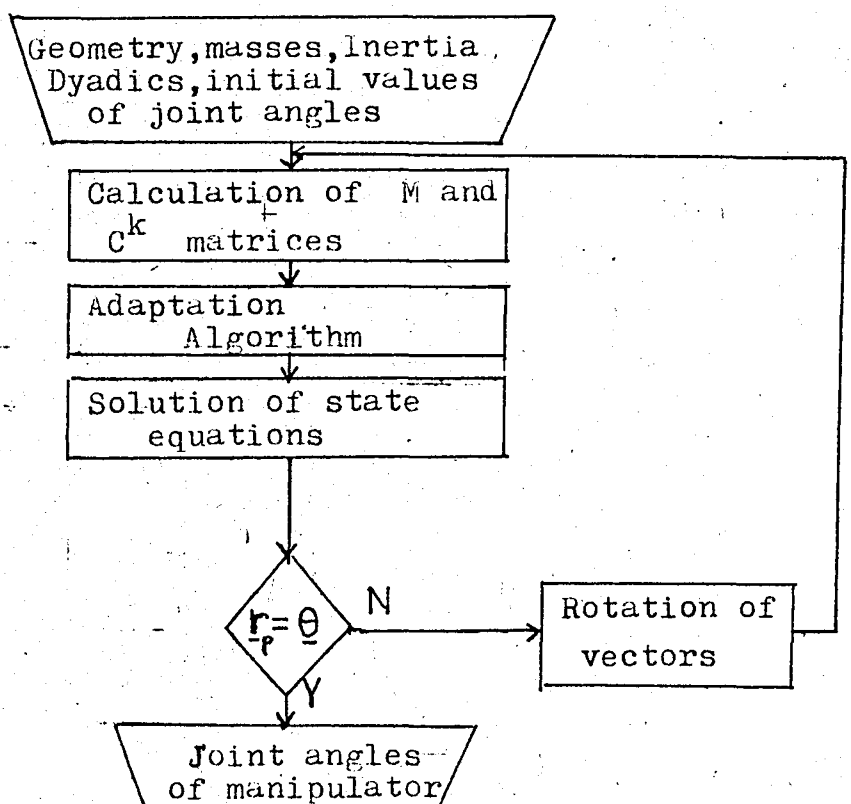


Figure 5.2 Flow chart of the MRAS manipulator control algorithm.

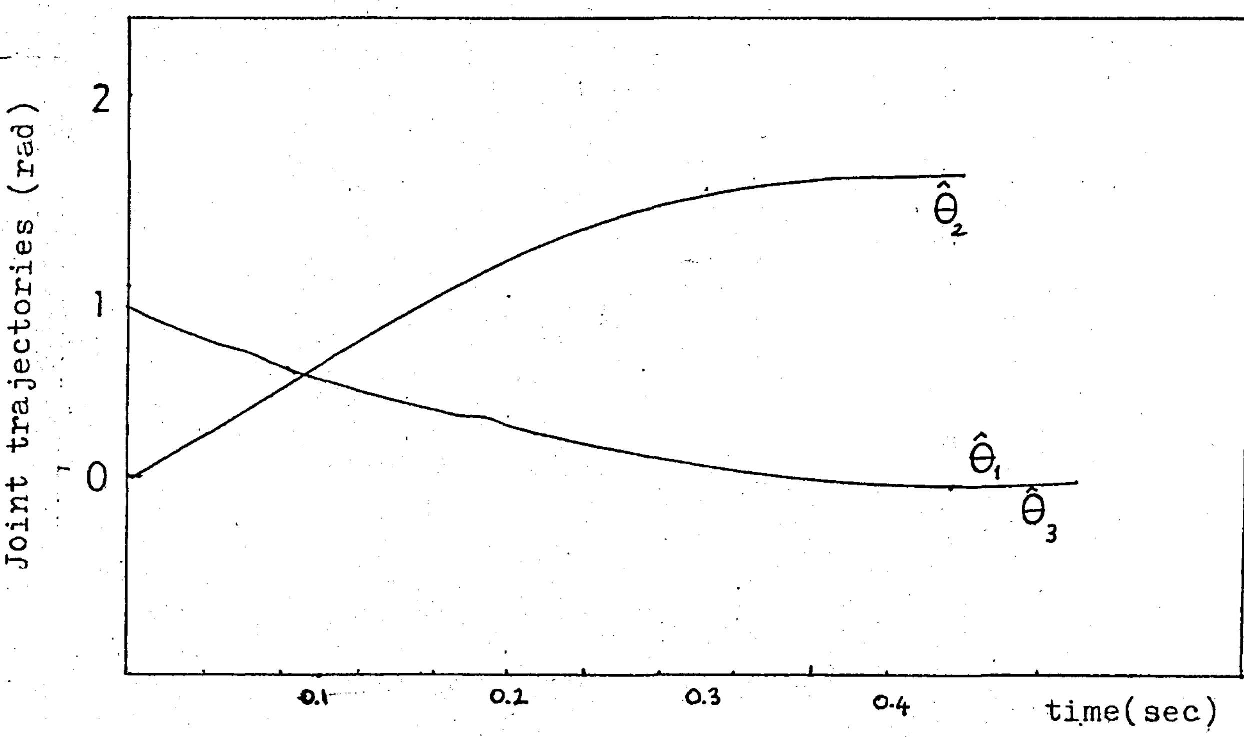


Figure 5.3 Joint angle trajectories of reference model for $r_p = [0\ 1.57\ 0]$ rad(without payload)

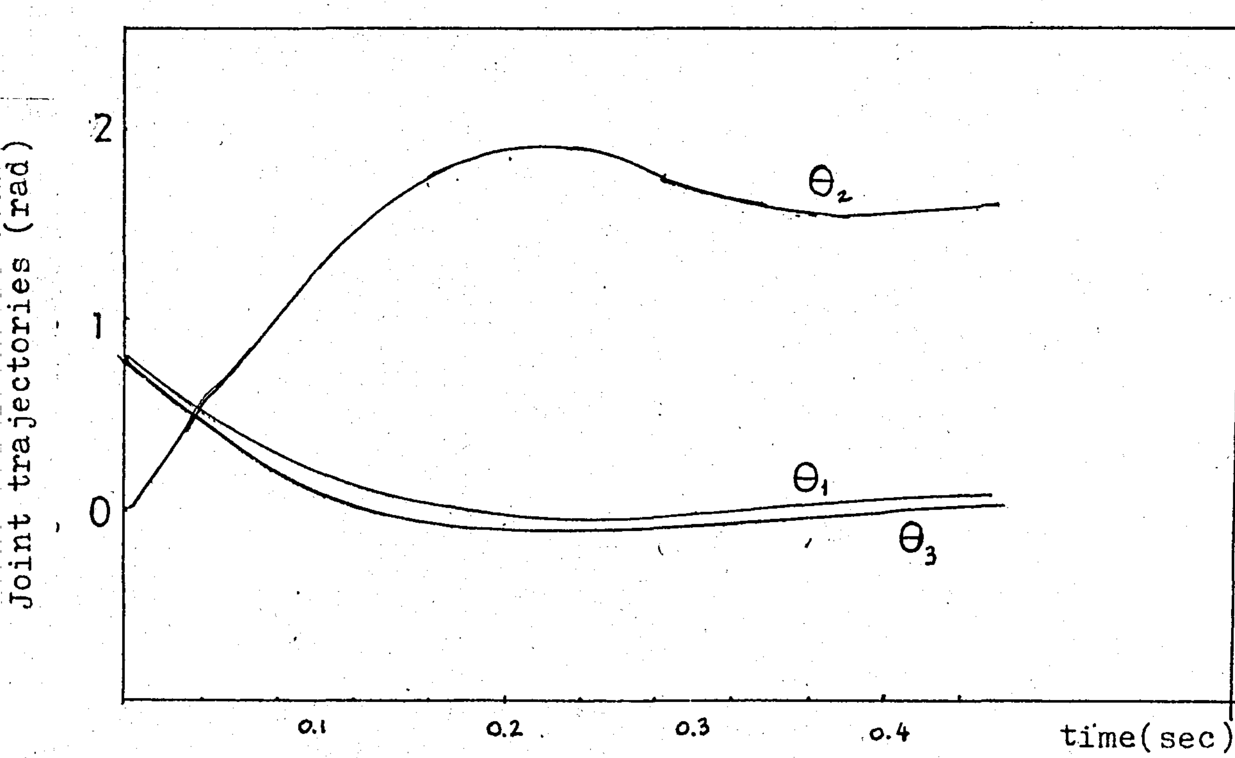


Figure 5.4 Joint angle trajectories of manipulator control system for $r_p = [0\ 1.57\ 0]$ rad. (without payload)

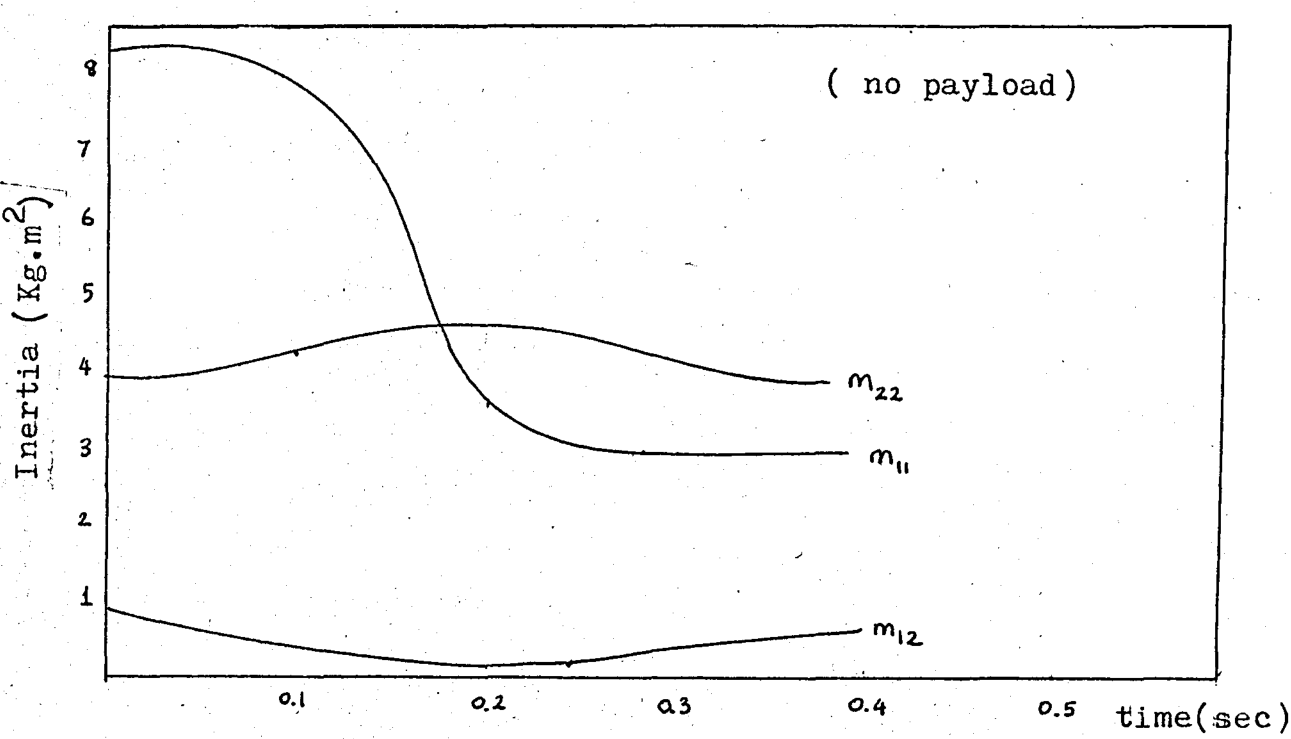


Figure 5.5 Elements of the inertia matrix during the motion from $\theta = [0.78\ 0\ 0.78]$ to $r_p = [0\ 1.57\ 0]$ rad

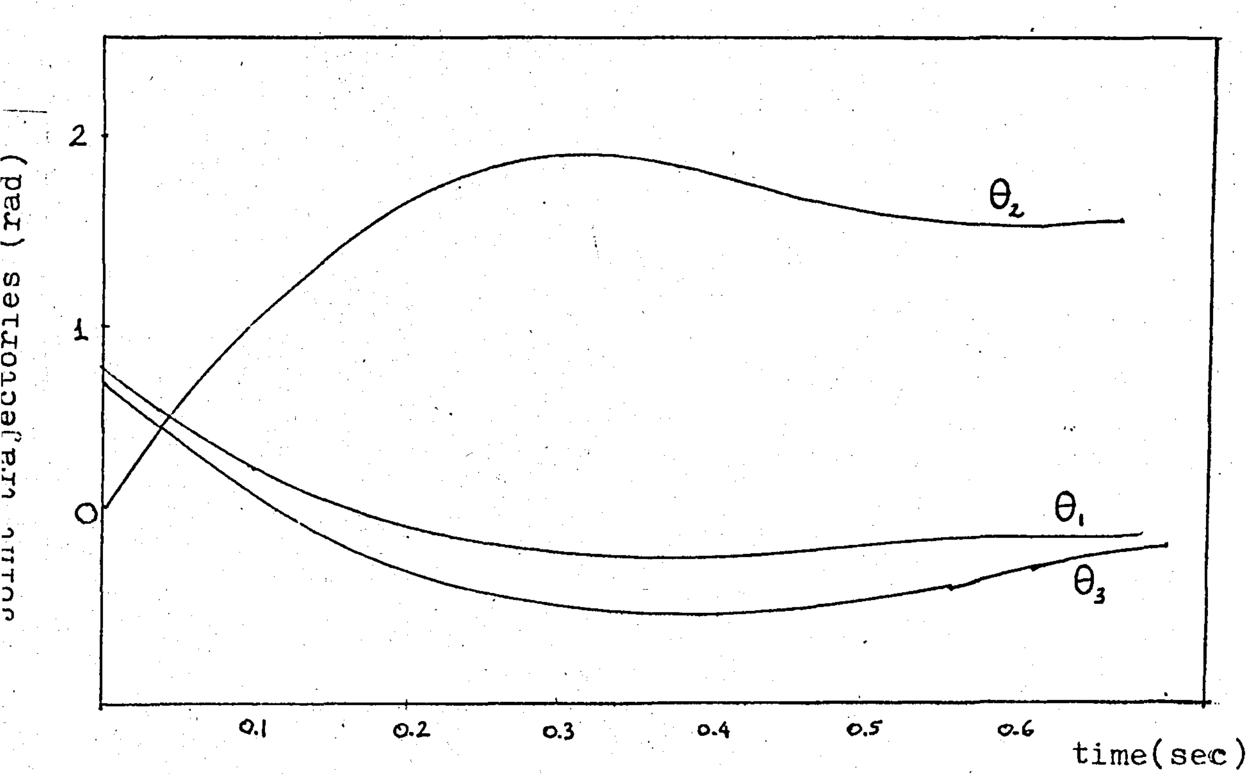


Figure 5.6 Joint angle trajectories of manipulator control system for $r_p = [0\ 1.57\ 0]$ rad.(payload= 5 kg)

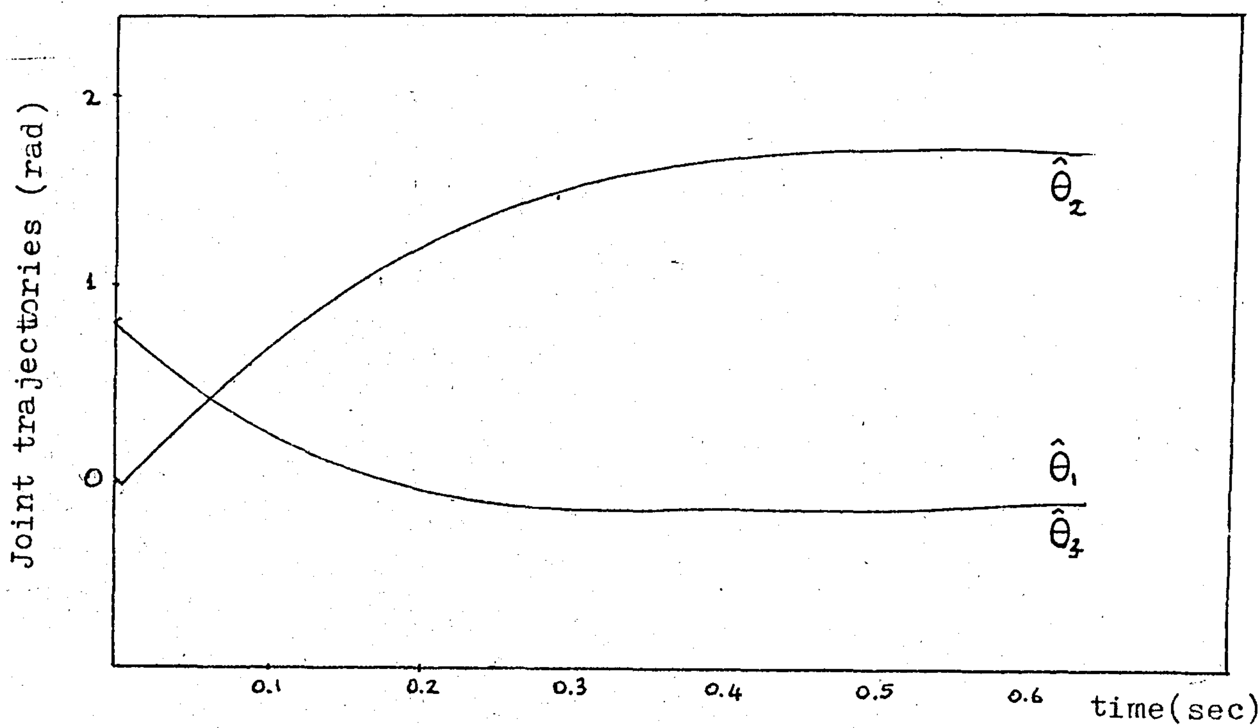


Figure 5.7 Joint angle trajectories of reference model for r_p=[0 1.57 0]rad (payload = 5 kg)

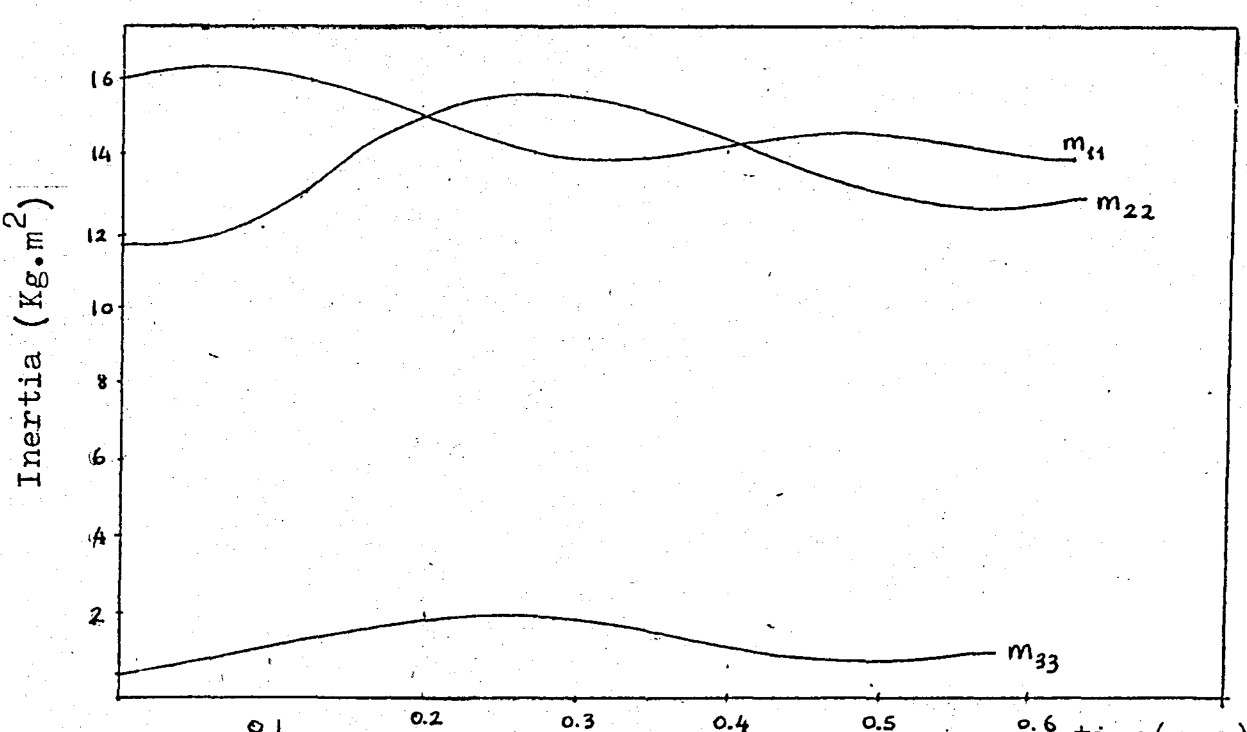


Figure 5.8 Elements of inertia matrix during the motion from θ =[0.78 0 0.78] to r_p=[0 1.57 0]rad (payload = 5 kg.)

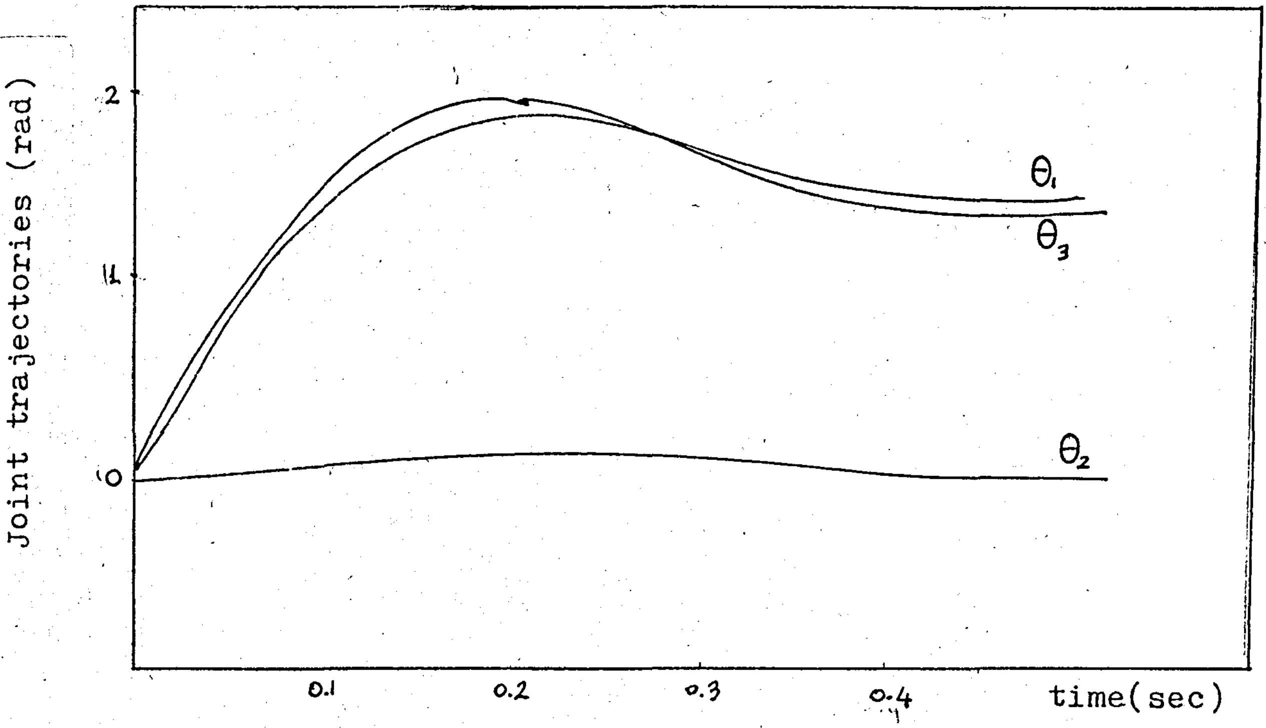


Figure 5.9 Joint angle trajectories of manipulator control system for $r_p=[1.57\ 0\ 1.57]$ rad (payload= 8 kg)

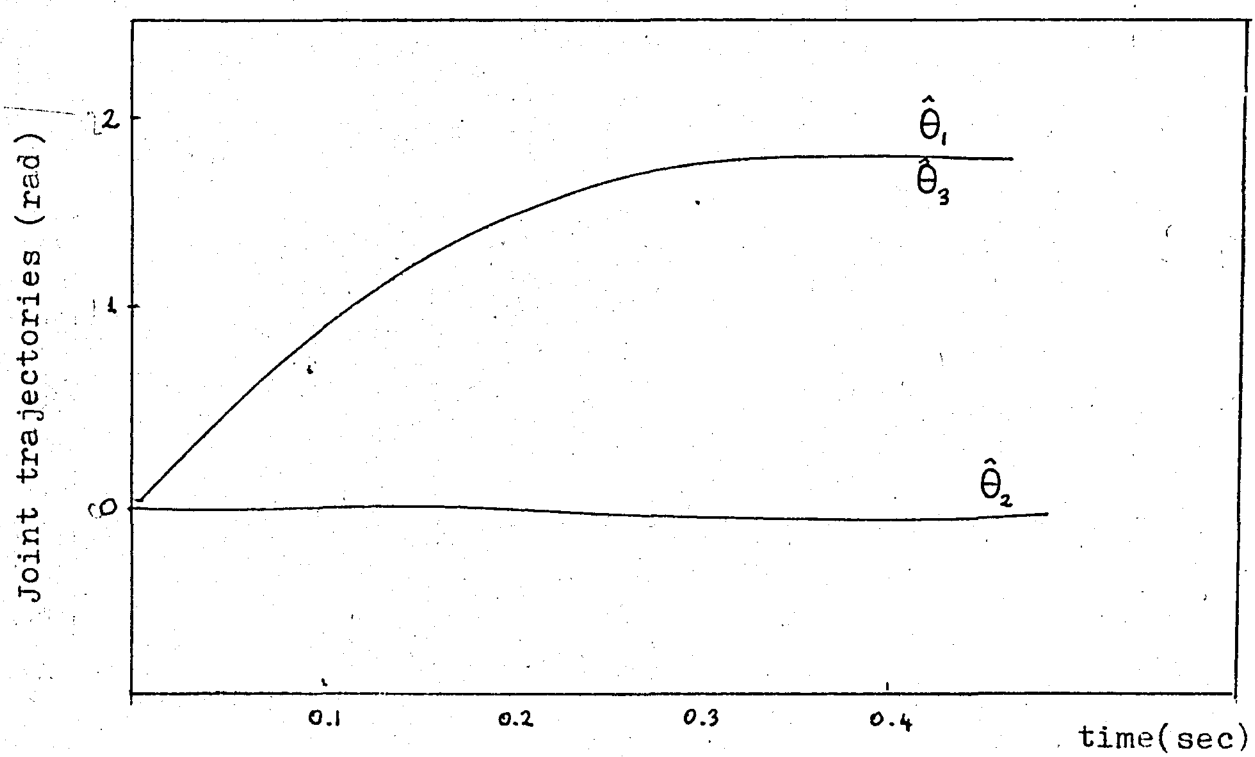


Figure 5. 10 Joint angle trajectories of reference model for $r_p=[1.57\ 0\ 1.57]$ rad (payload= 8 kg)

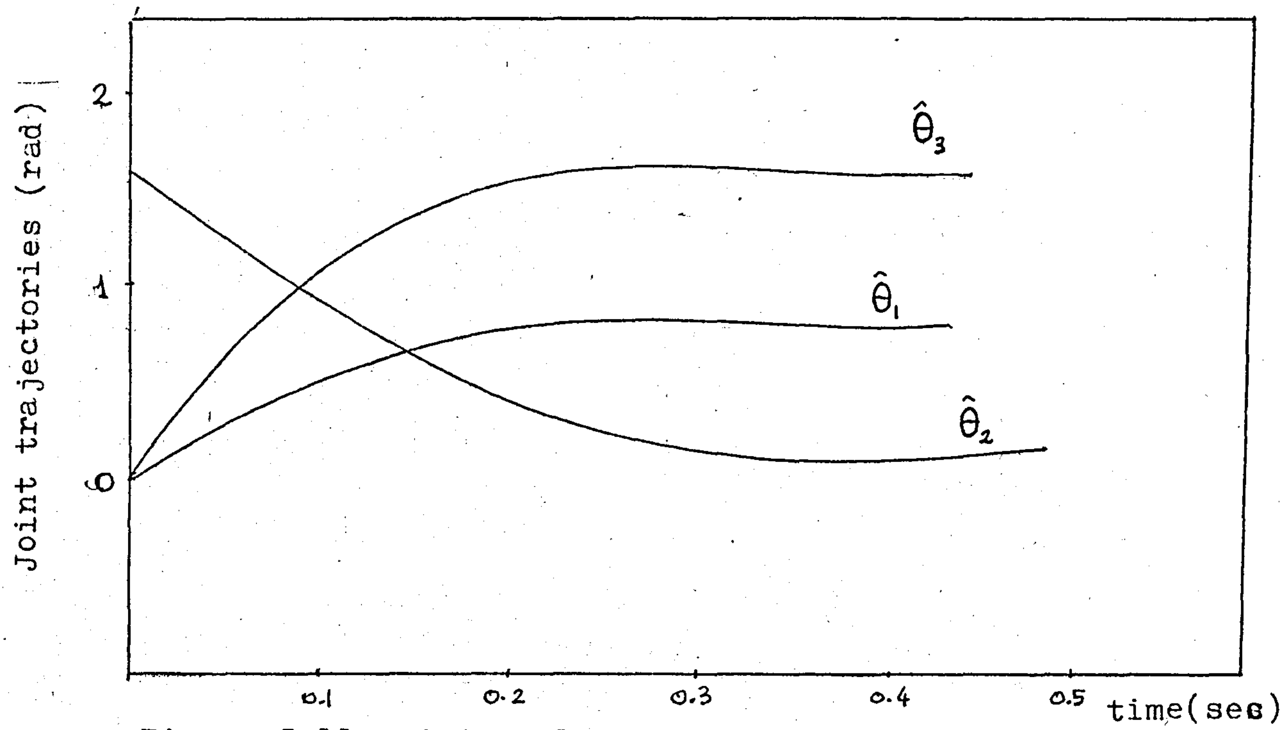


Figure 5.11 Joint angle trajectories of reference model for $r_p=[0.78\ 0\ 1.57]$ rad.(payload=10 kg)

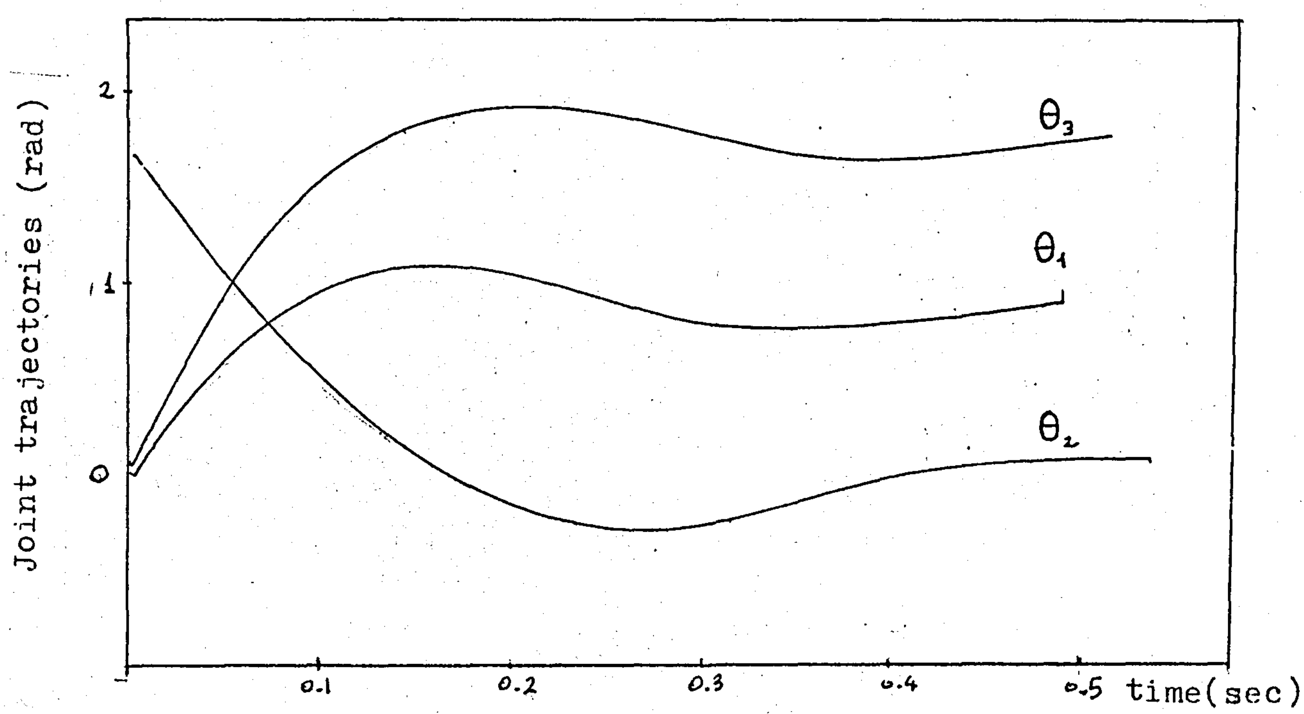


Figure 5.12 Joint angle trajectories of manipulator control system for $r_p=[0.78\ 0\ 1.57]$ rad. (payload= 10 kg)

$\theta^T=[\theta_1\ \theta_2\ \theta_3]$ (rad)	$r_p^T=[r_{p1} r_{p2} r_{p3}]$ (rad)	$P_i^T=[x_i y_i z_i]$ (m.)	$P_d^T=[x_d y_d z_d]$ (m.)	$P_a^T=[x_a y_a z_a]$ (m.)
0.78 0 0.78	0 1.57 0	- 0.625 1.281 0.642	0 0 1.804	- 0.007 0.003 1.802
(no payload)				
0.78 0 0.78	0 1.57 0	- 0.625 1.281 0.642	0 0 1.804	- 0.01 0.008 1.801
(payload= 5 kg.)				
0 0 0	1.57 0 1.57	0 1.804 0	- 0.89 0 0.914	- 0.878 0.011 0.912
(payload= 8 kg.)				
0 11.57 0	0.78 0 1.57	0 0 1.804	- 0.625 0.632 0.914	- 0.612 0.648 0.910
(payload= 10 kg.)				

$\theta^T=[\theta_1\ \theta_2\ \theta_3]$ initial values of joint angles

$r_p^T=[r_{p1} r_{p2} r_{p3}]$ desired input vector of joint angles

$P_i^T=[x_i\ y_i\ z_i]$ initial position of end-effector

$P_d^T=[x_d\ y_d\ z_d]$ desired position of end-effector

$P_a^T=[x_a\ y_a\ z_a]$ actual position of end-effector

Table 5.2 Initial and final positions of end-effector for different motions.

VI. CONCLUSION

Simply stated, a manipulator is a positioning device which consists of rigid links and several types of joints. The selection of proper manipulator configuration requires the examination of kinematics and dynamics properties. As mentioned in Chapter II, the dynamics of an n degrees of freedom manipulator can be described by n coupled, second order, nonlinear ordinary differential equations. Physically, the coupling terms represent gravitational torques which depend on positions of the joints, reaction torques due to the accelerations of other joints and Coriolis and centrifugal torques. The significance of coupling depends on the physical parameters of the manipulator and the load to be carried. In a manipulator control technique all these effects are taken into consideration.

The presented model reference Adaptive control method makes use of the basic properties of the dynamics equations of manipulator. In this method, an ideal form of the manipulator equations is selected as the reference model.The manipulator is used as the adjustable model such that it is tried to be forced to behave similar to the reference model.To achieve this type of control a term is added for cancelling the nonlinearity of the system.Another term is used to accomplish decoupling so that each position and acceleration pair is related by a double integrator. Making use of the hyperstability theory,additional design parameters are used for guaranteeing the stability of the control system.A computer simulation study was performed to evaluate the performance of the proposed control algorithm. It was observed that the response of the manipulator system is virtually identical to the response of the reference model for various payloads.The method,however,should be extended to consider other effective terms in the dynamic equations of the manipulator.

Several manipulator control methods were presented and examined to make a comparison with the given method. In the Resolved Motion Rate control algorithm the joint angle rates are determined to execute a motion in cartesian coordinates expressed in the world coordinates which describes the position and orientation of the end-effector. The main drawback of this method is the additional computational effort and the singularity problem of the Jacobian matrix.

Due to the nonlinearity and complexity of the dynamic equations of the manipulator, an optimal control law is very difficult to obtain.However, as mentioned in Chapter IV, an approximate feedback control law can be determined but the method is not applicable for the manipulators with four or more degrees of freedom.

The Cerebellar Model Articulation controller method is a table look-up control technique.The control functions are obtained by referring to a table rather than by solving the dynamic equations of the manipulator.It is a memory management technique which causes the similar inputs resulting in similar outputs called the generalization property.However,dissimilar inputs lead to independent outputs.These outputs are actuator drive signals for each joint.The major problems in this method are memory size and the accuracy of the outputs for practical applications. Therefore, some other memory organization techniques should be developed to solve this type of problems.

APPENDIX. A

COMPUTER PROGRAM OF SIMULATION STUDY

The variables used in the program are given below;

N	: Number of degrees of freedom
N2	: 2XN
U	: Axes of rotation
R	: Position vector of joints
G	: Vector of center of mass of links
IM	: Inertia Dyadics of links
MASS	: Masses of links
MHT	: Initial values of $\hat{M}$ matrix
NHT	: Initial values of $\hat{C}^k$ matrices
COFM	: Coefficients used in adaptation algorithm(for $\hat{M}$ matrix)
COFN	: Coefficients used in adaptation algorithm(for $\hat{C}^k$ matrices)

KP,KV,AI	: Feedback gain matrices
XP	: Initial values of joint angles
RP	: Desired values of joint angles
XPFIN 1,XPFIN 2, XPFIN 3	: Final values of joint angles of manipulator
XPH	: The output vector of reference model
MT	: Moment of inertia matrix

Given the initial values of the vectors and matrices given above, the trajectories of joint angles are printed out.Some of these vectors and matrices are three dimensional. The first dimension is used as index of any vector and matrix.The list of subprograms used in this program is given below:

SUBROUTİNE CROS	: It is used to obtain the cross product of two vectors.(three dimensional)
SUBROUTİNE TRANS	: It takes the transpose of any matrix.(three dimensional)
SUBROUTİNE TRACE	: It is used to obtain the trace of any matrix(three dimensional)

SUBROUTİNE MULT	: It is used to obtain the multiplication of two matrices. (three dimensional)
SUBROUTİNE EXPO	: It is used to obtain the state transition matrix of the system.
SUBROUTİNE MULTIQ	: It is used to obtain the multiplication of two matrices.(two dimensional)
SUBROUTİNE EVRİK	: It takes the inverse of any matrix.
SUBROUTİNE MVECA	: It is used to obtain the multiplication of a matrix (two dimensional) and a vector(one dimensional).
SUBROUTİNE VECA	: It multiplies two vectors. (one dimensional)
SUBROUTİNE FUBRİ	: It takes the difference of two matrices.(two dimensional)
SUBROUTİNE ROT 1	: It rotates any vector about the Z axis.
SUBROUTİNE ROT 2	: It rotates any vector about the X axis.

```
      PROGRAM NEJ(KILINC1,RUTPUT,TAPE5=KILINC1,TAPE6=RUTPUT)
      DIMENSION G(4,3,3),R(4,3,3),U(4,3,3),IM(4,3,3),A(6,6),B(6,3)
      DIMENSION C1(4,3,3),C2(4,3,3),TR1(4,3,3),TR2(4,3,3),Z1(4,3,3)
      DIMENSION Z2(4,3,3),Z3(4,3,3),F2(4,3,3),F1(4,3,3),S(3,3)
      DIMENSION F3(4,3,3),C3(4,3,3),TR3(4,3,3),F4(4,3,3),C4(4,3,3)
      DIMENSION C5(4,3,3),Z4(4,3,3),C6(4,3,3),CL(4,3,3),Z5(4,3,3)
      DIMENSION C7(4,3,3),TO(4,3,3),TR4(4,3,3),Z6(4,3,3),SE(3,3)
      DIMENSION MASS(4),MT(3,3),NTH(3,3,3),CV(3,3),CP(3,3),FV(3,3)
      DIMENSION FP(3,3),IDEN(3,3),FS(3,3),FT(3,3),FMAT(6,6),GMAT(6,3)
      DIMENSION NHT(3,3,3),MHT(3,3),COFM(3,3),COFN(3,3,3),INTEGA(6,6)
      DIMENSION R1(3),R2(3),R3(3),U1(3),U2(3),U3(3),G1(3),G2(3),G3(3)
      DIMENSION XVD(3),YVEK(3),CMAT(3,6),XF(6),XV1(3),XV2(3),XV3(3)
      DIMENSION XV6(3),XV7(3),MF(3,3),XVHD(3),XFS(3),X(6),XP(3),XV(3)
      DIMENSION RP(3),MG(3),NF1(3,3),NF2(3,3),NF3(3,3),SOM(3,3),XT6(3)
      DIMENSION XSON(3),XT1(3),XT2(3),XT3(3),XT4(3),XPFIN1(100),XSOL(3)
      DIMENSION XPFIN3(100),XPFIN2(100),XSOM(3),XV4(3),XV5(3),XPH(3)
      DIMENSION XVH(3),PI(3,1),PO(3,3),XE(6),ROM(3,1),NH1(3,3),XT5(3)
      DIMENSION NH3(3,3),NT1(3,3),NT2(3,3),NT3(3,3),SV(3)
      DIMENSION NH2(3,3),ST(6,6),KP(3,3),KD(3,3),AI(3,3),XVDS(6)
      DIMENSION XIP(3),XIV(3),XES(6),XET(6),RQ(3),XPVS(6),XPV(6)
      DIMENSION XEW(6),XEV(3),XEVD(3),X1(3),X2(3),XIT(3),SW(3)
      DIMENSION XA(6),XB(6),XC(6),XD(6),SY(3),GI(3),GA(3),X3(3)
      DIMENSION Y1(3),Y2(3),Y3(3),V1(3),V2(3),V3(3),S1(3),S2(3)
      DIMENSION S3(3),P1(3),P2(3),P3(3),O1(3),O2(3),O3(3),X4(3)
      REAL IM,MT,MASS,NTH,IDEN,NHT,MHT,INTEGA,MG,MF,NH1,NH2,NH3,NT1
      REAL NF1,NF2,NF3,NT2,NT3,KP,KD
      N=3
      N2=2*N
      ERR=0.1
      LFI=1
      HL=0.1
      M=1
      COFN(1,1,2)=0.050
      COFN(1,1,3)=0.10
      COFN(1,2,2)=0.10
      COFN(1,3,3)=0.020
      COFN(2,3,3)=0.10
      READ(5,*) ((G(NM,I,M),I=1,N),NM=1,N+1)
      READ(5,*) ((R(NM,I,M),I=1,N),NM=1,N+1)
      READ(5,*) ((U(NM,I,M),I=1,N),NM=1,N+1)
      READ(5,*) ((IM(M,I,J),J=1,N),I=1,N)
      READ(5,*) ((IM(2,I,J),J=1,N),I=1,N)
      READ(5,*) ((IM(3,I,J),J=1,N),I=1,N)
      READ(5,*) ((IM(4,I,J),J=1,N),I=1,N)
      READ(5,*) (MASS(I),I=1,N+1)
      READ(5,*) ((IDEN(I,J),J=1,N),I=1,N)
      READ(5,*) ((FMAT(I,J),J=1,N2),I=1,N)
      READ(5,*) ((GMAT(I,J),J=1,N),I=1,N)
      READ(5,*) ((NHT(1,I,J),J=1,N),I=1,N)
      READ(5,*) ((NHT(2,I,J),J=1,N),I=1,N)
      READ(5,*) ((NHT(3,I,J),J=1,N),I=1,N)
      READ(5,*) ((MHT(I,J),J=1,N),I=1,N)
      READ(5,*) ((COFM(I,J),J=1,N),I=1,N)
      READ(5,*) ((KP(I,J),J=1,N),I=1,N)
      READ(5,*) ((KD(I,J),J=1,N),I=1,N)
      READ(5,*) ((AI(I,J),J=1,N),I=1,N)
      READ(5,*) ((PO(I,J),J=1,N),I=1,N)
      READ(5,*) ((PI(I,J),J=1,M),I=1,N)
      READ(5,*) (XP(I),I=1,N)
      READ(5,*) (RP(I),I=1,N)
      XPFIN1(1)=XP(1)
      XPFIN2(1)=XP(2)
      XPFIN3(1)=XP(3)
      TH1=XP(1)
      TH2=XP(2)
      TH3=XP(3)
```

```
      DO 3100 I=1,N
      U1(I)=U(1,I,1)
      U2(I)=U(2,I,1)
      U3(I)=U(3,I,1)
      R1(I)=R(1,I,1)
      R2(I)=R(2,I,1)
      R3(I)=R(3,I,1)
      G1(I)=G(1,I,1)
      G2(I)=G(2,I,1)
      G3(I)=G(3,I,1)
3100  CONTINUE
C*********************************************************************
C  ROTATION OF VECTORS ACCORDING TO INITIAL CONDITIONS               *
C                                                                    *
C                        OF JOINT ANGLES                             *
C*********************************************************************
      CALL ROT1(U1,TH1,X1)
      CALL ROT1(U2,TH1,X2)
      CALL ROT1(U3,TH1,X3)
      CALL ROT1(R1,TH1,Y1)
      CALL ROT1(R2,TH1,Y2)
      CALL ROT1(R3,TH1,Y3)
      CALL ROT1(G1,TH1,V1)
      CALL ROT1(G2,TH1,V2)
      CALL ROT1(G3,TH1,V3)
      CALL ROT2(X1,TH2,S1)
      CALL ROT2(X2,TH2,S2)
      CALL ROT2(X3,TH2,S3)
      CALL ROT2(Y1,TH2,P1)
      CALL ROT2(Y2,TH2,P2)
      CALL ROT2(Y3,TH2,P3)
      CALL ROT2(V1,TH2,O1)
      CALL ROT2(V2,TH2,O2)
      CALL ROT2(V3,TH2,O3)
      DO 3300 I=1,N
      U(1,I,1)=S1(I)
      U(2,I,1)=S2(I)
      U(3,I,1)=S3(I)
      R(1,I,1)=P1(I)
      R(2,I,1)=P2(I)
      R(3,I,1)=P3(I)
      G(1,I,1)=O1(I)
      G(2,I,1)=O2(I)
      G(3,I,1)=O3(I)
3300  CONTINUE
      DO 37 I=1,N
      XV(I)=0.
      XPH(I)=XP(I)
      XVH(I)=XV(I)
      XVD(I)=0.
      XVHD(I)=0.
 37   CONTINUE
      SIGMV=3.
      SIGMP=1.
      ROV=0.35
      ROP=0.25
      DO 40 I=1,N
      DO 40 J=1,N
      CV(I,J)=SIGMV*IDEN(I,J)
      CP(I,J)=SIGMP*IDEN(I,J)
      FV(I,J)=ROV*IDEN(I,J)
      FP(I,J)=ROP*IDEN(I,J)
 40   CONTINUE
      DO 80 K=1,N
      X(K)=XP(K)-XPH(K)
      X(K+N)=XV(K)-XVH(K)
 80   CONTINUE
      DO 3700 I=1,N2
      XES(I)=X(I)
3700  CONTINUE
```

```
C*****************************************************************************
C                                                               K            *
C CALCULATION   OF   INERTIA MATRIX  M    AND  C    MATRICES                 *
C                                                                            *
C*****************************************************************************
 900    DO 2 I=1,N
        DO 2 J=1,N
        K=MAX(I,J)
        MT(I,J)=0.0
 3      CALL FARK (G,R,F1,K,I)
        CALL FARK(G,R,F2,K,J)
        CALL CROS(U,F2,C2,J,J)
        CALL CROS(U,F1,C1,I,I)
        CALL TRANS(C1,TR1,I,3,1)
        CALL MULT(TR1,C2,Z1,I,1,3,J,1)
        DA=MASS(K)*Z1(I,1,1)
        CALL MULT(IM,U,Z2,K,3,3,J,1)
        CALL TRANS(U,TR2,I,3,1)
        CALL MULT(TR2,Z2,Z3,I,1,3,K,1)
        MT(I,J)=DA+Z3(I,1,1)+MT(I,J)
        IF(K.EQ.N+1) GO TO 2
        K=K+1
        GO TO 3
  2     CONTINUE
        DO 9 K=1,N
        DO 9 I=1,N
        DO 9 J=1,N
        L=MAX(K,J)
        IF(J.LT.I) GO TO 15
        NTH(K,I,J)=0.0
  6     CALL FARK(G,R,F3,L,K)
        CALL CROS(U,F3,C3,K,K)
        CALL TRANS(C3,TR3,K,3,1)
        CALL FARK(G,R,F4,L,J)
        CALL CROS(U,F4,C4,J,J)
        CALL CROS(U,C4,C5,I,J)
        CALL MULT(TR3,C5,Z4,K,1,3,I,1)
        BT=MASS(L)*Z4(K,1,1)
        CALL TRACE(IM,L,N,T1)
        T1=0.5*T1
        CALL CROS(U,U,C6,I,J)
        DO 7 IS=1,N
        CL(I,IS,1)=C6(I,IS,1)*T1
  7     CONTINUE
        CALL MULT(IM,U,Z5,L,3,3,I,1)
        CALL CROS(U,Z5,C7,J,L)
        DO 17 JS=1,N
        TO(I,JS,1)=CL(I,JS,1)+C7(J,JS,1)
 17     CONTINUE
        CALL TRANS(U,TR4,K,3,1)
        CALL MULT(TR4,TO,Z6,K,1,3,I,1)
        NTH(K,I,J)=BT+Z6(K,1,1)+NTH(K,I,J)
 16     IF(L.EQ.N+1) GO TO 9
        L=L+1
        GO TO 6
 15     NTH(K,I,J)=NTH(K,J,I)
  9     CONTINUE
        DO 50 I=1,N
        DO 50 J=1,N
        NH1(I,J)=NHT(1,I,J)
        NH2(I,J)=NHT(2,I,J)
        NH3(I,J)=NHT(3,I,J)
        NT1(I,J)=NTH(1,I,J)
        NT2(I,J)=NTH(2,I,J)
        NT3(I,J)=NTH(3,I,J)
 50     CONTINUE
        DO 41 I=1,N
        DO 41 J=1,N
        S(I,J)=MT(I,J)
 41     CONTINUE
```

```
      CALL RINV(N,S,N,RQ,IFAIL)
      IF(IFAIL.EQ.(-1.)) GO TO 3467
      CALL MULTIQ(S,FP,FS,N,N,N)
      CALL MULTIQ(S,FV,FT,N,N,N)
      DO 42 I=1,N
      DO 42 J=1,N
      FS(I,J)=-1.0*FS(I,J)
      FT(I,J)=-1.0*FT(I,J)
  42  CONTINUE
      DO 43 I=1,N
      DO 43 J=1,N
      NL=I+N
      ML=J+N
      FMAT(NL,J)=FS(I,J)
      FMAT(NL,ML)=FT(I,J)
      GMAT(NL,J)=S(I,J)
 43   CONTINUE
      CALL EXPO(FMAT,GMAT,A,B,6,3,0.1)
C**********************************************************
C     ADAPTATION         ALGORITHM                        *
C                                                         *
C**********************************************************
      DO 86 I=1,N
      MHT(I,I)=MHT(I,I)-(HL*COFM(I,I)*(YVEK(I)*XVHD(I)))
  86  CONTINUE
      OX=YVEK(1)*XVHD(2)+YVEK(2)*XVHD(1)
      MHT(1,2)=MHT(1,2)-(HL*COFM(1,2)*OX)
      MHT(2,1)=MHT(1,2)
      OY=YVEK(1)*XVHD(3)+YVEK(3)*XVHD(1)
      MHT(1,3)=MHT(1,3)-(HL*COFM(1,3)*OY)
      MHT(3,1)=MHT(1,3)
      OQ=YVEK(2)*XVHD(3)+YVEK(3)*XVHD(2)
      MHT(2,3)=MHT(2,3)-(HL*COFM(2,3)*OQ)
      MHT(3,2)=MHT(2,3)
      OM=YVEK(2)*(XV(1)**2.)
      OL=2.*YVEK(1)*XV(2)*XV(1)
      NH1(1,2)=NH1(1,2)-(HL*COFN(1,1,2)*(OL-OM))
      XSL=2.*YVEK(1)*XV(1)*XV(3)
      XSM=YVEK(3)*(XV(1)**2.)
      NH1(1,3)=NH1(1,3)-(HL*COFN(1,1,3)*(XSL-XSM))
      NH1(2,2)=NH1(2,2)-(HL*COFN(1,2,2)*(YVEK(1)*(XV(2)**2.)))
      YSL=2.*YVEK(1)*XV(2)*XV(3)
      YSM=YVEK(1)*(XV(3)**2.)
      NH1(3,3)=NH1(3,3)-(HL*COFN(1,3,3)*(YSL+YSM))
      WSL=2.*YVEK(2)*XV(2)*XV(3)
      WSM=YVEK(2)*(XV(3)**2.)
      WSN=YVEK(3)*(XV(2)**2.)
      NH2(3,3)=NH2(3,3)-(HL*COFN(2,3,3)*(WSL+WSM-WSN))
      NHT(1,1,2)=NH1(1,2)
      NHT(1,1,3)=NH1(1,3)
      NHT(1,2,2)=NH1(2,2)
      NHT(1,3,3)=NH1(3,3)
      NHT(2,3,3)=NH2(3,3)
      NHT(1,2,1)=NHT(1,1,2)
      NHT(1,3,1)=NHT(1,1,3)
      NHT(1,2,3)=NHT(1,3,3)
      NHT(1,3,2)=NHT(1,3,3)
      NHT(2,1,1)=-NHT(1,1,2)
      NHT(2,2,3)=NHT(2,3,3)
      NHT(2,3,2)=NHT(2,3,3)
      NHT(3,1,1)=-NHT(1,1,3)
      NHT(3,2,2)=-NHT(2,3,3)
      DO 51 I=1,N
      DO 51 J=1,N
      NH1(I,J)=NHT(1,I,J)
      NH2(I,J)=NHT(2,I,J)
      NH3(I,J)=NHT(3,I,J)
      NT1(I,J)=NTH(1,I,J)
      NT2(I,J)=NTH(2,I,J)
      NT3(I,J)=NTH(3,I,J)
  51  CONTINUE
```

```
 420    FORMAT(20X,'X(',I2,')=',F10.3,/)
C*****************************************************************
C      CALCULATION OF INPUT VECTOR  FOR THE ERROR STATE MODEL     *
C                                                                 *
C*****************************************************************
        CALL FUBRI(MHT,MT,MF,N,N)
        CALL FUBRI(NH1,NT1,NF1,N,N)
        CALL FUBRI(NH2,NT2,NF2,N,N)
        CALL FUBRI(NH3,NT3,NF3,N,N)
        CALL MVECA(MF,XVHD,XFS,N,N)
        CALL MVECA(NF1,XV,XV1,N,N)
        CALL MVECA(NF2,XV,XV2,N,N)
        CALL MVECA(NF3,XV,XV3,N,N)
        CALL VECA(XV,XV1,WT,N)
        CALL VECA(XV,XV2,WU,N)
        CALL VECA(XV,XV3,WV,N)
        MG(1)=XFS(1)+WT
        MG(2)=XFS(2)+WU
        MG(3)=XFS(3)+WV
        DO 591 LOP=1,N
 591    WRITE(6,592) LOP,MG(LOP)
 592    FORMAT(30X,'MG(',I2,')=',F10.3,/)
        CALL MVECA(A,X,XE,N2,N2)
        CALL MVECA(B,MG,XF,N2,N)
        DO 82 I=1,N2
        X(I)=XE(I)+XF(I)
  82    CONTINUE
        DO 450 JHL=1,N2
 450    WRITE(6,460) JHL,X(JHL)
 460    FORMAT(25X,'X(',I2,')=',F10.3,/)
        DO 83 I=1,N
        DO 83 J=1,N
        IH=J+N
        CMAT(I,J)=CP(I,J)
        CMAT(I,IH)=CV(I,J)
  83    CONTINUE
        CALL MVECA(CMAT,XES,YVEK,3,6)
        DO 492 I=1,N
 492    WRITE(6,1100) I,YVEK(I)
 1100   FORMAT(29X,'YVEK(',I2,')=',F9.3,/)
        DO 89 I=1,N
        U1(I)=U(1,I,1)
        U2(I)=U(2,I,1)
        U3(I)=U(3,I,1)
        R1(I)=R(1,I,1)
        R2(I)=R(2,I,1)
        R3(I)=R(3,I,1)
        G1(I)=G(1,I,1)
        G2(I)=G(2,I,1)
        G3(I)=G(3,I,1)
  89    CONTINUE
C*****************************************************************
C     OBTAIN    NEW     XP HAT  VECTOR                            *
C    (OUTPUT OF THE   REFERENCE MODEL )                           *
C ****************************************************************
        CALL EXPO(PO,PI,SOM,ROM,3,1,0.1)
        DO 2106 I=1,N
        XT4(I)=ROM(I,1)*RP(1)
        XT5(I)=ROM(I,1)*RP(2)
        XT6(I)=ROM(I,1)*RP(3)
2106    CONTINUE
        XIP(1)=XPH(1)
        XIP(2)=XVH(1)
        XIP(3)=XVHD(1)
        XIV(1)=XPH(2)
        XIV(2)=XVH(2)
        XIV(3)=XVHD(2)
        XIT(1)=XPH(3)
        XIT(2)=XVH(3)
        XIT(3)=XVHD(3)
```

```
      CALL MVECA(SOM,XIP,SW,3,3)
      CALL MVECA(SOM,XIV,SV,3,3)
      CALL MVECA(SOM,XIT,SY,3,3)
      DO 5030 I=1,N
      XIP(I)=SW(I)+XT4(I)
      XIV(I)=SV(I)+XT5(I)
      XIT(I)=SY(I)+XT6(I)
5030  CONTINUE
      XPH(1)=XIP(1)
      XPH(2)=XIV(1)
      XPH(3)=XIT(1)
      XVH(1)=XIP(2)
      XVH(2)=XIV(2)
      XVH(3)=XIT(2)
      XVHD(1)=XIP(3)
      XVHD(2)=XIV(3)
      XVHD(3)=XIT(3)
      WRITE(6,5060)(XPH(I),I=1,N)
5060  FORMAT(10X,'XPH ;',3F9.3,//)
      WRITE(6,5070)(XVH(I),I=1,N)
5070  FORMAT(5X,'XVH  ;',3F9.3,//)
C**************************************************************
C   CALCULATION  OF  NEW  XP   VECTOR                         *
C   ( VECTOR  OF  THE  JOINT  ANGLES  )                       *
C *************************************************************
      DO 4100 I=1,N
      XP(I)=X(I)+XPH(I)
      XV(I)=X(I+N)+XVH(I)
 4100 CONTINUE
      XSOL(1)=XP(1)
      XSOL(2)=XV(1)
      XSOL(3)=XVHD(1)
      XSOM(1)=XP(2)
      XSOM(2)=XV(2)
      XSOM(3)=XVHD(2)
      XSON(1)=XP(3)
      XSON(2)=XV(3)
      XSON(3)=XVHD(3)
      CALL MVECA(SOM,XSOL,XT1,3,3)
      DO 2103 J=1,N
2103  XSOL(J)=XT1(J)+XT4(J)
      CALL MVECA(SOM,XSOM,XT2,3,3)
      DO 2105 J=1,N
      XSOM(J)=XT2(J)+XT5(J)
2105  CONTINUE
      CALL MVECA(SOM,XSON,XT3,3,3)
      DO 2108 J=1,N
2108  XSON(J)=XT3(J)+XT6(J)
      XP(1)=XSOL(1)
      XP(2)=XSOM(1)
      XP(3)=XSON(1)
      XV(1)=XSOL(2)
      XV(2)=XSOM(2)
      XV(3)=XSON(2)
      XVD(1)=XSOL(3)
      XVD(2)=XSOM(3)
      XVD(3)=XSON(3)
      DO 1512 I=1,N
      X(I)=XP(I)-XPH(I)
      X(I+N)=XV(I)-XVH(I)
      XES(I)=X(I)
      XES(I+N)=X(I+N)
 1512 CONTINUE
      TH1=XP(1)
      TH2=XP(2)
      TH3=XP(3)
      LFI=LFI+1
      XPFIN1(LFI)=TH1
      XPFIN2(LFI)=TH2
      XPFIN3(LFI)=TH3
```

```
      ZA=RP(1)-TH1
      ZB=RP(2)-TH2
      ZC=RP(3)-TH3
      TH1=ZA
      TH2=ZB
      TH3=ZC
      ZD=ABS(ZA)
      ZE=ABS(ZB)
      ZF=ABS(ZC)
      IF(ZD.LE.ERR) GO TO 928
C**********************************************************
C    ROTATION  OF  VECTORS TO OBTAIN NEW POSITIONS      *
C              AFTER DESIRED  MOVEMENT                  *
C**********************************************************
 972  CALL ROT1(U1,TH1,X1)
      CALL ROT1(U2,TH1,X2)
      CALL ROT1(U3,TH1,X3)
      CALL ROT1(R1,TH1,Y1)
      CALL ROT1(R2,TH1,Y2)
      CALL ROT1(R3,TH1,Y3)
      CALL ROT1(G1,TH1,V1)
      CALL ROT1(G2,TH1,V2)
      CALL ROT1(G3,TH1,V3)
 928  IF(ZE.LE.ERR) GO TO 937
2078  CALL ROT2(X1,TH2,S1)
      CALL ROT2(X2,TH2,S2)
      CALL ROT2(X3,TH2,S3)
      CALL ROT2(Y1,TH2,P1)
      CALL ROT2(Y2,TH2,P2)
      CALL ROT2(Y3,TH2,P3)
      CALL ROT2(V1,TH2,O1)
      CALL ROT2(V2,TH2,O2)
      CALL ROT2(V3,TH2,O3)
 937  IF((ZF.LE.ERR).AND.(ZE.LE.ERR).AND.(ZD.LE.ERR))GO TO 800
 761  DO 60 I=1,N
      U(1,I,1)=S1(I)
      U(2,I,1)=S2(I)
      U(3,I,1)=S3(I)
      R(1,I,1)=P1(I)
      R(2,I,1)=P2(I)
      R(3,I,1)=P3(I)
      G(1,I,1)=O1(I)
      G(2,I,1)=O2(I)
      G(3,I,1)=O3(I)
  60  CONTINUE
      GO TO 900
 800  WRITE(6,801)
 801  FORMAT(20X,'THETA 1',13X,'THETA 2',11X,'THETA 3',//)
      DO 700 I=1,LFI
      WRITE(6,701) XPFIN1(I),XPFIN2(I),XPFIN3(I)
 701  FORMAT(20X,F7.4,12X,F7.4,12X,F7.4)
 700  CONTINUE
      TH1=XPFIN1(LFI)
      TH2=XPFIN2(LFI)
      TH3=XPFIN3(LFI)
3467  WRITE(6,3451)
3451  FORMAT(15X,'   INERTIA MATRIX IS SINGULAR ',//)
      STOP
      END
      SUBROUTINE FARK(A,B,CF,KL,LT)
      DIMENSION A(4,3,3),B(4,3,3),CF(4,3,3)
      DO 1 II=1,3
      DO 1 JJ=1,1
   1  CF(LT,II,JJ)=A(KL,II,JJ)-B(LT,II,JJ)
      RETURN
      END
```

```
      SUBROUTINE CROS(D,F,CR,KX,IX)
      DIMENSION D(4,3,3),F(4,3,3),CR(4,3,3)
      CR(KX,1,1)=D(KX,2,1)*F(IX,3,1)-D(KX,3,1)*F(IX,2,1)
      CR(KX,2,1)=D(KX,3,1)*F(IX,1,1)-D(KX,1,1)*F(IX,3,1)
      CR(KX,3,1)=D(KX,1,1)*F(IX,2,1)-D(KX,2,1)*F(IX,1,1)
      RETURN
      END
      SUBROUTINE TRANS(ZL,CT,LI,NN,MM)
      DIMENSION ZL(4,3,3),CT(4,3,3)
      DO 1 II=1,NN
      DO 1 JJ=1,MM
      CT(LI,JJ,II)=ZL(LI,II,JJ)
 1    CONTINUE
      RETURN
      END
      SUBROUTINE TRACE(P,LZ,NZ,T)
      DIMENSION P(4,3,3)
      T=0.
      DO 1 II=1,NZ
      T=T+P(LZ,II,II)
 1    CONTINUE
      RETURN
      END
      SUBROUTINE MULT(X,Y,Z,N1,N2,N3,N4,N5)
      DIMENSION X(4,3,3),Y(4,3,3),Z(4,3,3)
      DO 1 II=1,N2
      DO 1 JJ=1,N5
      TT=0.
      DO 10 KK=1,N3
      TT=TT+X(N1,II,KK)*Y(N4,KK,JJ)
 10   CONTINUE
 1    Z(N1,II,JJ)=TT
      RETURN
      END
      SUBROUTINE EXPO(F,D,A,Y,NN,MM,TX)
      DIMENSION F(NN,NN),D(NN,MM),A(NN,NN),Y(NN,MM),INTEGA(6,6)
      DIMENSION ST(6,6)
      REAL INTEGA
      INTEGER POWER
      NORMFT=0.0
      DO 1 II=1,NN
      DO 1 JJ=1,NN
      ST(II,JJ)=F(II,JJ)*TX
 1    A(II,JJ)=ST(II,JJ)
      POWER=100.
      DO 7 II=2,POWER
      FPOWR=POWER-II+2
      DO 5 JJ=1,NN
      DO 3 KK=1,NN
 3    INTEGA(JJ,KK)=A(JJ,KK)/FPOWR
 5    INTEGA(JJ,JJ)=INTEGA(JJ,JJ)+1.0
      DO 4 JL=1,NN
      DO 4 JK=1,NN
      A(JL,JK)=0.
      DO 4 JH=1,NN
 4    A(JL,JK)=A(JL,JK)+ST(JL,JH)*INTEGA(JH,JK)
 7    CONTINUE
      DO 9 JJ=1,NN
      A(JJ,JJ)=A(JJ,JJ)+1.0
      DO 9 KK=1,NN
 9    INTEGA(JJ,KK)=TX*INTEGA(JJ,KK)
      DO 10 II=1,NN
      DO 10 JJ=1,MM
      Y(II,JJ)=0.
      DO 10 KK=1,NN
 10   Y(II,JJ)=Y(II,JJ)+INTEGA(II,KK)*D(KK,JJ)
      RETURN
      END
```

```
      SUBROUTINE MULTIQ(AB,CD,EF,N1,N2,N3)
      DIMENSION AB(N1,N2),CD(N2,N3),EF(N1,N3)
      DO 1 II=1,N1
      DO 1 JJ=1,N3
      TX=0.
      DO 2 KK=1,N2
      TX=TX+AB(II,KK)*CD(KK,JJ)
    2 CONTINUE
    1 EF(II,JJ)=TX
      RETURN
      END
      SUBROUTINE EVRIK(S,OR,N)
      DIMENSION S(6,6),OR(3,3)
      I=1
      NX=N+1
      NY=2*N
      DO 80 J=NX,NY
      S(I,J)=1.0
   80 I=I+1
      L=1
      K=2
  110 XM=S(L,L)
      DO 140 J=L,NY
  140 S(L,J)=S(L,J)/XM
      DO 190 I=K,N
      X=S(I,L)
      DO 190 J=L,NY
  190 S(I,J)=S(I,J)-S(L,J)*X
      L=L+1
      K=K+1
      IF(L-N)110,110,230
  230 L=N
  235 LZ=L-1
      DO 290 K=1,LZ
      I=L-K
      Y=S(I,L)
      DO 290 J=L,NY
  290 S(I,J)=S(I,J)-S(L,J)*Y
      L=L-1
      IF(L.GT.1) GO TO 235
      DO 45 I=1,N
      DO 45 J=NX,NY
      LN=J-N
      OR(I,LN)=S(I,J)
   45 CONTINUE
      RETURN
      END
      SUBROUTINE MVECA (A,B,C,LI,MI)
      DIMENSION A(LI,MI),B(MI),C(LI)
      DO 15 IK=1,LI
      TS=0.
      DO 10 JK=1,MI
      TS=TS+A(IK,JK)*B(JK)
   10 CONTINUE
   15 C(IK)=TS
      RETURN
      END
      SUBROUTINE VECA (A,B,WS,NX)
      DIMENSION A(NX),B(NX)
      WS=0.
      DO 10 I=1,NX
      WS=A(I)*B(I)+WS
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FUBRI (A,B,C,LS,LM)
      DIMENSION A(LS,LM),B(LS,LM),C(LS,LM)
      DO 8 I=1,LS
      DO 8 J=1,LM
      C(I,J)=A(I,J)-B(I,J)
    8 CONTINUE
      RETURN
      END
      SUBROUTINE ROT1 (A,TH1,B)
      DIMENSION A(3),B(3)
      B(1)=A(1)*COS(TH1)-A(2)*SIN(TH1)
      B(2)=A(1)*SIN(TH1)+A(2)*COS(TH1)
      B(3)=A(3)
      RETURN
      END
      SUBROUTINE ROT2 (A,TH2,B)
      DIMENSION A(3),B(3)
      B(1)=A(1)
      B(2)=A(2)*COS(TH2)-A(3)*SIN(TH2)
      B(3)=A(2)*SIN(TH2)+A(3)*COS(TH2)
      RETURN
      END
      SUBROUTINE ROT3 (A,TH3,B)
      DIMENSION A(3),B(3)
      B(1)=A(1)
      B(2)=A(2)*COS(TH3)-A(3)*SIN(TH3)
      B(3)=A(2)*SIN(TH3)+A(3)*COS(TH3)
      RETURN
      END

READY.
```

BIBLIOGRAPHY

1. William R. Ferrel, Thomas B. Sheridon., "Supervisory Control of Remote Manipulation." IEEE Spectrum, pp. 81-88, October 1967.

2. Cvetkovic, V., Vukobratovic M. "Computer-Oriented Algorithm for Modeling Active Spatial Mechanisms for Robotics Applications." IEEE Trans. Syst., Mon. Cybern., Vol SMC-12, pp. 838-847, 1982.

3. Hollerbach, J.M, "A Recursive Lagrangian Formulation of Manipulator Dynamics and A comparative Study of Dynamics Formulation Complexity." IEEE Trans. Syst., Mon. Cybern., Vol SMC-10, pp. 730-736, 1980.

4. Paul, R.P.C, "Modelling, Trajectory Calculation, and Servoing of a Computer Controlled Arm.", Stanford Artificial Intelligence Laboratory Memo, No. 177-1972

5. Luh, J., Walter, M.W., and Paul, R.P.C. "On-Line Computational Scheme for Mechanical Manipulators", J. Dynamic. Syst. Measurement, Contr., Vol. 102, pp. 69-72, 1980.

6. Whitney D.E., "Resolved Motion Rate Control of Manipulators and Human Prostheses", IEEE Trans. Vol-MMS-10 pp. 47-53, 1969.

7. Whitney D.E, "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators" , J. Dynamic. Syst., Measurement, Contr. pp. 303-309, December,1972.

8. Saridis G.n., Stephanou H.E "A Hierarchical Approach to the Control of a Prosthetic Arm.", IEEE Trans, Vol. SMC-7 pp. 407-420, 1977.

9. Hemami H., Camana P. "Nonlinear Feedback in Simple Locomotion systems". IEEE Trans. an Automatic Control, pp. 855-860, December, 1976.

10. Saridis G.N. Lee C.S.G. "An Approximation Theory of Optimal Control for Trainable Manipulators", IEEE Trans.Vol. SMC-9., pp. 152-159, 1979.

11. Rekasius Z.V., "Suboptimal design of Intentionally Nonlinear Controllers.", IEEE TRans. on Automatic Control, pp. 380-386, October 1964.

12. Landau I.D., "A Survey of Model Reference Adaptive Techniques-Theory and applications." Automatica, Vol-10 pp. 353-379, 1979.

13. Landau I.D. "Model Reference Adaptive Systems-A Survey (MRAS) - What is Possible and Why?"., J. Dynamic Syst. Measurement and Contr., pp. 119-132, June 1972.

14. Albus J.S., "A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller(CMAC) " J.Dynamic Syst., Measurement and Contr.,pp, 220-227, September 1975.

15. Albus. J.S.,"Data Storage in the Cerebellar Model Articulation Controller (CMAC) ", J. Dynamic Syst, Measurement and Contr., pp. 228-233, September 1975.

16. Young D.K.,"Controller Design for a Manipulator Using Theory of Variable Structure Systems.", IEEE Trans. Vol. SMC-8., pp.101-109, 1978.

17. Vukobratovic M., Potkonjak V. "Contribution of Forming of Computer Methods for Automatic Modelling of Spatial Mechanisms Motions", Mechanism and Machine Theory Vol.14, pp.179-188, 1979.

18. Lee G.C.S.,"Robot Arm Kinematics, Dynamics and Control", Computer, pp.62-80,1982.

19. Vukobratovic M., Potkonjak V."Two New Methods for Computer Forming of Dynamic Equations of Active Mechanisms" , Mechanism and Machine Theory, Vol.14, pp.189-200, 1979.

20. Dubowsky S., DesForges D.T.,"The Application of Model Referenced Adaptive Control to Robotic Manipulators" J. Dynamic Syst. Measurement and Contr. Vol.101 pp.193-200, 1979.

21. Vukobratovic M., "Computer Method for Dynamic Model Construction of Active Articulated Mechanisms Using Kinetostatic Approach" Mechanism and Machine Theory. Vol-13 pp.19-39, 1978.

22. Landau I.D."Adaptive Control: The Model Reference Approach", Marcel Dekker, New York, 1979.

23. Popov V.M. "Hyperstability of Control Systems", Editura Academiei Bucuresti and Springer-Verlag, Berlin, Heidelberg. New York, 1973

24. Lefschetz S. "Stability of Nonlinear Control Systems" Mathematics In Science and Engineering, Vol.13. Academic Press, 1965.

25. Horowitz R., Tomizuka M., "An Adaptive Control Scheme for Mechanical Manipulators - Compensation of Nonlinearity and Decoupling Control". Dynamic Syst. and Control Division of ASME, July.1980.

26. Bass R.W., Webber R.F."Optimal Nonlinear Feedback Control Derived from Quadratic and Higher-Order Performance Criteria". IEEE Trans. on Automatic Control, Vol. AC-11, 1966

27. Kahn. M.E. "The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains". J.Dynamic Syst. Measurement and Contr. pp.164-172, September 1971

28. Rissanen J.J. "Performance Deterioration of Optimum Systems". IEEE an Automatic Control, Vol.AC-11 pp.530-532, July 1966.

29. Mageirou E.F., "On the Comparison of Optimal Control Systems", IEEE on Automatic Control, pp.877-878, December, 1976.

30. Bailey F.N., Laub A.J. "Suboptimality Bounds and Stability in the Control of Nonlinear Dynamic Systems" IEEE on Automatic Control, pp.396-399, June 1976.

31. Hartenberg R.S., Denavit J., "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices" J.of Applied Mechanics, pp.215-221, June, 1955.

32. Vukobratovic M. "Computer Method for Dynamic Model Construction of Active Articulated Mechanisms by using Euler's Angles". Mechanism and Machine Theory, 1978.

33. Vukobratovic M. "Synthesis of Functional Artificial Motion " Mechanism and Machine Theory, Vol.13 pp.41-56, 1978.

34. Boni G., Tomovic. R., "An Adaptive Artificial Hand", IRE Trans. on Automatic Control. pp.3-10, April 1962.

35. Starr S.P., "A Comparison of Control Modes for Time-Delayed Remote Manipulation" IEEE Trans. Vol SMC-9, pp.241-246, 1979.

36. Okada T., "Object Handling System for Manual Industry" IEEE. Trans. Vol.SMC-9, pp.79-89, 1979.

37. Paul R. "Manipulator Cartesian Path Control", IEEE Trans Vol.SMC-9, pp.702-711, 1979.

38. Luh. J.Y.S, Walker M.W., Paul R., "Resolved-Acceleration Control of Mechanical Manipulators", IEEE Trans. on Automatic Control, Vol.AC-25, pp.468-474, 1980.

39. Freund E. "Nonlinear Control with Hierarchy For Coordinated Operation of Robots". Electrical Engineering Dep. of Unv. of Southern California., Los Angeles, California.

40. Kohonen T., Associative Memory-A system Theoretical Approach., Springer-Verlag Berlin Heidelberg New York, 1977.