AN ABSTRACT OF THE THESIS OF

YOUNUS VORA for the degree of Master of Science in Industrial Engineering presented on July 18, 1984

Title: DYNAMIC SIMULATION OF ROBOT MANIPULATORS

Abstract approved: _____

Dr. Eugene F. Fichter

A program has been written to investigate the dynamics of robot manipulators during task execution. The program simulates robot motion along a path specified by the user. A smooth trajectory is generated by interpolation in joint space. Forces and torques on actuators are calculated at intermediate points, using a recursive Lagrangian formulation of manipulator dynamics.

Plots and tables of position, velocity, acceleration and generalized force, versus time may be output to evaluate various kinematic arrangements, and to improve the trajectory plan for a known application. A base has been provided for mechanical design of links and for actuator sizing. Manipulator configuration drawn at each specified position may be studied to analyse the workspace and interaction with other elements.

Calculations were verified by deriving inertia and gravity terms for a three degree of freedom articulated arm and comparing results. The methodology developed was applied to three six degree of freedom configurations, which are included as library solutions.

DYNAMIC SIMULATION OF ROBOT MANIPULATORS

by

Younus Vora

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed July 18, 1984

Commencement June, 1985

APPROVED:

Redacted for Privacy

Assistant Professor of Industrial and General Engineering in charge of major

Redacted for Privacy

Head of Department of Industrial and General Engineering

Redacted for Privacy

Dean of Graduate School

Date thesis is presented July 18, 1984

Typed by the author

ACKNOWLEDGEMENTS

I would like to thank Dr Eugene Fichter for his help and encouragement in initiating this project and on numerous occasions during the course of this study. I am grateful to him for providing a research assistantship to support my work here.

I am thankful to Dr Jeffery Arthur in Statistics, who inspired my interest in simulation and modelling, and to Dr Kenneth Funk in Industrial Engineering, for his help and advice on problems concerning the PDP 11.

I owe much to my father and to my family for their support and encouragement.

TABLE OF CONTENTS

INTRODUCTION	1
REVIEW OF LITERATURE	5
2.1 Introduction	5
2.2 Kinematic and Dynamic Formulations	6
2.3 Trajectory Planning	9
2.4 Simulation	10
2.5 Scope of this Thesis	11
ROBOT ARM KINEMATICS AND DYNAMICS	12
3.1 Introduction	12
3.2 End Effector Location and Orientation	13
3.3 The Kinematic Solution	16
3.4 The Dynamic Solution	20
TRAJECTORY GENERATION AND WORKSPACE	28
4.1 Introduction	28
4.2 Task Specification	29
4.3 Trajectory Generation	30
4.4 Workspace Limitations	36
SIMULATION ALGORITHM AND PROGRAMMING	37
5.1 Introduction	37
5.2 Description of the Main Algorithm	41
5.3 Program Organization and Coding	45
MANIPULATOR SPECIFICATIONS FOR RUNNING THE SIMULATION	58
6.1 Introduction	58
6.2 Assumptions	60
6.3 Data Files	61
VERIFICATION OF RESULTS	69
7.1 Introduction	69
7.2 RRR Manipulator Solution	69
7.3 Task Specification	78
7.4 Results	80
SUMMARY AND CONCLUSIONS	94
BIBLIOGRAPHY	96
APPENDIX	100

LIST OF FIGURES

FIGURE		PAGE
3.1.	Link coordinate frames	15
3.2.	Notation for dynamics	21
4.1.	Trajectory generation	33
5.1.	Menu options	38
5.2.	Main Algorithm	39
5.3.	Computation of inertia matrices	40
5.4.	Subroutine calls	46
5.5.	File assignment	48
5.6.	Main menu	50
5.7.	Submenu selections	51
6.1.	Standard link	59
6.2.	Manipulator specification file	62
6.3.	Task specification file	65
6.4.	Segment of graphics data file	67
7.1.	Coordinate frames for RRR manipulator	71
7.2.	Kinematic solution	73
7.3.	Pseudo inertia matrices	76
7.4.	Trajectory specification	79
7.5.	End points of trajectory segments	82
7.6.	Joint position versus simulated time	84
7.7.	Joint velocity versus simulated time	85
7.8.	Joint acceleration versus simulated time	86
7.9.	Joint force/torque versus simulated time	87

A-1.	End points of trajectory segments	105
A-2.	Joint position versus simulated time	107
A-3.	Joint velocity versus simulated time	109
A-4.	Joint acceleration versus simulated time	111
A-5.	Joint force/torque versus simulated time	113
A-6.	End points of trajectory segments	117
A-7.	Joint position versus simulated time	119
A-8.	Joint velocity versus simulated time	121
A-9.	Joint acceleration versus simulated time	123
A-10.	Joint force/torque versus simulated time	125
A-11.	End points of trajectory segments	129
A-12.	Joint position versus simulated time	131
A-13.	Joint velocity versus simulated time	133
A-14.	Joint acceleration versus simulated time	135
A-15.	Joint force/torque versus simulated time	137

LIST OF TABLES

TABLE		PAGE
7.1.	Link parameters and dimensions	72
7.2.	Link properties	75
7.3.	Summary results	81
7.4.	Tabulated results (simulation)	88
7.5.	Tabulated results (equations 7.14)	91
A-1.	Link parameters and dimensions RRPRRR (Stanford) manipulator	103
A-2.	Summary results	104
A-3.	Link parameters and dimensions PPPRRR manipulator	115
A-4.	Summary results	116
A-5.	Link parameters and dimensions RRRRRR manipulator	127
A-6.	Summary results	128

DYNAMIC SIMULATION OF ROBOT MANIPULATORS

CHAPTER I

INTRODUCTION

Flexible automation offers a possible solution to the problems of lagging productivity and high labour costs that plague industry today. Robot Manipulators play an important role in Flexible Manufacturing Systems. In the past few years the robot industry has grown rapidly and intensive research efforts have been directed towards integration of robots in manufacturing.

Typically industrial robots have performed the tasks of pick and place, spot welding, spray painting and foundry operation. With the development of vision systems and other sensors, applications have been found in assembly and inspection tasks, and in a variety of manufacturing operations. In spite of intensive research, robots still perform in a limited field in industry [Luh,1983]. An accurate description of capabilities and limitations of a

robot is usually not available when particular applications are being studied. A better evaluation of the basic structure of the manipulator is required during the design process. Better task specifications are necessary in the selection of a design for a given application.

Robot manipulators consist of multiple degree of freedom kinematic chains, and their design and analysis must start from mechanical considerations. This portion of the design process includes Kinematics, Dynamics, Locomotion and Control. A kinematic solution can usually be obtained for most manipulators of practical interest. The dynamic solution uses knowledge of joint coordinates, velocity and acceleration, along with the physical characteristics of the manipulator to generate torques or forces which must be applied to execute the desired motion. The dynamic problem is often complex and difficult to solve in real time. Even when solution is possible, the cumulative effects of disturbances such as modeling and computational inaccuracies, frictional effects, etc., may cause errors in the actual traversed path. When feedback control is used, errors in the command sequence are measured and used to modify the inputs. However, gross modeling inaccuracies must be corrected at the design stage. It is usually not cost effective to build a prototype to check these. Simulation provides an excellent

tool for improvement of the basic kinematic arrangement and interactive design and structural detailing of links. Kinematic and dynamic formulations used for control may be verified. Trajectory planning for task execution and time and motion analysis may be performed.

This thesis is an attempt to provide a simulation package for the above purposes. The program is capable of evaluating some configurations by using built up solution libraries. User written subroutines may be included to analyse new configurations. A trajectory generation scheme is used to lead the manipulator smoothly through a series of user defined points which specify the task. Generalized forces or torques are calculated at discrete time intervals. To evaluate the feasibility of a control algorithm the frequency of these computations must correspond to the structural resonant frequency of the robot. Manipulator configuration is drawn at each specified point on the trajectory, so that the feasibility of the task sequence may be analysed. This can also be used for comparing different kinematic formulations. At the end of each simulation run, plots and tables of joint position, velocity, acceleration and forces/torques may be output. With this information, the menu options may be used to modify the physical characteristics of the robot, or change trajectory specifications to achieve a better

design and task implementation. Successive applications of this procedure can be used to improve the solution.

This report describes the algorithms and techniques used in programming and provides guidelines for using the simulation. It is structured as follows:

Chapter two presents a survey of current literature in robot kinematics and dynamics, and in simulation studies in this field.

Chapters three and four are a brief tutorial on manipulator analysis and trajectory planning. Existing techniques which have been used in this simulation are described.

Chapter five is a detailed description of the simulation algorithm and programming, and also contains instructions for user written subroutines. Data structures, program modules and utilities are described, and areas of possible expansion are identified. Chapter six describes the format for the data files and interactive input.

Chapter seven contains verification of results obtained from the simulation, using a three degree of freedom articulated arm. Chapter eight presents a summary of the work done and recommendations for expansion of the program. Examples using library solutions are appended.

CHAPTER II

REVIEW OF LITERATURE

2.1 INTRODUCTION

The development of robot manipulators may be traced over the past few decades. Work on "master slave" mechanical manipulator systems was started at the Argonne Laboratories in 1947. In the fifties, force sensing teleoperators and "man amplifiers" were developed [Sullivan, 1971]. These devices are not classified as robots since they cannot function independently of an operator. However these developments led directly to research on computer controlled arms in the sixties. In 1961, a computer controlled mechanical hand was developed at MIT and in the mid sixties vision and touch sensors were used to navigate a mobile platform at the Stanford Research Institute. The first industrial robot was demonstrated and found application during this period [Engelberger, 1980]. Work on automatic assembly started at the Draper Laboratories and in 1972 a water pump was assembled at Stanford University using both visual and force feedback.

A number of advances have since been made in the fields of control and sensor application, and in integrating robots into manufacturing systems. In 1983 FANUC Limited started production at a new servomotor plant in Japan using over a hundred robots for machine loading and assembly [Urbaniak, 1983].

2.2 KINEMATIC AND DYNAMIC FORMULATIONS

Most mathematical models for multiple degree of freedom kinematic chains use notation developed by Denavit and Hartenberg [1955]. The kinematic configuration of a series of links can be expressed by embedding a coordinate system in each link and expressing the relationship between these coordinate systems as homogeneous transformations. Solution for most manipulators of practical importance may be obtained by using a method based on this approach [Paul, 1981]. In general the solution for a kinematic chain may be obtained algebraically by using a high degree polynomial. Pieper [1969] presented the kinematic solution of a six degree of freedom kinematic chain where the last three joint axes intersect in a point. Paul [1981] presented an algebraic algorithm for solving the same class of manipulators. However multiple solutions are obtained and the selection of the best one must be based on

intuition. A recursive kinematic solution has also been presented [Milenkovic, 1983] but the solution algorithm requires more computation and is not gauranteed to converge.

Dynamic equations for multiple degree of freedom kinematic chains were first formulated by Uicker [1967] for a more general purpose problem. Equations for open loop kinematic chains were later formulated, based on this approach [Kahn and Roth, 1971]. Derivation of these equations is based on Lagrangian dynamics. The Uicker Kahn equations are extremely cumbersome to derive and computationally expensive and were found to be unsuitable for real time applications [Luh, 1980]. Various simplifications have been used to render the equations soluble in real time. One approach was to neglect centripetal and coriolis components of generalized Torques. However, present day manipulators are capable of attaining speeds at which this assumption is not justified, but acceptable results were obtained for special cases [Bejczy and Paul, 1981]. Another method was to tabulate the results for a series of points which may be encountered during trajectory execution. Solutions for the entire region under consideration may then be obtained by using an interpolation scheme [Raibert, 1977]. Although the equations have been parameterized over some of the

variables, this approach still requires enormous amounts of memory. Moreover a given set of solutions becomes useless if loading conditions change. Albus has taken this approach further by tabulating all possible points in a state space and storing data by an iterative learning process. [Albus, 1975].

A different approach to developing the dynamic equations was taken by using the Newton-Euler equations of motion [Luh, 1980]. The derivation results naturally in a recursive formulation which may be utilized to evaluate the generalized forces and torques from the base to the tip of the manipulator. It has been shown that the recursive Newtonian formulation yields a solution which can be implemented in real time. Hollerbach [1980] showed that the Uicker Kahn equations could be reformulated into a recursive relationship almost as efficient as the Newtonian formulation. In fact it has been demonstrated that the two formulations are equivalent [Silver, 1982]. Generalized D'Alembert's equations [Lee, 1983], and the principle of Virtual Work [Walker and Orin, 1982] have also been used to derive these equations. The recursive Lagrangian and the Newton-Euler approaches have, however, been identified as the best methods for calculating the dynamics of robot arms [Hollerbach, 1980].

2.3 TRAJECTORY PLANNING

The actual execution of a task moves each link of the manipulator from an initial position to a final position providing appropriate inputs to take care of changing velocities and loading conditions. Some trajectories may be inadmissable due to motion constraints and workspace limitations. Whitney [1972], presented a solution for coordinated rate and position control of manipulators. Paul [1979,1981] has developed algorithms for trajectory planning both in joint space and in Cartesian space. Joint motion is very efficient and coordinated but it is not along straight lines or along any other simple path. In Cartesian motion the path between end points is along straight lines but this scheme is computationally more expensive. Taylor's bounded deviation method [Taylor, 1979] is a recursive method for trajectory generation. This method computes the joint mid-points corresponding to a trajectory segment and compares it to the Cartesian midpoint. The trajectory is subdivided and midpoints are recalculated until deviation between the two is within bounds.

Collision free trajectory planning is now recieving considerable attention [Lozano Perez, 1979]. Incorporating dynamics into trajectory planning is an open research area.

2.4 SIMULATION

With rapid decrease in the cost of computing power and memory, digital simulation is becoming a powerful tool for design and evaluation in many engineering fields. In robotics, simulation and modeling are being used for computer aided design and task analysis, as well as for workspace design and economic evaluation. Both these functions are included in large scale simulations written at McDonnell Douglas [Shumaker, 1980]. Automated manufacturing cells are evaluated using Q GERT [Medieros and Sadowski, 1983], and a conceptual model for simulation and workplace analysis is presented in [Welch, 1983].

Available simulations for computer aided design and task analysis of robot manipulators are usually a set of special purpose programs, which may be accessed for particular configurations [Heginbotham, 1979]. Large data base systems have been used to provide readily accessible information on arm designs and commercially available subsystems [Warnecke, 1978,1981]. A computer graphics and simulation package for design of robot arms has been written in MACLISP [Soroka, 1982]. Links are considered to be generalized cones and a number of solution libraries may be used. A computer graphics simulation written at the University of Minnesota [Sjolund, 1983], provides a tool

for offline task planning. Task planning is facilitated by generation of "Constraint Maps" at any point, indicating all possible movements within the workspace from that point. These simulations do not investigate dynamic characteristics of robot arms. Dynamics, where included, makes use of simplifications [Liegois, 1980], or is restricted to special cases [Sata, 1981]. Kinematics and Elastodynamics may be analysed using a large simulation program in FORTRAN [Derby, 1980]. The program also animates trajectory execution using a vector refresh terminal. An interactive procedure for selection of manipulator characteristics using exact dynamics has recently been presented [Potkonjak, 1983].

2.5 SCOPE OF THIS STUDY

Most of the simulations cited above are implemented on large mainframe computers. Non standard features often make these difficult to implement on other systems. A need exists for a relatively general purpose simulation which would incorporate dynamic analysis during trajectory generation. Implementation of this program on a mini-computer (PDP 11/23) would make it available to users interested in designing for small applications.

CHAPTER III

ROBOT ARM KINEMATICS AND DYNAMICS

3.1 INTRODUCTION

Homogeneous transformation matrices are used to define a manipulator configuration uniquely. The direct kinematic problem is to find coordinate transformation matrices, which relate link coordinate frames to the base, or to any inertial coordinate system. These can be used to determine end effector position when values of the joint variables are known. Inverse kinematics determines joint variables, given end effector location and orientation in space. The task is often stated in terms of base coordinates and joint variables are used to drive the arm. An algorithm which may be used to derive the inverse kinematic solution is described in this chapter. Of the various methods of obtaining a dynamic solution, the recursive Lagrangian formulation has been adopted and is described here. Lagrangian mechanics offers a simple method of analysing

complex dynamic systems such as manipulators. Generalized coordinates used in this formulation offer some advantages in programming. Homogeneous transformation matrices have been retained because of their intuitive appeal, although they are wasteful of memory space and computation time.

3.1 END EFFECTOR LOCATION AND ORIENTATION

The position and orientation of a rigid body in space may be completely specified if the following information is available:

A position vector from a reference coordinate frame to a coordinate frame fixed in the body is specified.

Rotation angles which describe the orientation of the body coordinate system with respect to the base coordinate system are specified.

This information may be provided by using Cartesian coordinates and Roll, Pitch and Yaw angles. In the following discussion, end effector position will mean the (x,y,z) coordinates of a predefined point on the end effector. Orientation of the end effector will mean Roll, Pitch and Yaw angles of the end effector coordinate system specified in base coordinates. With this information, a transformation matrix may be formulated to represent the coordinate system of the end effector in base coordinates.

Let POS represent this transformation. If (ϕ, θ, ψ) are the orientation angles, and $(p_x, p_y, p_z, 1)$ is the homogeneous vector defining the end effector coordinate system in base coordinates, then POS may be written as:

$$POS = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.1)$$

where the first three columns are the direction cosines of the axes comprising the end effector coordinate system, and the last column vector gives the coordinates of the origin of this coordinate system. The elements of the first three columns may be written as:

$$\begin{aligned} n_x &= \cos\phi \cos\theta \\ n_y &= \sin\phi \cos\theta \\ n_z &= -\sin\theta \\ o_x &= \cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi \\ o_y &= \sin\phi \sin\theta \sin\psi + \cos\phi \cos\psi \\ o_z &= \cos\theta \sin\psi \\ a_x &= \cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi \\ a_y &= \sin\phi \sin\theta \cos\psi - \cos\phi \sin\psi \\ a_z &= \cos\theta \cos\psi \end{aligned} \qquad (3.2)$$

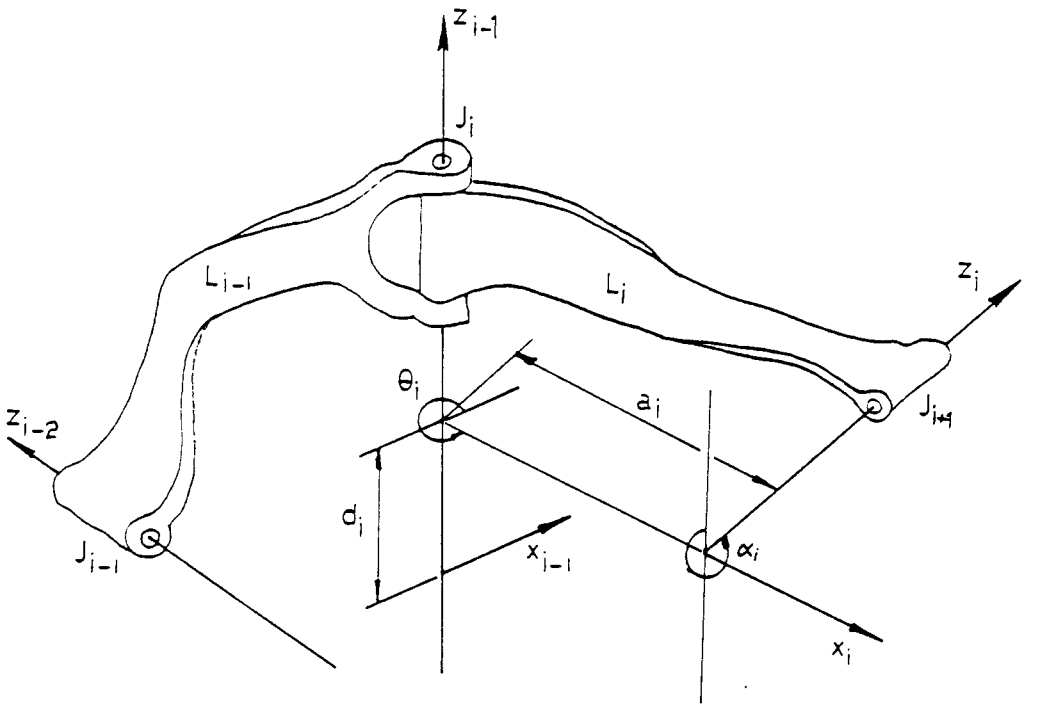


Figure 3.1. Link Coordinate Frames

3.2 THE KINEMATIC SOLUTION

Derivation of the inverse kinematic solution is described in some detail because this derivation must be performed by a user if a new configuration is to be analysed. The recursive dynamics also depends on the kinematic formulation used.

3.2.1 Link Coordinate Systems and Parameters

A coordinate system must be set up in each link according to the following rules:

1. The base of the manipulator is link zero and is not considered to be one of the n links. Link one is connected to the base link by joint one and there is no joint at the end of the nth link.

2. An orthogonal coordinate system is fixed in each link as follows:
   z_i is directed along the axis of joint i+1
   x_i lies along the common normal from z_{i-1} to z_i
   y_i is selected to complete the right handed system.

3. The relationship between two adjacent links is completely and uniquely defined by four link

parameters:

θ_i joint angle measured from x_{i-1} to x_i about z_{i-1}

d_i joint offset measured from x_{i-1} to x_i along z_{i-1}

α_i link twist angle measured from z_{i-1} to z_i about x_i

a_i the link length measured from z_{i-1} to z_i along x_i .

If joint i is revolute then θ_i will be the joint variable; if joint i is prismatic then d_i will be the joint variable. Multiple degree of freedom joints may be considered as single degree of freedom joints connected by links of zero length.

3.2.2 Specification of A Matrices and T Matrices

A homogeneous transformation relating the coordinate frame of link i to the coordinate frame of link i-1 is known as an A matrix. Most industrial robots employ special purpose attachments at the end of the last link. The task is specified in terms of some point on this attachment. A frame embedded in this point is defined in link n coordinate system, by the coordinate transformation 'TOOL'. The A matrix for each link may be written by making the following transformations:

$$A_i = Rot(z_{i-1}, \theta_i)Trn(0,0,d_i)Trn(a_i,0,0)Rot(x_i, \alpha_i) \quad (3.3)$$

where Rot(p,q) denotes a rotation of q angular units about axis p and Trn(a,b,c) denotes translation of a, b and c units along axes X, Y and Z respectively. Or in homogeneous matrix notation

$$A_i = \begin{bmatrix} \cos\theta_i & \sin\theta_i\cos\alpha_i & -\sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

Relationships between link i and link j

where

$i = 0,1,\ldots,n-1$

$j = i+1,\ldots,n$

are described by matrices of the form

$$^iT_j = A_{i+1} A_{i+2} \ldots A_j \quad (3.5)$$

The last link of the manipulator is described with respect to base coordinates

$$T_n = A_1 A_2 \ldots A_n \quad (3.6)$$

the leading superscript is omitted if it is zero.

3.2.3 Obtaining the Kinematic Solution

For this simulation the end effector position and orientation is specified as a six component vector:

$$(x, y, z, \phi, \theta, \psi)$$

Using the components of this vector the origin of coordinate system in link n may be described, with respect to base coordinates, by equation 3.1. For an n degree of freedom system the T_n matrix may be formulated from equation 3.6. The end effector coordinate system may be described in base coordinates by the transformation

$$T_{n+1} = T_n \, TOOL \qquad (3.7)$$

The inverse kinematic solution is obtained by comparing the matrices of equations 3.6 or 3.7, and 3.1. Inverting the A matrices and premultiplying equation 3.1 yields additional equations. For a six degree of freedom system six matrix equations can be obtained as follows:

$$
\begin{aligned}
A_1^{-1}\, T_6 &= {}^1T_6 \\
A_2^{-1}\, A_1^{-1}\, T_6 &= {}^2T_6 \\
&\cdot \\
&\cdot \\
A_5^{-1}\, A_4^{-1}\, A_3^{-1}\, A_2^{-1}\, A_1^{-1}\, T_6 &= {}^5T_6
\end{aligned}
\qquad (3.8)
$$

Using these a number of kinematic formulations are possible for most manipulators. Examples of kinematic solutions are given in chapter seven and in appendix A.

3.3 THE DYNAMIC SOLUTION

Manipulator dynamic equations relate the motion of a kinematic chain to torques applied at the actuators. In this case position, velocity, and acceleration of each link are known from the trajectory generation algorithm. Actuator torques and forces are required under these conditions. A recursive solution procedure based on Lagrangian dynamics is used. The equations can be derived by a reformulation of the Uicker Kahn equations.

3.3.1 Manipulator Dynamics Equation

A generalized coordinate system is used to designate the position of each joint.

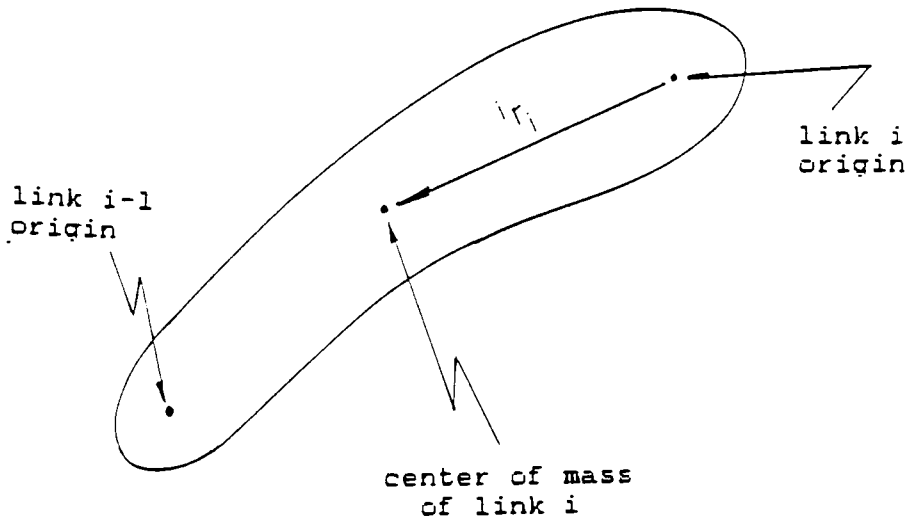


Figure 3.2. Notation for Dynamics

Let

q_i = position of joint i

$\dot{q}_i$ = velocity of center of mass of link i

$\ddot{q}_i$ = acceleration of center of mass of link i

${}^i r_i$ = vector from origin of link i to the center of mass of link i, (x ,y ,z)

m_i = mass of link i

J_i = inertia matrix for link i

F_i = generalized torque/force for link i

If joint i is revolute q_i ,$\dot{q}_i$ and $\ddot{q}_i$ will represent angular position, velocity, and acceleration and F_i will be the torque applied to rotate the link i. If joint i is prismatic q_i ,$\dot{q}_i$ and $\ddot{q}_i$ will be linear and F_i will be the force required for linear motion of link i. Deriving kinetic and potential energy for each link, formulating and differentiating the Lagrangian, the Uicker Kahn equations for generalized forces may be written as:

$$F_i = \sum_{j=i}^{n} \left[\sum_{k=1}^{j} \left(tr\left(\frac{\partial T_j}{\partial q_i} J_j \frac{\partial T_j^T}{\partial q_k} \right) \right) \ddot{q}_k + \sum_{k=1}^{j} \sum_{l=1}^{j} \left(tr\left(\frac{\partial T_j}{\partial q_i} J_j \frac{\partial^2 T_j^T}{\partial q_k \partial q_l} \right) \dot{q}_k \dot{q}_l \right) - m_j g^T \frac{\partial T_j}{\partial q_i} {}^j r_j \right] \quad (3.9)$$

where 'tr' is the trace operator. Inertial and velocity dependent components may be easily identified. The last

term in equation 3.9 may also be identified as the gravity loading term.

3.3.2 Recursive Lagrangian Dynamics

The generalized forces were written more compactly by Waters [1979] and this was used by Hollerbach [1980] to formulate a recursive procedure for the calculation of forces.

$$F_i = \sum_{j=i}^{n} \left[\left(\frac{\partial T_j}{\partial q_i} J_j \ddot{T}_j^T \right) - m_j g^T \frac{\partial T_j}{\partial q_i} {}^j r_j \right] \qquad (3.10)$$

where

$$\ddot{T}_j = \sum_{k=1}^{j} \frac{\partial T_j}{\partial q_k} \ddot{q}_k + \sum_{k=1}^{j} \sum_{l=1}^{j} \frac{\partial^2 T_j}{\partial q_k \partial q_l} \dot{q}_k \dot{q}_l \qquad (3.11)$$

Futhermore since

$$T_j = T_i \, {}^i T_j \qquad (3.12)$$

equation 3.10 may be written as

$$F_i = \mathrm{tr} \left(\frac{\partial T_i}{\partial q_i} D_i \right) - g^T \frac{\partial T_i}{\partial q_i} c_i \qquad (3.13)$$

where

$$D_i = \sum_{j=i}^{n} {}^{i}T_j \, J_j \, \ddot{T}_j^{T}$$

$$= J_i \ddot{T}_i^{T} + A_{i+1} D_{i+1} \qquad (3.14)$$

and

$$c_i = \sum_{j=i}^{n} m_j \, {}^{i}T_j \, {}^{j}r_j$$

$$= m_i \, {}^{i}r_i + A_{i+1} c_{i+1} \qquad (3.15)$$

3.3.3 Recursive Procedure for Torques/Forces

The recursive procedure for obtaining generalized forces given position, velocity and acceleration may now be stated as follows:

1. Calculate all A matrices at current values of the joint variables.

2. Calculate T,$\dot{T}$ and $\ddot{T}$ matrices using the equations

$$T_i = T_{i-1} A_i \qquad (3.16)$$

$$\dot{T}_i = \dot{T}_{i-1} A_i + T_{i-1} (\partial A_i / \partial q_i) \dot{q}_i \qquad (3.17)$$

$$\ddot{T}_i = \ddot{T}_{i-1} A_i + 2\dot{T}_{i-1}(\partial A_i / \partial q_i)\dot{q}_i + T_{i-1}(\partial^2 A_i / \partial q_i^2)\dot{q}_i^2 + T_{i-1}(\partial A_i / \partial q_i)\ddot{q}_i \qquad (3.18)$$

This recursion starts from link 1 and proceeds to link n. $\dot{T}_0$ and $\ddot{T}_0$ are both [0] since T_0 is defined as the identity matrix.

3. Calculate all D_i and c_i terms proceeding from link i+1 to link 1 using equations 3.14 and 3.15.

4. The generalized torques/forces may finally calculated using equation 3.13.

3.3.4 Differentiating A Matrices

The derivative of A_i with respect to q_i is obtained by using a matrix operator Q defined by

$$\partial A_i(q_i)/\partial q_i = Q\ A_i \qquad (3.19)$$

where Q takes the form Q_θ or Q_d depending on whether the joint is revolute or prismatic [Uicker, 1967].

$$Q_d = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Q_\theta = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

3.3.5 The Inertia Matrix

Use of homogeneous transformation matrices in the derivation of kinetic energy leads to an a matrix of

integrals known as the Pseudo Inertia matrix [Paul, 1981]. This matrix is specified for each link with respect to the coordinate system embedded in that link, and remains constant once the dimensions of the link are given. Using conventional notation for moments of inertia, the inertia matrix may be written as:

$$J_i = \begin{bmatrix} J_{11} & J_{12} & J_{13} & J_{14} \\ J_{21} & J_{22} & J_{23} & J_{24} \\ J_{31} & J_{32} & J_{33} & J_{34} \\ J_{41} & J_{42} & J_{43} & J_{44} \end{bmatrix} \qquad (3.20)$$

where

$$J_{11} = (-I_{xx_i} + I_{yy_i} + I_{zz_i})/2$$

$$J_{12} = I_{xy_i}$$

$$J_{13} = I_{xz_i}$$

$$J_{14} = m_i \bar{x}_i$$

$$J_{21} = I_{xy_i}$$

$$J_{22} = (I_{xx_i} - I_{yy_i} + I_{zz_i})/2$$

$$J_{23} = I_{yz_i}$$

$$J_{24} = m_i \bar{y}_i$$

$$J_{31} = I_{xz_i}$$

$$J_{32} = I_{yz_i}$$

$$J_{34} = m_i \bar{z}_i$$

$$J_{41} = m_i \bar{x}_i$$

$$J_{42} = m_i \bar{y}_i$$

$$J_{43} = m_i \bar{z}_i$$

$$J_{44} = m_i$$

CHAPTER IV

TRAJECTORY GENERATION AND WORKSPACE

4.1 INTRODUCTION

To perform a given task, the end effector of the manipulator must be constrained to follow a path, defined by a series of points, within acceptable error limits. Trajectory generation converts a description of the desired motion into a time sequence of intermediate configurations and velocity and acceleration information at these configurations. The outputs of trajectory generation dictate the actual motion of the arm and constitute the input to a feedback control system. Incorporating dynamics into trajectory generation involves calculation of torques and forces needed at the actuators to execute the trajectory.

4.2 TASK SPECIFICATION

A task may be specified as a series of end effector locations and orientations in terms of the base, or a world coordinate system. Task execution requires the end effector coordinate system to pass through these points with given orientation. If POS describes the end effector coordinate frame in base coordinates then it has been shown that

$$POS = T_n \, TOOL \qquad (4.1)$$

To drive the arm through its trajectory, POS transformations are formulated from Cartesian and Roll, Pitch, Yaw (R,P,Y) descriptions of the end points of the trajectory. The required number of intermediate positions may then be found by interpolation. The obvious advantage of using cartesian coordinates is that the trajectory can be easily visualized and the end effector can be kept clear of obstacles. In manufacturing, end points may be a series of pick and place positions or points defining a contour, as in arc welding or spray painting applications. Operations may have to be performed at some of these points. In this simulation, no provision is made for stopping at intermediate points. A task which requires such stops may, however, be analysed piecewise.

4.3 TRAJECTORY GENERATION

There are many ways of moving a manipulator from one position to another. Every system must provide continuity of position, velocity and acceleration. The most natural controls for a robot arm are actuator torques, computation of which requires a study of dynamics during trajectory generation. Torques generated at the actuators must not exceed design values. In real time control systems this information must be available at the joint servo rate. Although it is easy to specify end points in Cartesian coordinates, generation of intermedate positions may present some problems. A collision free path for the end effector does not guarantee that movements of other links will also be collision free. It must be ensured that joint coordinates corresponding to intermediate points do not violate physical limits. If intermediate positions are obtained by interpolating in joint space, the feasibility of a move under motion constraints may be studied. This simulation generates the trajectory by using a quartic polynomial to interpolate between end points of a trajectory. The end points are first transformed into their respective joint coordinates and interpolation is carried out in joint space [Paul, 1981].

4.3.1 Trajectory Generation in Joint Space

For a given initial and final configuration, position may be defined as an appropriate function of time, under given boundary conditions.

Let

P = current position of end effector (x,y,z,ϕ,θ,ψ)

PA = position at t=-TACC

P0 = starting position of current trajectory segment

P1 = end point of current trajectory segment

P2 = end point of next trajectory segment

Q = current joint coordinates

QA = joint coordinates at t=-TACC

Q0 = joint coordinates corresponding to P0

Q1 = joint coordinates corresponding to P1

TACC = time to accelerate from rest to maximum velocity

Vmax = maximum velocity of link i

t = current time

T1 = time to travel from P1 to P2

Consider figure 4.1; at time t=-TACC the end effector is at point P=PA and approaching the end of the current trajectory segment. A transition must be made to the next trajectory segment defined by the end points P1 and P2. The joint coordinates corresponding to P1 and P2 are available at this time. Using the boundary conditions of position, velocity and acceleration at both ends of the transition a polynomial q(t) defining joint position at time t can be found which interpolates smoothly in joint space. Because of the symmetry of the transition Paul[1981] uses only a quartic.

$$q(t) = a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0 \qquad (4.2)$$

To determine the values of the coefficients the following boundary conditions are available:

1. $q(t) = Q1$ at $t = 0$

2. $q(t) = Q2$ at $t = T1$

3. $\dot{q}(t) = (Q1(.)-QA(.))/T1$ at $t = -TACC$

4. $\dot{q}(t) = (Q2(.)-Q1(.))/T2$ at $t = TACC$

5. $\ddot{q}(t) = 0$ at $t = -TACC$

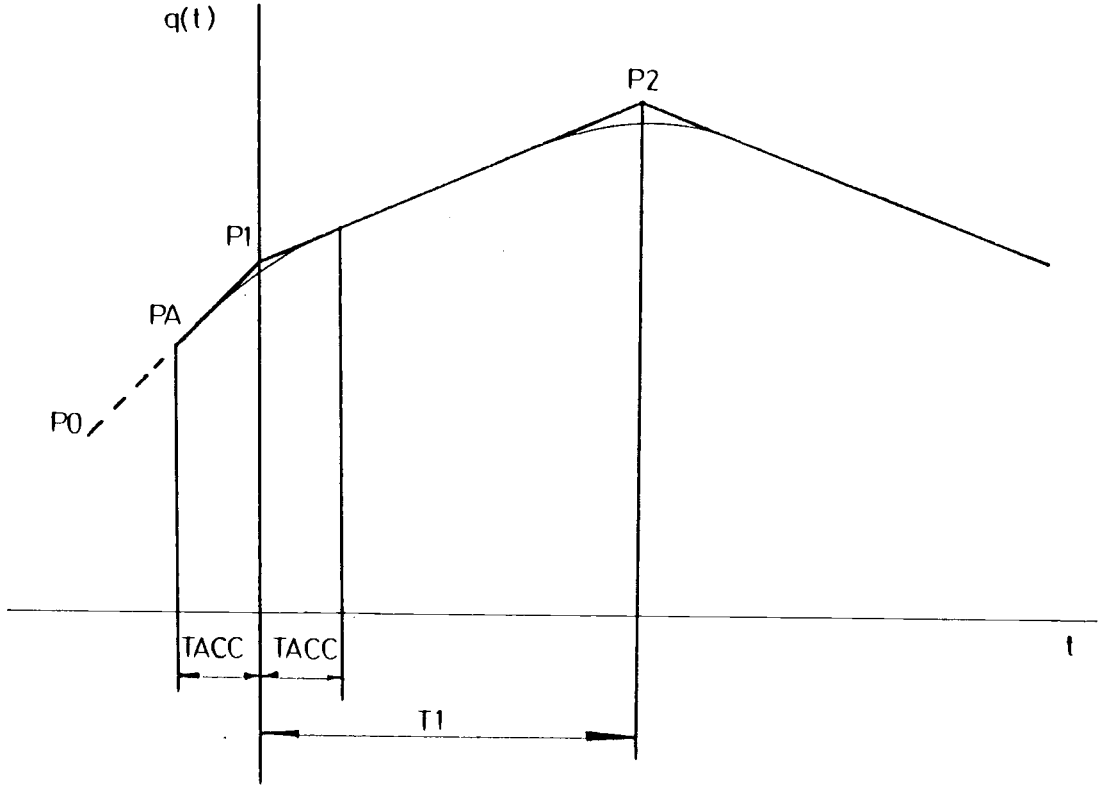


Figure 4.1. Trajectory Generation

6. $\ddot{q}(t) = 0$ at $t = TACC$

Where the dot in parentheses indicates that each joint is considered individually. Applying the boundary conditions, the equations used to interpolate for joint position, velocity and acceleration during the transition period are: [Paul, 1981]

$$q = [(\Delta Q2(TACC/T1)+\Delta Q1)(2-h)h - 2\Delta Q1]h + Q1 + \Delta Q1 \qquad (4.3)$$

$$\dot{q} = [(\Delta Q2(TACC/T1)+\Delta Q1)(1.5-h)2h - \Delta Q1]/TACC \qquad (4.4)$$

$$q = [(\Delta Q2(TACC/T1) + \Delta Q1)(1-h)3h]/TACC \qquad (4.5)$$

where

$\Delta Q1 = Q1 - QA$

$\Delta Q2 = Q2 - Q1$

$h = (t+TACC)/(2TACC)$

Once the transition period is over i.e. when t is greater than or equal to TACC, the joint position, velocity and acceleration are calculated from the equations below:

$$q = \Delta Q2h + Q1 \qquad (4.6)$$

$$\dot{q} = \Delta Q2/T1 \qquad (4.7)$$

$$\ddot{q} = 0 \qquad (4.8)$$

where h = t/T1

For this simulation a fixed acceleration time has been assumed for all joints. However the value is treated as a parameter which may be changed at the beginning of a simulation run.

Time to travel from P1 to P2 is calculated when the end effector is at the point PA. At least one of the actuators travels at maximum velocity. The time to traverse a trajectory segment must be at least 2TACC to allow for acceleration from -Vmax to Vmax. Thus T1 is calculated from the equation

$$T1 = \max\{ABS[\ Q2(.)-Q1(.)]/Vmax(.)\ ,\ 2TACC\} \qquad (4.9)$$

3.2.4 Other Functions for Trajectory Generation

In principle q(t) may be any continous function satisfying the boundary conditions. A number of functions

sine on linear ramp, etc. [Brady, 1982]. It may also be noted that a polynomial trajectory is probably not an ideal choice. The reason for this is that a polynomial of degree n crosses an arbitrary straight line n times. This may cause path deviations, and checking for workspace violations may prove extremely cumbersome. This problem has not been considered in the course of this study.

4.4 WORKSPACE LIMITATIONS

A manipulator consists of links of finite length and the mechanical structure presents a number of constraints. There is a limited workspace available. Trajectory end points which are out of reach must be identified and modified before simulation runs. Although obstacle avoidance in trajectory planning is beyond the scope of this thesis, a cursory examination of the points defining the trajectory is made to ensure that the specified task does not include positions which are out of reach. Only the dimensions of the first three links are used to determine the work envelope. This will give satisfactory results because all manipulators to be considered have the last three joint axes intersecting in a point.

CHAPTER V

SIMULATION ALGORITHM AND PROGRAMMING

5.1 INTRODUCTION

This simulation has been implemented on a PDP 11/23 minicomputer, using an RSX-11M BL26 operating system. The program is written in FORTRAN IV. PDP FORTRAN has some non-standard features, and some library functions provided on the system have been used. An attempt has been made to keep these at a minimum. The program consists of about three thousand lines of code divided into modules dealing with different aspects of the simulation. Due to memory limitations on the PDP 11/23 the program modules are overlaid. Memory requirements for running the program are thus kept within allowable user limits.

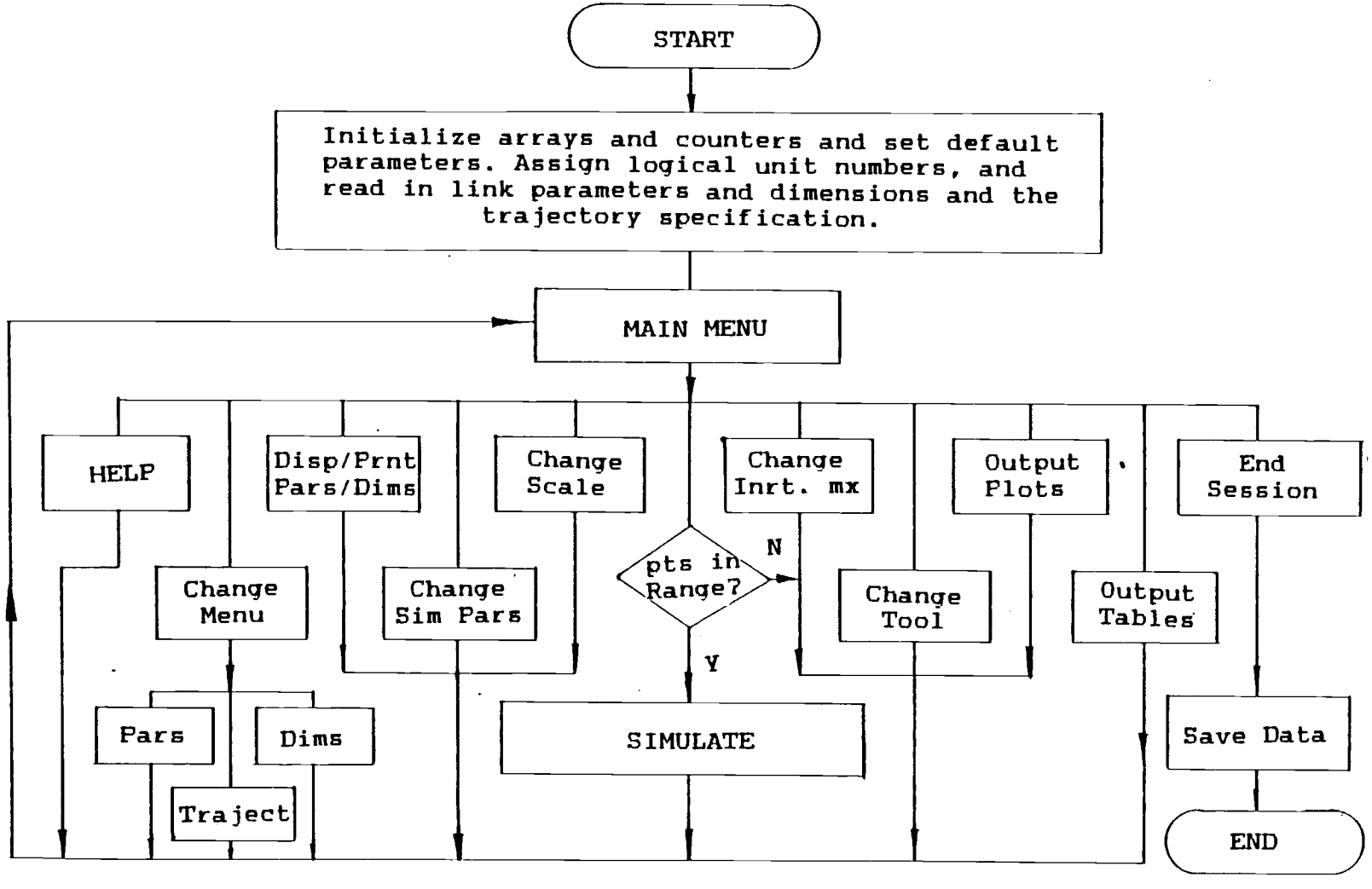


Figure 5.1. Menu Options

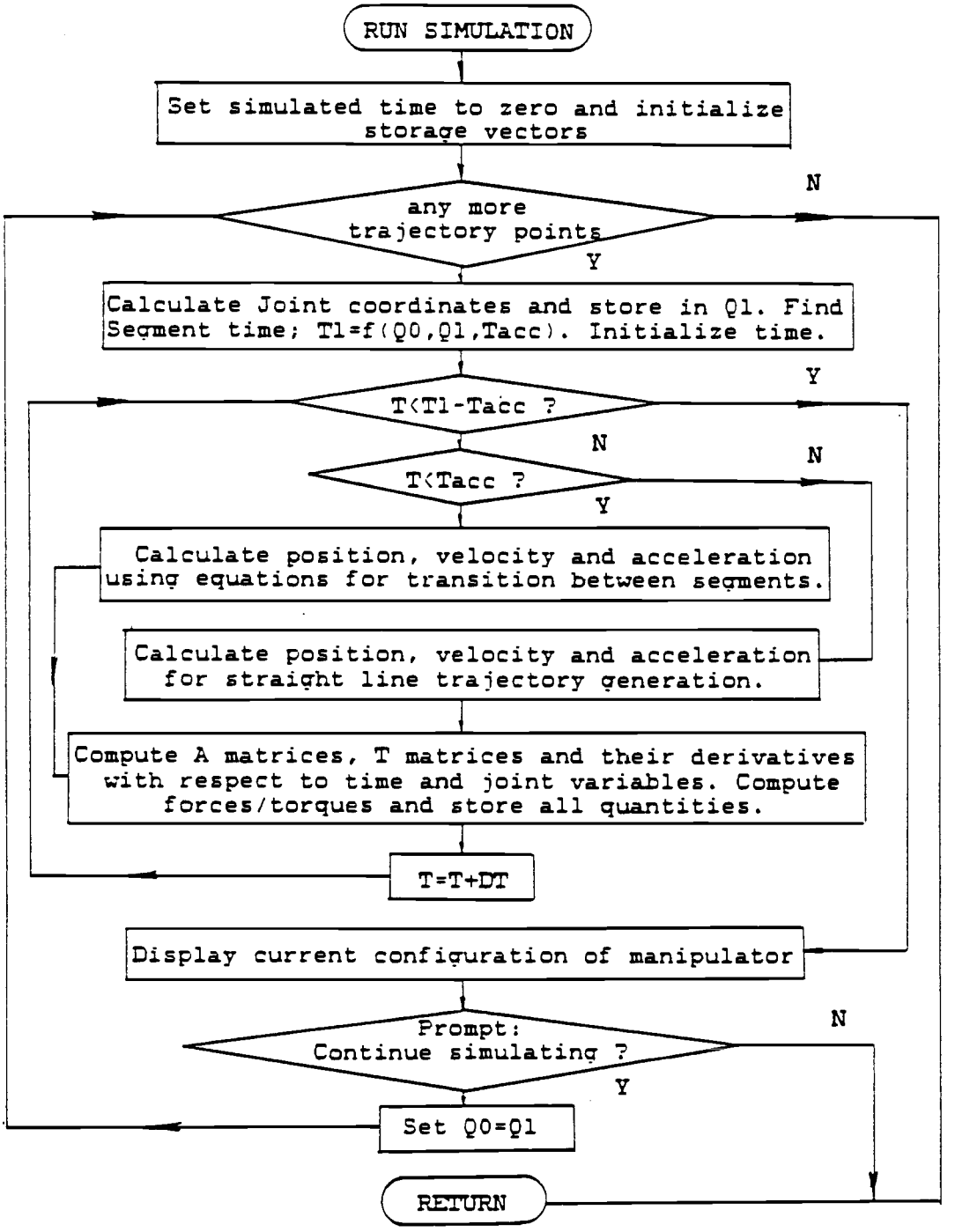


Figure 5.2. Main Algorithm

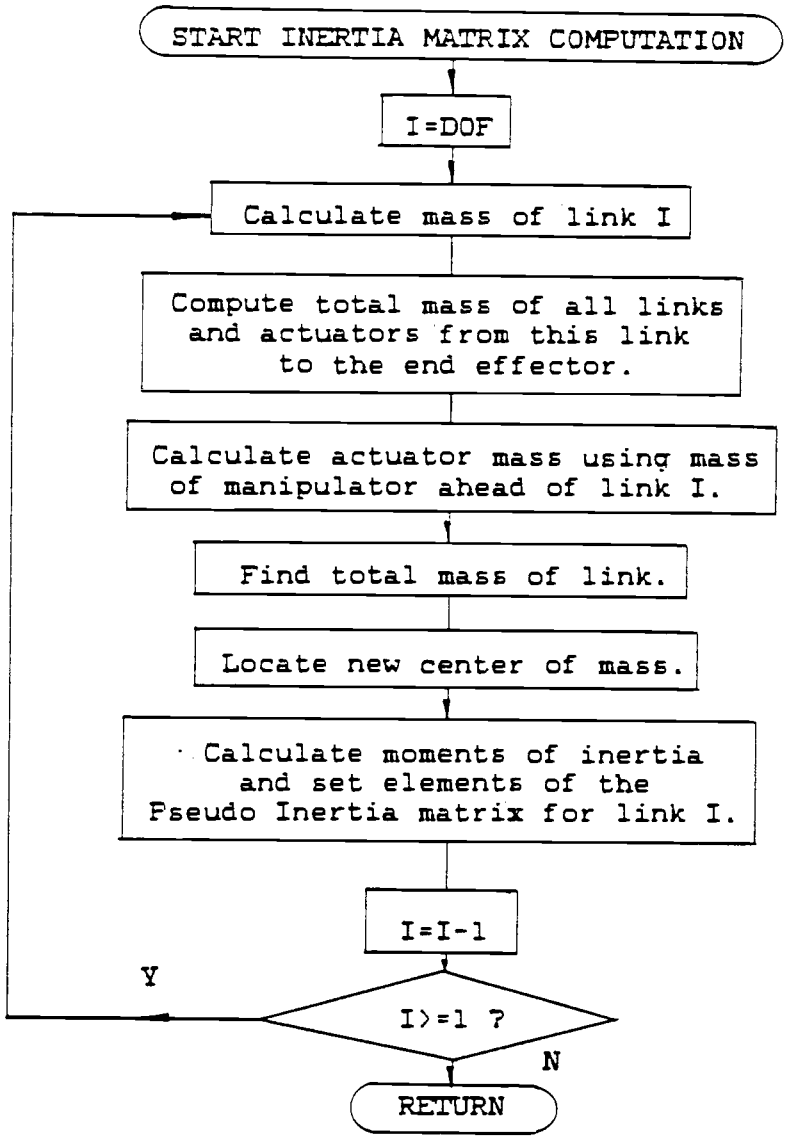


Figure 5.3. Computation of Inertia Matrices

4.2 DESCRIPTION OF THE MAIN ALGORITHM

In this section the main algorithm and flow of control through the simulation are described. The main menu and input routines are not described in detail. There are eleven branches which may be selected on the main menu. All branches appearing after the 'SIMULATE' option require at least one run to be made. Only the main branch which contains the simulation procedure will be described in this section. Individual branches of the menu are discussed in later sections. The steps of the algorithm are :

1. Start the simulation by calling up the main program. (The user does this by typing in '@START'). Set up environmental parameters for input and display. Initialize display file, and storage arrays and counters. Assign logical unit numbers to hardcopy devices and to data files according to the type of manipulator selected.

2. Read in link parameters, link dimensions and end points of the trajectory segments from data files. (Data files already exist for library manipulator types. If the user is analysing a new manipulator type and does not want to write up data files, he may input data interactively by making appropriate

selections on the menus. This data will be saved, if desired, at the end of the program. However the file containing information for graphic displays must be written before the program is started. Formats for writing data files are given in section 6.4).

3. Display main menu and allow user to make changes in data. (These changes affect only the storage arrays and not the data files. New data may be entered into data files only at the end of the program. Twelve options are available on the main menu which are described in the section on menu subprograms).
   IF option 9 is selected go on to step 4,
   ELSE IF option 10 or 11 is selected go to step 17.
   ELSE IF option 12 is selected go to step 18.

4. On selection of the option to SIMULATE tranfer control to subroutine SIMULT and start trajectory generation.

5. Calculate Joint coordinates for the first point on the trajectory and store in array Q0. (This involves calling up subroutines for the inverse kinematic solution).

6. IF there are no more destination points go to step 15,
   ELSE calculate joint coordinates for next destination point and store in array Q1.

7. Calculate time (T1) for execution of this trajectory segment.

8. Initialize simulated time (t) to zero or to -TACC depending on whether this is the beginning of the path or of a new segment.

9. IF T>T1-TACC go to step 14
   ELSE IF T<TACC calculate values for joint positions, velocity and acceleration at time T according to the equations for transition between segment end points. (This routine utilizes a quartic polynomial to interpolate between joint positions).
   ELSE calculate the above quantities using equations for straight line trajectory between path segment transitions.

10. Calculate all A matrices and T matrices, and their derivatives with respect to time and with respect to joint coordinates at this instant.

11. Calculate generalized forces (torques).

12. Store quantities calculated in steps 10 and 12 in appropriate disc file.

13. Set t=t+DT and return to step 9 (note that the time increment DT may be specified by the user).

14. The manipulator is now approaching the end of a trajectory segment. Display current configuration. (When all displays and hardcopies are complete, the user has the option to continue simulating or return to the main menu for modifications).
    IF user wishes to continue go on to step 15
    ELSE return to step 3.

15. Set Q0=Q1 and return to step 6.

16. Display final configuration and return to main menu.

17. If options 10 or 11 are selected, output plots or tables of quantities calculated during simulation. (Desired quantities are output and control is returned to the main menu.

18. Before ending the program save new data in data files if desired by the user. (Modified data may thus be used again at another session to continue the analysis).

19. STOP.

5.3 PROGRAM ORGANIZATION AND CODING

FORTRAN code for the simulation has been divided into seven major modules which are stored as separate files. Each module deals with a different stage of the simulation and contains a number of subprograms. Because of memory limitations on the PDP 11/23 the modules are overlaid in physical memory The modules are organized according to the functional nature of the subroutines, i.e.:

1. Main program and input
2. Menu subprograms
3. Trajectory generation
4. Kinematic solution
5. Dynamic solution
6. Matrix manipulation
7. Graphic display generator

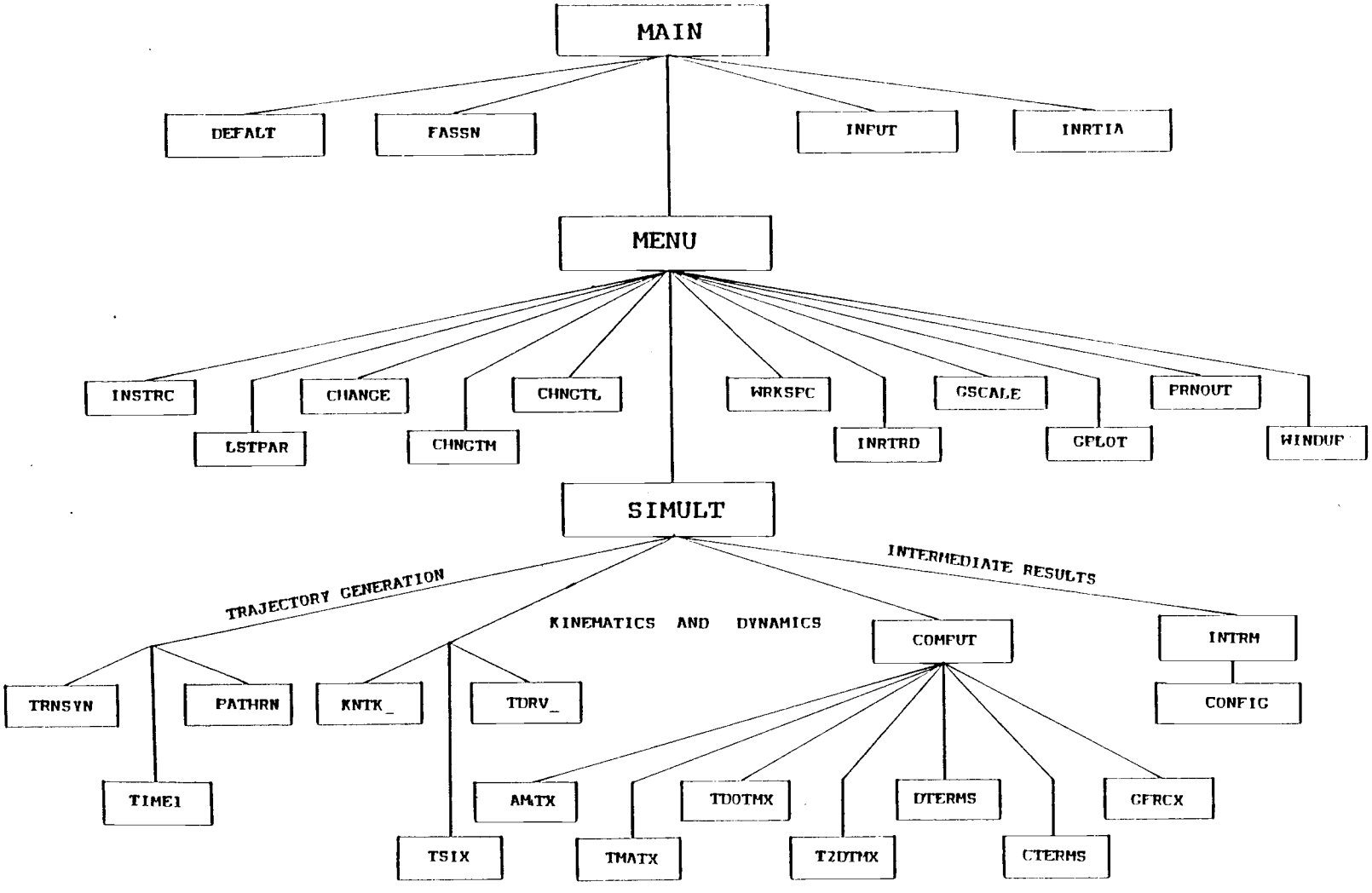


Figure 5.4. Subroutine Calls

These modules are partitioned into separate object code files. The RSX-11M operating system uses a task builder to combine object code modules into a task image which can be run by the user. Input to the task builder is an overlay description file. Procedure for running the task depends on whether the user is analysing one of the library solutions or investigating a new kinematic arrangement. In the former case no further additions have to be made. In case of a new configuration, subroutines containing the new kinematic solution, and derivatives of T matrices with respect to joint variables must be written and the task image rebuilt.

5.3.1 Main Program and Input

The MAIN program starts up the simulation and initializes all storage arrays and counters. Environmental Parameters for graphics are set by calling subroutine DEFALT. Default values for drawing scale, windowing, viewport and perspective viewpoint are assigned. Subroutine FASSN is called up to assign logical unit numbers to data files and allocate device numbers for input and output. On entering FASSN the user is presented with the information in figure 5.5. If a new kinematic arrangement is to be investigated, user written subroutines must be included in the task image being run. Data files for the new manipulator must be available. User written

DYNAMIC SIMULATION OF ROBOT MANIPULATORS

Instructions for Selecting Manipulator for analysis:

You may select a library manipulator type or specify a new configuration. In case you are simulating a new configuration, data files for link parameters and dimensions and trajectory end points must be available. Null data files may be used if data is to be input interactively. If you do not want to see graphics create a null graphics data file.

Data file names must be of the form 'NAME.DAT', where NAME is a valid FORTRAN variable name.

Press L for library configuration
N for new manipulator type
E to exit the program

and press RETURN :

Figure 5.5. File Assignment

subroutines and data file formats are explained in chapter VI. If a library configuration is selected the program can proceed without these. Subroutine INPUT reads in manipulator specifications and end points of trajectory segments which define the task. Subroutine INRTIA is called to compute inertia matrices and control is transferred to subroutine MENU.

5.3.2 Menu Subprograms

Control is retained by the main menu subprogram, MENU until the end of the session. Twelve menu options are available on the main menu (figure 5.6). The user may select any option but must return to the main menu before traveling down another branch. The first eight options on the menu allow the user to change values of various variables defining the manipulator and the task. These changes may be made before running the simulation or between simulation runs. Option nine allows the user to simulate and options ten and eleven output plots and tabulated results.

Selection of options one and two call up subroutines INSTRC and DSPLAY which display the required information. Option three outputs manipulator specifications to the line printer. Option four calls up subroutine CHANGE. Submenu selections are available at this point and the user may

DYNAMIC SIMULATION OF ROBOT MANIPULATORS

MAIN MENU

1. Instructions for running the program
2. Display parameters and dimensions
3. Print Parameters and dimensions
4. Change parameters, dimensions or trajectory points
5. Input or change Inertia Matrices
6. Change simulation parameters
7. Change Tool (End effector)
8. Change drawing scale factor
9. Simulate
10. Display Plots
11. Print Tables
12. End the program

Enter number of desired menu item and press RETURN :

Figure 5.6. Main Menu

```
CHANGE MENU

  1. Change link parameters
  2. Change link dimensions
  3. Change trajectory points
  4. Display current parameters and dimensions
  5. Return to main menu


MENU for parameter changes

  1. Change theta
  2. Change d
  3. Change alpha
  4. Change a
  5. No more changes


MENU for dimension changes

  1. Change link material (density)
  2. Change dimensions along X-axis
  3. Change dimensions along Y-axis
  4. Change dimensions along Z-axis
  5. No more changes


MENU for trajectory changes

Current coordinates for point   1
1    0.300   0.100  -0.100   0.000   90.000  60.000

  1. Change X-coordinate
  2. Change Y-coordinate
  3. Change Z-coordinate
  4. Change Roll (phi)
  5. Change Pitch (theta)
  6. Change Yaw (psi)
  7. Next destination point
  8. Add a new trajectory point
  9. No more changes
```

Figure 5.7. Submenu Selections

change dimensions, parameters and trajectory points. All changes affect current storage arrays. Any number of changes may be made before returning to the main menu. If the user does not wish to use simplified link structures he may input the inertia matrices by calling up option five. Any element of current inertia matrices may be changed or new values may be input for all elements. Simulation parameters include time increment and acceleration time from rest to maximum velocity. These may be changed by calling subroutine CHNGTM when option six is selected. Option seven changes drawing scale factors. If a new end effector is to be attached to the end of the sixth link, option eight allows the user to change the 'TOOL' transformation.

After all changes are complete and a simulation run is to be made subroutine SIMULT is called by selecting option nine. Trajectory generation routines are descibed in the next section. On completing trajectory execution control is once again returned to MENU and ouptput may be obtained in the form of tables or graphs by using option ten and eleven. Subroutine GPLOT reads the computed values stored on disk and plots desired graphs. The session may be ended by selecting option twelve. Subroutine WINDUP is called to store modified data, close all data files and exit.

5.3.3 Trajectory Generation

The algorithm used for trajectory execution has been described in chapter IV. Subroutine SIMULT is the main driver of the simulation because it keeps track of time and calls appropriate subroutines. When the simulation is started, execution time for the first trajectory segment is determined by calling subroutine TIME1. Simulated time is set to zero and incremented in steps of DT. At intermediate trajectory points, time is initialized to -TACC. During the transition period subroutine TRNSYN uses a quartic polynomial to interpolate in joint space. Subsequently PATHRN executes a linear trajectory. As position, velocity and acceleration at each time increment are obtained. Torques or forces are computed using the kinematic and dynamic solution routines and STOREV is called to store these values. When the end point of a trajectory segment is reached manipulator configuration is drawn on the screen by calling subroutines INTRM and CONFIG. A transition is then made to the next trajectory segment and the process is repeated until the end of the task when a return is made to the main menu.

5.3.3 Kinematic Solution Routines

Computation of the inverse kinematic solution and the derivatives of T matrices with respect to joint variables depends on manipulator configuration. Routines for library

manipulator types have been included in the module KINEM. Subroutines KNTK01 and TDRV01 are called for MTYP equal to one; KNTK02 and TDRV02 are called for MTYP equal to two and so on. This list can easily be expanded in the future. When a new manipulator type is to be investigated the kinematic solution must be coded in subroutine KNTKUS and the derivatives of T matrices in TDRVUS. These routines are saved in a file called KINEMUS.FTN and the task image rebuilt.

5.3.4 Dynamic Solution Routines

At each generated point on the path, joint positions, velocities and accelerations are calculated. Subroutine AMTX is called to compute A matices for current values of joint variables. TMTX, TD0TMX, and T2DTMX compute T matrices and their first and second time derivatives respectively. Derivatives of T matrices with respect to joint variables requires the appropriate routines already discussed. Subroutines DTERMS, CTERMS and GFRCX compute torques and forces at the current trajectory point.

5.3.5 Matrix Manipulation Routines

A number of matrix manipulation routines are required during computation of the inverse kinematic and dynamic solutions. One feature of matrix manipulations such as multiplication is that all transformation matrices are

declared as 6x4x4 arrays. Each 4x4 submatrix represents the transformation matrix for a particular link. Thus operations are performed by accessing the 4x4 submatrix of a given link from the 6x4x4 array, performing the required operations and storing the submatrix back into place. All matrix operations are performed with homogeneous transformation matrices and vectors.

5.3.6 Graphics

A small graphics module has also been written to generate a three dimensional image of manipulator configuration at trajectory segment end points and for plotting graphs. Graphic displays are obtained on a Tektronix 4052 graphics computer which has been interfaced to the PDP-11, and is used in terminal mode. The graphics package uses a display file to store points as they are generated from the simulation. A number of evironmental parameters and conditions may be set by the user. On start up, all of these are set to default values. A brief description of some of the features is given below

1. Normalized screen coordinates are used. These coordinates correspond to the screen size on the 4052 and provide a drawing space extending from 0-130 units on the x axis and from 0 to 100 units on the y-axis. When drawing the image on the

screen a final transformation must be made on points defined with respect to the base coordinate system of the robot, in order to redefine the points in screen coordinates. This transformation is stored in the matrix BSTFRM which is formulated at the beginning of the program by calling MKBTRN. A new transformation may be needed for each manipulator type.

2. Origin of the screen coordinate system may be translated to any location on the screen by using subroutine ORIGIN. All subsequent points are drawn relative to the new origin.

3. Drawing scale may be set as desired.

4. Two dimensional clipping is done on the image after a perspective transformation has been performed on the image. Subroutines CLFT,CRGT,CTOP and CBOT are called to clip at each boundary and VWTRAN is called to transform the points according to the specified viewport. Coordinates of clipped points are saved so that a subsequent point inside the viewport may be connected from the viewport boundary.

5. Before displaying the image, subroutine DENTR stores all transformed points in a display file alongwith the specified command option. When all points to be displayed have been written into the display file, subroutine DSPLAY is called to display the image on the screen.

CHAPTER VI

MANIPULATOR SPECIFICATIONS FOR RUNNING THE SIMULATION

6.1 INTRODUCTION

The kinematic arrangement and physical attributes of the manipulator which is being analysed must be specified in a particular format before a simulation run can begin. Geometry of the links must be defined so that the robot can be drawn on the screen. An attempt has been made to keep batch data input at a minimum and allow interactive input. Three data files must be created before the simulation can begin. All input specifications can be changed within the program by making appropriate selections on the menus. A number of assumptions concerning manipulator characteristics and structure have also been made to simplify input and computation.

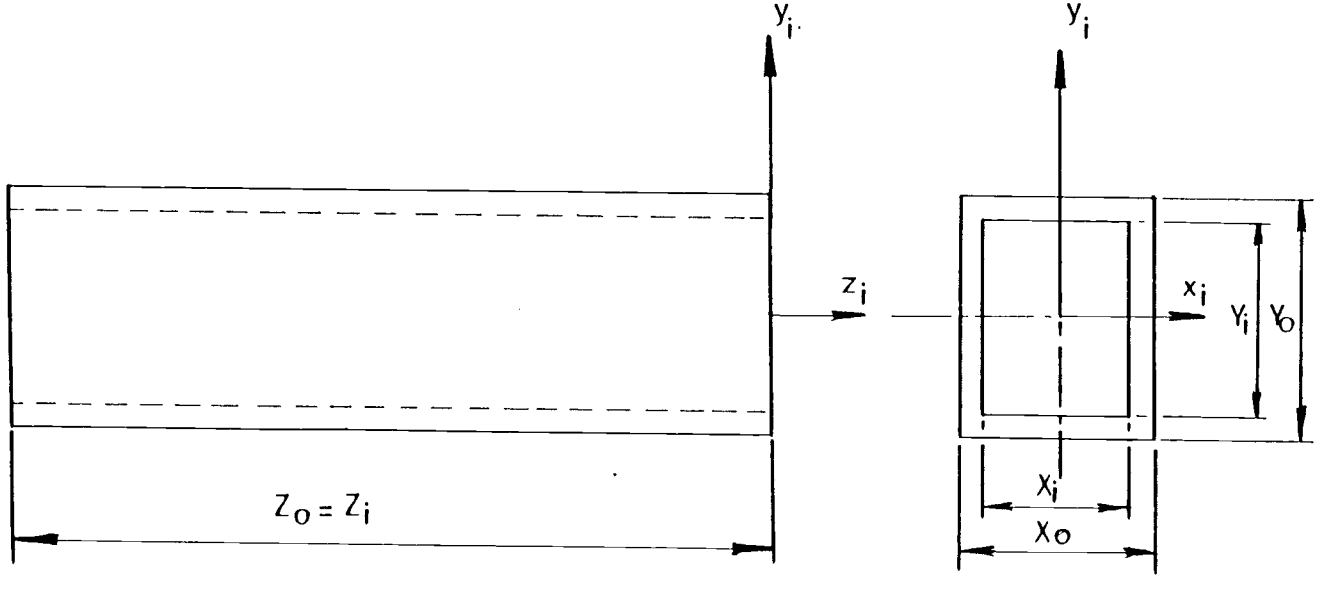


Figure 6.1. Standard Link

6.2 ASSUMPTIONS

Assumptions concerning the physical attributes of the links were made to allow easy computation procedures for moments of Inertia and centers of mass.

1. All links are assumed to be hollow bars of uniform rectangular section. Within this restriction the dimensions may be changed as desired.

2. The actuator for link i is assumed to be a point mass acting at the coordinate origin of link i-1. The actuators thus make no contribution to the moments of inertia of the link. The mass of the actuator for link i is assumed to be twenty five percent of the mass of all links and actuators ahead of link i, including the mass of link i, i.e.

$$ma_i = 0.25 \left(\sum_{j=i}^{n} m_j + \sum_{j=i+1}^{n} ma_j \right)$$

where ma is the mass of the actuator for link i and m is the mass of link i.

To facilitate analysis of complex links using exact physical attributes, an option is provided to input the inertia matrices, interactively or by using a data file.

The value of any element of the inertia matrices may also be changed between simulation runs.

6.3 DATA FILES

Data files are set up as permanent files in the user storage area. The input routines use free format to read in variables although these have to be arranged in a specified order. Data may also be input interactively by using the menu options.

6.3.1 Manipulator Specification

Figure 6.2 shows a data file for the RRR manipulator. A description of the elements is given below:

1. The first element is the number of degrees of freedom.

2. LINK1R: The first five characters are for ease of writing and reading data files. The sixth character is R if joint i is revolute or P if joint i is prismatic. No other character is allowed in the sixth place.

3. Link parameters in the order θ_i ,d_i, α_i,a_i and

```
3
LINK1R
0.0,0.0,90.0,0.0
2460.0
.05,.4,.05,.045,.4,.045
0.0,-.25,0.0,1.0
1.0
LINK2R
0.0,0.0,0.0,0.5
2460.0
.5,.05,.05,.5,.045,.045
-.25,0.0,0.0,1.0
1.0
LINK3R
0.0,0.0,0.0,0.5
2460.0
.5,.05,.05,.5,.045,.045
-.25,0.0,0.0,1.0
1.25
0.0,0.0,-9.8,0.0
.025
.3
```

Figure 6.2. Manipulator Specification file

are in degrees and d and a are in meters.

4. Density of the material of link in kilograms per cubic meter.

5. Dimensions of the link in meters. Since links are hollow bars of rectangular section the (see figure 6.1) dimensions must be given in the following format:

Xo,Yo,Zo,Xi,Yi,Zi

where X,Y and Z correspond to the link coordinate system and subscripts o and i indicate outer and inner dimensions of the hollow section. For the longitudnal dimension outer and inner dimensions are shown to be equal in figure 6.1 although this is not necessary.

6. Center of mass of link i in link i coordinates. This does not include the mass of the actuator and serves to locate the coordinate origin of link i.. The actual center of mass is calculated within the program.

7. Maximum velocity of the actuator for link i.

8. Direction of the gravity vector with respect to the zero coordinate system.

9. Time required to accelerate from rest to maximum velocity (TACC).

10. Increment for advancing simulated time (DT).

Data items two through seven must be specified successively for each link, starting from link one upto link n.

6.3.2 Trajectory Specification

End points for the trajectory segments may be listed in a data file or entered interactively. Any point may be changed between simulation runs, and points may be added to the end of the trajectory. Each line of this data file (figure 6.3) must be of the format:

X,Y,Z,R,P,Y

where X,Y,Z are the Cartesian coordinates in meters, and R,P,Y are the roll pitch and yaw angle in degrees. If a manipulator with less than six degrees of freedom is being investigated the remaining components should be entered as zero. The name of the trajectory file must be of the form NAME.DAT where NAME is a valid FORTRAN variable name.

```
0.3,0.1,-0.1,45.0,90.0,90.0
0.6,0.05,-0.295,60.0,90.0,90.0
0.61,0.05,-0.295,60.0,90.0,90.0
0.6,0.05,-0.295,60.0,90.0,90.0
0.5125,0.4,-0.24,60.,45.,90.0
0.5124,0.4,-0.255,60.0,45.0,90.0
0.5125,0.4,0.24,60.0,45.0,90.0
0.3,0.1,-0.1,45.0,90.0,90.0
```

Figure 6.3. Task Specification File

6.3.3 Data Files for Graphics

To draw a manipulator configuration a data file containing description of the links must be available to the program. Each link is specified as a series of points in three dimensions which are to be connected in sequence. The points are specified with respect to the base coordinate system. An option to choose different scaling factors allows the same data file to be used when dimensions are changed proportionately. Each line in the data file consists of four numbers. The last three are the x,y,z coordinates of the point. The first number may be any one of five integers, and specifies an option for the graphics software. These are:

1 - Move

2 - Draw

3 - Relative Move

4 - Relative Draw

20 - End of data for current link

Data entered before the first occurance of option 20, are the points comprising the base of the robot and the workspace.

6.3.4 Interactive Input

Figures 5.6 and 5.7 show the menu selections available before a simulation run is started. Options may be selected to enter data interactively or to change current

```
1,0.,0.,0.
1,-.20,-.20,-.4
3,.20,-.20,-.4
3,.20,.20,-.4
3,-.20,.20,-.4
3,-.20,-.20,-.4
3,.20,.20,-.4
1,-.20,.20,-.4
3,.20,-.20,-.4
20,0.,0.,0.
1,0.,0.,0.
1,-.05,0.,.05
3,.05,0.,.05
3,.05,0.,-.05
3,-.05,0.,-.05
3,-.05,0.,.05
3,-.05,-.4,.05
3,.05,-.4,.05
3,.05,-.4,-.05
3,-.05,-.4,-.05
3,-.05,-.4,.05
1,.05,-.4,.05
3,.05,0.,.05
1,.05,0.,-.05
3,.05,-.4,-.05
1,-.05,-.4,-.05
3,-.05,0.,-.05
20,0.,0.,0.
1,0.,0.,0.
1,0.,-.05,.05
```

Figure 6.4. Segment of a Graphics Data File

values. All data except that in the graphic data files may be changed interactively. The changes do not affect the data files immediately, and the original data is preserved until the user decides to end the session. At this point the new values may be saved or discarded. The design process may thus be continued over a number of sessions.

CHAPTER VII

VERIFICATION OF RESULTS

7.1 INTRODUCTION

To verify results obtained from calculations using the recursive Langrangian formulation for manipulator dynamics, a three degree of freedom articulated arm was used. The dynamic solution was obtained using the Uicker Kahn equations directly. A program was written to compute generalized torques at the interpolated points on the trajectory using these equations. Results were compared and found to be in agreement.

7.2 RRR MANIPULATOR SOLUTION

Following assumptions have been made in simplifying the derivation:

1. All links were considered to be hollow bars of uniform square cross section.

2. Actuators were assumed to be point masses acting at the coordinate origin of the links.

3. The off diagonal terms of the inertia matrix, i.e. the cross products of inertia were assumed to be zero. This assumption follows directly from the first two.

4. Centripetal and coriolis components of the forces, which are velocity dependent, were neglected in the derivation.

5. The mass of any object picked up by the manipulator was considered to be negligible compared to the mass of links and actuators.

7.2.1 Kinematic Solution

Figure 7.1 shows the link coordinate system and table 7.1 gives the link parameters for the manipulator. The kinematic solution may be obtained directly from the geometry of the links or by using Paul's method.

$$\theta_1 = \text{atan2}(p_y, p_x) \qquad (7.1)$$

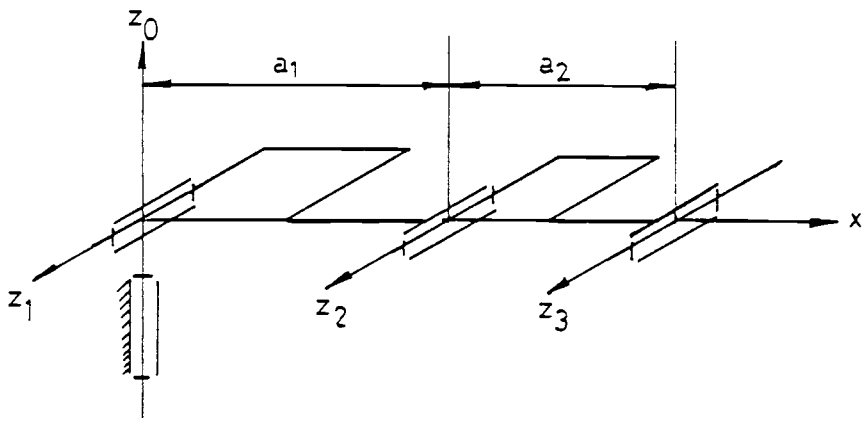


Figure 7.1. Coordinate Frames for RRR Manipulator

LINK PARAMETERS:

Link	Theta	d	Alpha	a
1	0.0000	0.0000	90.0000	0.0000
2	0.0000	0.0000	0.0000	0.5000
3	0.0000	0.0000	0.0000	0.5000

LINK DIMENSIONS:

Link	Density	Outer dimensions			Inner dimensions		
	kg/cu.m	m	m	m	m	m	m
1	2460.000	0.050	0.400	0.050	0.045	0.400	0.045
2	2460.000	0.500	0.050	0.050	0.500	0.045	0.045
3	2460.000	0.500	0.050	0.050	0.500	0.045	0.045

ACTUATOR SPECIFICATIONS:

Joint	Type	Max Velocity
1	R	1.0000
2	R	1.0000
3	R	1.2500

Acceleration Time (TACC) : 0.3000
Time Increment (DT) : 0.0250

Table 7.1. Link Parameters and Dimensions

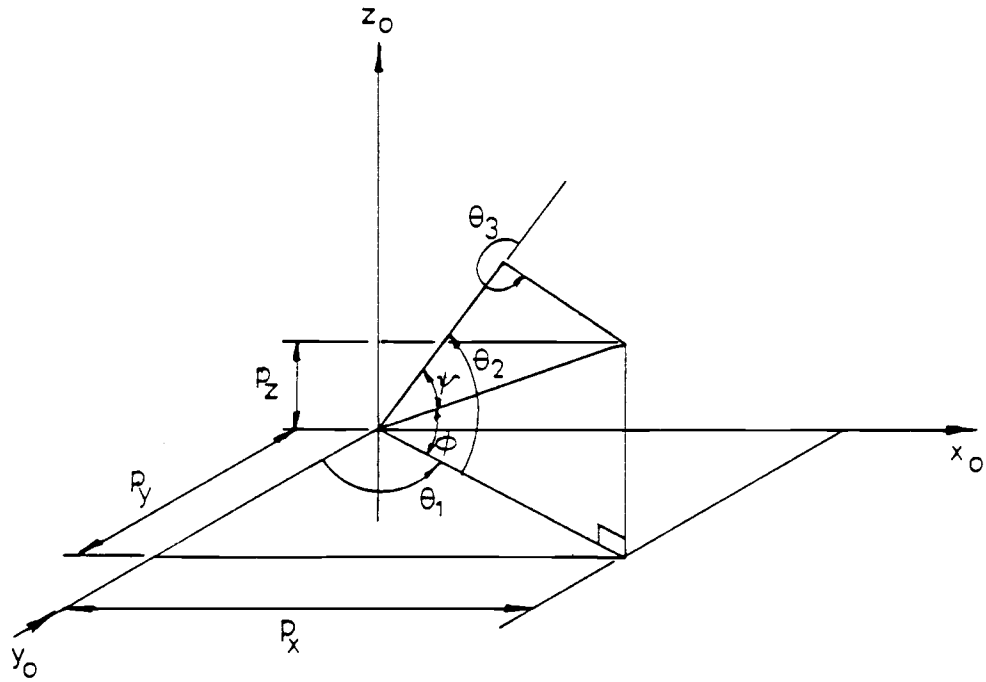


Figure 7.2. Kinematic Solution

$$r = \sqrt{(p_x^2 + p_y^2 + p_z^2)}$$

$$\cos(\phi) = \sqrt{(p_x^2 + p_y^2)}/r$$

$$\sin(\phi) = \sqrt{(1 - (\cos^2))}$$

$$\phi = \text{atan2}(\sin\phi , \cos\phi) \qquad (7.2)$$

$$\cos(\gamma) = (a_1^2 + r^2 - a_2^2)/(2a_1 r)$$

$$\sin(\gamma) = \sqrt{(1 - (\cos^2\gamma))}$$

$$\gamma = \text{atan2}(\sin\gamma , \cos\gamma) \qquad (7.3)$$

$$\theta_2 = \phi + \gamma$$

for up elbow and

$$\theta_2 = \phi - \gamma$$

for down elbow

$$\cos(\theta_3) = -\cos(\theta_3 - 180)$$

$$= (a_1^2 + a_2^2 - r^2)/(2a_1 a_2)$$

$$\sin(\theta_3) = -\sin(\theta_3 - 180)$$

$$= -\sqrt{(1 - \cos^2(\theta_3))}$$

$$\theta_3 = \text{atan2}(\sin\theta_3 , \cos\theta_3) \qquad (7.4)$$

7.2.2 Link Dimensions and Inertias

Table 7.1 shows the link dimensions. Masses, moments of inertia and centers of mass are given in table 7.2. Psuedo inertia matrices for each link are presented in figure 7.3.

7.2.3 Dynamic Solution

Effective and coupling inertias, and gravity loading term are given below:

Link	Mass	Act.Mass	Ixx	Iyy	Izz	x	y	z
1	0.796	0.528	0.0356	0.0003	0.0356	0.0	-0.15	0.0
2	0.730	0.329	0.0004	0.0489	0.0489	-0.2	0.0	0.0
3	0.584	0.146	0.0004	0.0489	0.0489	-0.25	0.0	0.0

Table 7.2. Link Properties

$$J_1 = \begin{bmatrix} 0.00018 & 0.00000 & 0.00000 & 0.00000 \\ 0.00000 & 0.03544 & 0.00000 & -0.11685 \\ 0.00000 & 0.00000 & 0.00018 & 0.00000 \\ 0.00000 & -0.11685 & 0.00000 & 0.79604 \end{bmatrix}$$

$$J_2 = \begin{bmatrix} 0.04869 & 0.00000 & 0.00000 & -0.14606 \\ 0.00000 & 0.00022 & 0.00000 & 0.00000 \\ 0.00000 & 0.00000 & 0.00022 & 0.00000 \\ -0.14606 & 0.00000 & 0.00000 & 0.73031 \end{bmatrix}$$

$$J_3 = \begin{bmatrix} 0.04869 & 0.00000 & 0.00000 & -0.14606 \\ 0.00000 & 0.00022 & 0.00000 & 0.00000 \\ 0.00000 & 0.00000 & 0.00022 & 0.00000 \\ -0.14606 & 0.00000 & 0.00000 & 0.58425 \end{bmatrix}$$

Figure 7.3 Psuedo Inertia Matrices

Effective inertias

$$D_{11} = \bar{I}_{1YY} + (S_2^2 I_{2XX} + C_2^2 I_{2YY}) + m_2 C_2^2 a_2 (a_2 + 2\bar{x}_2) + (S_{23}^2 I_{3XX} + C_{23}^2 I_{3YY}) + + m_3 (a_3 C_{23} + a_2 C_2)[(a_3 C_{23} + a_2 C_2) + 2x_3 C_{23}] \quad (7.5)$$

$$D_{22} = I_{2ZZ} + I_{3ZZ} + m_2 a_2 (a_2 + 2\bar{x}_2) + m_3 [a_3^2 + a_2^2 + a_2 a_3 C_3 + 2\bar{x}_3 (a_3 + a_2 C_3)] \quad (7.6)$$

$$D_{33} = I_{3ZZ} + m_3 a_3 (a_3 + 2\bar{x}_3) \quad (7.7)$$

Coupling inertias:

$$D_{12} = 0 \quad (7.8)$$

$$D_{13} = 0 \quad (7.9)$$

$$D_{23} = I_{3ZZ} + m_3 [(a_3 + a_2 C_3)(a_3 + \bar{x}_3) + \bar{x}_3 a_3] \quad (7.10)$$

Gravity loading terms: since z is vertical

$$g = (\ 0\ 0\ -g\ 0\ )$$

and

$$D_1 = 0 \quad (7.11)$$

$$D_2 = g[m_2 C_2 (\bar{x}_2 + a_2) + m_3 (C_{23} (\bar{x}_3 + a_3) + a_2 C_2)] \quad (7.12)$$

$$D_3 = g[m_3 C_{23} (\bar{x}_3 + a_3)] \quad (7.13)$$

Finally the generalized torques may be written as:

$$F_1 = D_{11}\ddot{q}_1$$

$$F_2 = D_{22}\ddot{q}_2 + D_{23}\ddot{q}_3 + D_2 \qquad (7.14)$$

$$F_3 = D_{33}\ddot{q}_3 + D_3$$

7.3 TASK SPECIFICATION

A simple task was specified and torques were calculated during trajectory execution using both the recursive Lagrangian formulation and the solution in section 7.2. The task is to move over a contoured weld seam, and return to the original position. Motion primitives for the task are:

1. Move from initial position to beginning of weld seam

2. Move to first point defining contour

3. Move to second point

4. Move to third point

5. Return to original position

Coordinates for the points making up the trajectory are

```
.5,0.,-.1,0.,0.,0.
.5,0.1,-0.1,0.,0.,0.
.7,0.1,-.4,0.,0.,0.
.60,-.15,-.35,0.,0.,0.
.5,-.20,-.25,0.,0.,0.
.40,-.25,-.20,0.,0.,0.
.5,0.,-.1,0.,0.,0.
```

Figure 7.4. Trajectory Specification

given in figure 7.4. No provision is made for stopping at any point although this may have to be done in a real system. Since the effect of any load carried is assumed to be negligible, values of the torques would be constant during this interval.

7.4 RESULTS

The simulation was run with a time increment of 0.025 seconds giving a sampling frequency of forty Hertz. It took 3.725 seconds to complete the task. Configurations of the manipulator as it executes the task are shown in figure 7.5. Plots of torque, acceleration, velocity and position versus simulated time are also given in figures 7.6 to 7.9. Table 7.4 gives the values of these quantities as calculated by the simulation using recursive Lagrangian dynamics. Table 7.5 lists the values of the torques at the same conditions using the equations presented in this chapter. The discrepancy is due to the velocity dependent components, which are neglected in the equations derived in this chapter. Since the simulation was run at at low velocities, this discrepancy is quite small and the values obtained agree well.

```
      DYNAMIC SIMULATION OF ROBOT MANIPULATORS
      ----------------------------------------

SUMMARY RESULTS :
---------------

Time taken for task execution (sec.) :       0.3725005E+01

Maximum values for actuator Torque (Nm)/Force (N) :
-------------------------------------------------

  Joint( 1)   (R)  :                          0.5220423E+00
  Joint( 2)   (R)  :                          0.6255348E+01
  Joint( 3)   (R)  :                          0.5695611E+00

Extreme Positions in the trajectory (m/rad) :
-------------------------------------------

  Joint( 1)   (R) :       -0.5032491E+00 to  0.1823285E+00
  Joint( 2)   (R) :        0.1576130E+00 to  0.8376182E+00
  Joint( 3)   (R) :       -0.2071214E+01 to -0.1328831E+01
```

Table 7.3. Summary Results
(Recursive Lagrangian Method)

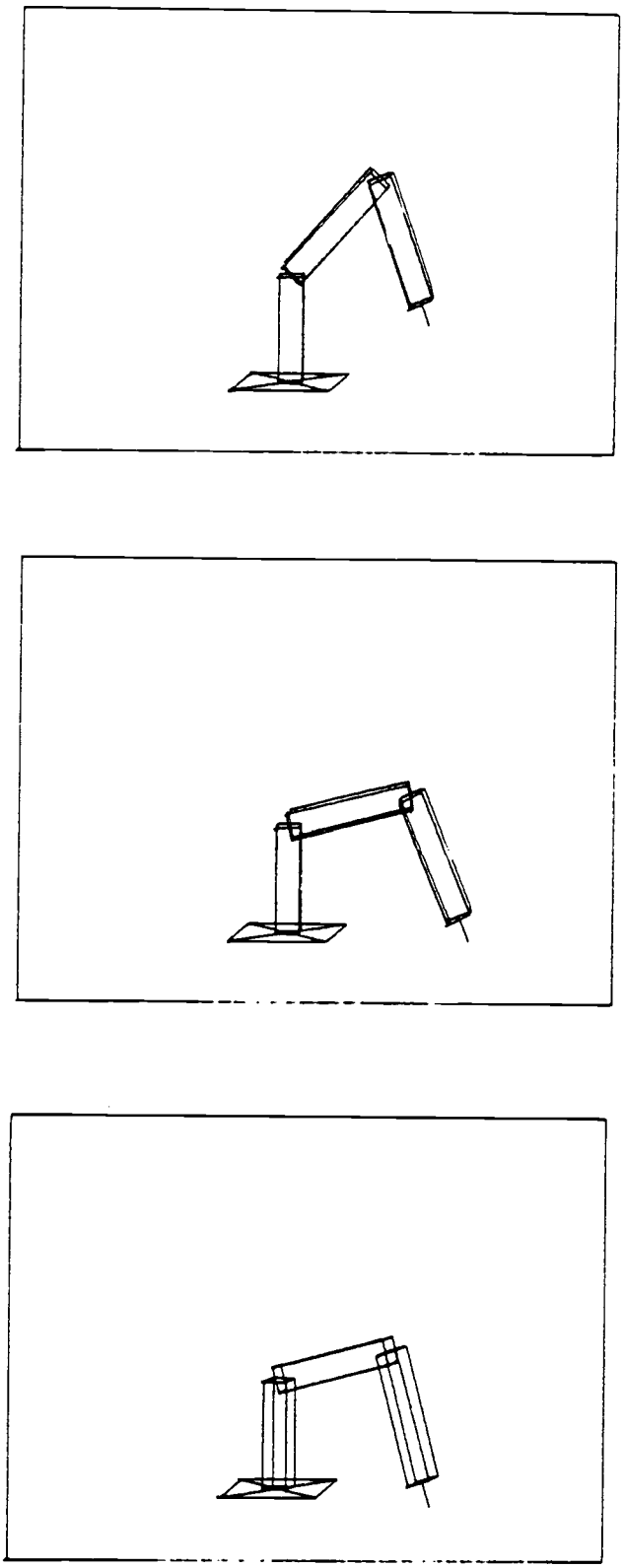
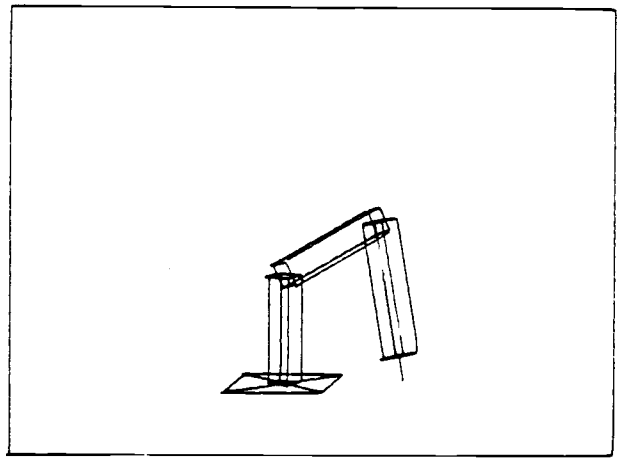
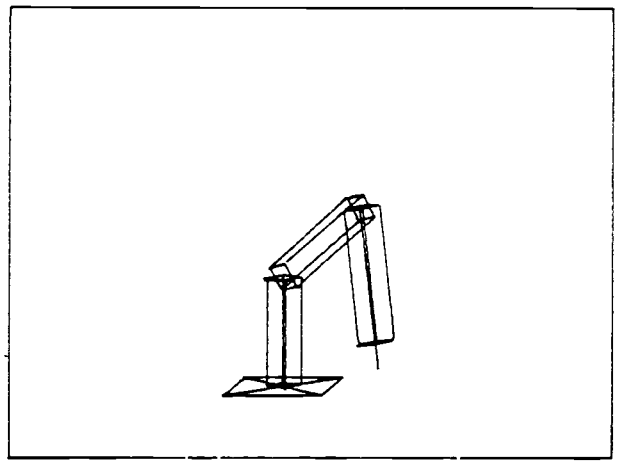


Figure 7.5. End Points of Trajectory Segments





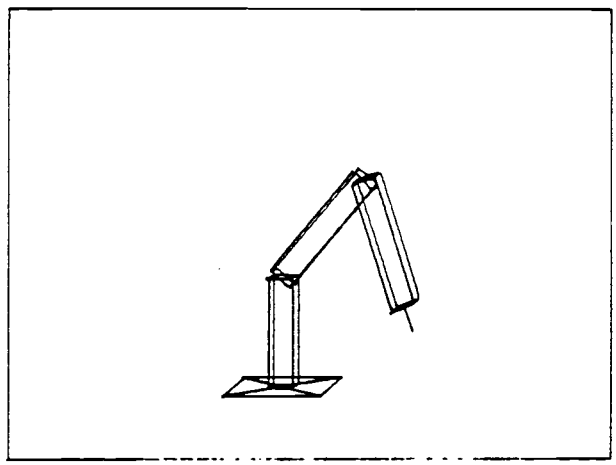


Figure 7.5. (Contd.)

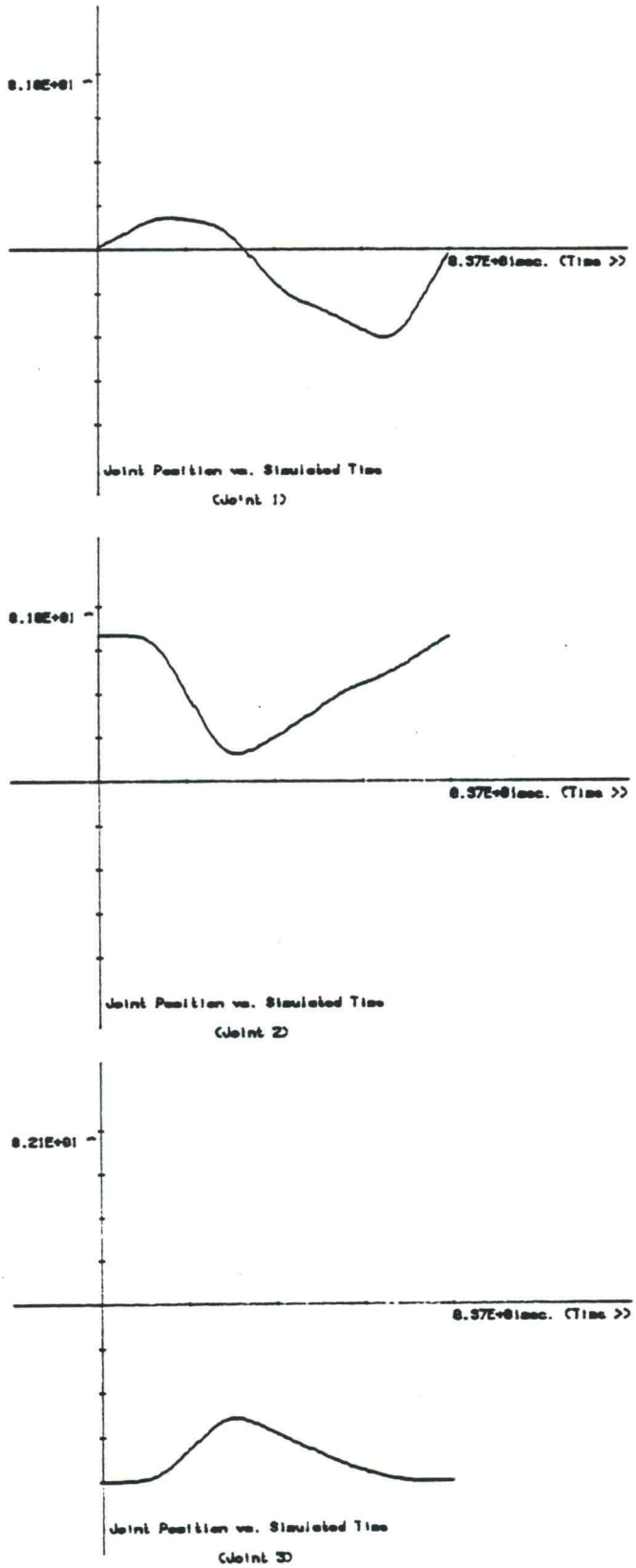


Figure 7.6.
Joint position versus simulated time

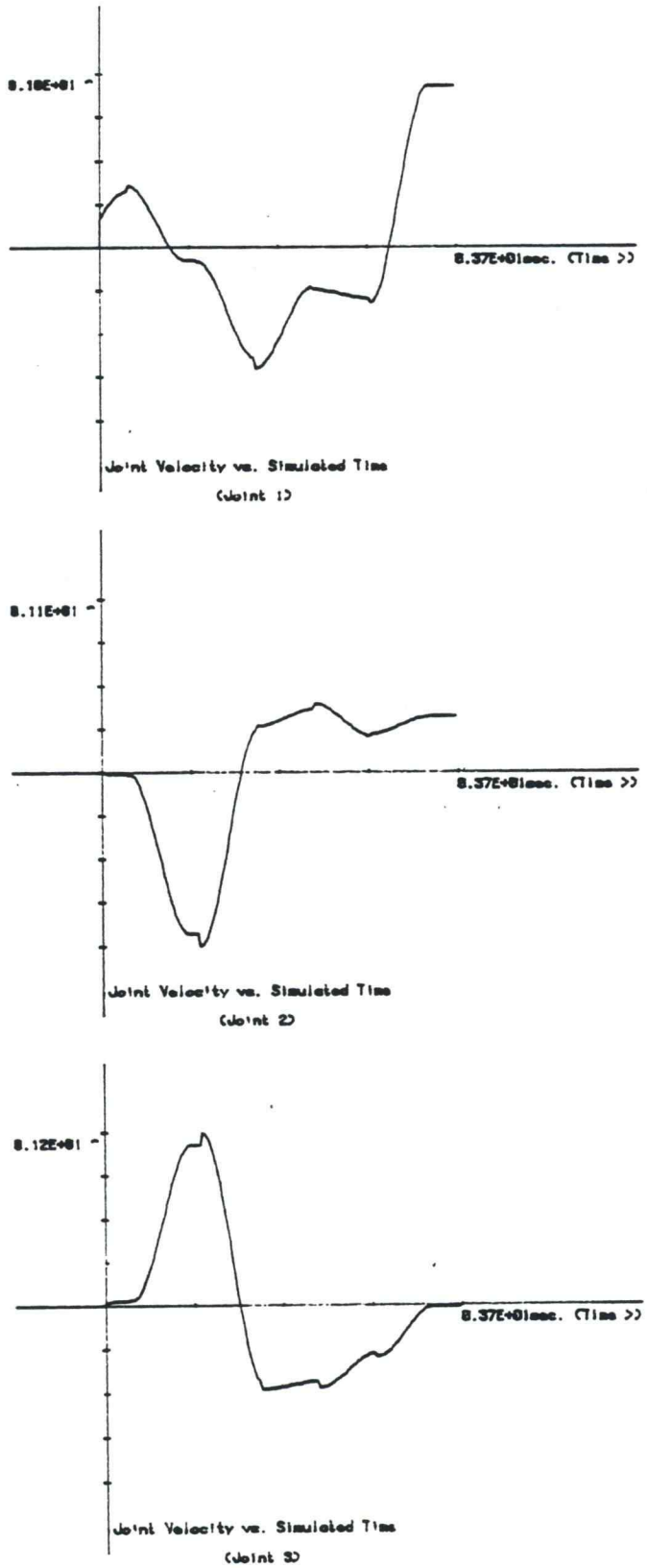


Figure 7.7.
Joint velocity versus simulated time

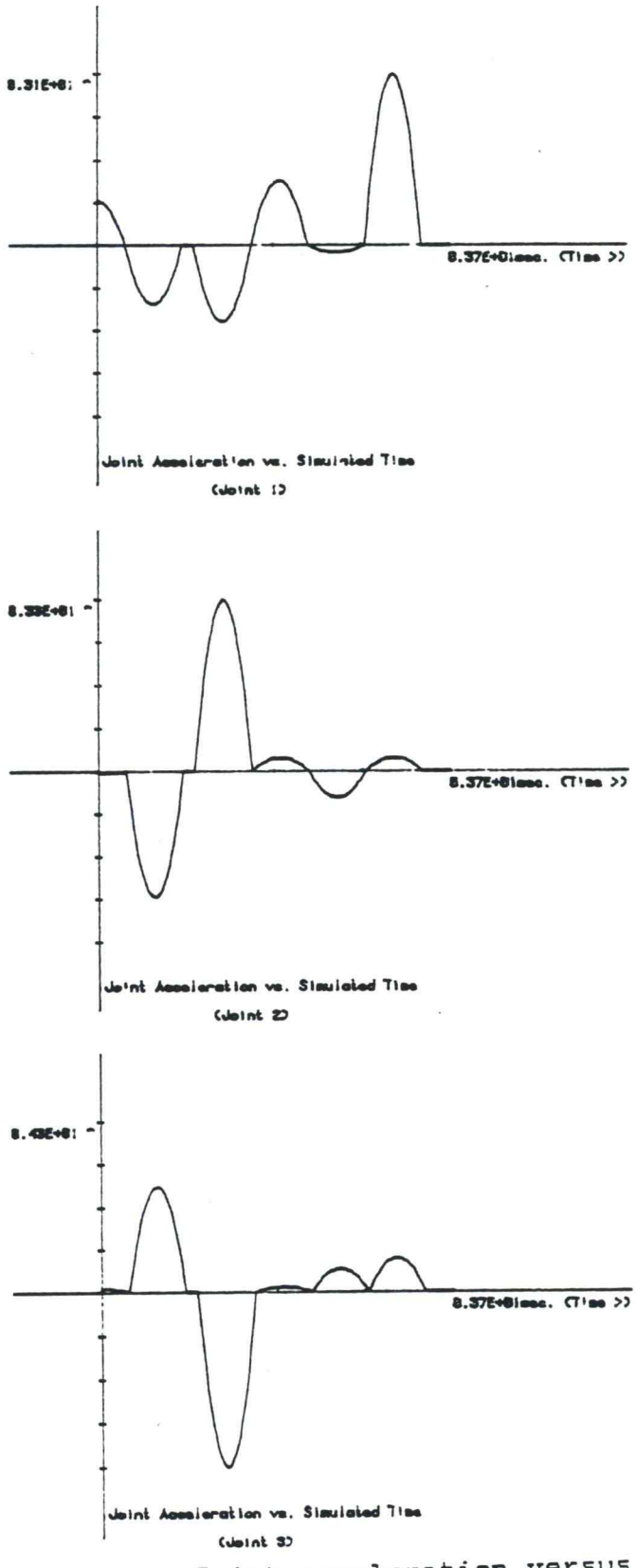


Figure 7.8.
Joint acceleration versus simulated time

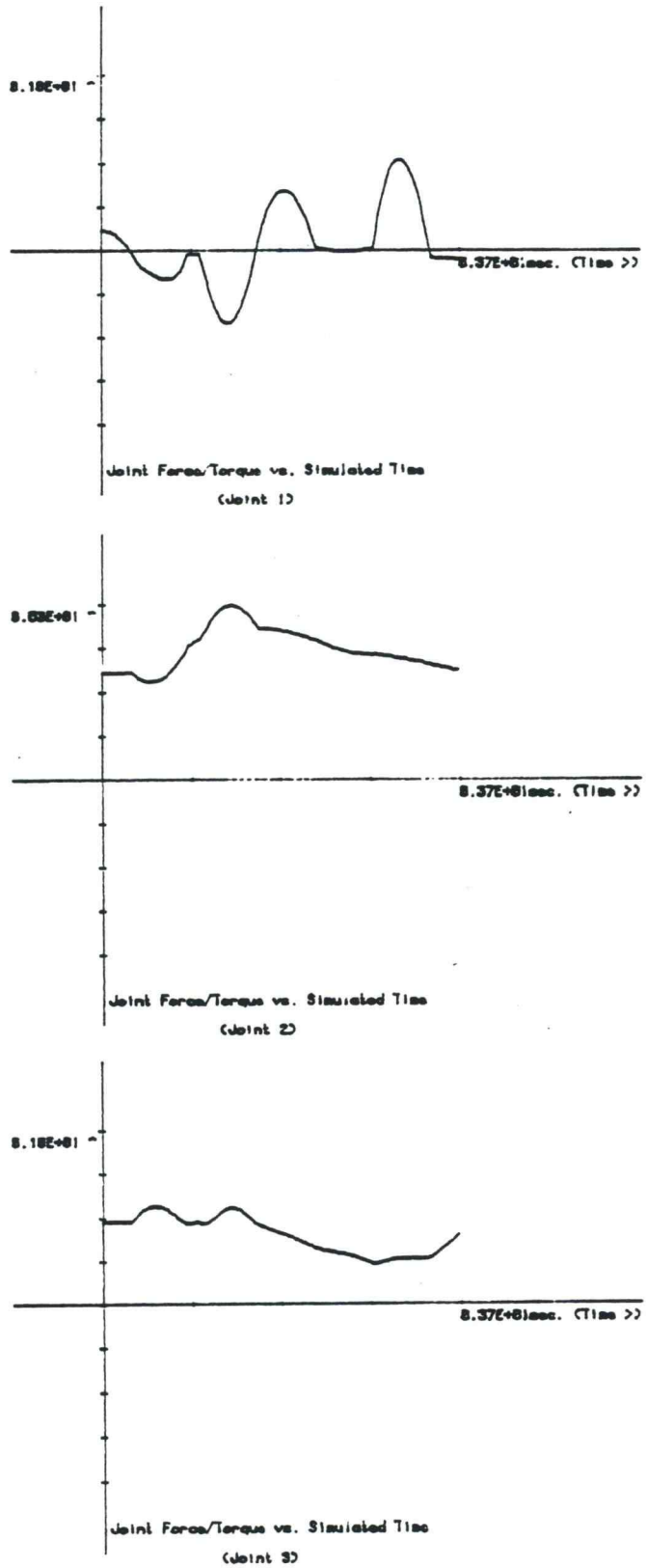


Figure 7.9.
Joint force/torque versus simulated time

Current Time = 0.000000E+00 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.185058E-01	0.246744E+00	0.185058E+01	0.265067E+00
2	0.837618E+00	-0.949063E-02	-0.711797E-01	0.382132E+01
3	-0.206933E+01	0.283241E-01	0.212431E+00	0.481783E+00

Current Time = 0.250000E-01 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.252513E-01	0.292768E+00	0.182167E+01	0.261671E+00
2	0.837359E+00	-0.112609E-01	-0.700675E-01	0.382495E+01
3	-0.206855E+01	0.336072E-01	0.209112E+00	0.480801E+00

Current Time = 0.500000E-01 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.331322E-01	0.337346E+00	0.173492E+01	0.250109E+00
2	0.837056E+00	-0.129755E-01	-0.667310E-01	0.382948E+01
3	-0.206765E+01	0.387244E-01	0.199154E+00	0.479440E+00

Table 7.4. Tabulated Results (Simulation)

Current Time = 0.750000E-01 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.420944E-01	0.379032E+00	0.159035E+01	0.230319E+00
2	0.836711E+00	-0.145789E-01	-0.611700E-01	0.383478E+01
3	-0.206662E+01	0.435096E-01	0.182558E+00	0.477786E+00

Current Time = 0.100000E+00 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.520477E-01	0.416381E+00	0.138794E+01	0.202219E+00
2	0.836328E+00	-0.160154E-01	-0.533848E-01	0.384068E+01
3	-0.206548E+01	0.477970E-01	0.159323E+00	0.475949E+00

Current Time = 0.125000E+00 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.628653E-01	0.447947E+00	0.112770E+01	0.165716E+00
2	0.835912E+00	-0.172296E-01	-0.433751E-01	0.384700E+01
3	-0.206423E+01	0.514205E-01	0.129450E+00	0.474054E+00

Table 7.4. (Contd.)

Current Time = 0.150000E+00 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.743848E-01	0.472284E+00	0.809630E+00	0.120711E+00
2	0.835469E+00	-0.181656E-01	-0.311411E-01	0.385351E+01
3	-0.206291E+01	0.542142E-01	0.929385E-01	0.472238E+00

Current Time = 0.175000E+00 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.864073E-01	0.487947E+00	0.433730E+00	0.671000E-01
2	0.835006E+00	-0.187681E-01	-0.166827E-01	0.385998E+01
3	-0.206153E+01	0.560121E-01	0.497885E-01	0.470635E+00

Current Time = 0.200000E+00 sec.

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.864073E-01	0.554941E+00	0.000000E+00	0.604929E-02
2	0.835006E+00	-0.213450E-01	0.000000E+00	0.386782E+01
3	-0.206153E+01	0.637031E-01	0.000000E+00	0.464135E+00

Table 7.4. (Contd.)

Current Time = 0.000000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.185058E-01	0.246744E+00	0.185058E+01	0.263874E+00
2	0.837618E+00	-0.949063E-02	-0.711797E-01	0.381197E+01
3	-0.206933E+01	0.283241E-01	0.212431E+00	0.486516E+00

Current Time = 0.025000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.252512E-01	0.292768E+00	0.182167E+01	0.259992E+00
2	0.837359E+00	-0.112608E-01	-0.700675E-01	0.381390E+01
3	-0.206855E+01	0.336072E-01	0.209112E+00	0.487049E+00

Current Time = 0.050000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.331322E-01	0.337346E+00	0.173492E+01	0.247879E+00
2	0.837056E+00	-0.129755E-01	-0.667310E-01	0.381665E+01
3	-0.206765E+01	0.387244E-01	0.199154E+00	0.487373E+00

Table 7.5. Tabulated Results (Equations 7.14)

Current Time = 0.075000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.420944E-01	0.379032E+00	0.159035E+01	0.227502E+00
2	0.836711E+00	-0.145789E-01	-0.611700E-01	0.382019E+01
3	-0.206662E+01	0.435096E-01	0.182558E+00	0.487484E+00

Current Time = 0.100000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.520477E-01	0.416381E+00	0.138794E+01	0.198818E+00
2	0.836328E+00	-0.160154E-01	-0.533848E-01	0.382451E+01
3	-0.206548E+01	0.477970E-01	0.159323E+00	0.487373E+00

Current Time = 0.125000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.628653E-01	0.447947E+00	0.112770E+01	0.161779E+00
2	0.835912E+00	-0.172296E-01	-0.433751E-01	0.382956E+01
3	-0.206423E+01	0.514205E-01	0.129450E+00	0.487026E+00

Table 7.5. (Contd.)

Current Time = 0.150000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.743848E-01	0.472284E+00	0.809630E+00	0.116332E+00
2	0.835469E+00	-0.181656E-01	-0.311411E-01	0.383531E+01
3	-0.206291E+01	0.542142E-01	0.929385E-01	0.486425E+00

Current Time = 0.175000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.864073E-01	0.487947E+00	0.433730E+00	0.624231E-01
2	0.835007E+00	-0.187681E-01	-0.166827E-01	0.384170E+01
3	-0.206153E+01	0.560121E-01	0.497885E-01	0.485551E+00

Current Time = 0.200000sec. (VERIFICATION)

Joint	Position	Velocity	Acceleration	Force/Torque
1	0.864073E-01	0.554941E+00	0.000000E+00	0.000000E+00
2	0.835007E+00	-0.213450E-01	0.000000E+00	0.384565E+01
3	-0.206153E+01	0.637031E-01	0.000000E+00	0.483116E+00

Table 7.5. (Contd.)

CHAPTER VIII

SUMMARY AND CONCLUSIONS

This simulation is primarily a tool for selecting a kinematic configuration and designing a trajectory plan for a known application. It can prove to be a valuable aid in the selection of a robot from options available on the market. With known kinematic configurations it provides an interactive tool for trajectory planning to optimize loading conditions on actuators. A logical work station layout can be based on the results. During the design process it can be used to compute loads at extreme conditions for actuator sizing. A base has been provided for mechanical design of link structures, although this aspect of the program needs considerable expansion before it can be practically applied. Since length of code and storage requirements are within 64000 bytes, the program may be run on small systems. All graphics and mathematical operations are integral with the program and no additional software is needed.

Manipulator trajectory is generated by interpolation in joint space and this is an important limitation of the program. It would be useful to include interpolation in Cartesian space as an option. Other functions should also be included for interpolation as polynomial trajectories present some disadvantages. There should be some provision for stopping the manipulator at any point on the path and at the end of the trajectory. It may be worthwhile to have a large number of library solutions available which could be accessed when needed. The program has been structured to allow this expansion. Finally there is considerable potential for improving the graphics. Hidden line removal would be helpful and animation of the trajectory on the screen would provide insight into the actual interpolated motion.

BIBLIOGRAPHY

Albus, J. S., "Data Storage in the Cerebellar Model Articulated Computer (CMAC)", Journal of Dynamic Systems, Measurement and Control, Sept. 1975, pp. 228-233.

Bejczy, A. K., and Paul, R. P., "Simplified Robot Arm Dynamics for Control," Proc. IEEE Conference on Decision and Control, 1981, pp. 261-262.

Brady, M., et al., (ed.) Robot Motion, The MIT Press Series in Artificial Intelligence, 1982.

Colson, J. C., "Kinematic Arrangements used in Industrial Robots," Proc. 13th International Symposium on Industrial Robots, 1983, pp. 20-1 to 20-18.

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

Derby, S. J., "Kinematic Elasto-Dynamic Analysis and Computer Graphics Simulation of General Purpose Robot Manipulators," Ph.D. Thesis, Rensselaer Polytechnic Institute, August 1981.

Engelberger, J. F., Robotics in Practice, IFS Publications Ltd., Kempston, England, 1980.

Harrington, S., Computer Graphics A Programming Approach, McGraw-Hill Book Company, New York, 1983.

Heginbotham, W. B., Dooner, M., and Case., K., "Rapid Assesment of Industrial Robots Performance by Interactive Computer Graphics," Proc. 9th International Symposium on Industrial Robots, March 1979, pp. 563-574.

Hemami, H., Jaswa, V. C., and McGhee, R. B., "Some Alternative Formulations of Manipulator Dynamics for Computer Simulation Studies," Proc. 13th Allerton Conference on Circuit and System Theory, October 1975, pp. 124-140.

Hollerbach, J. M., "A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Fomulation Complexity," IEEE Transactions on Systems Man and Cybernetics, Vol. SMC-10, No. 11, November 1980, pp. 730-736.

Kahn, M. E., and Roth, B., "The Near Minimum Time Control of Open Loop Articulated Kinematic Chains," Journal of Dynamic Systems, Measurement and Control, September 1971, pp. 164-172.

Lee, C. S., Lee, B. H., and Nigam, R., "Development of the Generalised d'Alembert Equations of Motion for Mechanical Manipulators," Proc. 22nd Conference on Decision and Control, December 1983.

Ligeois, A., Khalil, W., and Dumas, J. M., "Mathematical and Computer Models of Interconnected Mechanical Systems," 2nd CISM-IFToMM Symposium on Theory and Practice of Robots and Manipulators, 1978, pp. 5-17.

Lozano-Perez, T., and Wesley, M. A., "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles," Communications of ACM, V22, N10, October 1979, PP. 560-570.

Luh, J. Y. S., Walker, M. W., and Paul, P. C., "On Line Computational Scheme for Mechanical Manipulators," Journal of Dynamic Systems Measurement and Control, 1980, pp. 69-76.

Medeiros, D. J., and Sadowski, R. P., "Simulation of a Robotic Manufacturing Cell, a Modular Approach," Simulation, V40 N1, January, 1983.

Milenkovic, V., and Huang, B., "Kinematics of Major Robot Linkage," Proc. 13th International Symposium on Industrial Robots and Robots 7, 1983, pp. 16-31 to 14-47.

Paul, R. P., "Cartesian Coordinate Control of Robots in Joint Coordinates," Proc. 3rd International CISM IFToMM Symposium on the Theory and Practice of Robots and Manipulators, 1978, pp. 228-250.

Paul, R. P. "Manipulator Cartesian Path Control," IEEE Transactions on Systems, Man and Cybernetics, V SMC-9 N11, November 1979, pp. 702-711.

Paul, R. P., Robot Manipulators: Mathematics, Programming, and Control," The MIT Press series in Artificial Intelligence, 1981.

Pieper, D. L., "The Kinematics of Manipulators under Computer Control," Ph.D. Thesis, Stanford University, 1969.

Potkonjak, V., Vukobratovic, M., and Hristic, D., "Interactive Procedure for Computer-Aided Design of Industrial Robot Mechanisms," Proc. 13th International Symposium on International Robots and Robots 7, 1983, pp. 16-85 to 16-94.

Raibert, N. H., "Analytical Equations versus Table Look up for Manipulation: A Unifying Concept," Proc. IEEE Conference on Decision and Control, 1977, 576-579.

Sata, T., Fumihiko, K., and Amuno, A., "Robot Simulation System as a Task Programming Tool," Proc. 11th International Symposium on Industrial Robots, 1981, pp. 595-602.

Shumaker, G. C., "Robotics - Air Force Project," Computer World, March 17, 1980.

Silver, W. M., "On the Equivalence of Lagrangian and Newton-Euler Dynamics for Manipulators," The International Journal of Robotics Research, V1, N2, Summer 1982, pp. 60-70.

Sjolund, P., and Donath, M., "Robot Task Planning: Programming Using Interactive Computer Graphics," Proc. 13th International Symposium on Industrial Robots and Robots 7, 1983, pp. 7-122 to 7-135.

Sullivan, G., Rise of the Robots, Dodd, Mead and Company, New York, 1971.

Taylor, R. H., "Planning and Execution of Straight Line Manipulator Trajectories," IBM Journal of Reaearch and Development, V23 N4, July 1979, pp. 253-264.

Uicker, J., J., Jr., "Dynamic Force Analysis of Spatial Linkages," Journal of Applied Mechanics, June 1967, pp. 418-424.

Urbaniak, D. F., "The Unattended Factory FANUC's New Flexibility Automated Plant Using Industrial Robots," Proc. 13th International Symposium on Industrial Robots and Robot 7, 1983, pp. 1-18 to 1-24.

Walker, M. W., and Orin, D. E., "Efficient Dynamic Computer Simulation of Robotic Mechanisms," Journal of Dynamic Systems Measurement and Control, V104 N3, September 1982, pp. 205-211.

Warnecke, H. J., and Schraft, R. D., "A Computer Aided Method to Design an Industrial Robot," Proc. 2nd International CISM IFToMM Symposium on the Theory and Practice of Robots and Manipulators, September 1976, pp. 85-92.

Warnecke, H. J., Schraft, R. D., and Schmidt-Streier, U., "Computer Graphics Planning of Industrial Robot Application," 3rd International Symposium on the Theory and Practice of Robots and Manipulators, 1978.

Welch, T. L., "Evaluating Robotic Systems Through Simulation," Proc. 13th International Symposium on Industrial Robots and Robots 7, 1983, pp. 7-55 to 7-72.

Whitney, D. E., "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators," Journal of Dynamic Systems Measurement and Control, V122, December 1972, pp. 303-309.

APPENDIX

LIBRARY SOLUTIONS

A large number of arrangements are possible for a six degree of freedom kinematic chain, but very few are applicable to practical situations [Colson, 1983, Malenkovic, 1983]. However, if the complete kinematic and dynamic solution was derived for each, this would still represent a substantial task.

Solution libraries for this simulation contain only the kinematic solution and a routine to check for workspace limitations. The dynamic solution is obtained recursively once the values of joint variable, velocities and accelerations are determined for a given move. If a user wishes to investigate a new kinematic arrangement, only two subroutines need to be written, viz. KNTKUS and TDRVUS. These are described in chapter five. In addition, trajectory points can be checked for workspace limitations by including subroutines WKSPUS. Dummy subroutines exist for all of these.

Four library solutions are available at present, complete with data files for graphics, and sample trajectories.

1. RRR manipulator
2. RRPRRR (Stanford) manipulator
3. PPPRRR (Cartesian coordinate) manipulator
4. RRRRRR manipulator

Type one has been used for verification and has been described in chapter seven. They can all be solved by Paul's method and solutions are available in the literature [Paul, 1981; Derby, 1981]. Output obtained for a hypothetical trajectory through given segment end points is appended. It may be observed that in all cases joint motion is smooth and uniform. Since no provision is made for stopping at any point on the trajectory, the velocity does not become zero at the end of the trajectory. However motion at the end point is uniform and acceleration is accurately simulated. Forces and torques versus simulated time are also presented and the trajectory may be improved by studying these. It may be seen that in the present trajectory there are some jerks when moving from one trajectory segment to another. These would have to be studied alongwith link structure. A good estimate of actuator requirements can also be obtained from these graphs.

LINK PARAMETERS:

Link	Theta	d	Alpha	a
1	0.0000	0.0000	-90.0000	0.0000
2	0.0000	0.3700	90.0000	0.0000
3	0.0000	1.0000	0.0000	0.0000
4	0.0000	0.0000	-90.0000	0.0000
5	0.0000	0.0000	90.0000	0.0000
6	0.0000	0.0000	0.0000	0.0000

LINK DIMENSIONS:

Link	Density	Outer dimensions			Inner dimensions		
	kg/cu.m	m	m	m	m	m	m
1	2460.000	0.150	0.150	0.300	0.145	0.145	0.300
2	2460.000	0.150	0.150	0.370	0.145	0.145	0.370
3	2460.000	0.125	0.125	1.000	0.120	0.120	1.000
4	2460.000	0.030	0.030	0.100	0.000	0.000	0.100
5	2460.000	0.030	0.030	0.100	0.000	0.000	0.100
6	2460.000	0.030	0.030	0.100	0.000	0.000	0.100

ACTUATOR SPECIFICATIONS:

Joint	Type	Max Velocity
1	R	2.0000
2	R	2.5000
3	P	1.0000
4	R	3.0000
5	R	3.0000
6	R	3.5000

Acceleration Time (TACC) : 0.3000
Time Increment (DT) : 0.0250

Table A-1. Link Parameters and Dimensions
RRPRRR (Stanford) Manipulator

DYNAMIC SIMULATION OF ROBOT MANIPULATORS

SUMMARY RESULTS :

Time taken for task execution (sec.) : 0.4650009E+01

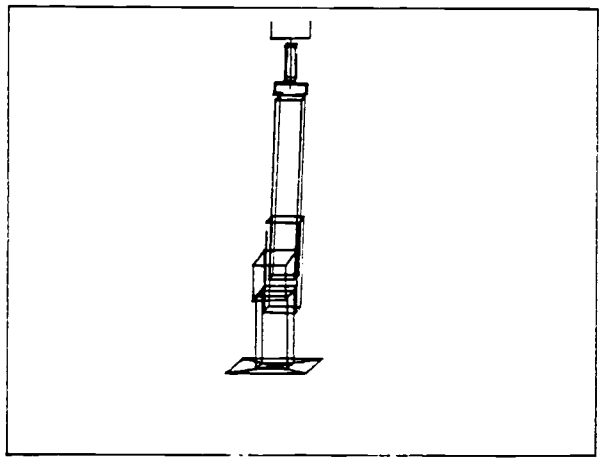
Maximum values for actuator Torque (Nm)/Force (N) :

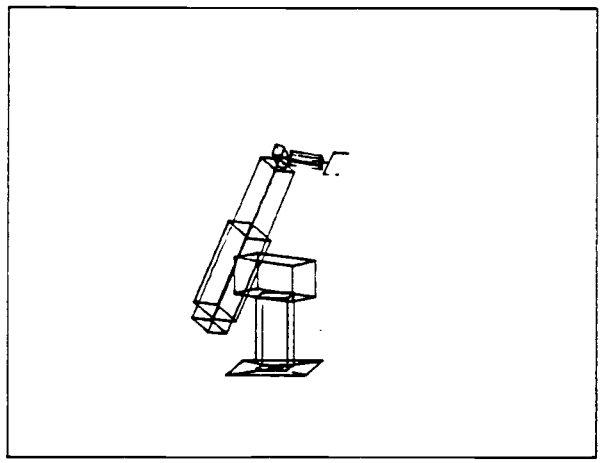
Joint(1)	(R)	:	-0.1304385E+02
Joint(2)	(R)	:	-0.9186593E+01
Joint(3)	(P)	:	0.1737053E+02
Joint(4)	(R)	:	0.6921432E+00
Joint(5)	(R)	:	0.6392899E+00
Joint(6)	(R)	:	-0.2646126E-03

Extreme Positions in the trajectory (m/rad) :

Joint(1)	(R)	:	-0.1339640E+01	to	0.1090283E+01
Joint(2)	(R)	:	0.0000000E+00	to	0.1408105E+01
Joint(3)	(P)	:	0.7062050E+00	to	0.1000000E+01
Joint(4)	(R)	:	-0.2284291E+01	to	-0.3534046E-01
Joint(5)	(R)	:	0.1044054E+00	to	0.1546464E+01
Joint(6)	(R)	:	0.3435701E-01	to	0.1723374E+01

Table A-2. Summary Results





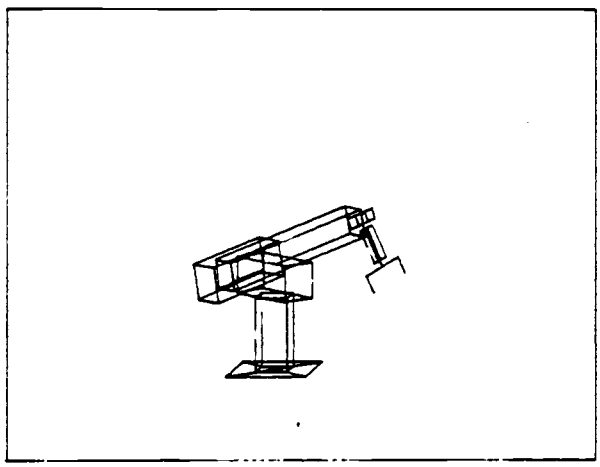
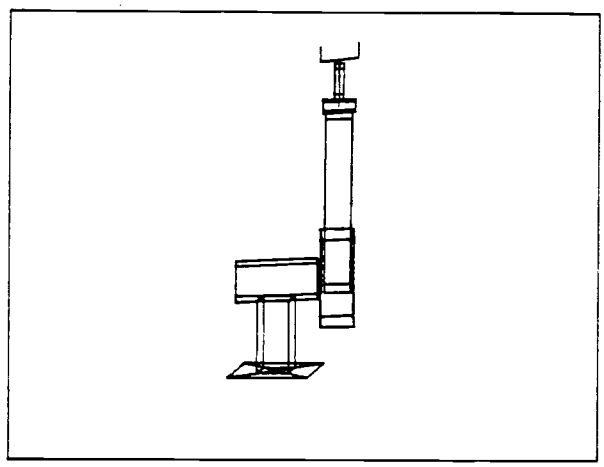
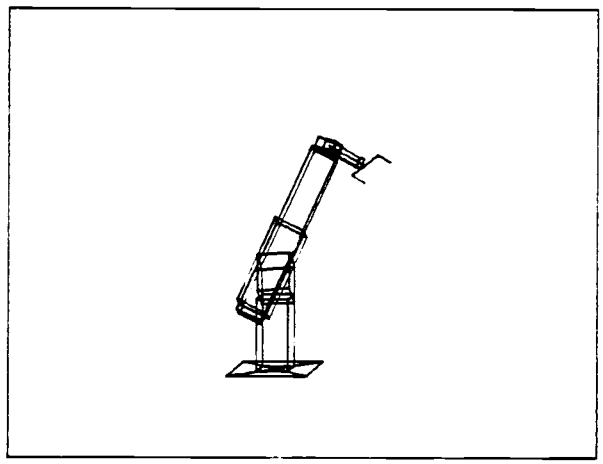


Figure A-1. End Points of Trajectory Segments





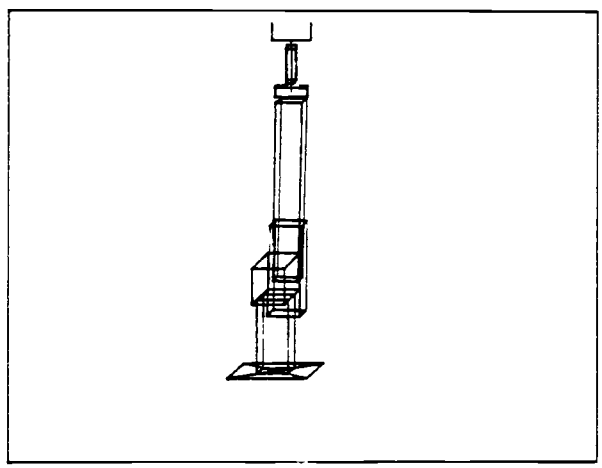


Figure A-1. (Contd.)

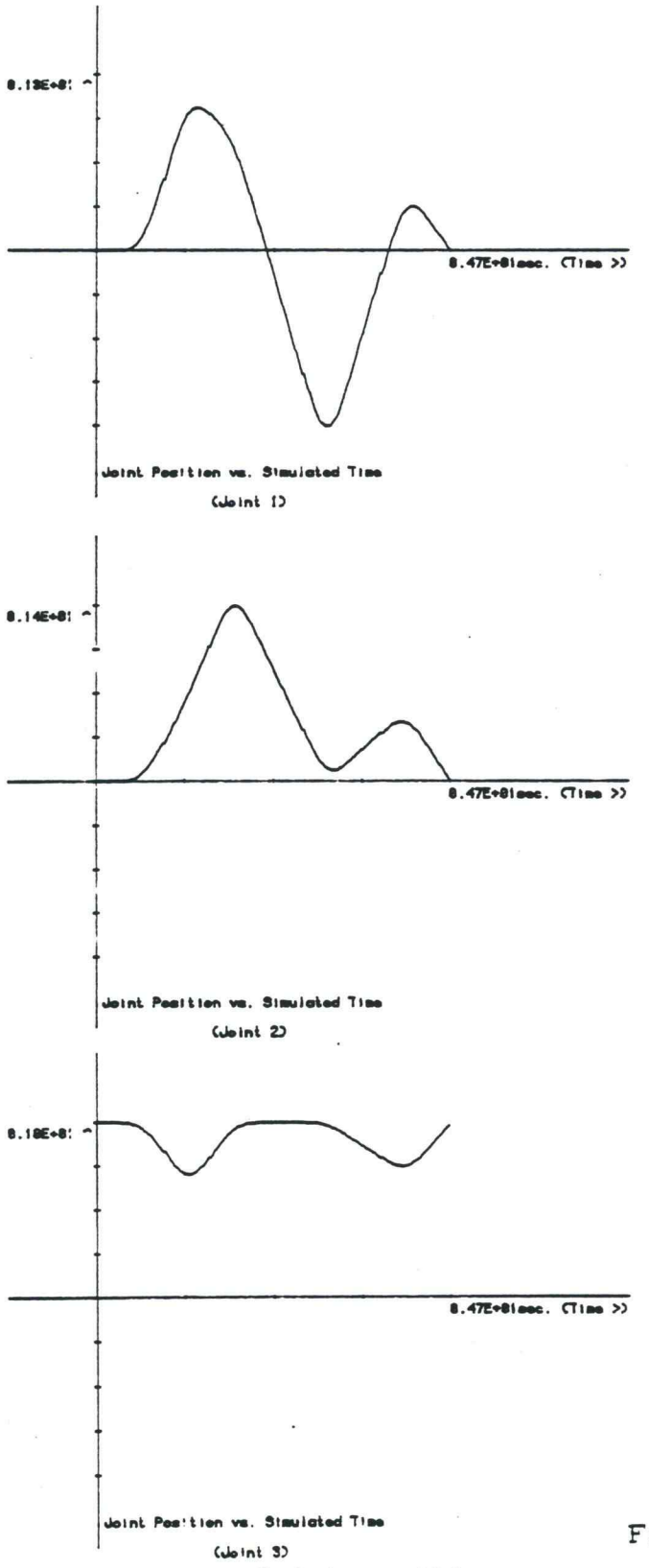
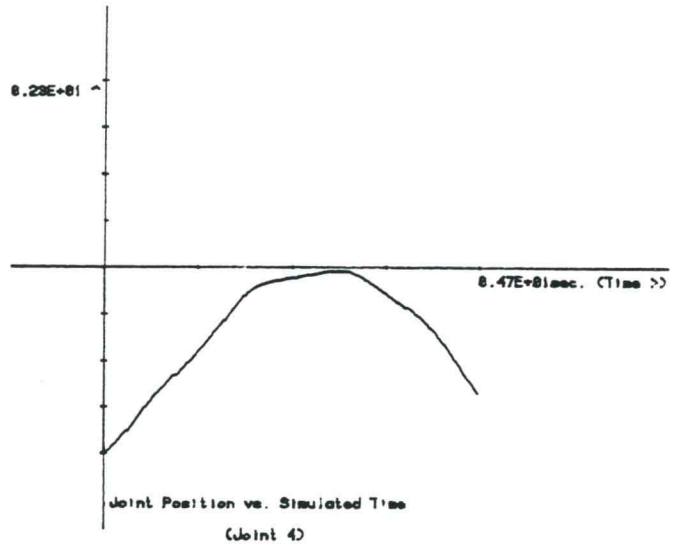
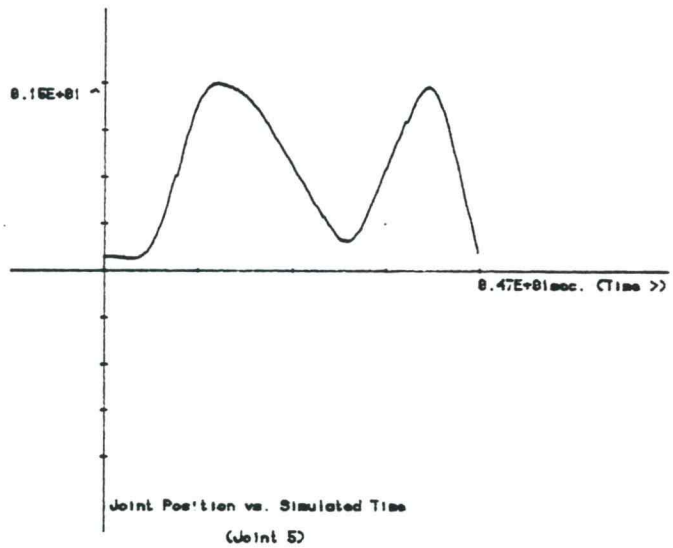


Figure A-2.
Joint position versus simulated time





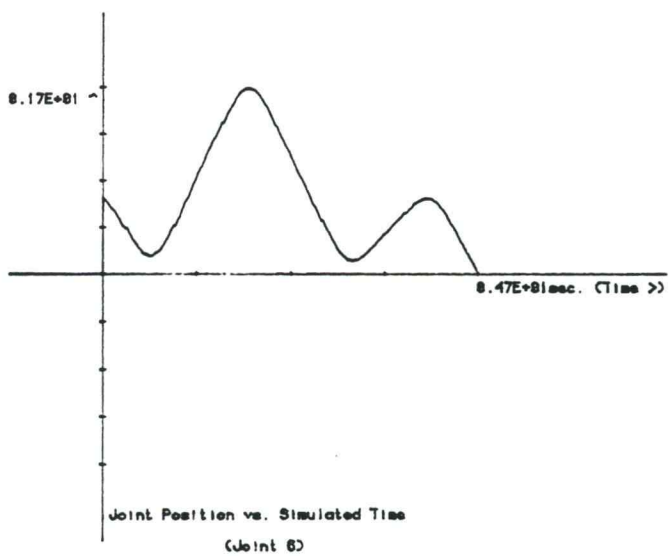


Figure A-2.
(Contd.)

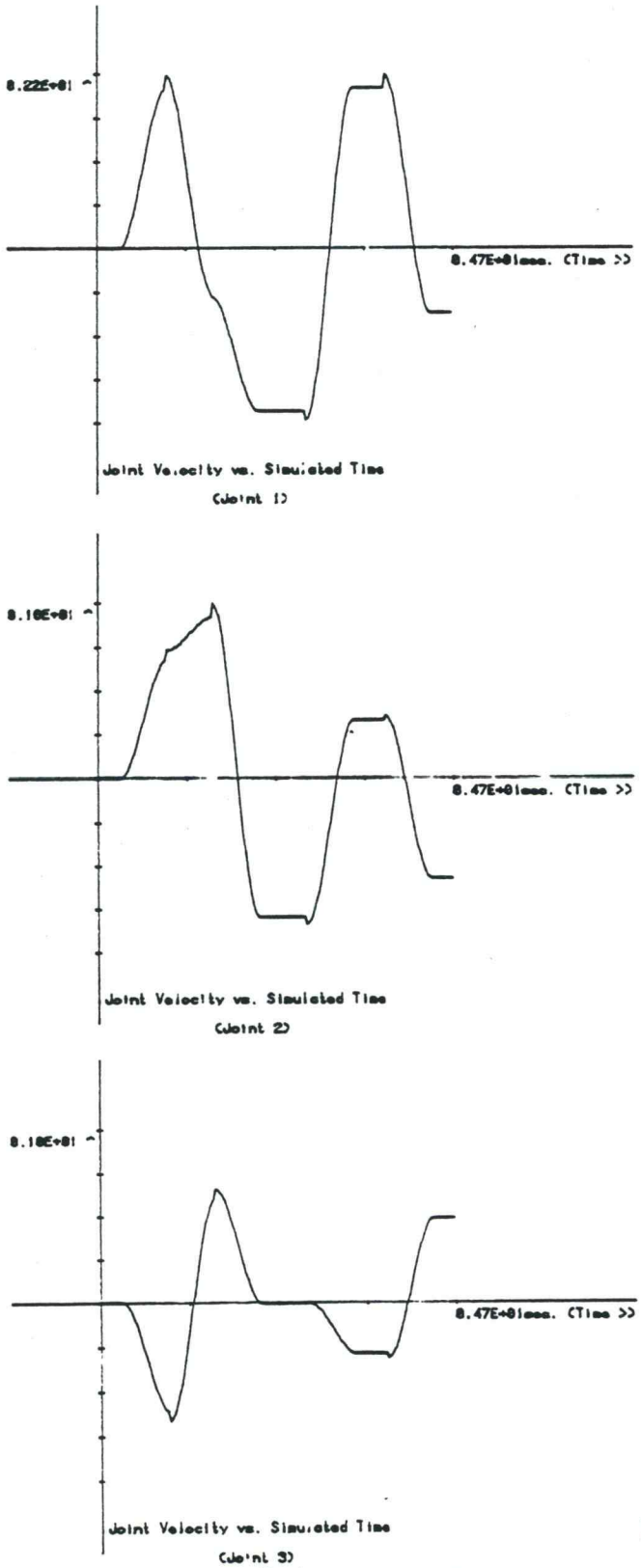
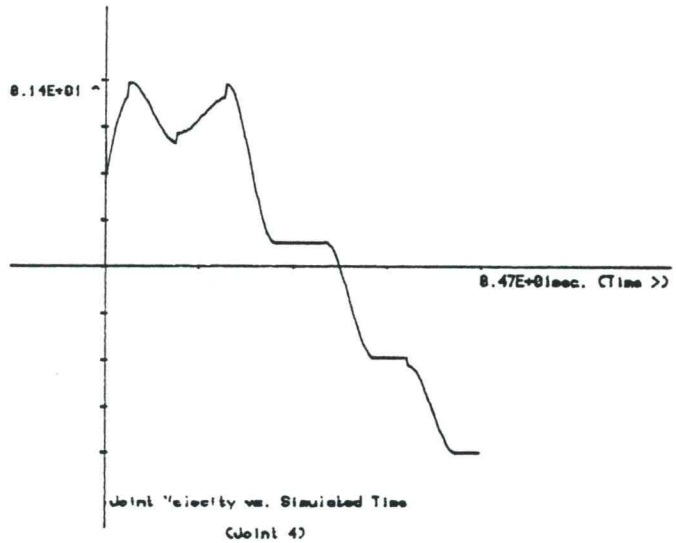
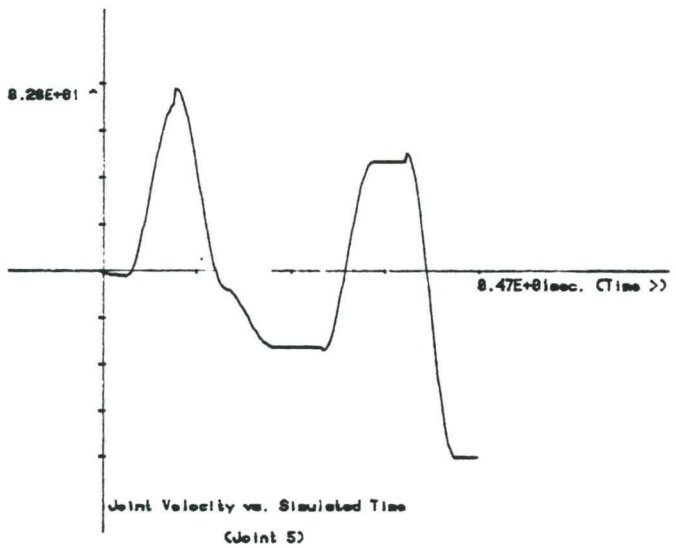


Figure A-3.
Joint velocity versus simulated time





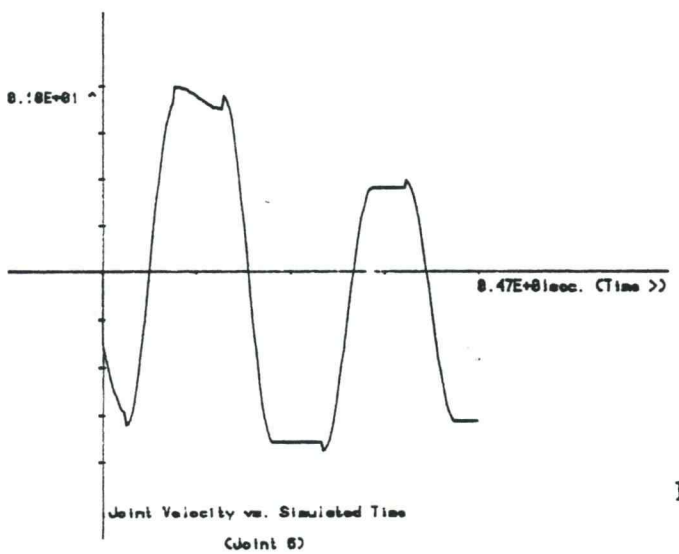


Figure A-3.
(Contd.)

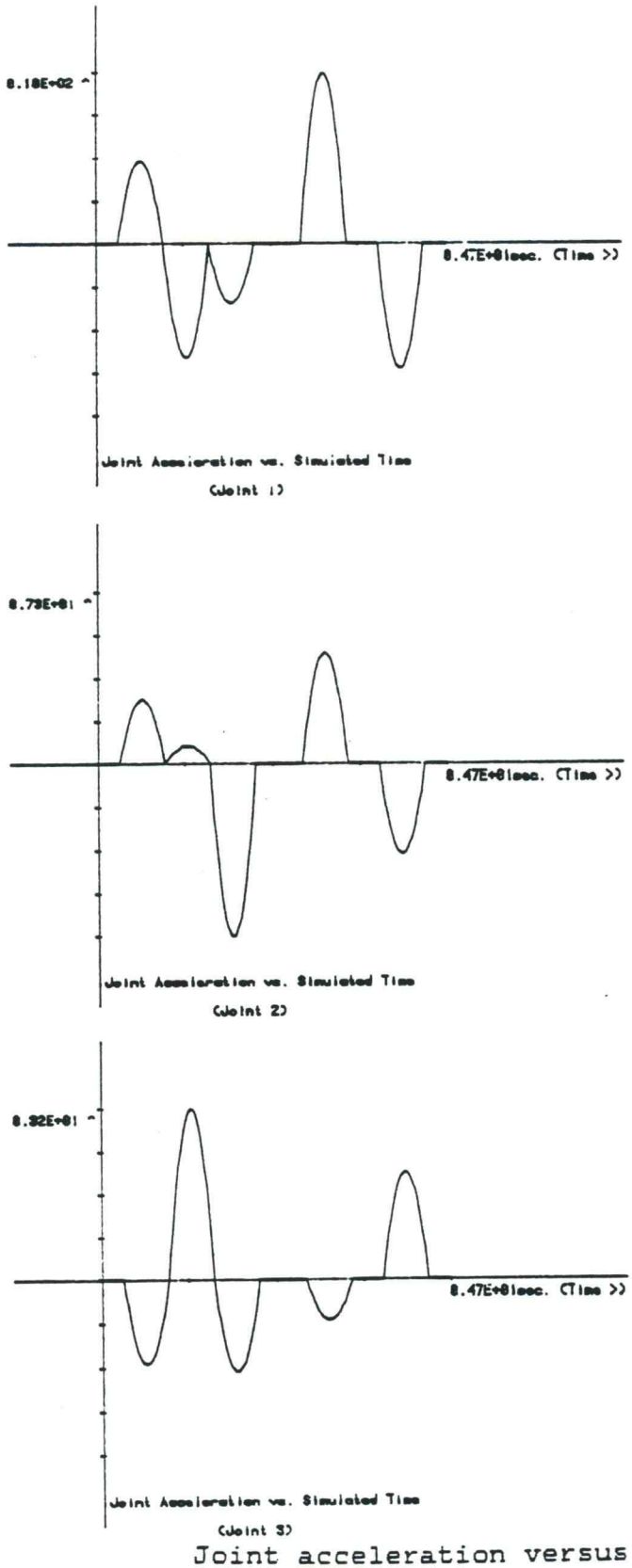
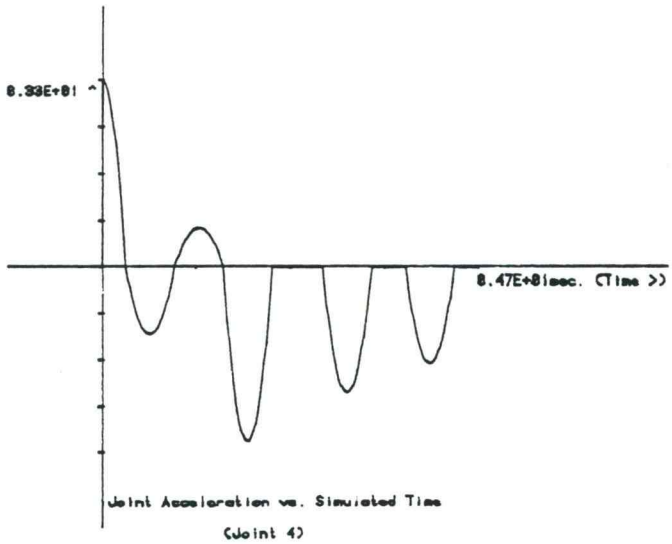
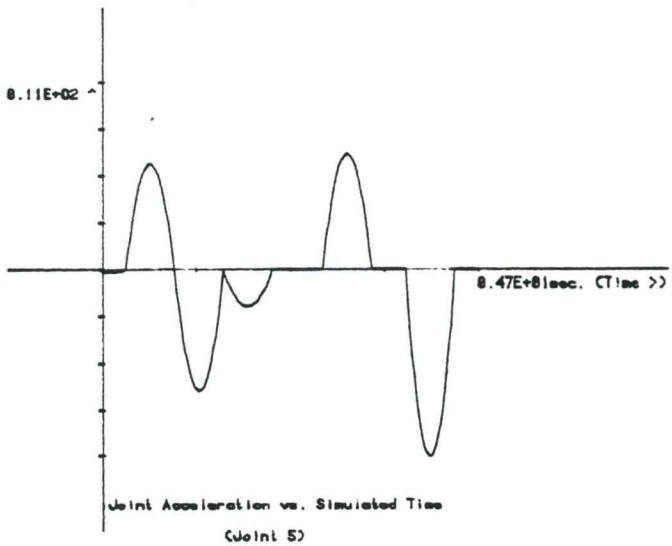


Figure A-4.
Joint acceleration versus simulated time





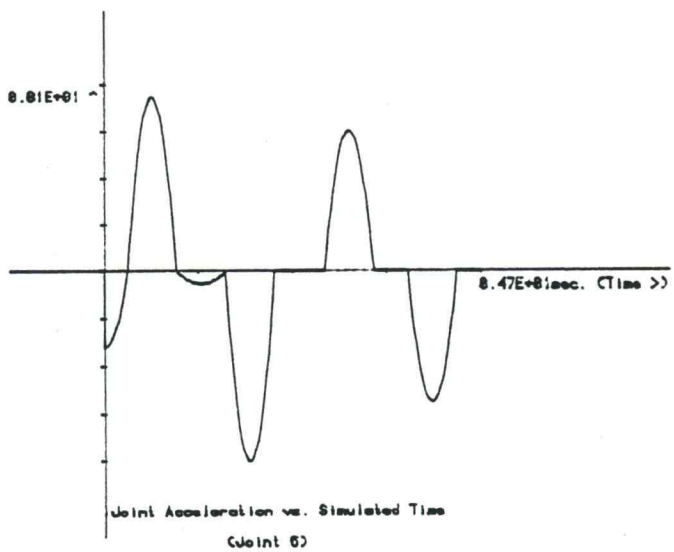


Figure A-4.
(Contd.)

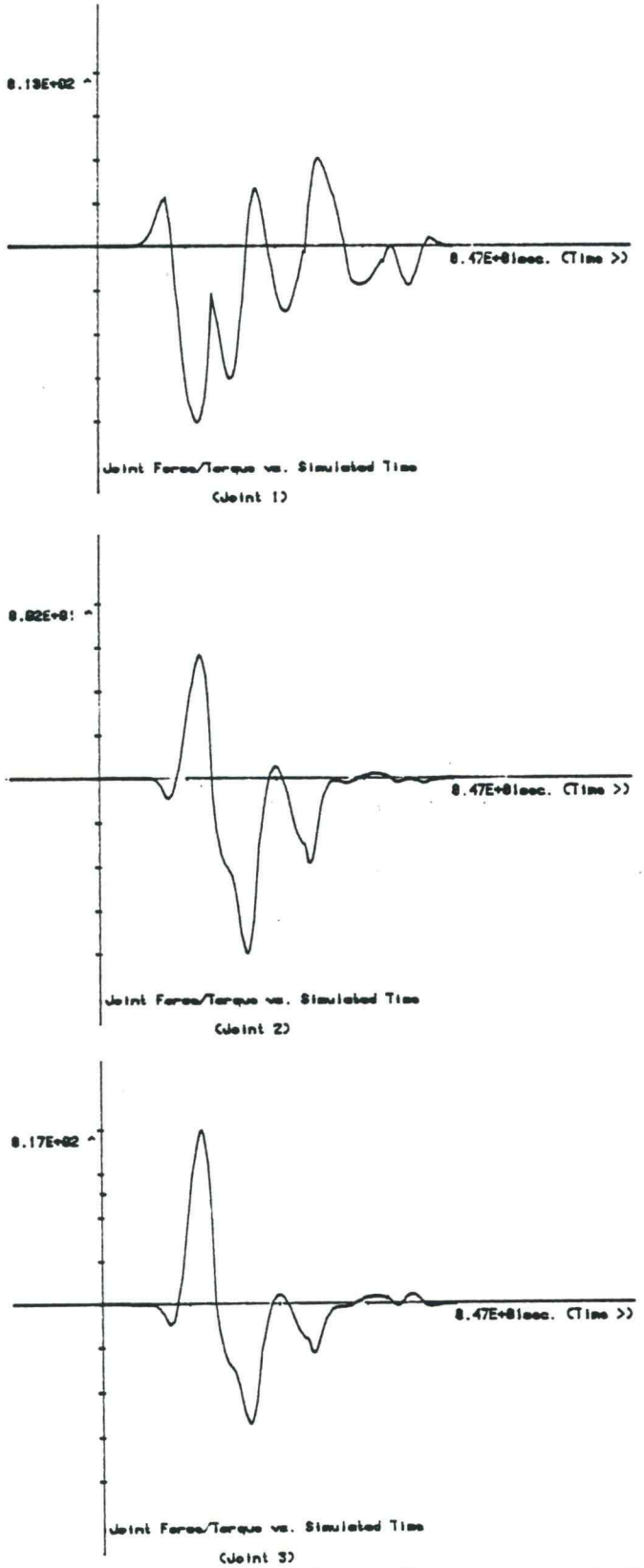
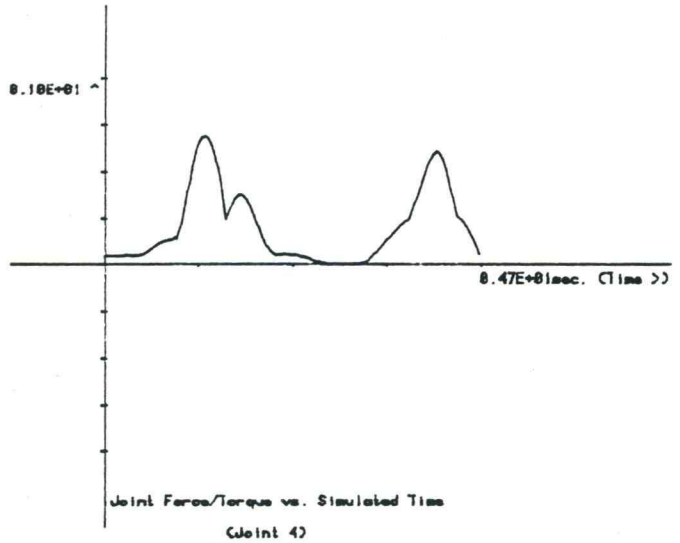
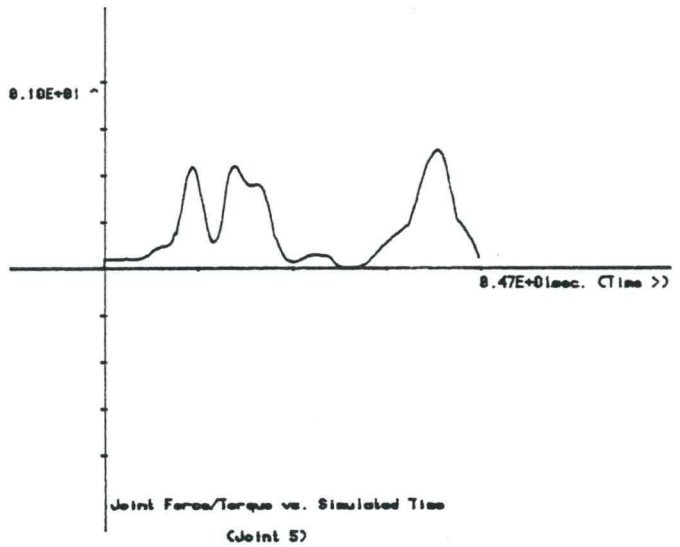


Figure A-5.
Joint force/torque versus simulated time





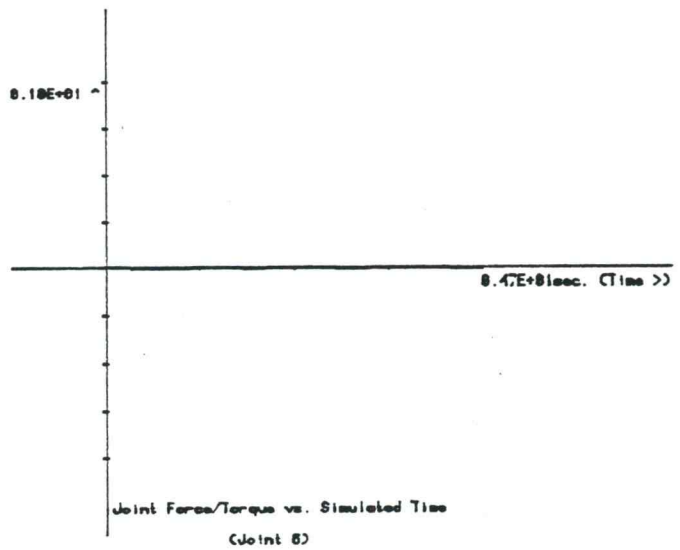


Figure A-5.
(Contd.)

LINK PARAMETERS:

Link	Theta	d	Alpha	a
1	0.0000	0.0000	90.0000	0.0000
2	90.0000	0.0000	90.0000	0.0000
3	90.0000	0.0000	90.0000	0.0000
4	0.0000	0.0000	90.0000	0.0000
5	0.0000	0.0000	90.0000	0.0000
6	0.0000	0.0000	0.0000	0.0000

LINK DIMENSIONS:

Link	Density	Outer dimensions			Inner dimensions		
	kg/cu.m	m	m	m	m	m	m
1	2460.000	0.300	0.100	0.300	0.200	0.100	0.200
2	2460.000	0.100	0.700	0.100	0.090	0.700	0.090
3	2460.000	0.100	0.700	0.100	0.090	0.700	0.090
4	2460.000	0.020	0.020	0.100	0.015	0.015	0.100
5	2460.000	0.020	0.020	0.100	0.015	0.015	0.100
6	2460.000	0.020	0.020	0.100	0.015	0.015	0.100

ACTUATOR SPECIFICATIONS:

Joint	Type	Max Velocity
1	P	1.0000
2	P	1.0000
3	P	1.0000
4	R	5.0000
5	R	5.0000
6	R	5.0000

Acceleration Time (TACC) : 0.3000
Time Increment (DT) : 0.0250

Table A-3. Link Parameters and Dimensions
PPPRRR Manipulator

DYNAMIC SIMULATION OF ROBOT MANIPULATORS

SUMMARY RESULTS :

Time taken for task execution (sec.) : 0.4225007E+01

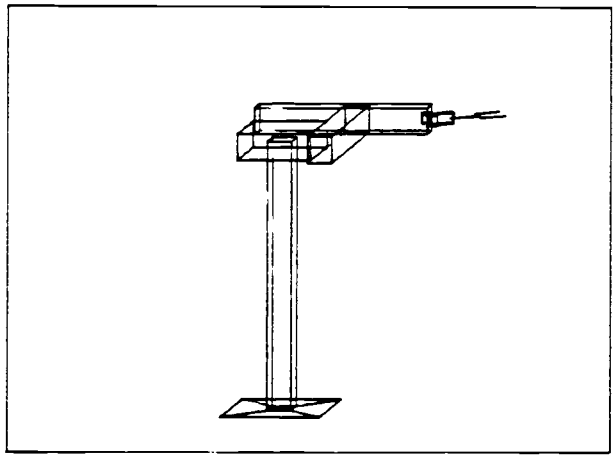
Maximum values for actuator Torque (Nm)/Force (N) :

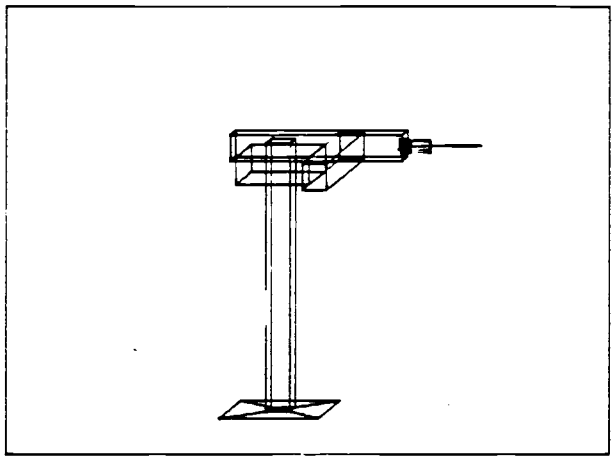
Joint(1)	(P)	:	0.2191642E+03
Joint(2)	(P)	:	-0.1321219E+02
Joint(3)	(P)	:	0.8939873E+01
Joint(4)	(R)	:	-0.3433928E-01
Joint(5)	(R)	:	0.2208594E+00
Joint(6)	(R)	:	0.5551437E-04

Extreme Positions in the trajectory (m/rad) :

Joint(1)	(P)	:	-0.1833334E+00	to	0.5000000E+00
Joint(2)	(P)	:	-0.4866584E+00	to	-0.2083333E+00
Joint(3)	(P)	:	0.3585883E+00	to	0.5875000E+00
Joint(4)	(R)	:	0.1188441E+01	to	0.2043266E+01
Joint(5)	(R)	:	0.7327282E-01	to	0.2729833E+01
Joint(6)	(R)	:	-0.2913444E+01	to	0.3141593E+01

Table A-4. Summary Results





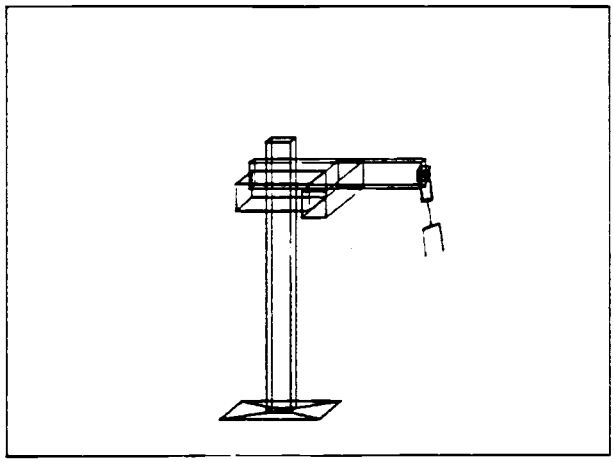
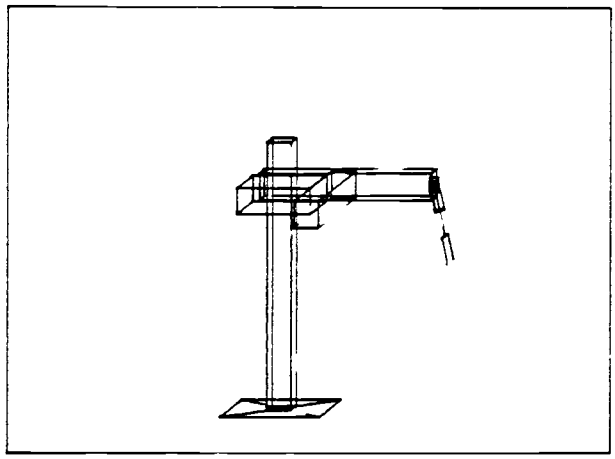
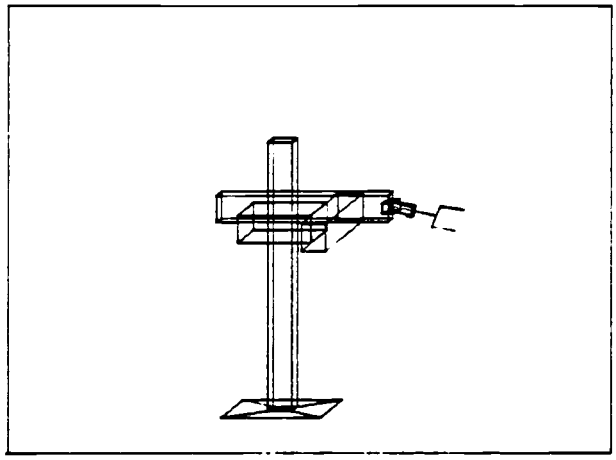


Figure A-6. End Points of Trajectory Segments





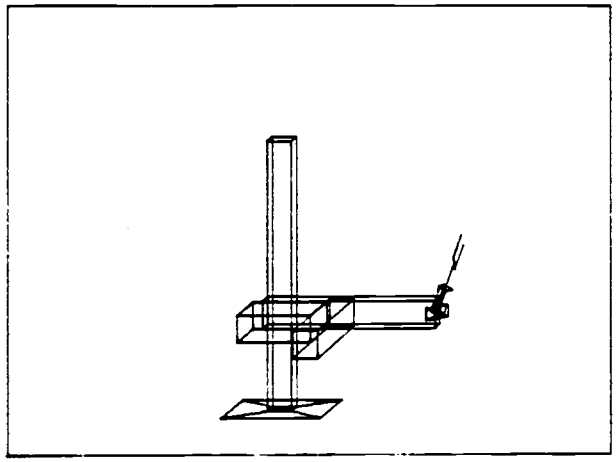


Figure A-6. (Contd.)

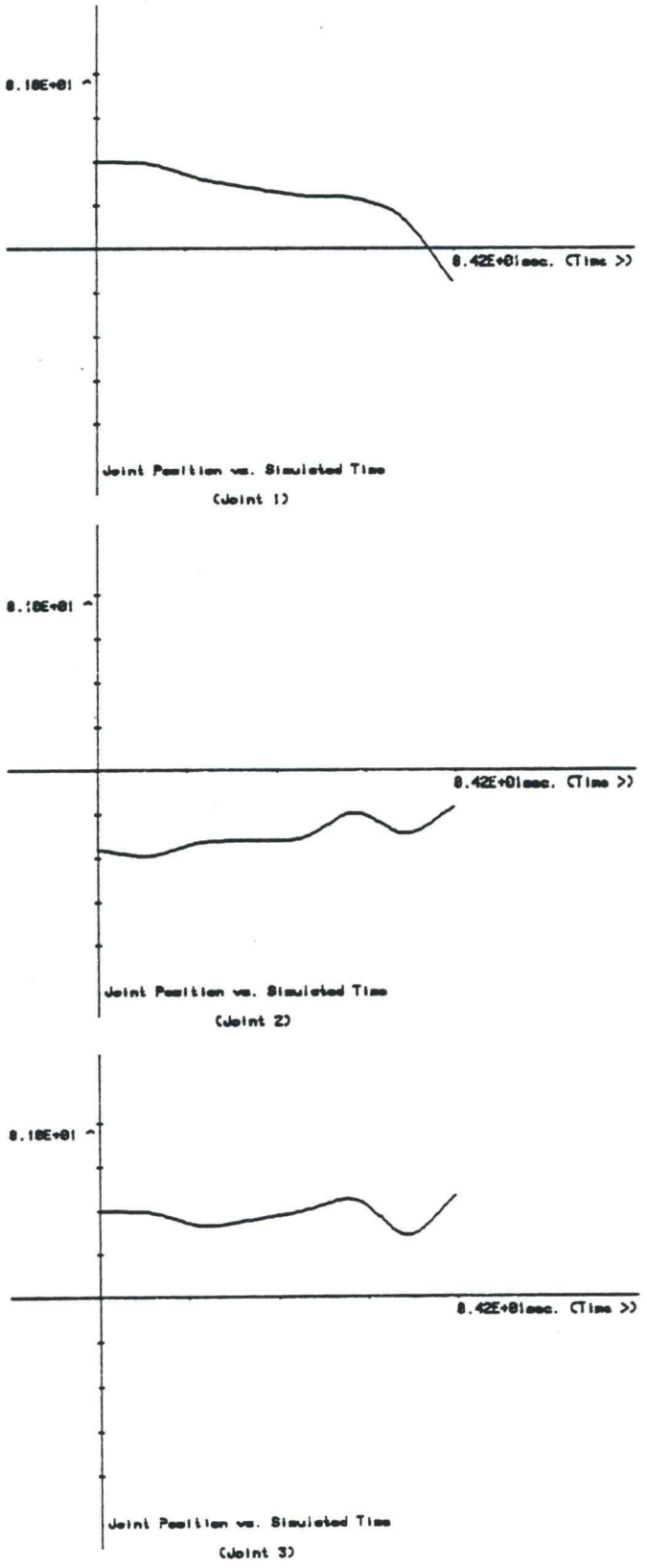
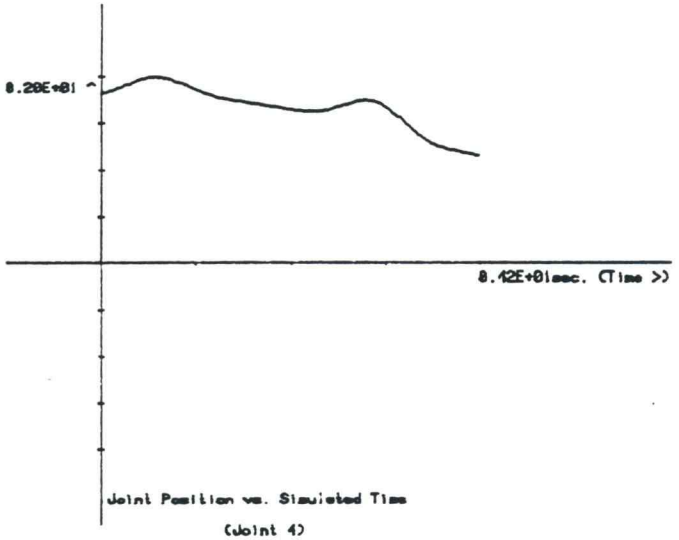
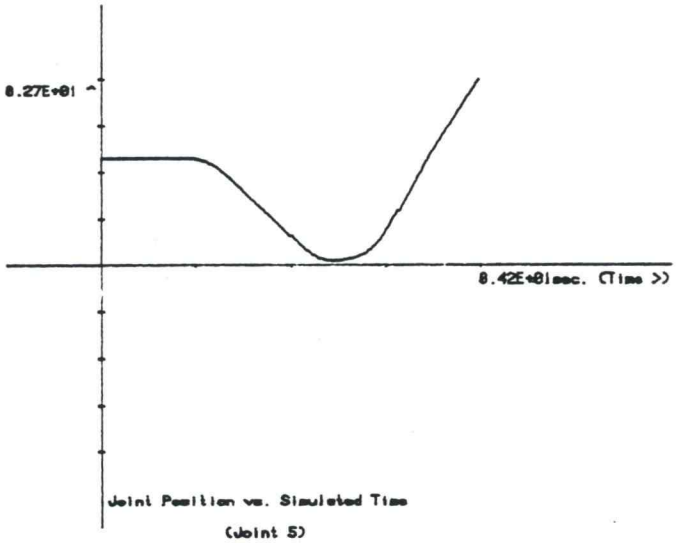


Figure A-7.
Joint position versus simulated time





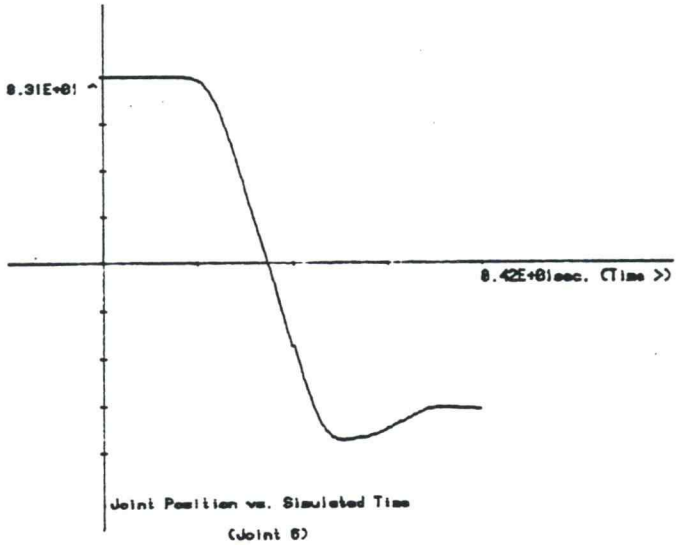


Figure A-7.
(Contd.)

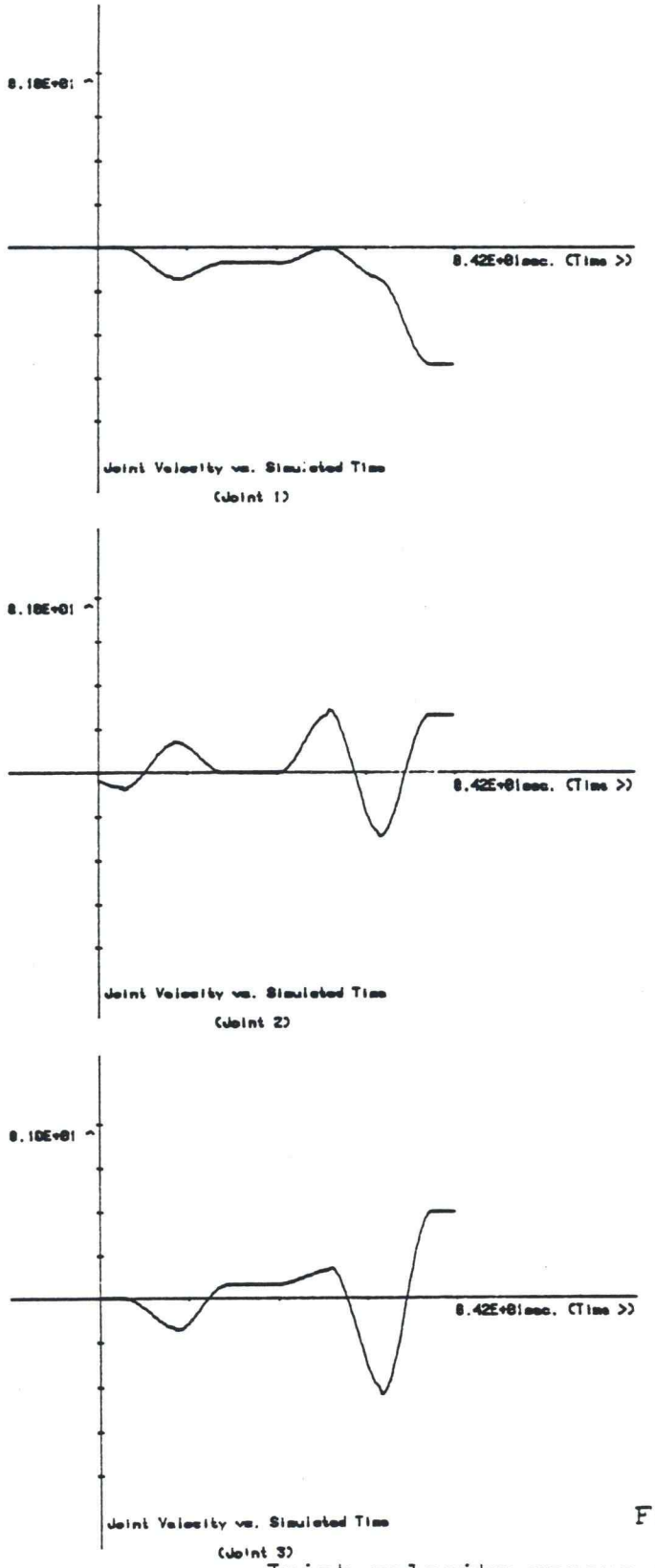
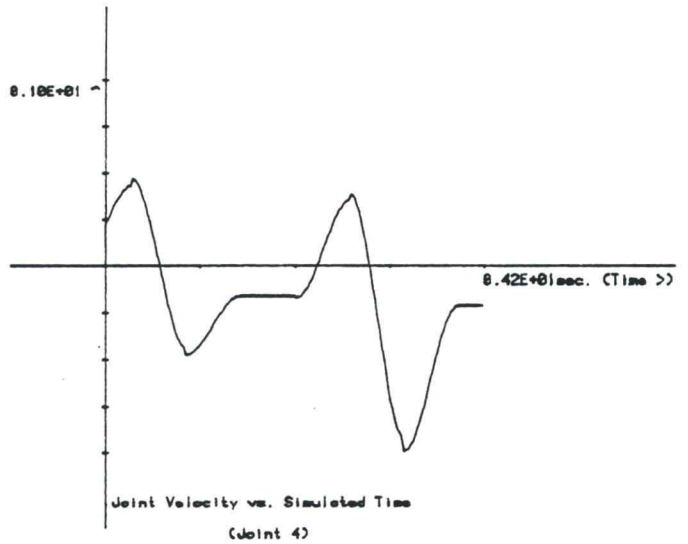
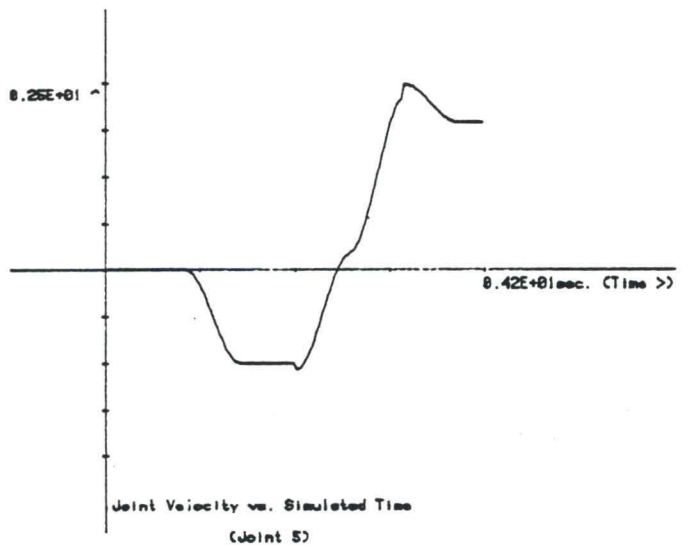


Figure A-8.
Joint velocity versus simulated time





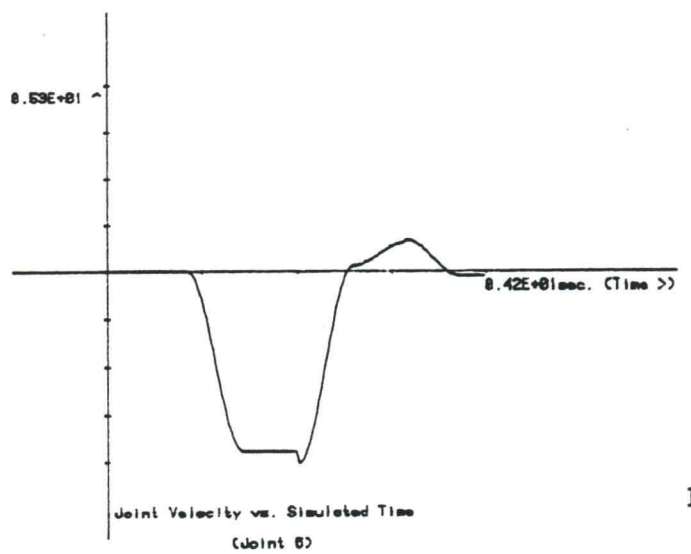


Figure A-8.
(Contd.)

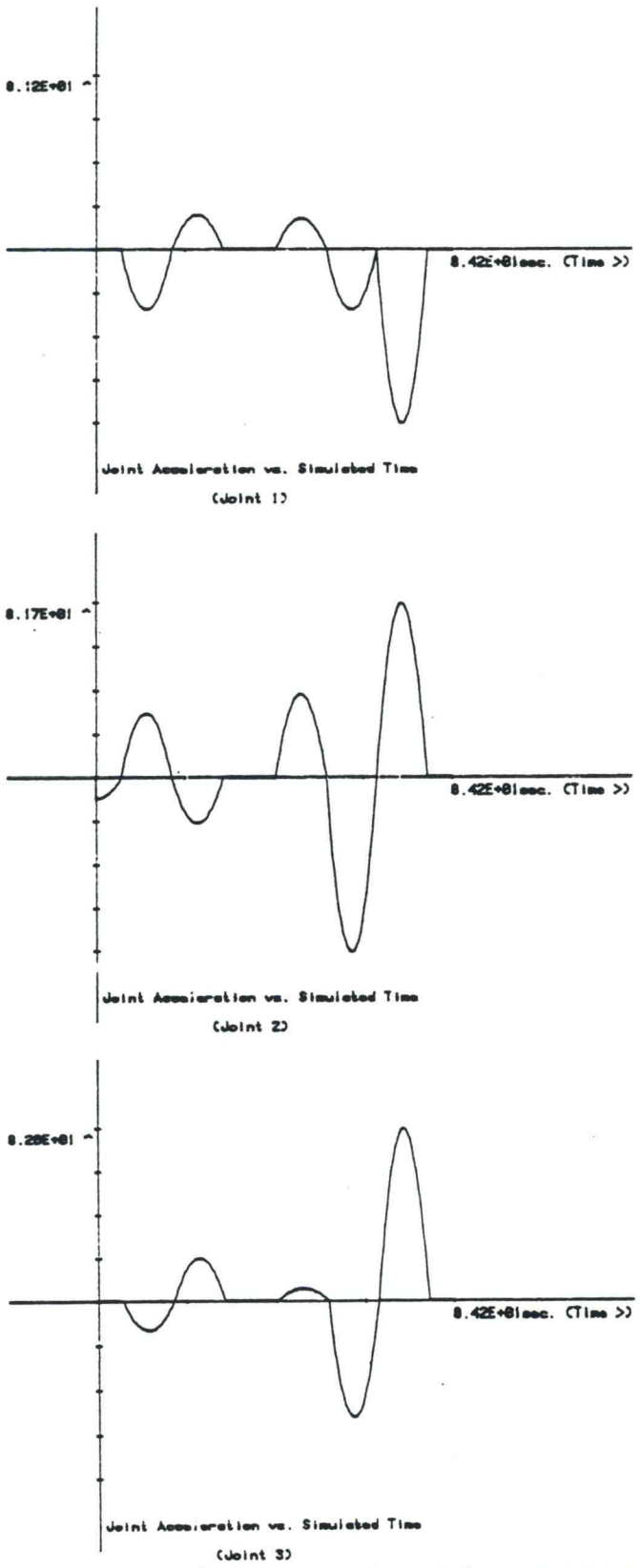
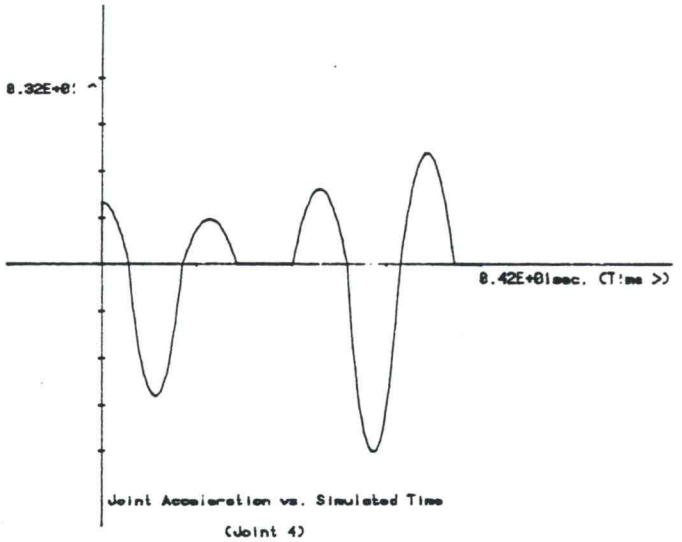
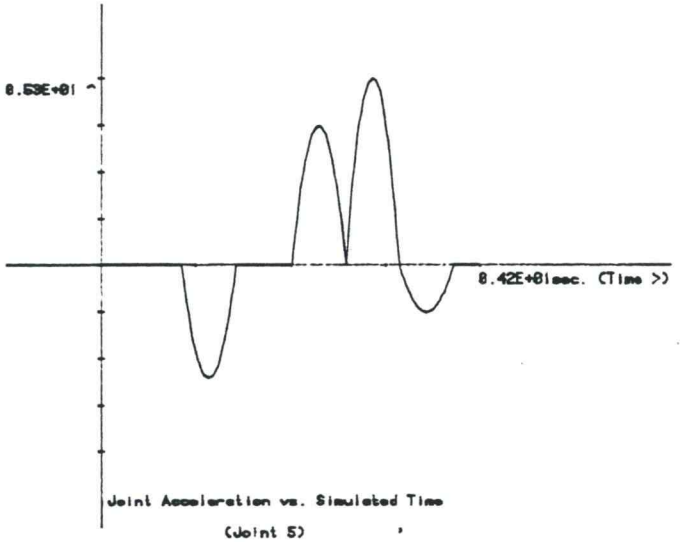


Figure A-9.
Joint acceleration versus simulated time





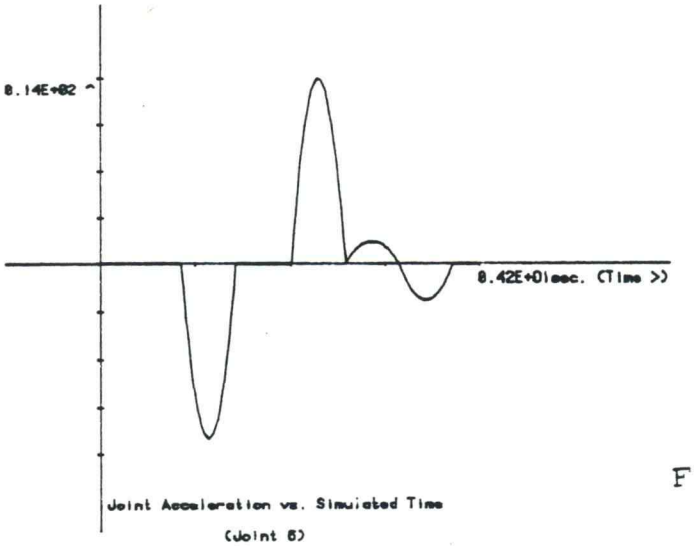


Figure A-9.
(Contd.)

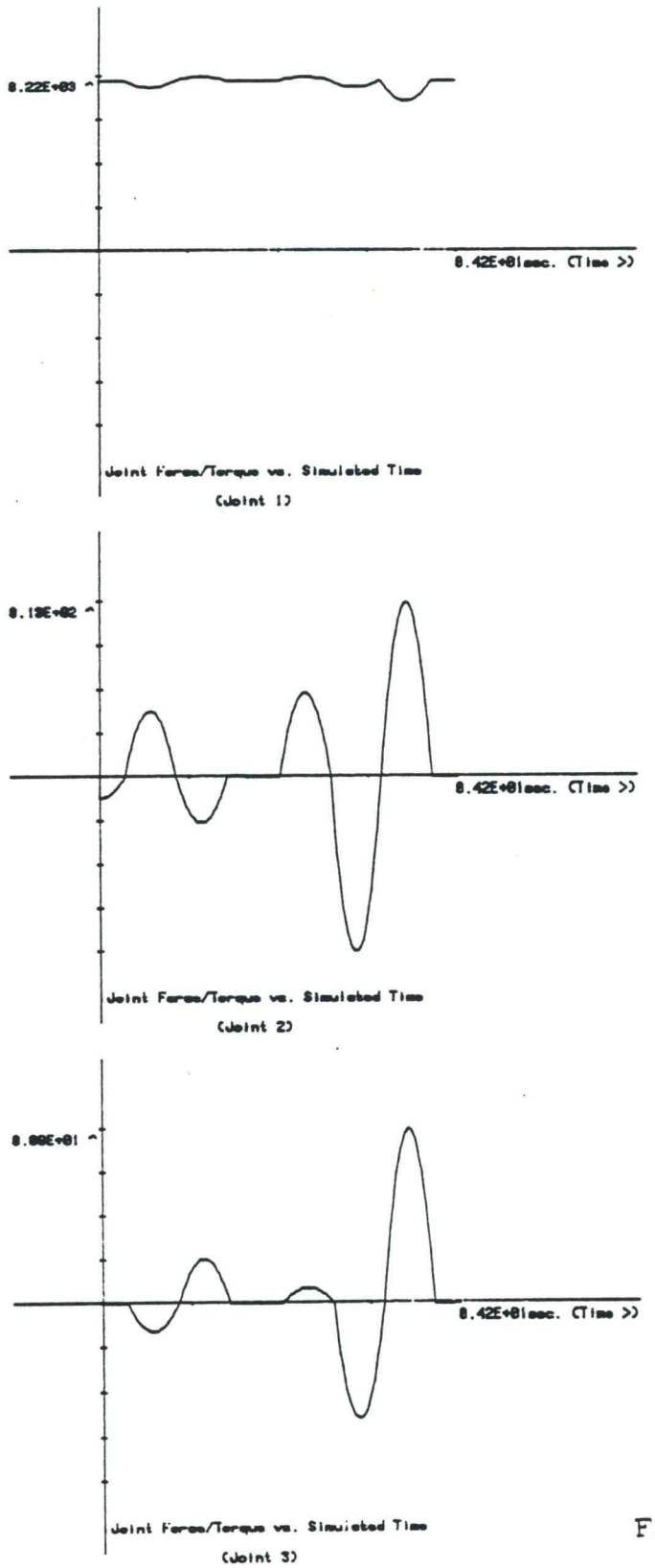


Figure A-10.
Joint force/torque versus simulated time

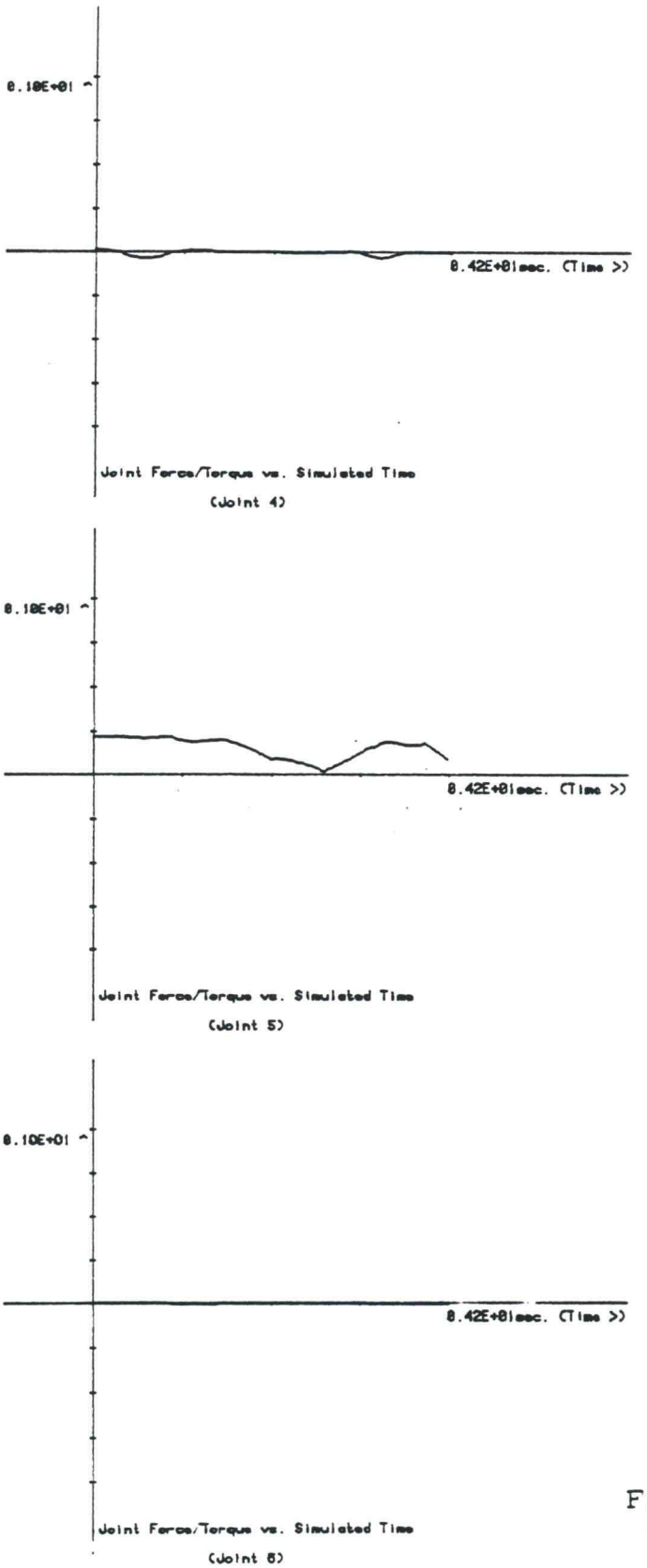


Figure A-10.
(Contd.)

LINK PARAMETERS:

Link	Theta	d	Alpha	a
1	0.0000	0.0000	90.0000	0.0000
2	0.0000	0.0000	0.0000	0.4000
3	0.0000	0.0000	0.0000	0.4000
4	0.0000	0.0000	-90.0000	0.0000
5	0.0000	0.0000	-90.0000	0.0000
6	0.0000	0.0000	0.0000	0.0000

LINK DIMENSIONS:

Link	Density	Outer dimensions			Inner dimensions		
	kg/cu.m	m	m	m	m	m	m
1	2460.000	0.050	0.200	0.050	0.045	0.200	0.045
2	2460.000	0.400	0.040	0.040	0.400	0.035	0.035
3	2460.000	0.400	0.035	0.035	0.400	0.030	0.030
4	2460.000	0.025	0.025	0.125	0.022	0.022	0.125
5	2460.000	0.025	0.025	0.125	0.022	0.022	0.125
6	2460.000	0.025	0.025	0.125	0.022	0.022	0.125

ACTUATOR SPECIFICATIONS:

Joint	Type	Max Velocity
1	R	1.0000
2	R	1.0000
3	R	1.2500
4	R	3.0000
5	R	3.0000
6	R	3.5000

Acceleration Time (TACC) : 0.3000
Time Increment (DT) : 0.0250

Table A-5. Link Parameters and Dimensions
RRRRRR Manipulator

DYNAMIC SIMULATION OF ROBOT MANIPULATORS

SUMMARY RESULTS :

Time taken for task execution (sec.) : 0.6150014E+01

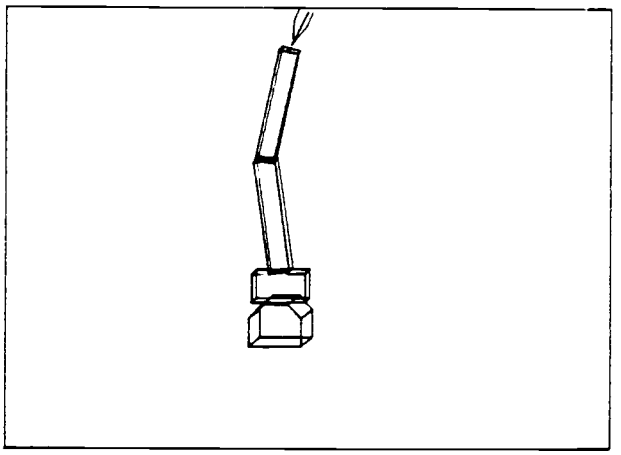
Maximum values for actuator Torque (Nm)/Force (N) :

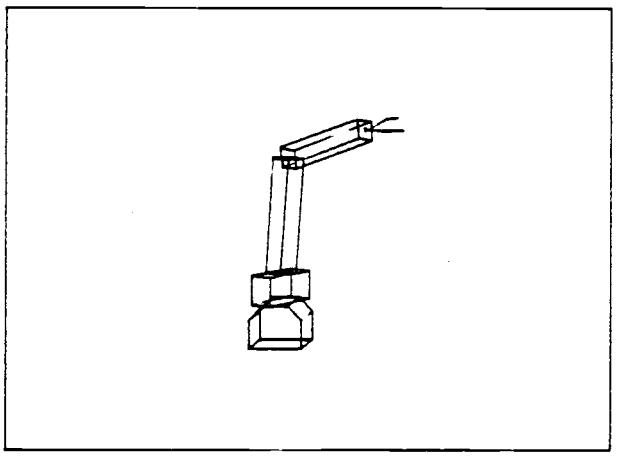
Joint(1)	(R)	:	0.5635633E+00
Joint(2)	(R)	:	0.4227077E+01
Joint(3)	(R)	:	0.1749426E+01
Joint(4)	(R)	:	0.1145901E+00
Joint(5)	(R)	:	-0.1060445E+00
Joint(6)	(R)	:	0.5921144E-04

Extreme Positions in the trajectory (m/rad) :

Joint(1)	(R)	:	-0.7195819E+00	to	0.9933083E+00
Joint(2)	(R)	:	0.7981268E+00	to	0.1775524E+01
Joint(3)	(R)	:	-0.1570796E+01	to	-0.3302598E+00
Joint(4)	(R)	:	-0.1980044E+00	to	0.2920215E+01
Joint(5)	(R)	:	-0.2857759E+01	to	0.1603589E+01
Joint(6)	(R)	:	-0.2312856E+01	to	-0.1056485E+01

Table A-6. Summary Results





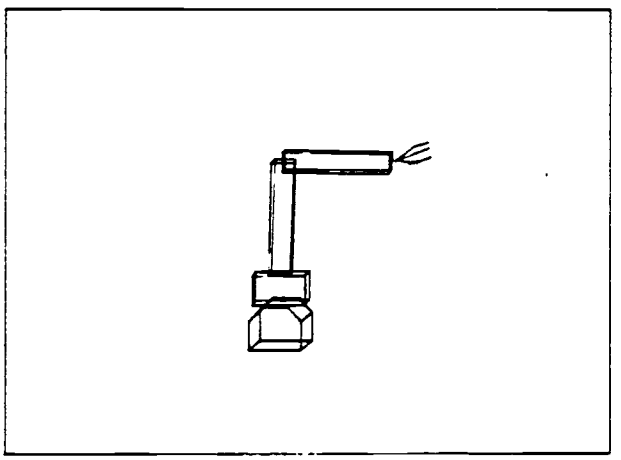
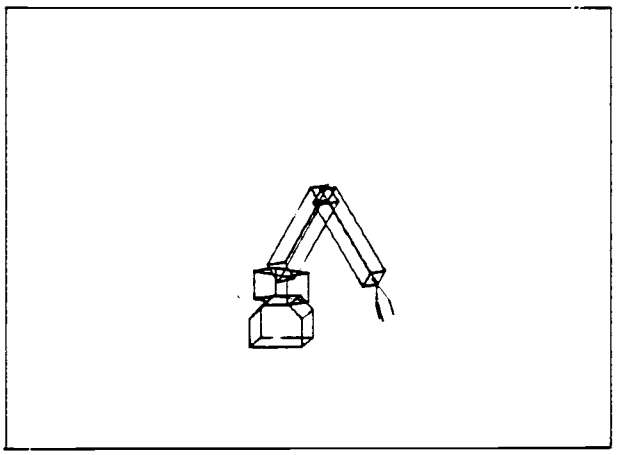
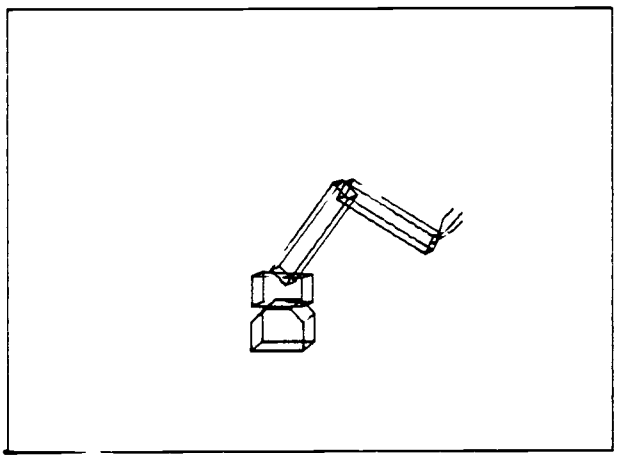


Figure A-11. End Points of Trajectory Segments





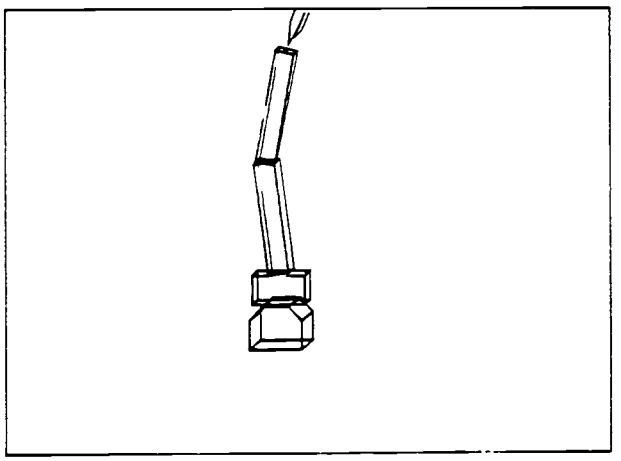


Figure A-11. (Contd.)

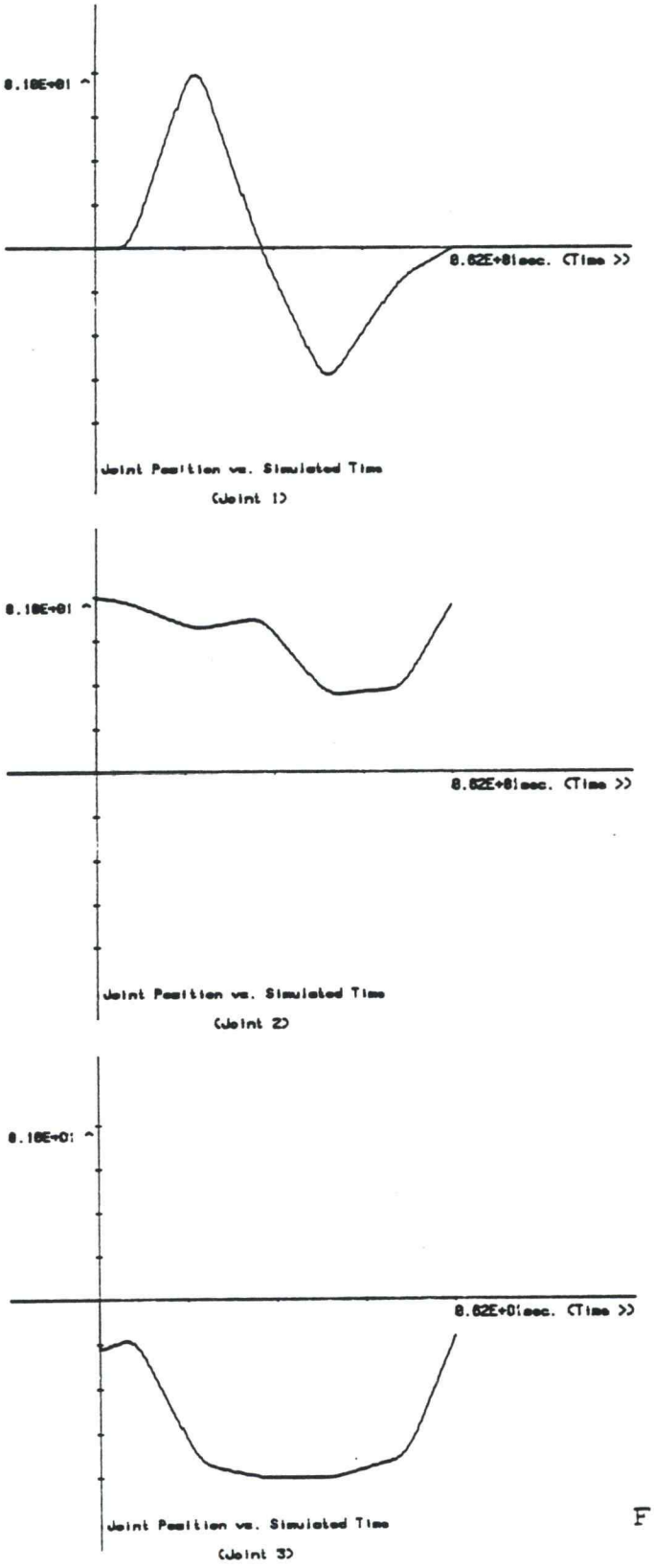


Figure A-12.
Joint position versus simulated time

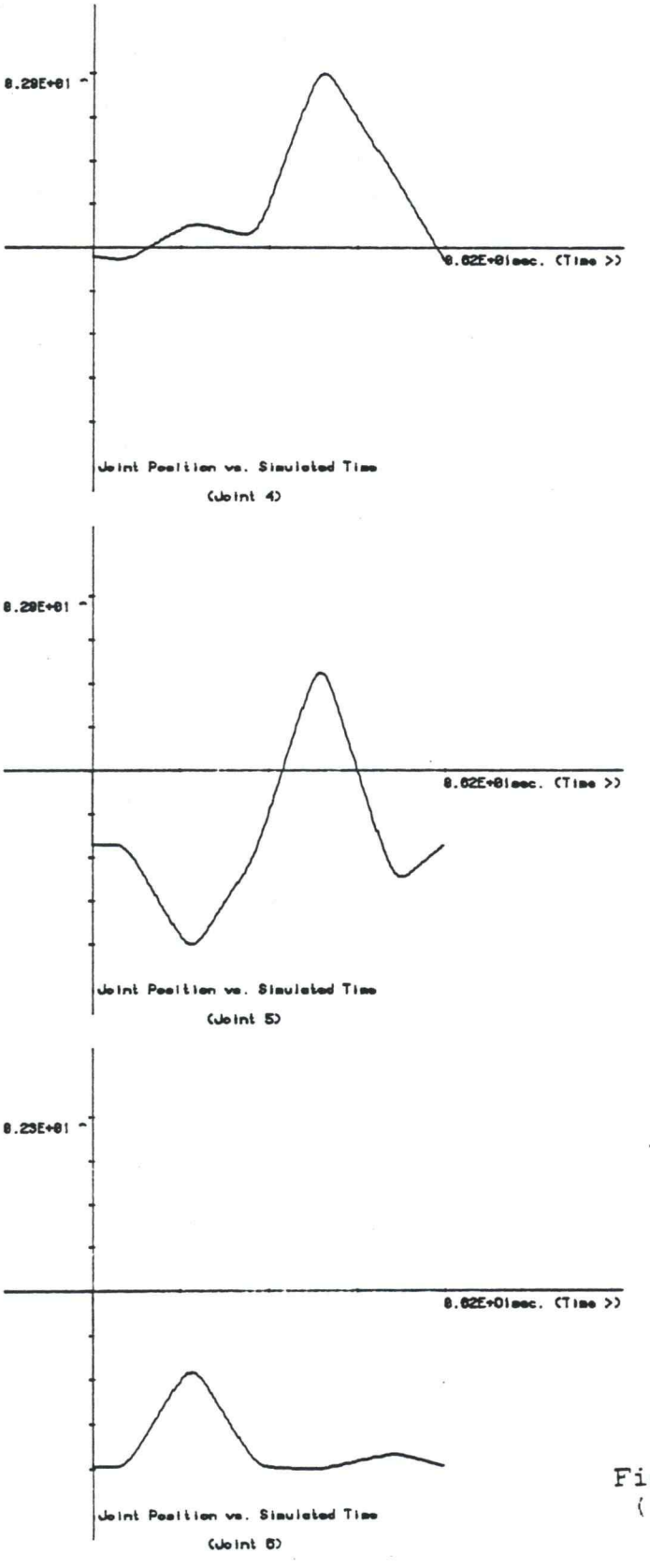


Figure A-12.
(Contd.)

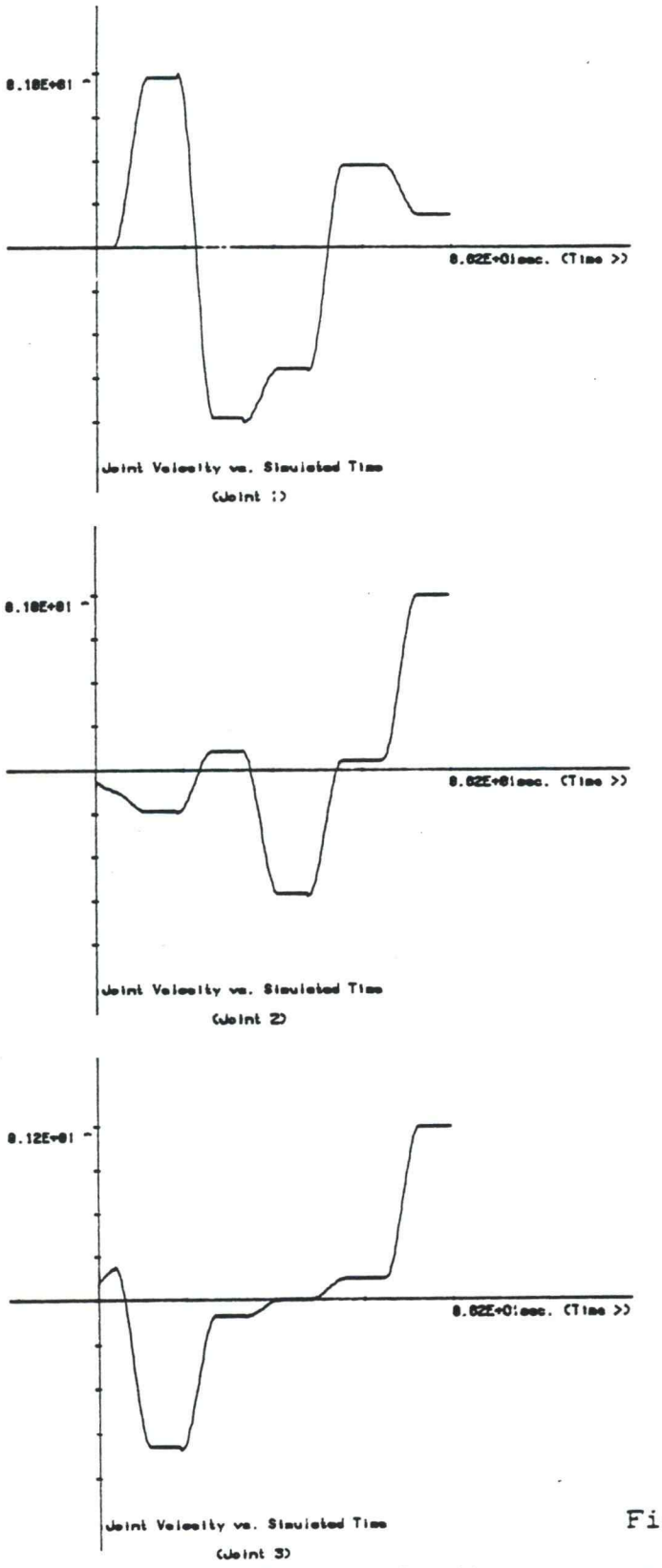
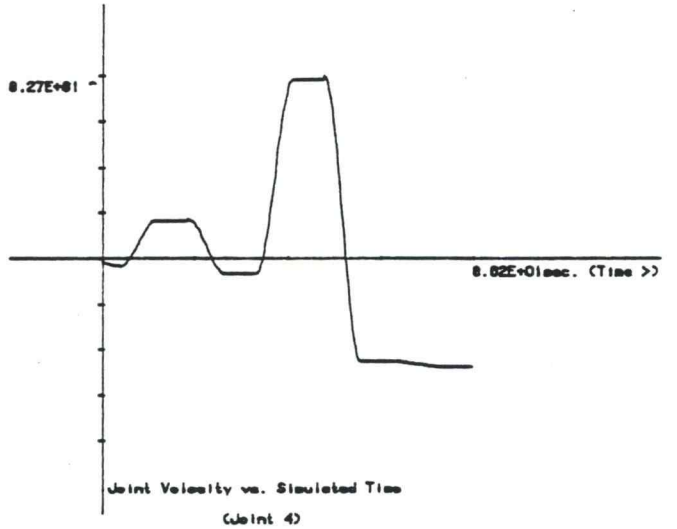
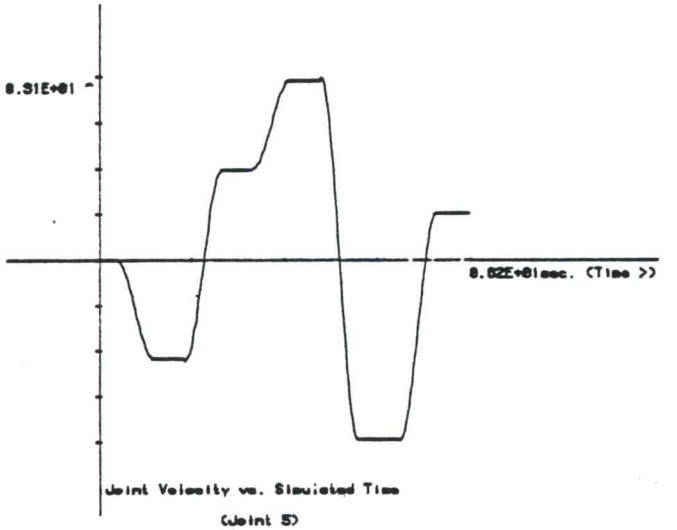


Figure A-13.
Joint velocity versus simulated time





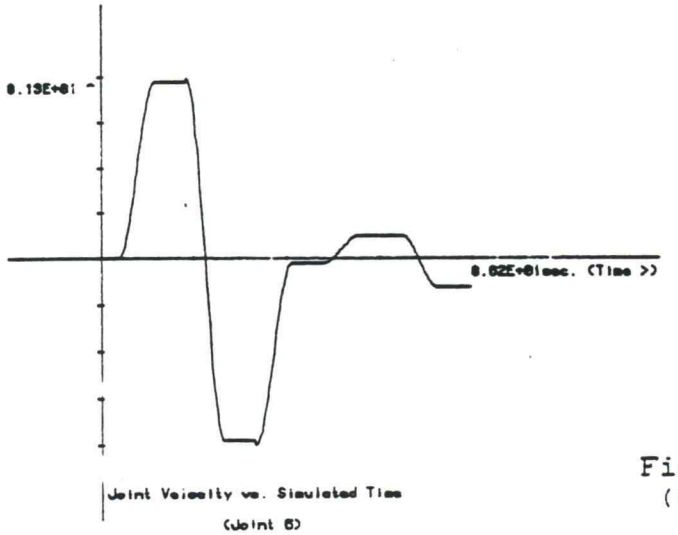


Figure A-13.
(Contd.)

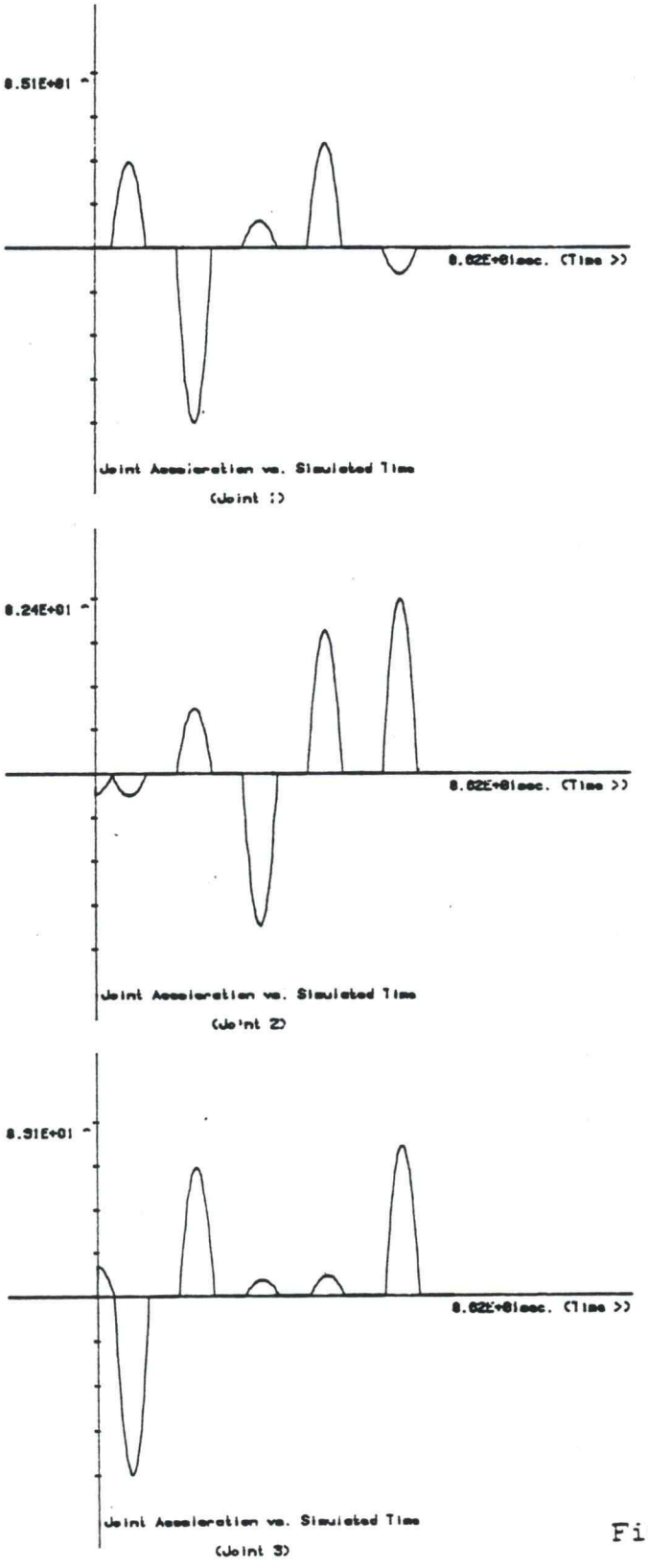


Figure A-14.
Joint acceleration versus simulated time

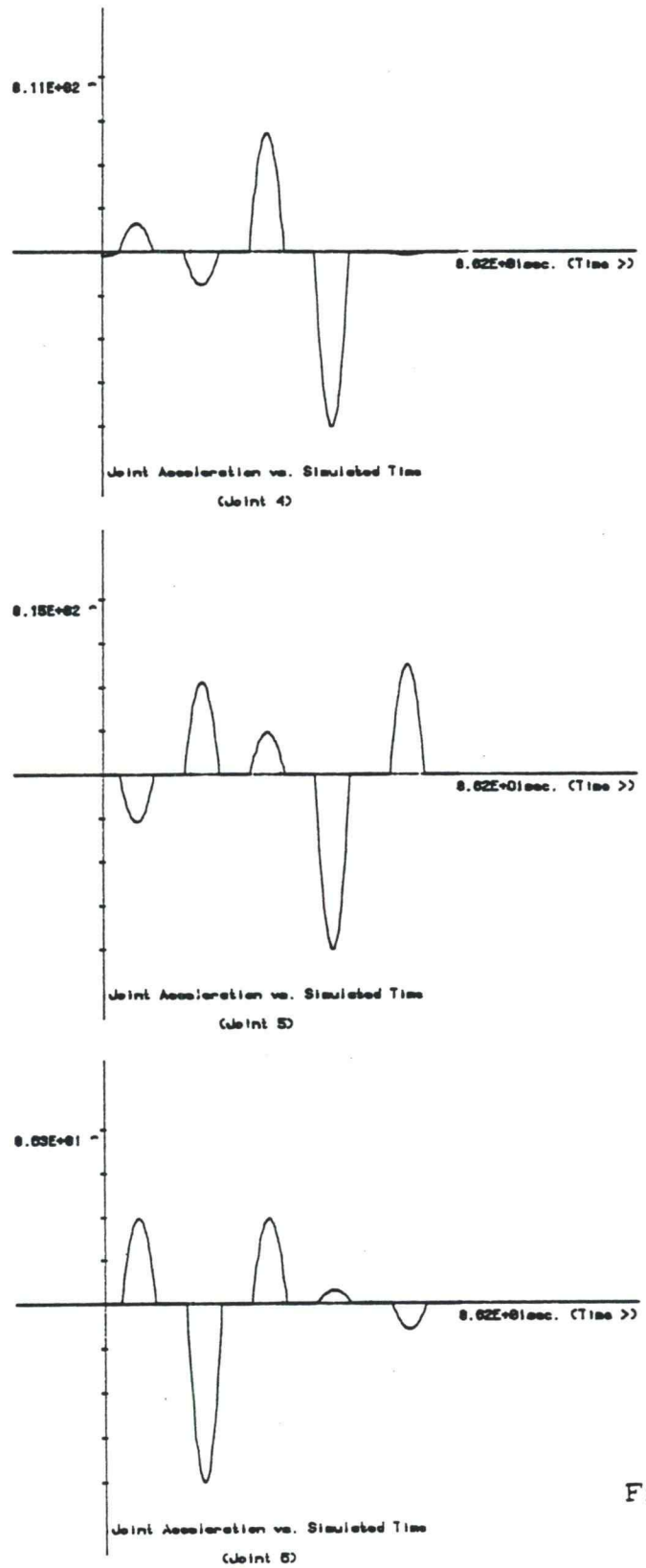


Figure A-14.
(Contd.)

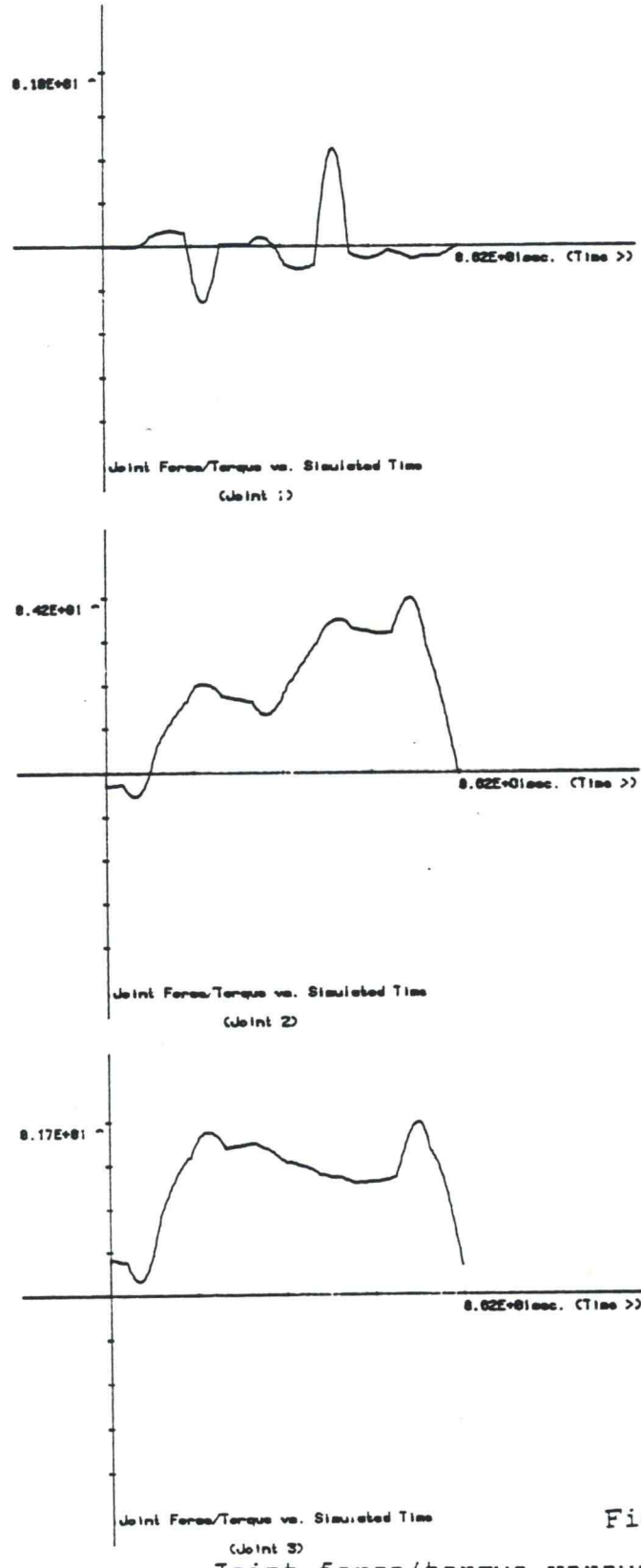
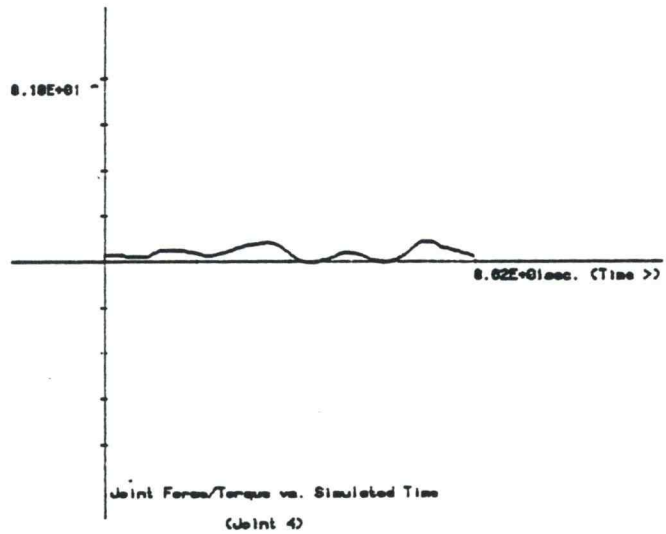
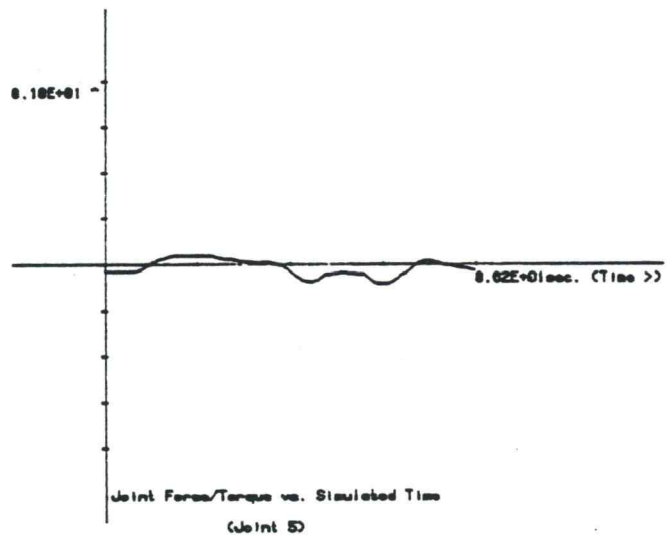


Figure A-15.
Joint force/torque versus simulated time





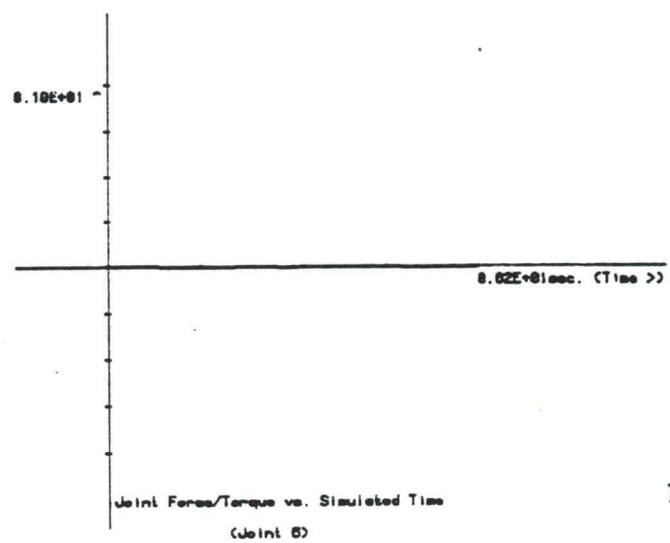


Figure A-15.
(Contd.)