SPACE MANIPULATOR KINEMATICS FOR DOCKING OPERATIONS

By

Osman Parlaktuna

Dissertation
Submitted to the Faculty of the
Graduate School of Vanderbilt University
In partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

May, 1990

Nashville, Tennessee

Approved: Date:

April 9, 1990

4/9/90

4/9/90

4/9/90

4/9/90

NASA EXT. April 9, 1990

ELECTRICAL ENGINEERING

SPACE MANIPULATOR KINEMATICS FOR DOCKING OPERATIONS

OSMAN PARLAKTUNA

Dissertation under the direction of Dr. George E. Cook

Robotic manipulators carried by future spacecraft are expected to perform important tasks in space, like satellite assembly. However, since the base of the manipulator (satellite) is not fixed in space, some problems will be encountered. Internal forces and torques applied to the satellite due to the motion of the manipulator will cause changes in the position and attitude of the base. The control method for space manipulators should take into account the motion of the base due to manipulator motions. In this study, the kinematics of a space manipulator system is investigated and a control method based on resolved motion rate control and conservation of angular momentum is developed. It is shown that a manipulator arm follows a prescribed path despite the translations and rotations of the base. It is also shown that manipulator arm motions can be used to correct the orientation of the base. An algorithm is developed assuming that manipulator motions cause small rotations in the attitude of the base and these rotations are additive. This algorithm is used to find the required motions of the manipulator to rotate the base from an initial orientation to a desired final orientation. Results are animated by using a computer graphic simulator program, ROBOSIM.

Approved ______________________________ Date ______________________________

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Prof. George E. Cook, for his invaluable support and continual guidance, and for his encouragement and advice throughout this study. I wish to extend my appreciation to Dr. Ken Fernandez for his suggestions.

Last but not least, my deep gratitude and appreciation go to my wife, İnci Parlaktuna, for her support and understanding. Without her support my years at Vanderbilt would have been more difficult.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

${}^{i-1}\mathbf{U}_i =$ 3×3 rotational transformation matrix between link $i-1$ and link i.

${}^{i-1}\mathbf{A}_i =$ 4×4 homogeneous transformation matrix between link $i-1$ and link i

$\mathbf{T}_i =$ 4×4 homogeneous transformation matrix between link i and inertial frame.

$\tau, \beta, \psi =$ attitude angles of satellite main body (yaw, pitch, roll).

$a_i =$ distance between the origins of coordinate systems $i-1$ and i measured along $\mathbf{x}_i$.

$d_i =$ distance between $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$ measured along $\mathbf{z}_{i-1}$.

$\alpha_i =$ angle between $\mathbf{z}_{i-1}$ and $\mathbf{z}_i$ axes measured about $\mathbf{x}_i$.

$\theta_i =$ angle between $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$ axes measured about $\mathbf{z}_{i-1}$.

$\mathbf{p}_i =$ position vector of coordinate frame i with reference to the inertial frame.

${}^{i-1}\mathbf{p}_i =$ position vector of coordinate frame i with reference to the inertial frame.

${}^{i}\mathbf{r}_i =$ position vector of the center of mass of link i with reference to coordinate frame i.

$\mathbf{r}_i =$ position vector of the center of mass of link i with reference to inertial frame.

$\mathbf{r}_G =$ position vector of the center of mass of the system with reference to the inertial frame.

$\mathbf{R} =$ position vector of the center of mass of the system with reference to a satellite fixed coordinate system.

$\mathbf{I}_i =$ inertia matrix of link i about a coordinate frame parallel to the inertial frame.

$\mathbf{w}_i =$ Angular velocity of link i with reference to the inertial frame.

$\mathbf{P} =$ 6×1 position and attitude vector of the end effector of the manipulator.

$\mathbf{q} =$ $n \times 1$ vector of joint variables of the manipulator.

$\mathbf{I}_s$= 3×3 inertia matrix of the satellite main body.

$\mathbf{I}_m$= $3 \times n$ inertia matrix of the manipulator.

$\mathbf{H}$= Jacobian matrix of a ground based manipulator.

$\mathbf{H}^*$= Jacobian matrix of a space manipulator.

$\mathbf{J}_i$= inertia dyadic of link i relative to a satellite fixed coordinate system.

$\mathbf{K}$=inertia dyadic of total system relative to a satellite fixed coordinate system.

$\mathbf{v}_i$= linear velocity of the center of mass of link i relative to a satellite fixed coordinate system.

$\mathbf{V}$= linear velocity of the center of mass of the system relative to a satellite fixed coordinate system.

$\mathbf{XYZ}$= Inertial frame.

$\mathbf{X}_0\mathbf{Y}_0\mathbf{Z}_0$= A coordinate frame attached to satellite main body.

INTRODUCTION

In this decade, NASA is planning to launch the first permanently manned U. S. space station. It is believed that "a key element of the right technology for the space station era is extensive use of advanced general-purpose automation and robotics [1]. The space station will be used to perform missions: collecting data from distant stars; satellite assembly, deployment and retrieval, servicing, maintenance and repair; and debris capture [2]. Most of these tasks require extra vehicular activities. It is proposed that manipulators be used to perform some of these tasks. By using manipulators, the following benefits can be obtained: [3]

- Astronauts spend their time as station managers rather than as operators carrying out routine functions, thus productivity increases.

- Lower cost of operations.

- Some of the tasks that cannot be performed by human power can be performed with robots- such as the assembly of large structures.

- Some of the tasks are hazardous to astronauts i.e., working in high orbits where radiation could be harmful. Using manipulators for these tasks will reduce the risk of exposure of astronauts to hazardous situations.

If the operations cannot be reduced to predetermined procedures, the handling of unpredicted events requires the presence of human beings. In cases where procedures are well defined manipulators can be used without human supervision.

A space station orbital maneuvering vehicle (OMV), which is a satellite-installed manipulator system, is being planned as a free flying teleoperator system to accomplish the tasks mentioned above [4]. To be able to accomplish these tasks, manipulators must be controlled by a very accurate control algorithm. Several control algorithms have been developed for ground-fixed manipulators. However, one of the characteristics of space based manipulators that is different from ground-fixed manipulators is the lack of a fixed base. If the space manipulator moves, the base of the manipulator, OMV, moves in reaction to the applied forces on it. The forces that cause the OMV to move are external and internal forces. External forces are the forces generated by the reaction jets, forces generated by contact with surrounding objects, forces caused by solar radiation and forces caused by gravitational gradients. Internal forces are the interactive forces between the manipulator and the OMV due to manipulator motions. These external and internal forces produce a translation of the center of mass of the OMV and a rotation of the OMV about its center of mass. If the algorithm that calculates the joint angles-which moves the manipulator to a prescribed end effector position and orientation- does not compensate for these internal and external forces, the end effector will miss the target. Ground-fixed manipulator control algorithms do not consider the motion of the base. Therefore these algorithms cannot be used with control methods unless the base is kept stationary.

The motion of the OMV as a result of manipulator motions can be compensated for by using reaction wheels and reaction jets. Reaction wheels add more weight and complexity to the system, and they cannot compensate for the translational changes in the base, therefore they should be used together with reaction jets. The life of reaction jets depends on the amount of reaction jet fuel carried by the spacecraft, and this amount is limited. Therefore, a method which eliminates or minimizes the use of reaction wheels and reaction jets will reduce the cost and increase the life of the spacecraft. In positions where the target is out of reach of the manipulator, reaction jets should be used to bring the system into the vicinity of the target. When the manipulator is in position to reach the target, reaction jets usage should be avoided.

In Chapter I, a control algorithm based on resolved motion rate control is developed. It is assumed that the base of the manipulator is free to translate and rotate, and reaction wheels and reaction jets will not be used to compensate for the motion in the base as a result of the manipulator motions. It is also assumed that the system is a collection of rigid bodies, and there are no external forces applied to the system. Using a modified resolved motion rate control algorithm and an angular momentum conservation law, a generalized Jacobian matrix of the manipulator, which gives the relation between the end effector velocities and joint angle rates, is derived. The algorithm is tested by a computer simulation of a space manipulator system. It is shown that the manipulator follows a prescribed path despite the translations and rotations of the base.

To service satellites, the OMV should dock with the satellite, for otherwise, they will start to drift apart as soon as there is a force of interaction between the OMV and the satellite. As mentioned above, the OMV will rotate about its center of mass as a result of external and internal forces. These deviations in the orientation of the OMV should be corrected to dock with the satellite. To avoid the usage of reaction jet gases, thrusters should not be used to correct these deviations.

It is well known that, astronauts can change their orientations in space by moving parts of their bodies. Also, divers, trampolinists, gymnasts and cats can perform rotation of their bodies in free fall without pushing against anything. This is possible because the angular momentum of the body is conserved. In this thesis, the same idea will be used to correct the orientation of the OMV for docking purposes.

A manipulator is assumed to be the arm of the OMV. If there is no external force applied to the system, the angular momentum of the system is conserved about its center of mass. The OMV will rotate as a result of manipulator motions. The orientation of the OMV will depend on the path the manipulator travels. If the manipulator moves along one path and returns to the starting point by another path, the orientation of the OMV will, in general, be different from its initial orientation before the manipulator motion. In Chapter II, an algorithm is developed to find the required manipulator motions to rotate the OMV from an initial orientation to a desired final orientation.

A manipulator control algorithm should be tested before applying it to real manipulators. Using real robots to test a control algorithm may result in an undesirable situation. For example, robots may collide accidentally with obstacles within a workcell. The control algorithm may be tested, however, using a computer graphic simulation program. Thus, the hazards involved in the algorithm can be observed and solved before applying it to real robots. A number of computer graphic simulation programs have been developed to solve such problems. Such a computer graphic simulator, ROBOSIM, was developed jointly at NASA and Vanderbilt University [5]. ROBOSIM will be used to animate the motion of the OMV as a result of manipulator motions. In Chapter III, an overview of ROBOSIM is given, then the results of the fine attitude control algorithm are animated.

We conclude in Chapter IV with a discussion of results and proposals for further research.

CHAPTER I

KINEMATIC ANALYSIS

Introduction

To control robot manipulators in space, the motion of the satellite (which is the base of the manipulator), as a result of the force exerted by the manipulator motion, should be considered. Most of the algorithms developed to control a robot manipulator assume that the manipulator is connected to a fixed base [6], [7]. Since the satellite moves due to the motion of the robot manipulator, the control schemes developed for the fixed-base manipulators cannot be used for space manipulators. If one wants to use the schemes for the fixed-base manipulators to control a space manipulator, one should keep the satellite stationary which requires an extra effort. Satellites can be kept stationary by using reaction jets and/or reaction wheels. Reaction jets require propellant to operate and their life depends on the amount of fuel carried by the satellite. Reaction wheels add more weight to the system which is a disadvantage during launch of the system, and they also increase the cost of the system. Reaction wheels cannot compensate for the translational disturbances and should be used together with reaction jets. Reaction jets can be used to bring the system into a position where the manipulator will be able to reach the target. If the manipulator is close to the target, reaction-jet usage should be avoided. The aim of this study is to find a control method which does not require the use of reaction jets and/or reaction wheels to move a space manipulator from an initial position to a desired final position.

Previous Studies

Studies have been made of space manipulator control under conditions of a nonstationary satellite base. Yamada *et al* [8] assumed that the base is free to translate but has attitude control to keep the orientation stationary. They derived the kinematic equations using a Newton-Euler formulation proposed by Luh [9]. They estimated the satellite linear acceleration by the use of the reaction force exerted on the satellite from the manipulator. They concluded that the satellite linear acceleration cannot be neglected in the case of controlling satellite attitudes.

Longman *et al.* [10] also assumed that the base is free to translate but assumed that reaction wheels were employed to keep the attitude stationary. They applied the method to a specific class of robot arms i.e., spherical-polar-coordinate robots, and showed that by controlling satellite attitude, the kinematics and dynamics of a free-flying space manipulator can be decoupled, and the control problem becomes a purely kinematic problem. They also computed the counter moments required to compensate for the turning moments of the satellite caused by manipulator arm motions. Their method is difficult to apply to an arbitrary manipulator configuration.

In these two methods, i. e., Yamada [8] and Longman [10] propellant consumption to keep the attitude fixed is very high, and the methods require a very high performance of attitude control.

Vafa and Dubowsky [11] proposed a 'Virtual Manipulator Concept' which is a massless kinematic chain. They defined the base of the virtual manipulator at the center of mass of the space manipulator. The virtual manipulator's end effector terminates at an arbitrary point on the real manipulator. They derived kinematic equations using the virtual manipulator and showed that the real manipulator kinematics is the same as the virtual manipulator kinematics.

Another method was developed by Umetani and Yoshida [12], [13]. This method assumes no position and attitude control of the satellite, so the satellite-based manipulator is a completely free-flying system. A generalized Jacobian matrix was derived using a modified resolved motion rate control [14] and momentum conservation laws. They used 3×3 rotational matrices to derive the equations.

In this chapter the method proposed by Umetani and Yoshida [12], [13] will be modified, and a generalized Jacobian matrix will be derived using 4×4 homogeneous transformation matrices. The method will be tested by computer simulation of a space manipulator system.

Fundamental Equations

To derive the equations, an n-degrees-of-freedom manipulator arm carried by a spacecraft is considered. The spacecraft has 6 degrees-of-freedom in space, 3 translational and 3 rotational degrees-of-freedom. The system has a total of n+6 degrees-of-freedom. An example of a space manipulator system is shown in Figure 1; the spacecraft is modelled as a cylinder.

The links and joints of the manipulator are assumed to be rigid bodies. The satellite is also assumed to be rigid and is represented by a single rigid body. Thus, the system can be modelled as a composition of rigid bodies. In many cases, space manipulators should be regarded as flexible bodies because of low stiffness. However, most of the formulations of flexible arms are based upon the small deformations from the virtual rigid body, so knowledge of the motion of rigid bodies is important. As a basic study for these advanced investigations, the movement of rigid links will be studied.

Each joint of the manipulator has one rotational degree-of-freedom and is rate controlled. The attitude and position of the main body is not controlled at all.

The forces due to gravitational gradients, solar radiation and aerodynamic drag are very small compared to the manipulator-satellite dynamic interaction forces. These small forces can accumulate in time and change the manipulator's position and orientation, but their short term effects on the system can be neglected. Therefore, it is assumed that there are no external forces applied on the system by external objects. The center of mass of the space manipulator system with respect to its orbit trajectory remains fixed during a manipulator maneuver. Hence, a coordinate frame attached to its orbit trajectory is an inertial frame, and the linear and angular momenta of the space manipulator system are conserved in the inertial frame.

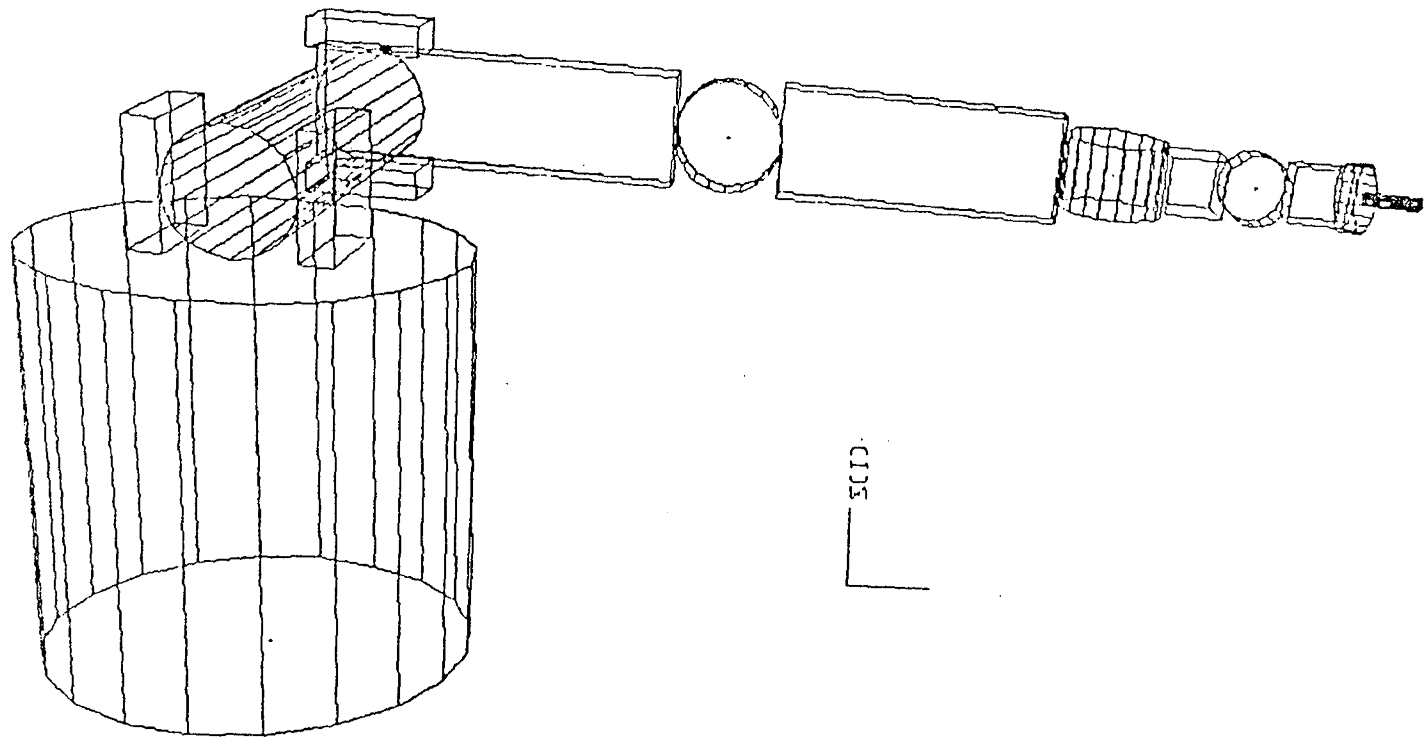

Figure 1. A space manipulator system.

ROBOSIM
NASA-MSFC

There is no relative motion between the center mass of the system and that of a target object at an initial position.

Let us denote the inertial frame by XYZ. An $X_0Y_0Z_0$ coordinate frame is fixed on the satellite main body, and its origin is at the center of mass of the satellite main body. The orientation of the satellite main body with respect to the inertial frame is defined by roll, pitch and yaw angles. Roll, pitch and yaw are a sequence of rotations which carries a moving coordinate frame from coincidence with the XYZ frame to coincidence with the coordinate frame $X_0Y_0Z_0$. The first rotation, roll, is a rotation ψ about the Z axis, pitch is a rotation β about the new Y axis , Y_1, and yaw is a rotation τ about the new X axis, X_2. These rotations are shown in Figure 2.

A rotational transformation between the $X_0Y_0Z_0$ coordinate frame and the inertial frame is given by

$$\mathbf{U}_0 = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\tau & -\sin\tau \\ 0 & \sin\tau & \cos\tau \end{pmatrix} \quad (1)$$

$$\mathbf{U}_0 = \begin{pmatrix} \cos\psi\cos\beta & \cos\psi\sin\beta\sin\tau - \sin\psi\cos\tau & \cos\psi\sin\beta\cos\tau + sin\psi\sin\tau \\ \sin\psi\cos\beta & \sin\psi\sin\beta\sin\tau + \cos\psi\cos\tau & \sin\psi\sin\beta\cos\tau - \cos\psi\sin\tau \\ -\sin\beta & \cos\beta\sin\tau & \cos\beta\cos\tau \end{pmatrix} \quad (2)$$

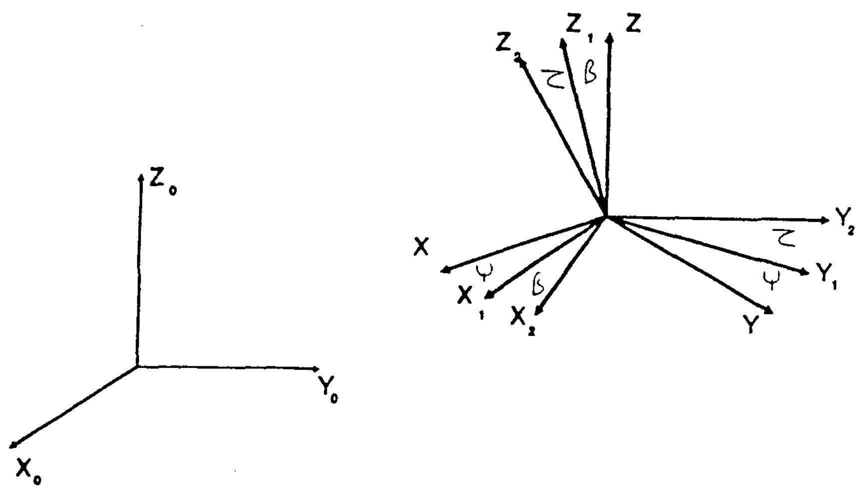

Figure 2. Roll, pitch and yaw rotations

The position of the center of mass of the satellite main body with respect to the inertial frame is given by a vector $\mathbf{r}_0$. The orientation and position of the $\mathbf{X}_0\mathbf{Y}_0\mathbf{Z}_0$ coordinate frame with respect to the inertial frame can be written using a 4×4 homogeneous transformation matrix $\mathbf{A}_0$:

$$\mathbf{A}_0 = \left(\begin{array}{ccc|c} & & & r_{0x} \\ & \mathbf{U}_0 & & r_{0y} \\ & & & r_{0z} \\ \hline 0 & 0 & 0 & 1 \end{array} \right) \tag{3}$$

A coordinate frame is assigned to each link of the manipulator using the Denavit-Hartenberg method [15]. The relation between two adjacent links of the manipulator is shown in Figure 3 [16].

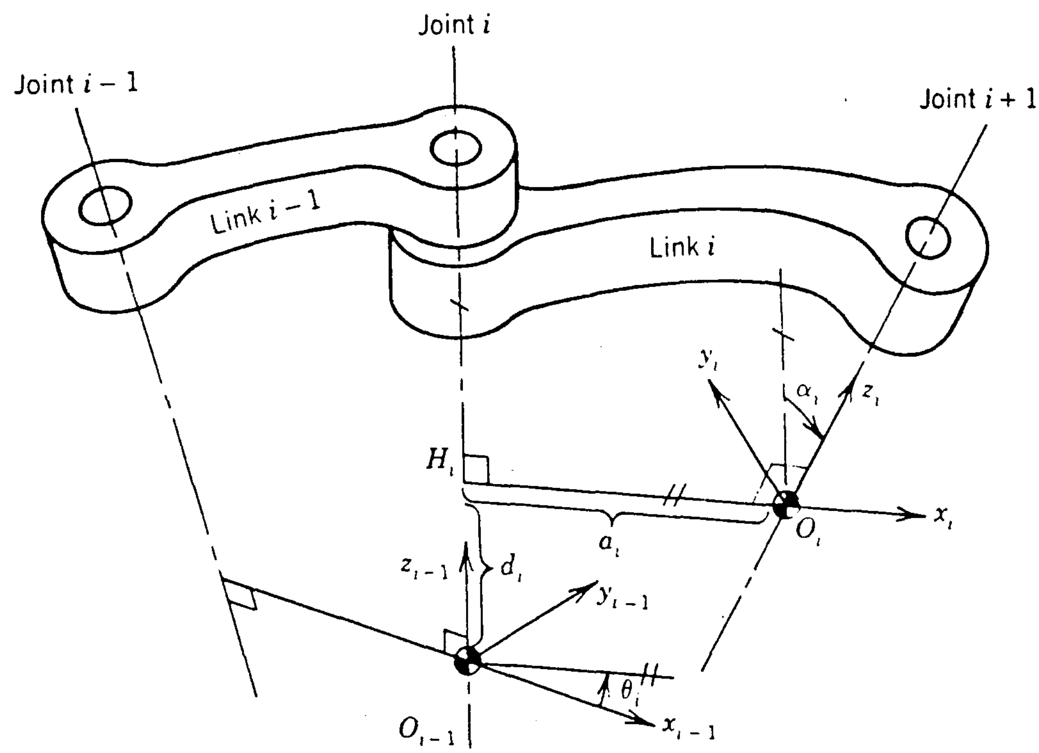

Figure 3. Relation between two links of the manipulator.

Definition of the link parameters are as follows:

a_i: distance between the origins of coordinate systems $i-1$ and i measured along $\mathbf{x}_i$.

d_i: distance between $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$ measured along $\mathbf{z}_{i-1}$.

α_i: angle between $\mathbf{z}_{i-1}$ and $\mathbf{z}_i$ axes measured about $\mathbf{x}_i$.

θ_i: angle between $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$ axes measured about $\mathbf{z}_{i-1}$.

The position and orientation of coordinate frame i with respect to coordinate frame $i-1$ can be defined by using 4×4 homogeneous transformation matrices ${}^{i-1}\mathbf{A}_i$, where

$$
{}^{i-1}\mathbf{A}_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4}
$$

The upper left 3×3 matrix of ${}^{i-1}\mathbf{A}_i$ gives the orientation information between the two coordinate systems and is denoted by ${}^{i-1}\mathbf{U}_i$. The first three elements of the fourth column give the translational information between the two links and is denoted by ${}^{i-1}\mathbf{p}_i$. θ_i and d_i are the joint variables for rotational and translational links, respectively.

Vector definitions giving locations of the center of mass for each link are shown in Figure 4.

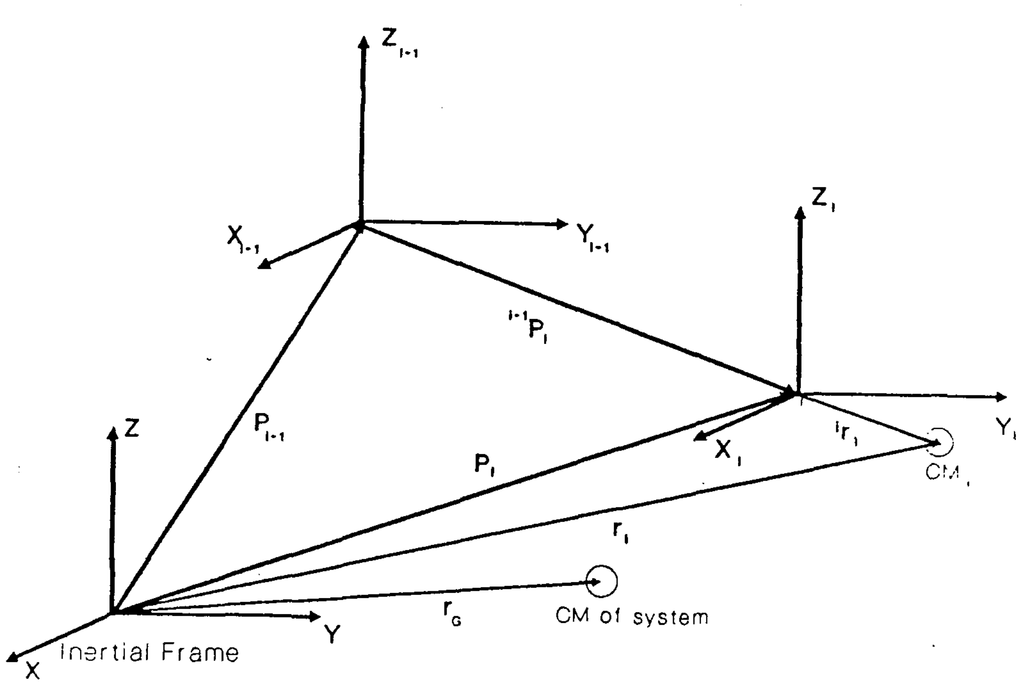

Figure 4. Vector definitions giving locations of the center of mass for each link.

Definitions of the vectors in Figure 4 are:

p_i: position vector of coordinate frame i, defined at joint i, with reference to the inertial frame.

${}^{i-1}\mathbf{p}_i$: position vector of the coordinate frame i with respect to the coordinate frame $i-1$.

CM_i: center of mass of link i.

${}^{i}\mathbf{r}_i$: position vector of the center of mass of link i with reference to the coordinate frame i.

$\mathbf{r}_i$: position vector of the center of mass of link i with reference to the inertial frame.

$\mathbf{r}_G$: position vector of center of mass of system with reference to the inertial frame.

The position and orientation of link i with reference to the inertial coordinate system can be found by the matrix multiplication of 4×4 homogeneous transformation matrices:

$$\mathbf{T}_i = \mathbf{A}_0 {}^{0}\mathbf{A}_1 \ldots {}^{i-1}\mathbf{A}_i \tag{5}$$

The position vector of the center of mass of link i with reference to the inertial coordinate frame is

$$\mathbf{r}_i = \mathbf{T}_i {}^{i}\mathbf{r}_i \tag{6}$$

Definition of the center of mass of the system is

$$\sum_{i=0}^{n} m_i \mathbf{r}_i = \mathbf{r}_G \sum_{i=0}^{n} m_i \tag{7}$$

where m_i is the mass of link i.

Since there are no external forces applied on the system, linear and angular momenta of the system are conserved in the inertial frame.

Linear momentum conservation is expressed by:

$$\sum_{i=0}^{n} m_i \dot{\mathbf{r}}_i = \text{constant} \tag{8}$$

Angular momentum conservation is expressed by:

$$\sum_{i=0}^{n} \mathbf{I}_i \mathbf{w}_i + m_i \mathbf{r}_i \times \dot{\mathbf{r}}_i = \text{constant} \tag{9}$$

where

$\mathbf{I}_i$= Inertia matrix of link i about a coordinate frame parallel to the inertial frame.

$\mathbf{w}_i$= Angular velocity of link i with reference to the inertial frame.

The angular velocity of the satellite main body with reference to the inertial frame, $\mathbf{w}_0$, in terms of changes in roll, pitch and yaw angles is given as follows

$$\begin{pmatrix} w_{0x} \\ w_{0y} \\ w_{0z} \end{pmatrix} = \begin{pmatrix} \cos\beta\cos\psi & -\sin\psi & 0 \\ \cos\beta\sin\psi & \cos\psi & 0 \\ -\sin\beta & 0 & 1 \end{pmatrix} \begin{pmatrix} \dot{\tau} \\ \dot{\beta} \\ \dot{\psi} \end{pmatrix} \tag{10}$$

and the angular velocity of link i with respect to the inertial frame can be written as

$$\mathbf{w}_i = \mathbf{w}_0 + \sum_{j=1}^{i} \mathbf{U}_j {}^j\mathbf{u}_j \dot{q}_j \tag{11}$$

where ${}^j\mathbf{u}_j$ is the unit vector about which rotation occurs, and q_j is the generalized coordinate representing the joint variable for link j.

To find the linear velocity of the center of mass of link i with respect to the inertial frame, differentiate equation (6) with respect to time. Then

$$\dot{\mathbf{r}}_i = \dot{\mathbf{T}}_i {}^i\mathbf{r}_i \tag{12}$$

where the differentiation of the homogeneous transformation matrix $\mathbf{T}_i$ is defined as

$$\dot{\mathbf{T}}_i = \frac{\partial \mathbf{T}_i}{\partial \tau}\dot{\tau} + \frac{\partial \mathbf{T}_i}{\partial \beta}\dot{\beta} + \frac{\partial \mathbf{T}_i}{\partial \psi}\dot{\psi} + \frac{\partial \mathbf{T}_i}{\partial r_{0x}}\dot{r}_{0x} + \frac{\partial \mathbf{T}_i}{\partial r_{0y}}\dot{r}_{0y} + \frac{\partial \mathbf{T}_i}{\partial r_{0z}}\dot{r}_{0z} + \frac{\partial \mathbf{T}_i}{\partial q_1}\dot{q}_1 + \ldots + \frac{\partial \mathbf{T}_i}{\partial q_i}\dot{q}_i \tag{13}$$

To eliminate $\dot{r}_{0x}, \dot{r}_{0y}$, and $\dot{r}_{0z}$ from the equations, substitute equations (12) and (13) into equation (8) and solve for the velocity of the center of mass of the satellite, $\dot{\mathbf{r}}_0$. Then

$$\begin{pmatrix} \dot{r}_{0x} \\ \dot{r}_{0y} \\ \dot{r}_{0z} \end{pmatrix} = -\begin{pmatrix} \mathbf{Q}_{r0x} & \mathbf{Q}_{r0y} & \mathbf{Q}_{r0z} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{Q}_\tau & \mathbf{Q}_\beta & \mathbf{Q}_\psi & \mathbf{Q}_{q1} \ldots \mathbf{Q}_{qn} \end{pmatrix} \begin{pmatrix} \dot{\tau} \\ \dot{\beta} \\ \dot{\psi} \\ \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix} \tag{14}$$

where

$$\mathbf{Q}_{r0x} = \sum_{k=1}^{n} m_k \frac{\partial \mathbf{T}_k}{\partial r_{0x}} {}^k\mathbf{r}_k + \begin{pmatrix} m_0 \\ 0 \\ 0 \end{pmatrix} \tag{15}$$

$$\mathbf{Q}_{r0y} = \sum_{k=1}^{n} m_k \frac{\partial \mathbf{T}_k}{\partial r_{0y}} {}^k\mathbf{r}_k + \begin{pmatrix} 0 \\ m_0 \\ 0 \end{pmatrix} \tag{16}$$

$$\mathbf{Q}_{r0z} = \sum_{k=1}^{n} m_k \frac{\partial \mathbf{T}_k}{\partial r_{0z}} {}^k\mathbf{r}_k + \begin{pmatrix} 0 \\ 0 \\ m_0 \end{pmatrix} \tag{17}$$

$$\mathbf{Q}_{\tau} = \sum_{k=1}^{n} m_k \frac{\partial \mathbf{T}_k}{\partial \tau} {}^k\mathbf{r}_k \tag{18}$$

$$\mathbf{Q}_{\beta} = \sum_{k=1}^{n} m_k \frac{\partial \mathbf{T}_k}{\partial \beta} {}^k\mathbf{r}_k \tag{19}$$

$$\mathbf{Q}_{\psi} = \sum_{k=1}^{n} m_k \frac{\partial \mathbf{T}_k}{\partial \psi} {}^k\mathbf{r}_k \tag{20}$$

$$\mathbf{Q}_{qi} = \sum_{k=i}^{n} m_k \frac{\partial \mathbf{T}_k}{\partial q_i} {}^k\mathbf{r}_k, \qquad i = 1, \ldots, n \tag{21}$$

are 3×1 vectors. Define a matrix $\mathbf{C}$ as

$$\mathbf{C} = -\left(\mathbf{Q}_{r0x} \quad \mathbf{Q}_{r0y} \quad \mathbf{Q}_{r0z} \right)^{-1} \left(\mathbf{Q}_{\tau} \quad \mathbf{Q}_{\beta} \quad \mathbf{Q}_{\psi} \quad \mathbf{Q}_{q1} \quad \ldots \quad \mathbf{Q}_{qn} \right) \tag{22}$$

Then, from equations (14) and (22), $\dot{r}_{0x}, \dot{r}_{0y}$ and $\dot{r}_{0z}$ can be written as

$$\dot{r}_{0x} = c_{11}\dot{\tau} + c_{12}\dot{\beta} + c_{13}\dot{\psi} + c_{14}\dot{q}_1 + \ldots + c_{1n+3}\dot{q}_n \tag{23}$$

$$\dot{r}_{0y} = c_{21}\dot{\tau} + c_{22}\dot{\beta} + c_{23}\dot{\psi} + c_{24}\dot{q}_1 + \ldots + c_{2n+3}\dot{q}_n \tag{24}$$

$$\dot{r}_{0z} = c_{31}\dot{\tau} + c_{32}\dot{\beta} + c_{33}\dot{\psi} + c_{34}\dot{q}_1 + \ldots + c_{3n+3}\dot{q}_n \tag{25}$$

where c_{kl} is the k^{th} row l^{th} column entry of the $\mathbf{C}$ matrix. Then the derivative of the homogeneous transformation matrix $\mathbf{T}_i$ can be written as

$$\dot{\mathbf{T}}_i = (\frac{\partial \mathbf{T}_i}{\partial \tau} + c_{11}\frac{\partial \mathbf{T}_i}{\partial r_{0x}} + c_{21}\frac{\partial \mathbf{T}_i}{\partial r_{0y}} + c_{31}\frac{\partial \mathbf{T}_i}{\partial r_{0z}})\dot{\tau}$$

$$+(\frac{\partial \mathbf{T}_i}{\partial \beta} + c_{12}\frac{\partial \mathbf{T}_i}{\partial r_{0x}} + c_{22}\frac{\partial \mathbf{T}_i}{\partial r_{0y}} + c_{32}\frac{\partial \mathbf{T}_i}{\partial r_{0z}})\dot{\beta}$$

$$+(\frac{\partial \mathbf{T}_i}{\partial \psi} + c_{13}\frac{\partial \mathbf{T}_i}{\partial r_{0x}} + c_{23}\frac{\partial \mathbf{T}_i}{\partial r_{0y}} + c_{33}\frac{\partial \mathbf{T}_i}{\partial r_{0z}})\dot{\psi}$$

$$+\sum_{j=1}^{n}(\frac{\partial \mathbf{T}_i}{\partial q_j} + c_{1j+3}\frac{\partial \mathbf{T}_i}{\partial r_{0x}} + c_{2j+3}\frac{\partial \mathbf{T}_i}{\partial r_{0y}} + c_{3j+3}\frac{\partial \mathbf{T}_i}{\partial r_{0z}})\dot{q}_j \tag{26}$$

<u>Generalized Jacobian Matrix</u>

In general the characteristic equation for an ordinary ground fixed manipulator can be written as

$$\mathbf{P} = f(\mathbf{q}) \tag{27}$$

where

$\mathbf{P} = (\mathbf{p}_e^T, \Theta_e^T)^T$: position and attitude vectors of the end effector of the manipulator.

$\mathbf{p}_e$: position vector of the end effector of the manipulator.

Θ_e: Attitude vector of the end effector of the manipulator.

$\mathbf{q} = (q_1, \ldots, q_n)^T$: vector of the joint variables.

It is easy to obtain the transformation that transforms the joint variables $\mathbf{q}$ into a set of task coordinates $\mathbf{P}$. The inverse transformation cannot be obtained easily even in ordinary cases, because the transformation is nonlinear and configuration dependent. However, by differentiating (27) with respect to time the transformation between $\dot{\mathrm{P}}$ and $\dot{\mathrm{q}}$ can be linearized, and the motion rate of the end effector in the task space can be resolved into that of joint variables in the configuration space.

$$\dot{\mathrm{P}} = \mathrm{H}(q)\dot{\mathrm{q}} \tag{28}$$

where $\mathrm{H}(q) = \frac{\partial f}{\partial q}$ is the Jacobian matrix. For a space manipulator, position coordinates and attitude angles of the satellite should be considered as state variables of the system because the posture of the satellite is influenced by a manipulator operation.

The relation between the position of the end effector with respect to the inertial frame $\mathbf{p}_e$ and the coordinate frame n, ${}^{n}\mathbf{p}_e$ is

$$\mathbf{p}_e = \mathbf{T}_n {}^{n}\mathbf{p}_e. \tag{29}$$

Then, the linear velocity of the end effector with respect to the inertial frame is given by

$$\dot{\mathbf{p}}_e = \dot{\mathbf{T}}_n {}^{n}\mathbf{p}_e \tag{30}$$

where

$$\begin{aligned}
\dot{\mathbf{T}}_n = &\left(\frac{\partial \mathbf{T}_n}{\partial \tau} + \frac{\partial \mathbf{T}_n}{\partial r_{0x}}c_{11} + \frac{\partial \mathbf{T}_n}{\partial r_{0y}}c_{21} + \frac{\partial \mathbf{T}_n}{\partial r_{0z}}c_{31}\right)\dot{\tau} \\
&+\left(\frac{\partial \mathbf{T}_n}{\partial \beta} + \frac{\partial \mathbf{T}_n}{\partial r_{0x}}c_{12} + \frac{\partial \mathbf{T}_n}{\partial r_{0y}}c_{22} + \frac{\partial \mathbf{T}_n}{\partial r_{0z}}c_{32}\right)\dot{\beta} \\
&+\left(\frac{\partial \mathbf{T}_n}{\partial \psi} + \frac{\partial \mathbf{T}_n}{\partial r_{0x}}c_{13} + \frac{\partial \mathbf{T}_n}{\partial r_{0y}}c_{23} + \frac{\partial \mathbf{T}_n}{\partial r_{0z}}c_{33}\right)\dot{\psi} \\
&+\sum_{i=1}^{n}\left(\frac{\partial \mathbf{T}_n}{\partial q_i} + \frac{\partial \mathbf{T}_n}{\partial r_{0x}}c_{1i+3} + \frac{\partial \mathbf{T}_n}{\partial r_{0y}}c_{2i+3} + \frac{\partial \mathbf{T}_n}{\partial r_{0z}}c_{3i+3}\right)\dot{q}_i
\end{aligned} \tag{31}$$

From equations (30) and (31) the linear velocity of the end effector with reference to the inertial frame can be written as a linear combination of $\dot{\tau}, \dot{\beta}, \dot{\psi}, \dot{q}_1, \ldots, \dot{q}_n$

$$\dot{\mathrm{p}}_e = \begin{pmatrix} \mathrm{H}_\tau & \mathrm{H}_\beta & \mathrm{H}_\psi & \mathrm{H}_{q1} & \ldots & \mathrm{H}_{qn} \end{pmatrix} \begin{pmatrix} \dot{\tau} \\ \dot{\beta} \\ \dot{\psi} \\ \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix} \tag{32}$$

where

$$\mathrm{H}_\tau = (\frac{\partial \mathrm{T}_n}{\partial \tau} + \frac{\partial \mathrm{T}_n}{\partial r_{0x}} c_{11} + \frac{\partial \mathrm{T}_n}{\partial r_{0y}} c_{21} + \frac{\partial \mathrm{T}_n}{\partial r_{0z}} c_{31})^n \mathrm{p}_e \tag{33}$$

$$\mathrm{H}_\beta = (\frac{\partial \mathrm{T}_n}{\partial \beta} + \frac{\partial \mathrm{T}_n}{\partial r_{0x}} c_{12} + \frac{\partial \mathrm{T}_n}{\partial r_{0y}} c_{22} + \frac{\partial \mathrm{T}_n}{\partial r_{0z}} c_{32})^n \mathrm{p}_e \tag{34}$$

$$\mathrm{H}_\psi = (\frac{\partial \mathrm{T}_n}{\partial \psi} + \frac{\partial \mathrm{T}_n}{\partial r_{0x}} c_{13} + \frac{\partial \mathrm{T}_n}{\partial r_{0y}} c_{23} + \frac{\partial \mathrm{T}_n}{\partial r_{0z}} c_{33})^n \mathrm{p}_e \tag{35}$$

$$\mathrm{H}_{qi} = (\frac{\partial \mathrm{T}_n}{\partial q_i} + \frac{\partial \mathrm{T}_n}{\partial r_{0x}} c_{1i+3} + \frac{\partial \mathrm{T}_n}{\partial r_{0y}} c_{2i+3} + \frac{\partial \mathrm{T}_n}{\partial r_{0z}} c_{3i+3})^n \mathrm{p}_e \tag{36}$$

From equations (10) and (11), the angular velocity of the end effector, w_e, can also be written as a linear combination of $\dot{\tau}, \dot{\beta}, \dot{\psi}, \dot{q}_1, \ldots, \dot{q}_n$

$$\mathrm{w}_e = \begin{pmatrix} \mathrm{w}_{0x} & \mathrm{w}_{0y} & \mathrm{w}_{0z} & \mathrm{U}_1{}^1\mathrm{u}_1 & \ldots & \mathrm{U}_n{}^n\mathrm{u}_n \end{pmatrix} \begin{pmatrix} \dot{\tau} \\ \dot{\beta} \\ \dot{\psi} \\ \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix} \tag{37}$$

where w_{0x}, w_{0y} and w_{0z} are the columns of w_0 matrix given by the equation (10). Combining equations (32) and (37) gives

$$\dot{\mathrm{P}} = \begin{pmatrix} \dot{\mathrm{p}}_e \\ \mathrm{w}_e \end{pmatrix} = \begin{pmatrix} \mathrm{H}_\tau & \mathrm{H}_\beta & \mathrm{H}_\psi & \mathrm{H}_{q1} & \ldots & \mathrm{H}_{qn} \\ \mathrm{w}_{0x} & \mathrm{w}_{0y} & \mathrm{w}_{0z} & \mathrm{U}_1{}^1\mathrm{u}_1 & \ldots & \mathrm{U}_n{}^n\mathrm{u}_n \end{pmatrix} \begin{pmatrix} \dot{\tau} \\ \dot{\beta} \\ \dot{\psi} \\ \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix} \tag{38}$$

Equation (38) can be divided into the satellite part and the manipulator part.

$$\dot{\mathbf{P}} = \mathbf{H}_s \dot{\mathbf{q}}_s + \mathbf{H}_m \dot{\mathbf{q}}_m \tag{39}$$

where

$\mathbf{q}_s$: 3×1 matrix of attitude angles of the satellite main body.

$\mathbf{q}_m$: $n \times 1$ matrix of joint variables of the manipulator.

$\mathbf{H}_s$ is an 6×3 matrix, and $\mathbf{H}_m$ is an $6 \times n$ matrix.

Equation (39) gives 6 equations in $n + 3$ unknowns. In order to solve this system of equations we need additional equations. These equations can be obtained using the angular momentum conservation equation. Substitute equations (11) and (12) into equation (9) to obtain

$$\begin{pmatrix} \mathbf{I}_\tau & \mathbf{I}_\beta & \mathbf{I}_\psi & \mathbf{I}_{q1} & \dots & \mathbf{I}_{qn} \end{pmatrix} \begin{pmatrix} \dot{\tau} \\ \dot{\beta} \\ \dot{\psi} \\ \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{pmatrix} = \text{constant} \tag{40}$$

where

$$\mathbf{I}_\tau = \sum_{j=0}^{n} (\mathbf{U}_j {}^j\mathbf{I}_j \mathbf{U}_j^{-1}) \mathbf{w}_{0x} + m_0 \mathbf{A}_0 {}^0\mathbf{r}_0 \times \mathbf{c}_1$$

$$+ \sum_{j=1}^{n} m_j \mathbf{T}_j {}^j\mathbf{r}_j \times \left(\frac{\partial \mathbf{T}_j}{\partial \tau} + c_{11} \frac{\partial \mathbf{T}_j}{\partial r_{0x}} + c_{21} \frac{\partial \mathbf{T}_j}{\partial r_{0y}} + c_{31} \frac{\partial \mathbf{T}_j}{\partial r_{0z}} \right) {}^j\mathbf{r}_j \tag{41}$$

$$\mathbf{I}_\beta = \sum_{j=0}^{n} (\mathbf{U}_j {}^j\mathbf{I}_j \mathbf{U}_j^{-1}) \mathbf{w}_{0y} + m_0 \mathbf{A}_0 {}^0\mathbf{r}_0 \times \mathbf{c}_2$$

$$+ \sum_{j=1}^{n} m_j \mathbf{T}_j {}^j\mathbf{r}_j \times \left(\frac{\partial \mathbf{T}_j}{\partial \beta} + c_{12} \frac{\partial \mathbf{T}_j}{\partial r_{0x}} + c_{22} \frac{\partial \mathbf{T}_j}{\partial r_{0y}} + c_{32} \frac{\partial \mathbf{T}_j}{\partial r_{0z}} \right) {}^j\mathbf{r}_j \tag{42}$$

$$\mathbf{I}_\psi = \sum_{j=0}^{n}(\mathbf{U}_j{}^j\mathbf{I}_j\mathbf{U}_j^{-1})\mathbf{w}_{0z} + m_0\mathbf{A}_0{}^0\mathbf{r}_0 \times \mathbf{c}_3$$

$$+ \sum_{j=1}^{n} m_j\mathbf{T}_j{}^j\mathbf{r}_j \times (\frac{\partial \mathbf{T}_j}{\partial \tau} + c_{13}\frac{\partial \mathbf{T}_j}{\partial r_{0x}} + c_{23}\frac{\partial \mathbf{T}_j}{\partial r_{0y}} + c_{33}\frac{\partial \mathbf{T}_j}{\partial r_{0z}})^j\mathbf{r}_j \quad (43)$$

$$\mathbf{I}_{qi} = \sum_{j=i}^{n}(\mathbf{U}_j{}^j\mathbf{I}_j\mathbf{U}_j^{-1})\mathbf{U}_i{}^i\mathbf{u}_i + m_0\mathbf{A}_0{}^0\mathbf{r}_0 \times \mathbf{c}_{j+3}$$

$$+ \sum_{j=1}^{n} m_j\mathbf{T}_j{}^j\mathbf{r}_j \times (\frac{\partial \mathbf{T}_j}{\partial q_i} + c_{1i+3}\frac{\partial \mathbf{T}_j}{\partial r_{0x}} + c_{2i+3}\frac{\partial \mathbf{T}_j}{\partial r_{0y}} + c_{3i+3}\frac{\partial \mathbf{T}_j}{\partial r_{0z}})^j\mathbf{r}_j \quad (44)$$

qwhere $\mathbf{c}_i$ is the i^{th} column of the C matrix defined by equation (22). ${}^i\mathbf{I}_i$ is the inertia matrix of link i with reference to coordinate frame i. This inertia matrix can be transformed into the inertial frame coordinates by the transformation [17].

$$\mathbf{I}_i = \mathbf{U}_i{}^i\mathbf{I}_i\mathbf{U}_i^{-1} \quad (45)$$

Equation (40) can also be divided into satellite part and manipulator part, as

$$\mathbf{I}_s\dot{\mathbf{q}}_s + \mathbf{I}_m\dot{\mathbf{q}}_m = \text{constant.} \quad (46)$$

$\mathbf{I}_s$ is a 3×3 inertia matrix of the satellite main body and $\mathbf{I}_m$ is a $3 \times n$ inertia matrix of the manipulator.

The system defined by equations (39) and (46) has 9 linear equations of $n+3$ variables $\dot{\tau}, \dot{\beta}, \dot{\psi}, \dot{q}_1 \ldots, \dot{q}_n$. By eliminating $\dot{\mathbf{q}}_s$ from equations (39) and (46) and assuming the system is at rest initially $\dot{\mathbf{q}}_m$ can be solved as

$$\dot{\mathbf{P}} = (\mathbf{H}_m - \mathbf{H}_s\mathbf{I}_s^{-1}\mathbf{I}_m)\dot{\mathbf{q}}_m \quad (47)$$

$$\dot{\mathbf{P}} = \mathbf{H}^*\dot{\mathbf{q}}_m \quad (48)$$

where $\mathbf{H}^*$ is the generalized Jacobian matrix for a space manipulator. The Jacobian matrix of the manipulator $\mathbf{H}_m$ is compensated for a disturbance of reactive movement of the base. The amount of compensation is proportional to the ratio of the manipulator inertia and satellite inertia matrices, $\mathbf{I}_m/\mathbf{I}_s$. This ratio approaches zero when the inertia of the satellite is much larger than the inertia of the manipulator. In this case the system can be treated as a ground fixed manipulator system. If the manipulator has 6 degrees-of-freedom there is no redundancy and $\mathbf{H}^*$ becomes a 6×6 square matrix and can be inverted if it is nonsingular. $\dot{\mathbf{q}}_m$ can be solved as

$$\dot{\mathbf{q}}_m = (\mathbf{H}^*)^{-1}\dot{\mathbf{P}} \tag{49}$$

By using equation (49) the required manipulator operation $\dot{\mathbf{q}}_m$ corresponding to the given trajectory $\dot{\mathbf{P}}$ for the capture of a free-flying target in space can be easily obtained. If the manipulator has a number of degrees-of-freedom other than 6, $\mathbf{H}^*$ can be inverted using a pseudo inverse.

A Test of the Control Algorithm

To test the method, a simulation program was written in FORTRAN. Listing of the program is given in Appendix A. A 6 degrees-of-freedom system was assumed. The specifications of the system are listed in Table 1.

Velocity of the end effector along the prescribed trajectory was assumed to be as shown in Figure 5. This velocity was applied along different axes of the end effector and the behavior of the system was observed. Figures 6a-6d show the response of the system when the velocity was applied along the X direction. Figure 6a shows the change in the position of the end effector. Figure 6b represents the change in joint angles of the manipulator to achieve this motion. Figures 6c and 6d show the changes in the position and attitude of the satellite main body. Figures 7a-7d show the response of the system when the velocity was applied along the Z direction. As seen from the results, the end effector follows the prescribed trajectory despite the rotational and translational motions of the satellite main body.

TABLE 1.

SPECIFICATIONS OF THE SPACE MANIPULATOR SYSTEM

	Link 0	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6
Mass (Lb)	23450.0	51.3	18.2	16.0	1.8	4.2	2.0
I_x(Lb in^2)	631330.0	15000.0	125.0	63.0	5.0	47.0	11.0
I_y	631330.0	425.0	3700.0	2525.0	30.0	140.0	11.0
I_z	2504500.0	12750.0	3725.0	2550.0	30.0	120.0	15.0
I_{xy}	0.0	0.0	160.0	-93.0	0.0	0.0	0.0
I_{xz}	0.0	0.0	0.0	0.0	0.0	19.0	0.0
I_{yz}	0.0	1080.0	0.0	0.0	0.0	0.0	0.0

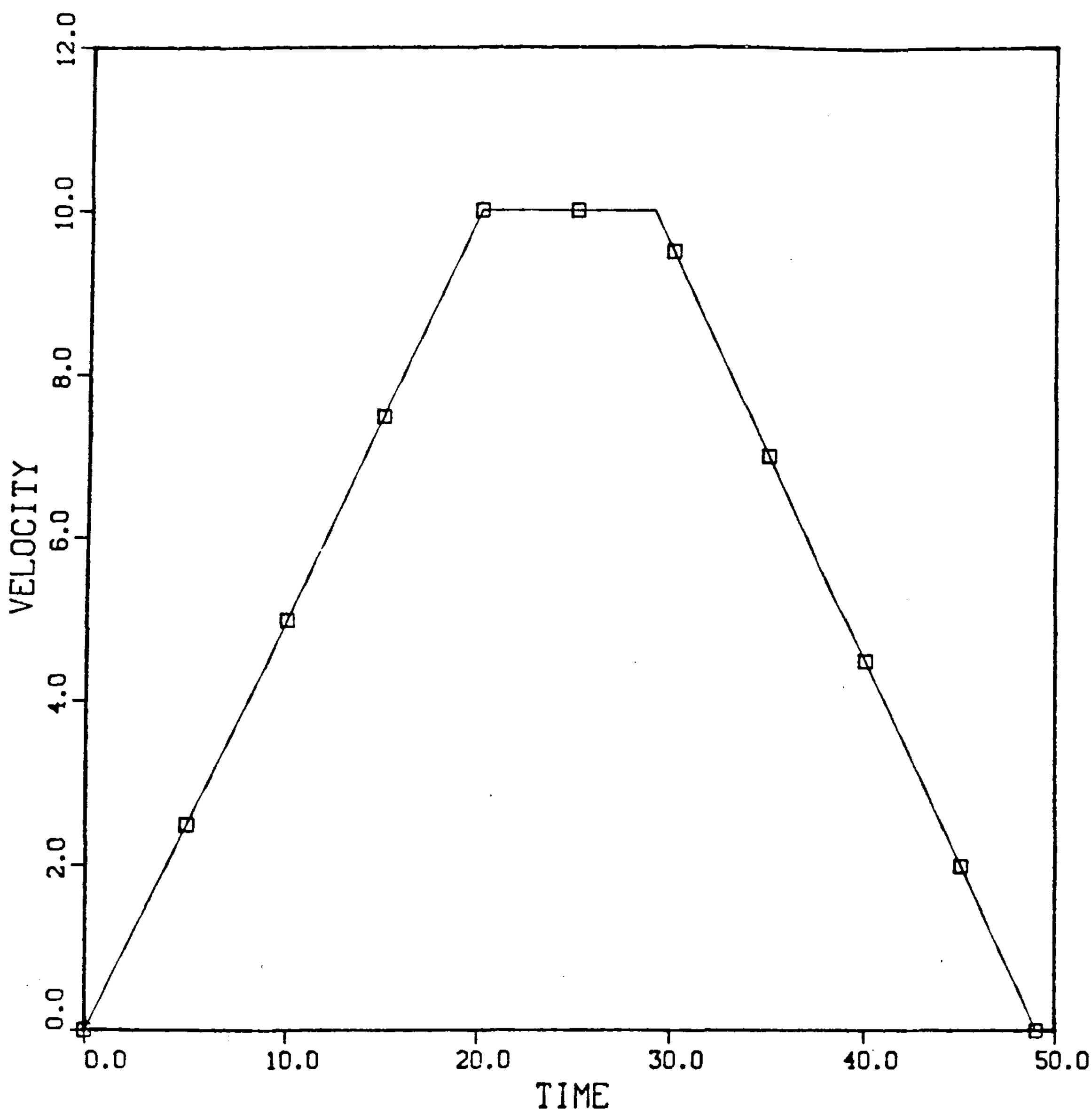

Figure 5. Velocity along trajectory.

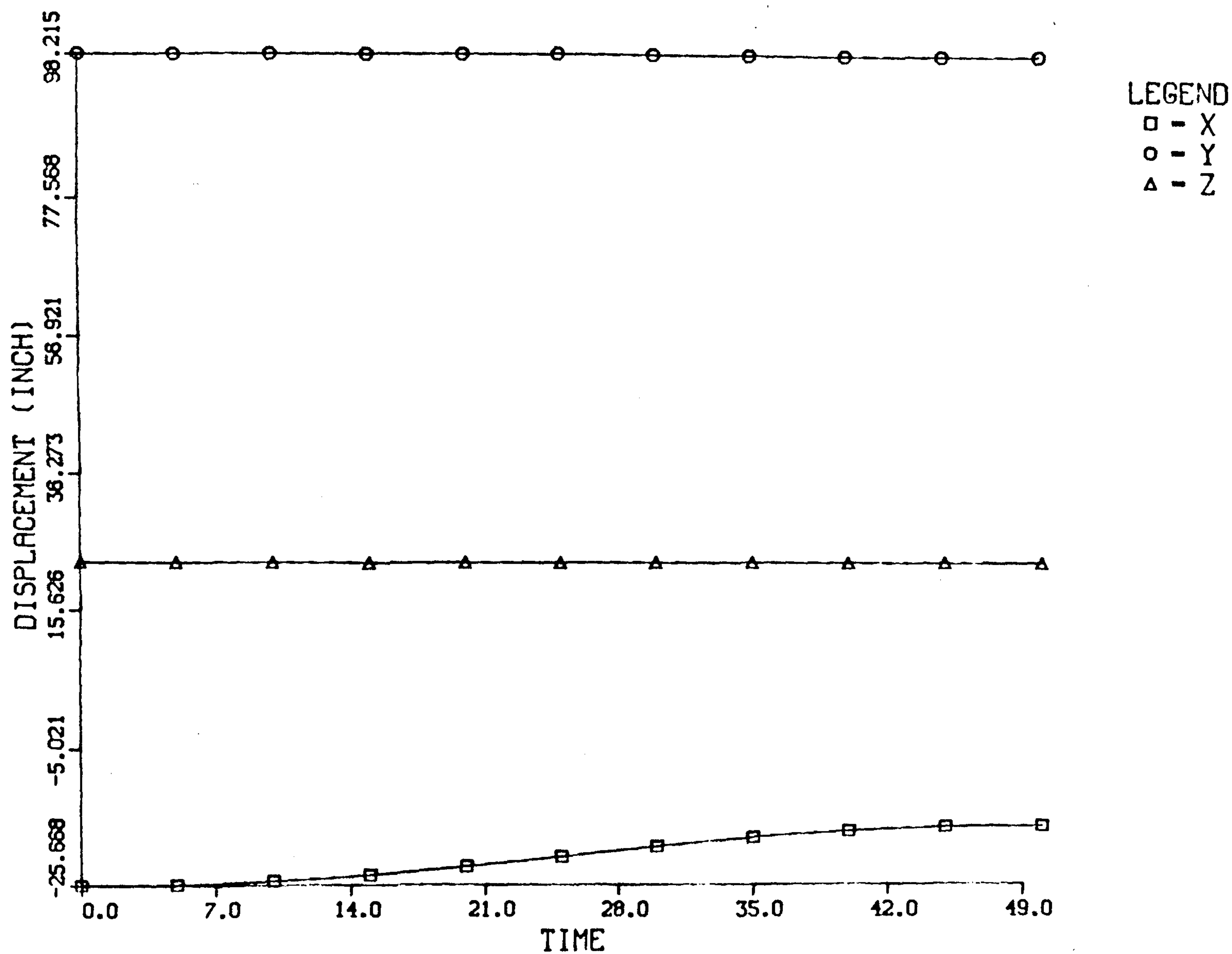

Figure 6a. Position of the end effector.

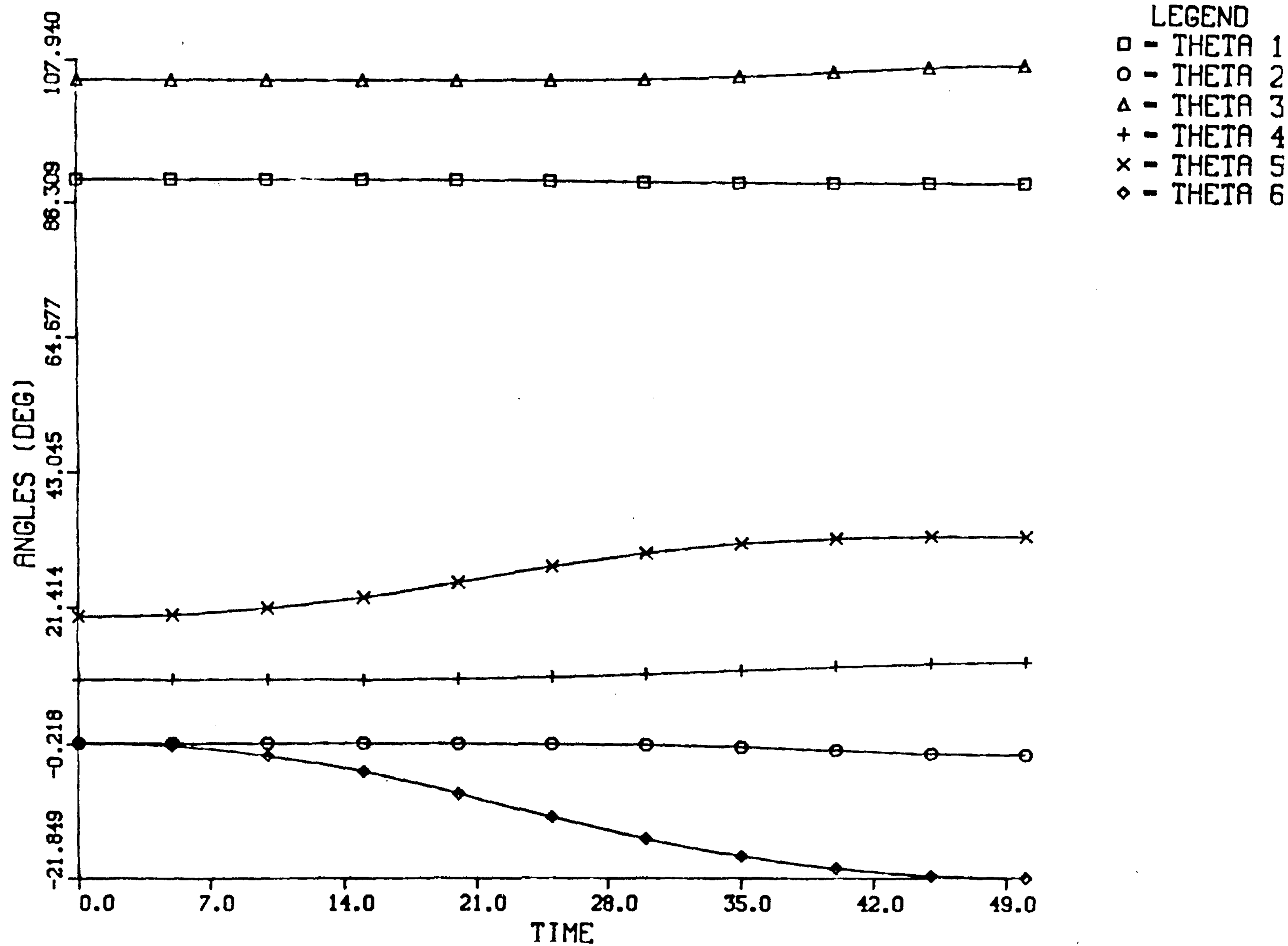

Figure 6b. Joint angles.

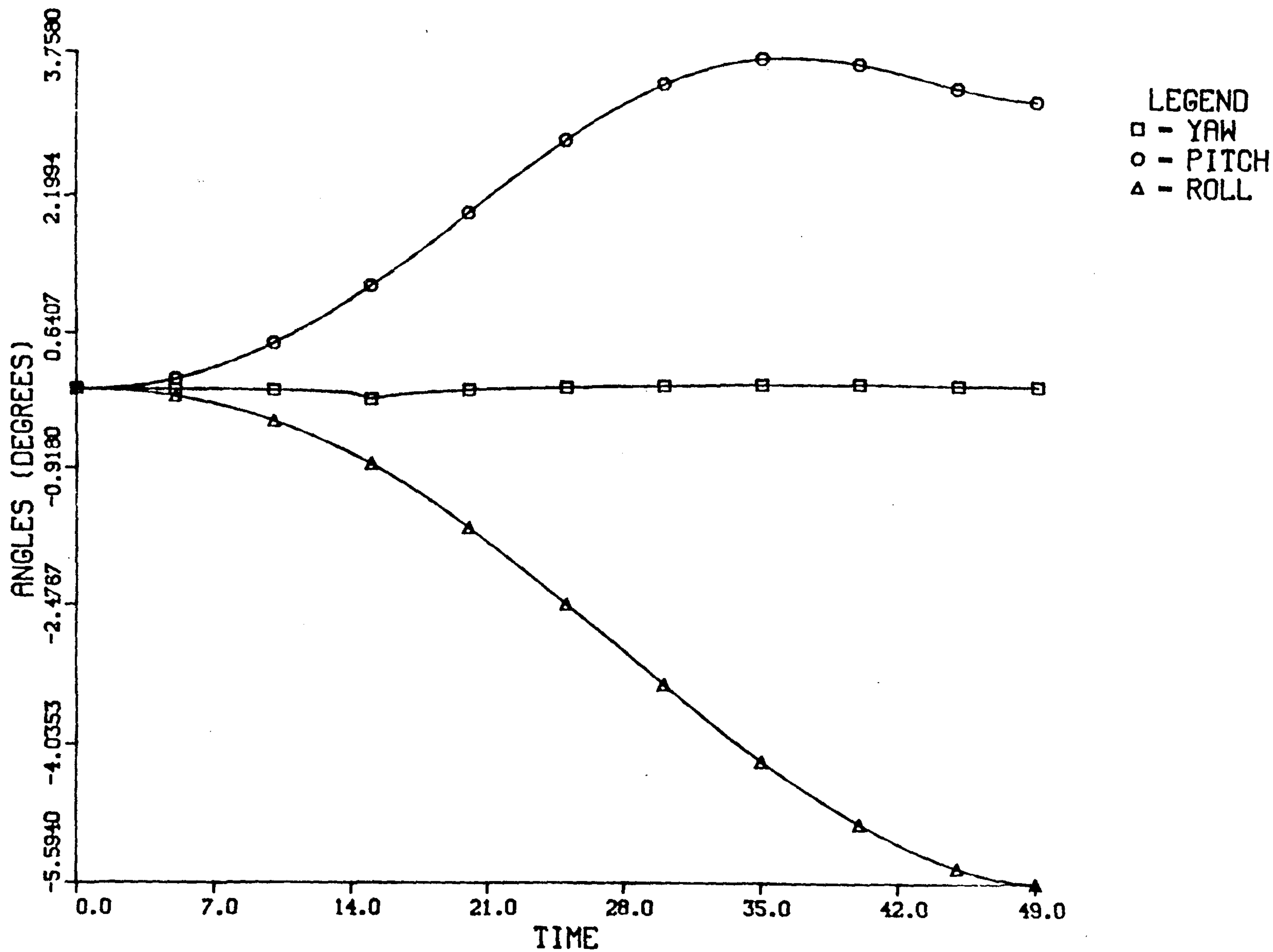

Figure 6c. Roll, pitch and yaw angles.

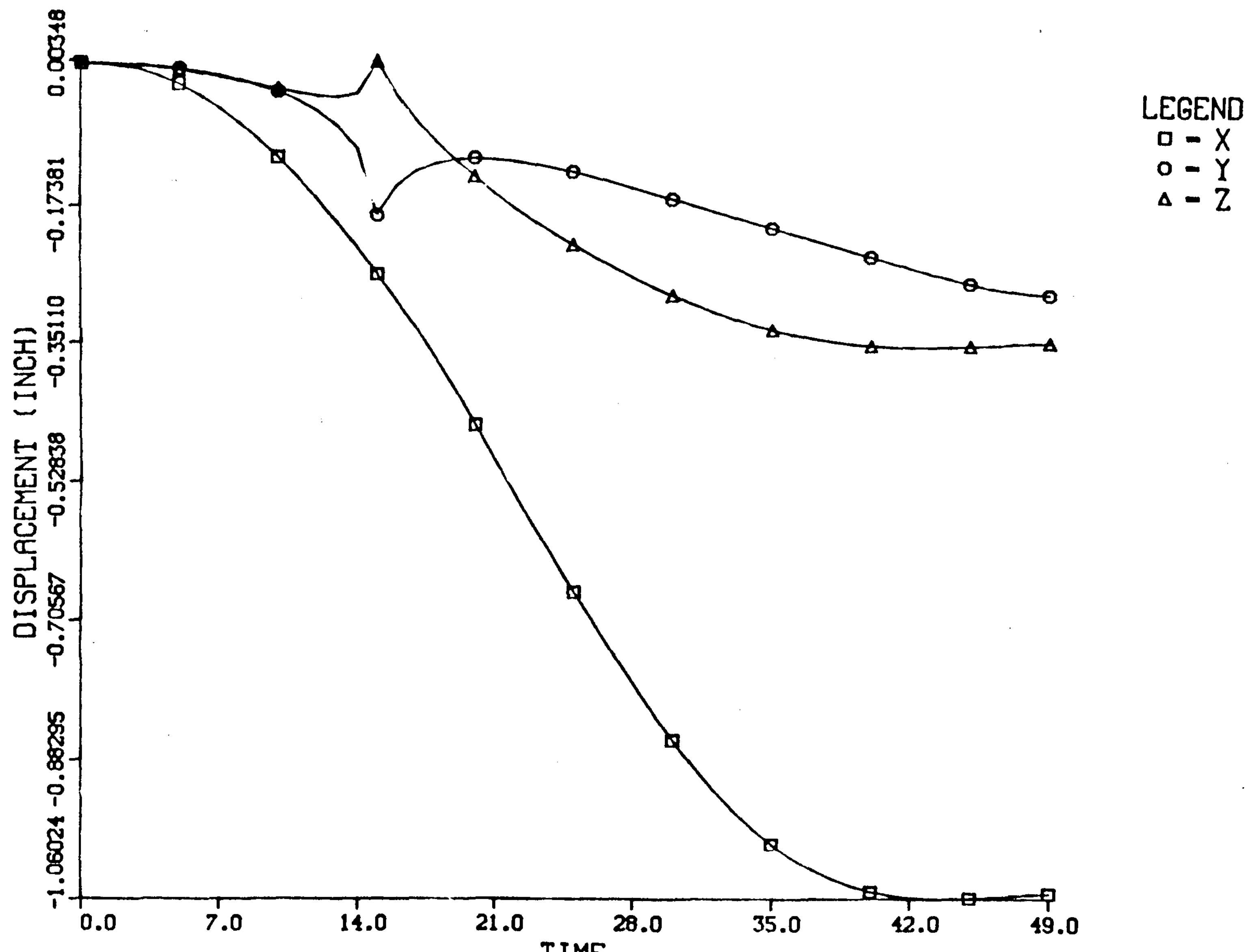

Figure 6d. Position of the base.

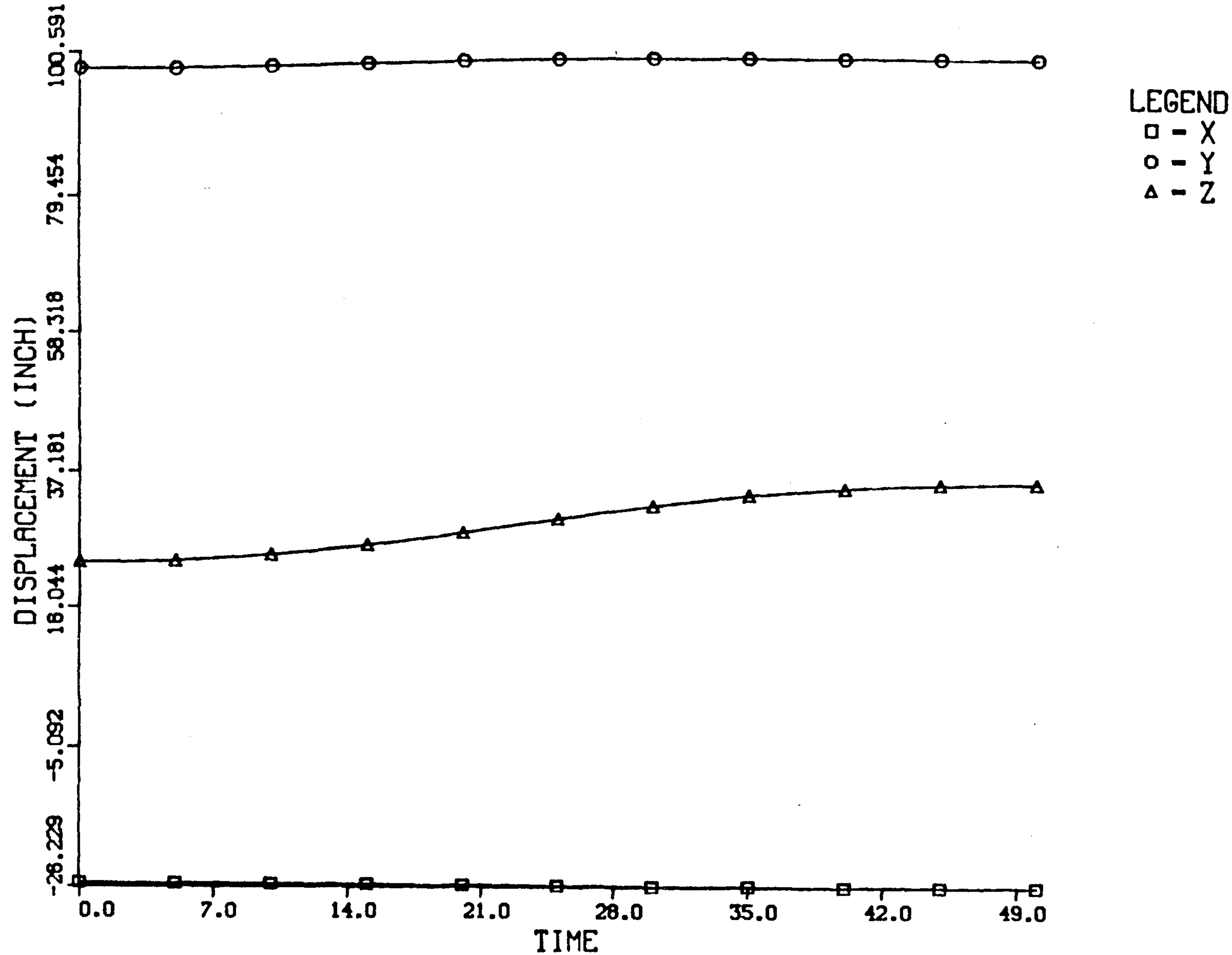

Figure 7a. Position of the end effector.

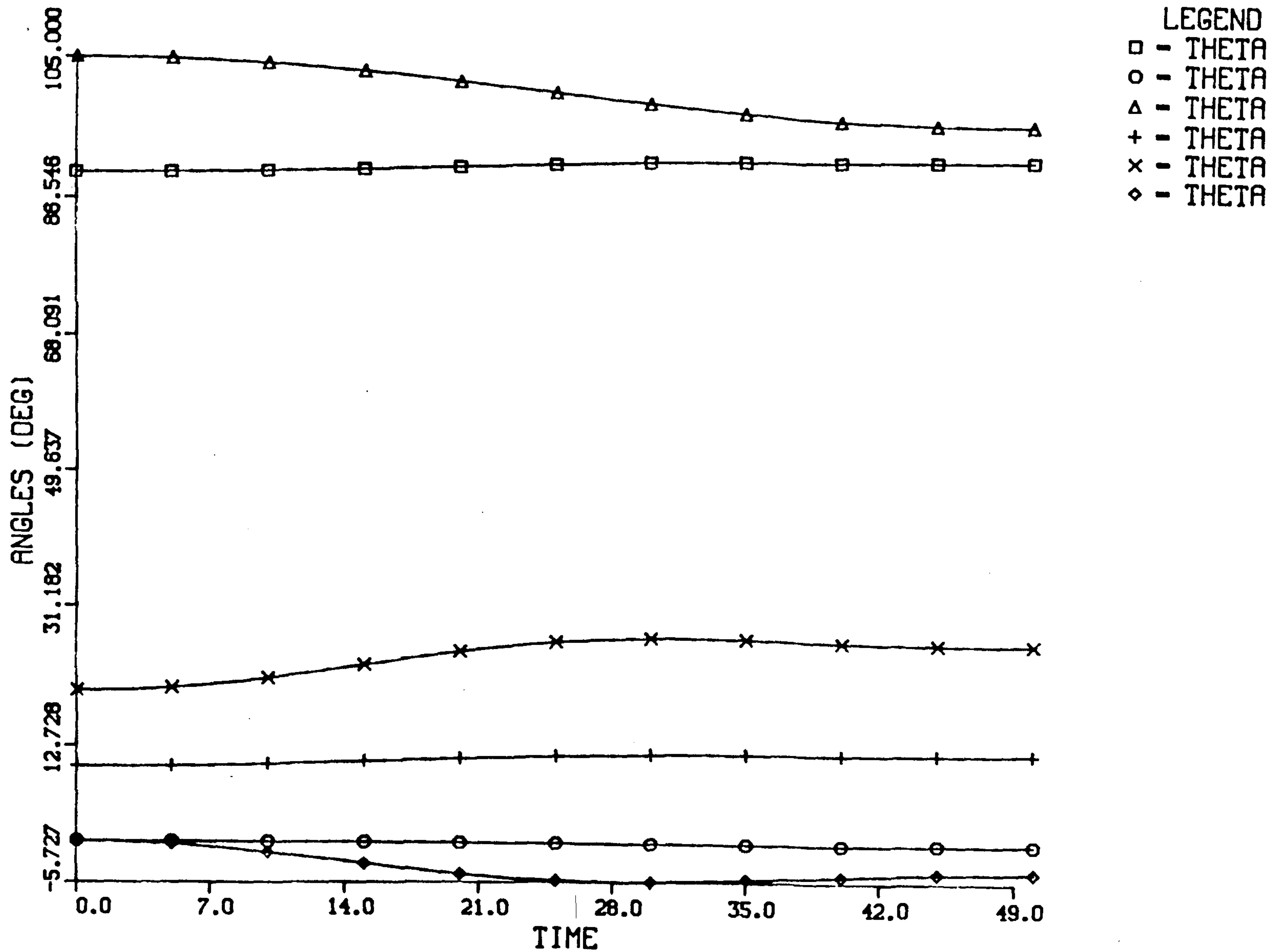

Figure 7b. Joint angles.

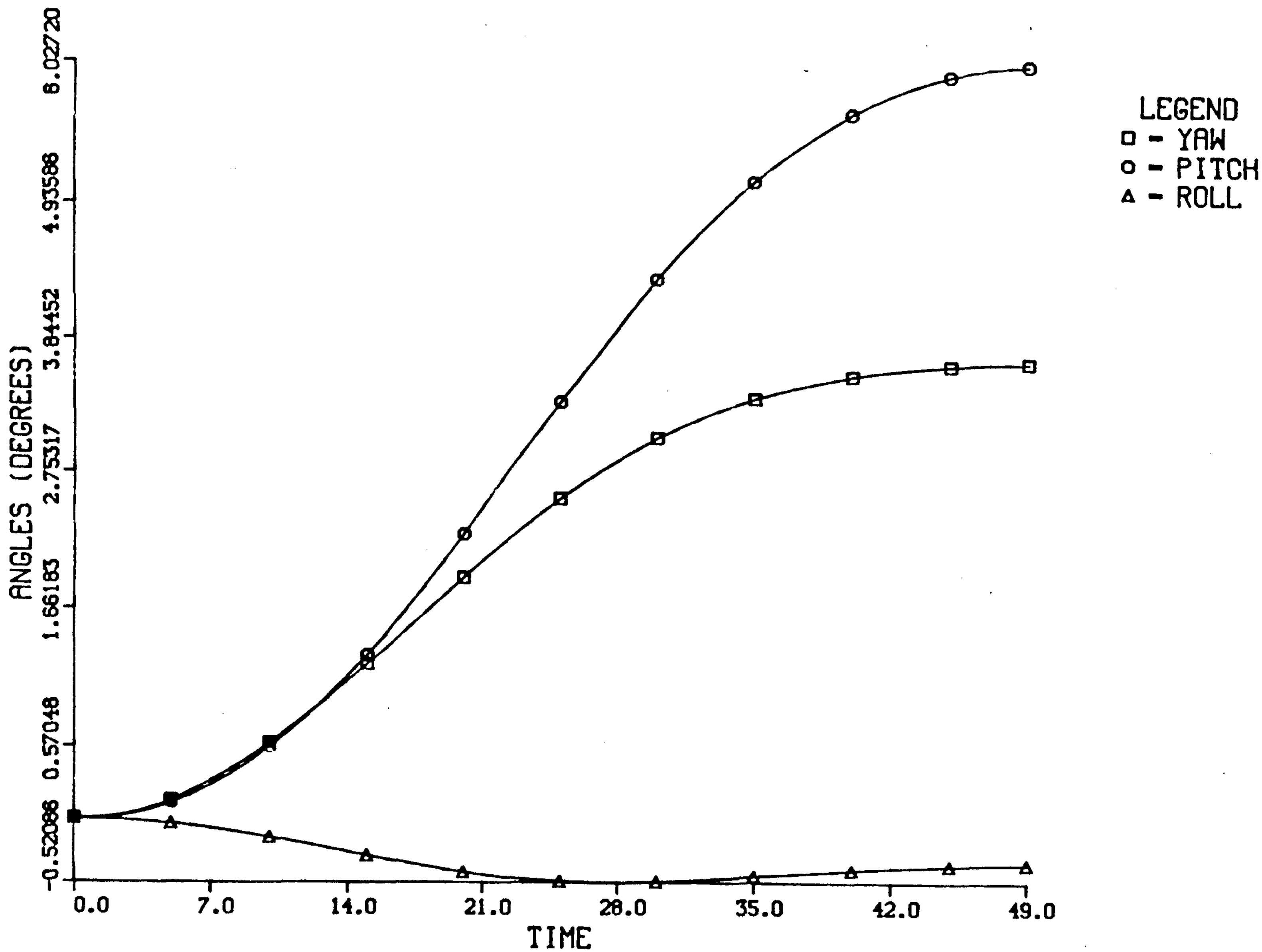

Figure 7c. Roll, pitch and yaw angles.

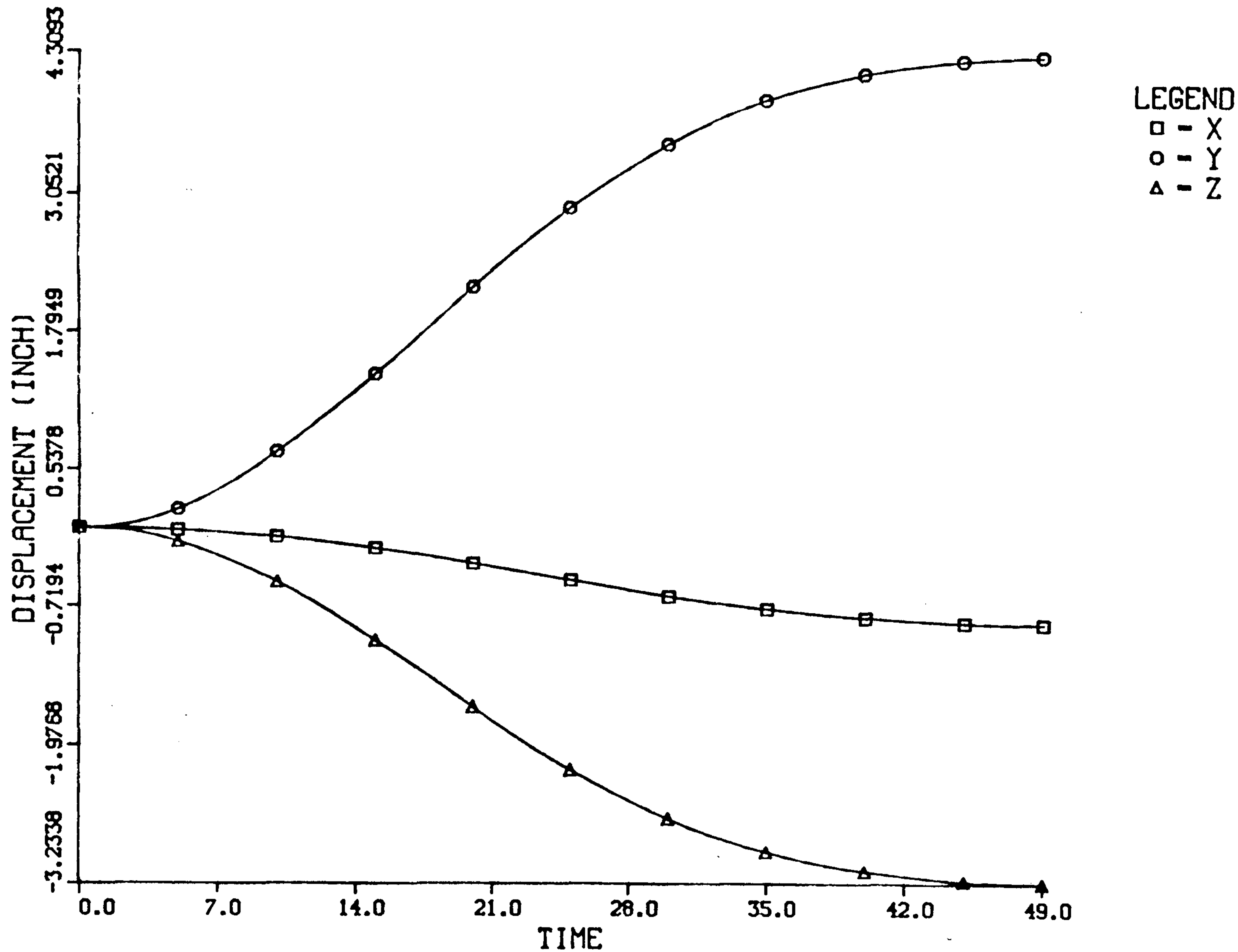

Figure 7d. Position of the base.

CHAPTER II

FINE ATTITUDE CONTROL

Previous Studies

Several researchers studied the problem of orientation of objects in free fall. McDonald [18] studied the problem of how a cat in free fall can rotate itself without pushing against anything. He took pictures of a falling cat and showed that by moving different parts of its body a cat is able to change his orientation and touch the ground on its feet. One year later [19] , he conducted the same experiment on a diver and showed that a diver can turn his body without any help from the board.

After manned space flights started, self-orientation of astronauts became an important subject. To perform certain tasks an astronaut must be able to change the orientation of his body in space. This is easy when the astronaut is in direct contact with some part of the space vehicle. However, if the astronaut is not in contact with the space vehicle he has two alternatives to change its orientation. The astronaut can carry a device which produces forces that help him to change his orientation, or as a second alternative he can move parts of his body to change his orientation.

Kulwicki *et al* [20] discussed self rotation techniques of astronauts in qualitative terms and showed that by a series of arm and leg motions astronauts can rotate their bodies in space without external help.

In 1964, McCrank [21] studied the problem of rotation of a body in free fall in his masters thesis. He derived the equations of motion of a body in free fall using momentum conservation laws and suggested some maneuvers for self-orientation of an astronaut.

V. I. Stephantsov and his colleagues [22], [23] studied methods of orienting the human body under conditions of weightlesness in the absence of a support. They concluded that if a man exerts muscular force and turns his arms through an angle α_1 then the rest of his body, according to conservation of angular momentum, turns in the opposite direction through an angle α_2. The angles of rotation from the starting point of the parts of the body are inversely proportional to their moments of inertia.

Smith and Kane [24] proposed an analytic solution to the self-orientation problem. They derived the differential equations of motion using the angular momentum conservation law. Since the number of equations of the system are smaller than the unknowns, they introduced an optimality criterion and obtained new equations to equate the number of equations with number of unknowns. The new problem turned out to be a nonlinear two point boundary value problem. They solved this boundary value problem using numerical methods.

Later, Kane and Scher [25], and Passerello and Huston [26] studied the orientation of human body problem in space without the aid of external control. Kane and

Scher suggested that if a man can perform both a pitch and a yaw rotation, he can acquire any desired orientation. Passerello and Huston derived the equations of the system using the principle of conservation of angular momentum. They integrated the equations assuming the motions of arms and legs are known and showed that astronauts can change their orientation by using predetermined limb motions.

Vafa and Dubowsky [27] studied the problem of deviations in the satellite orientations due to motions of the manipulator. They proposed that, if the orientation of the satellite exceeds a predetermined value while the manipulator moves, this deviation can be corrected through cyclic motions of the manipulator.

Equations of the System

To derive the equations, the same system in Chapter I is used. The center of mass of the system with reference to a satellite fixed coordinate system $\mathbf{X}_0\mathbf{Y}_0\mathbf{Z}_0$ is

$$\mathbf{R} = \frac{1}{m}\sum_{i=1}^{n} m_i {}^0\mathbf{r}_i \tag{50}$$

where m is the total mass of the system, m_i is the mass of the i^{th} link, ${}^0\mathbf{r}_i$ is the vector from the fixed body frame to the center of mass of link i, and $\mathbf{R}$ is the vector from the body fixed frame to the center of mass of the system.

The angular momentum of the system about the center of mass is

$$\mathbf{H}_{cm} = {}^0\mathbf{I}_0\mathbf{w}_0 + \sum_{i=1}^{n} {}^0\mathbf{I}_i(\mathbf{w}_0 + {}^0\mathbf{w}_i) + m_0(\mathbf{R}\times\dot{\mathbf{R}}) + \sum_{i=1}^{n} m_i[({}^0\mathbf{r}_i - \mathbf{R})\times({}^0\dot{\mathbf{r}}_i - \dot{\mathbf{R}})] \tag{51}$$

where ${}^0\mathbf{I}_0$ is the inertia matrix of the OMV about the $\mathbf{X}_0\mathbf{Y}_0\mathbf{Z}_0$ coordinate system and ${}^0\mathbf{I}_i$ is the inertia matrix of link i about a coordinate frame parallel to the

$X_0Y_0Z_0$ system. ${}^0\mathbf{w}_i$ is the angular velocity of link i with respect to the $X_0Y_0Z_0$ reference frame. It is assumed that no external force is applied on the system and the system is at rest initially. Then the angular momentum of the system relative to its center of mass is conserved and is equal to zero.

Equation (51) is expanded to obtain

$$\mathbf{H}_{cm} = {}^0\mathbf{I}_0\mathbf{w}_0 + \sum_{i=1}^{n} {}^0\mathbf{I}_i(\mathbf{w}_0 + {}^0\mathbf{w}_i) + m(\mathbf{R} \times \dot{\mathbf{R}})$$

$$-\mathbf{R} \times \sum_{i=1}^{n} m_i {}^0\dot{\mathbf{r}}_i - \sum_{i=1}^{n} m_i {}^0\mathbf{r}_i \times \dot{\mathbf{R}} + \sum_{i=1}^{n} m_i {}^0\mathbf{r}_i \times {}^0\dot{\mathbf{r}}_i = 0 \tag{52}$$

where ${}^0\dot{\mathbf{r}}_i$ is the velocity of the center of mass of link i and $\dot{\mathbf{R}}$ is the velocity of the center of mass of the total system relative to the inertial frame, respectively. Differentiating equation (50) with respect to time gives:

$$\dot{\mathbf{R}} = \frac{1}{m} \sum_{i=1}^{n} m_i {}^0\dot{\mathbf{r}}_i \tag{53}$$

Substitute equation (53) into equation (52). Then the third and fourth terms in equation (53) cancel and the equation reduces to

$$\mathbf{H}_{cm} = {}^0\mathbf{I}_0\mathbf{w}_0 + \sum_{i=1}^{n} {}^0\mathbf{I}_i(\mathbf{w}_0 + {}^0\mathbf{w}_i) - \sum_{i=1}^{n} m_i {}^0\mathbf{r}_i \times \dot{\mathbf{R}} + \sum_{i=1}^{n} m_i {}^0\mathbf{r}_i \times {}^0\dot{\mathbf{r}}_i \tag{54}$$

Using moving coordinate frames [28], ${}^0\dot{\mathbf{r}}_i$ and $\dot{\mathbf{R}}$ can be written as

$${}^0\dot{\mathbf{r}}_i = \mathbf{v}_i + \mathbf{w}_0 \times {}^0\mathbf{r}_i \tag{55}$$

$$\dot{\mathbf{R}} = \mathbf{V} + \mathbf{w}_0 \times \mathbf{R} \tag{56}$$

where $\mathbf{v}_i$ is the velocity of the center of mass of link i and $\mathbf{V}$ is the velocity of the center of mass of the total system relative to the $X_0Y_0Z_0$ frame, respectively. $\mathbf{w}_0$ is the angular velocity of the satellite main body with respect to the inertial frame. The orientation of the main body with respect to the inertial frame is given by the roll, pitch and yaw angles as shown by equation (2), and angular velocity of the main body with respect to the inertial frame in terms of rate of change of roll, pitch and yaw angles is given by equation (10).

Substituting equations (55) and (56) into equation (54) and expanding the terms gives

$$\begin{aligned}\mathbf{H}_{cm} = {}&{}^0\mathbf{I}_0\mathbf{w}_0 + \sum_{i=1}^{n} {}^0\mathbf{I}_i(\mathbf{w}_0 + {}^0\mathbf{w}_i) - m\mathbf{R} \times \mathbf{V} - m\mathbf{R} \times (\mathbf{w}_0 \times \mathbf{R}) \\ &+ \sum_{i=1}^{n} m_i {}^0\mathbf{r}_i \times \mathbf{v}_i + \sum_{i=1}^{n} m_i {}^0\mathbf{r}_i \times (\mathbf{w}_0 \times {}^0\mathbf{r}_i)\end{aligned} \tag{57}$$

Using the vector triple product property [29]

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \cdot \mathbf{c})\mathbf{b} - (\mathbf{b} \cdot \mathbf{a})\mathbf{c} \tag{58}$$

The $\mathbf{R} \times (\mathbf{w}_0 \times \mathbf{R})$ and ${}^0\mathbf{r}_i \times (\mathbf{w}_0 \times {}^0\mathbf{r}_i)$ terms can be written as

$$\mathbf{R}^2\mathbf{w}_0 - (\mathbf{w}_0 \cdot \mathbf{R})\mathbf{R} \tag{59}$$

$${}^0\mathbf{r}_i^2\mathbf{w}_0 - (\mathbf{w}_0 \cdot {}^0\mathbf{r}_i){}^0\mathbf{r}_i \tag{60}$$

Using vector dyadic product [30], the second terms in equations (59) and (60) can be written as $(\mathbf{RR})\mathbf{w}_0$ and $({}^0\mathbf{r}_i{}^0\mathbf{r}_i)\mathbf{w}_0$ where

$$\mathbf{RR} = \begin{pmatrix} R_xR_x & R_xR_y & R_xR_z \\ R_yR_x & R_yR_y & R_yR_z \\ R_zR_x & R_zR_y & R_zR_z \end{pmatrix} \tag{61}$$

Then equations (59) and (60) become

$$\mathrm{R} \times (\mathrm{w}_0 \times \mathrm{R}) = (\mathrm{R} \cdot \mathrm{RE} - \mathrm{RR})\mathrm{w}_0 \tag{62}$$

$$^0\mathbf{r}_i \times (\mathrm{w}_0 \times {}^0\mathbf{r}_i) = ({}^0\mathbf{r}_i \cdot {}^0\mathbf{r}_i\mathrm{E} - {}^0\mathbf{r}_i{}^0\mathbf{r}_i)\mathrm{w}_0 \tag{63}$$

where E is a unit dyadic. Then equation (57) becomes

$$[{}^0\mathrm{I}_0 + \sum_{i=1}^{n}({}^0\mathrm{I}_i + \mathrm{J}_i) - \mathrm{K}]\mathrm{w}_0 + \sum_{i=1}^{n} {}^0\mathrm{I}_i{}^0\mathrm{w}_i + \sum_{i=1}^{n} m_i{}^0\mathbf{r}_i \times \mathbf{v}_i - m\mathrm{R} \times \mathrm{V} = 0 \tag{64}$$

where

$$\mathrm{J}_i = m_i({}^0\mathbf{r}_i \cdot {}^0\mathbf{r}_i\mathrm{E} - {}^0\mathbf{r}_i{}^0\mathbf{r}_i) \tag{65}$$

$$\mathrm{K} = m(\mathrm{R} \cdot \mathrm{RE} - \mathrm{RR}) \tag{66}$$

J_i is the inertia dyadic of the i^{th} link relative to the $\mathrm{X}_0\mathrm{Y}_0\mathrm{Z}_0$ coordinate system as if the link were a point mass situated at the center of mass of the i^{th} link. K is the inertia dyadic of the total system relative to the $\mathrm{X}_0\mathrm{Y}_0\mathrm{Z}_0$ coordinate system as if the total system were a point mass situated at the center of mass of the total system. The linear velocity of the center of mass of link i, with respect to the $\mathrm{X}_0\mathrm{Y}_0\mathrm{Z}_0$ coordinate frame , $\mathbf{v}_i$, can be obtained using the relation given by equations (12) and (13)

$$\mathbf{v}_i = \sum_{j=1}^{i} \frac{\partial^0\mathbf{T}_i}{\partial q_j}\dot{q}_j{}^i\mathbf{r}_i. \tag{67}$$

Substituting equations (55) and (67) into equation (53) and equating corresponding terms of equation (56) gives the velocity of the center of mass of the system relative to the $X_0Y_0Z_0$ coordinate frame as:

$$V = \frac{1}{m}\sum_{i=1}^{n} m_i(\sum_{j=1}^{i} \frac{\partial^0 T_i}{\partial q_j}\dot{q}_j)^i r_i \tag{68}$$

The angular velocity of link i with respect to the $X_0Y_0Z_0$ coordinate frame is

$$^0w_i = \sum_{j=1}^{i} {}^0U_j{}^j u_j \dot{q}_j. \tag{69}$$

Substitute equations (67)-(69) into equation (61). Then

$$[^0I_0+\sum_{i=1}^{n}(^0I_i+J_i)-K]w_0+\sum_{i=1}^{n}[(\sum_{j=i}^{n}{}^0I_j)^0U_i{}^iu_i+\sum_{j=i}^{n}m_j[(^0r_j-R)\times(\frac{\partial^0 T_j}{\partial q_i}{}^j r_j)]]\dot{q}_i = 0 \tag{70}$$

Equation (70) gives 3 equations in $n+3$ unknowns for an n-degrees-of-freedom manipulator. If one knows the time history of joint angles then equation (70) becomes a set of 3 simultaneous, non-linear, non-homogeneous, ordinary differential equations with 3 unknowns. ψ, β and τ are the dependent variables, t is the independent variable, and joint angles are the intermediate variables.

The OMV will have different orientations depending on the manipulator path chosen from one position to another. Assume that manipulator moves along a closed path, i.e., it starts from a position and returns to the same position at the end of the motion. Since the OMV orientation depends on the manipulator motion path, the final OMV orientation will change if the manipulator moves along one path in joint space and returns to the initial position by another path. In this study functions used in specifying q_i are of the form suggested by Smith and Kane [23].

$$f(t) = f_0 + (f_1 - f_0)[t/T - \frac{1}{2\pi}\sin(\frac{2\pi t}{T})] \tag{71}$$

where T is the duration of angle change, and f_0 and f_1 are the values of f for $t = 0$ and $t = T$. This function has the property of vanishing first and second derivatives of the angles at $t = 0$ and $t = T$, so the system starts from rest and returns to the rest position at the end of the motion.

To find the desired base orientation corrections it is assumed that changes in joint angles are small and these small changes result in small changes in the orientation of the OMV. Since rotations are assumed small they can be classified as vectors and can be added vectorially [31]. Only one pair of joint angles are allowed to move at a time while the other joints are kept fixed. An example of the motion of a pair of joint angles is shown by Figure 8.

Computer Simulation of the Algorithm

A Fortran program was written to find the sequence of motions to bring the OMV to the desired orientation. Listing of the program is given in Appendix B. The algorithm of the program is as follows:

1) Read the initial and desired values of the attitude angles of the OMV.

2) Find the pair of joints, establish the order of change in these joints, and determine the magnitude and direction in these angles which rotates the OMV

towards the desired orientation. The following steps are applied to find the pair of joints which rotates the OMV towards the desired orientation

a) Pick up a pair of joints from a look-up table. This table contains pair of joints, order of change in these joints, magnitude and direction (positive or negative) of changes in the joint angles.

b) By changing this pair of joint angles according to equation (71) and integrating equation (70) using Runge-Kutta algorithm, find the orientation of the OMV at the end of the motion. Denote the new attitude angles of the OMV as τ_{new}, β_{new} and ψ_{new}.

c) If all angles converge to their desired values, i.e.,

$$|\tau_{\text{desired}} - \tau_{\text{new}}| < |\tau_{\text{desired}} - \tau_{\text{initial}}|$$

$$|\beta_{\text{desired}} - \beta_{\text{new}}| < |\beta_{\text{desired}} - \beta_{\text{initial}}|$$

$$|\psi_{\text{desired}} - \psi_{\text{new}}| < |\psi_{\text{desired}} - \psi_{\text{initial}}|$$

go to 3, else go back to a) to pick up another pair of joints.

3) Determine the number of cycles, k, for which this motion will be applied. The number is the minimum of the following divisions plus 1

$$\frac{|\tau_{\text{desired}} - \tau_{\text{initial}}|}{|\tau_{\text{new}} - \tau_{\text{initial}}|}$$

$$\frac{|\beta_{\text{desired}} - \beta_{\text{initial}}|}{|\beta_{\text{new}} - \beta_{\text{initial}}|}$$

$$\frac{|\psi_{\text{desired}} - \psi_{\text{initial}}|}{|\psi_{\text{new}} - \psi_{\text{initial}}|}$$

4) Apply the motion of the manipulator k times. Integrate equation (70) using the Runge-Kutta algorithm to find the orientation of the OMV at the end of the motion. Denote the new values of the attitude angles of the OMV as τ_k, β_k and ψ_k.

5) Calculate the sum of the error squared of attitude angles as

$$(\tau_{\text{desired}} - \tau_k)^2 + (\beta_{\text{desired}} - \beta_k)^2 + (\psi_{\text{desired}} - \psi_k)^2$$

Compare the sum of the error squared with the error criteria. If the criteria is satisfied stop, else assign

$$\tau_{\text{initial}} = \tau_k, \quad \beta_{\text{initial}} = \beta_k, \quad \psi_{\text{initial}} = \psi_k$$

and go back to step 2.

This program was tested for an OMV with a 6-degree-of-freedom manipulator. But, only the first 3 links of the manipulator were allowed to move, because the last 3 links of the manipulator have very little effects on the OMV due to very small inertias of these 3 links compared to the inertias of the rest of the system. Initial values for τ, β and ψ for Figures 9-11 are $\tau = 0.0^0$, $\beta = 0.0^0$ and $\psi = 0.0^0$. The desired final values for these angles are $\tau = 6.0^0$, $\beta = 10.0^0$ and $\psi = 2.0^0$.

Results of this run are shown in Figures 9-14. Figures 9-11 show the changes in angles τ, β and ψ, respectively. As can be seen from these figures all of the angles converge to the desired positions. The OMV oscillates in response to the manipulator motion. However, the mean orientation of the OMV changes continuously towards the desired orientation. Figure 12 shows the mean orientation of the

OMV after each cycle of the motion. Figure 13 represents the error squared of the attitude angles after each cycle of the manipulator motion and, as seen from the figure, squared error gets smaller after each cycle of the motion. To show the small rotations can be added vectorially and the order of motions will not effect the final orientation of the OMV, order of the motions were changed and the program run again. Results of these two runs are plotted in Figure 14. As seen from the figure, orientation of the OMV at the end of the manipulator motions is the same for both cases. In Table 2 the order, magnitude and direction of the joint angle changes are given. The number of cycles of each motion is also given in Table 2. Initially all joint angles are 0.0^0.

TABLE 2

CHANGES IN JOINT ANGLES TO ROTATE THE OMV TO THE DESIRED ORIENTATION

No of joint moves first	Change in joint angle	No of joint moves second	Change in joint angle	Number of cycles
1	10.0	2	10.0	8
2	-10.0	3	-10.0	6
1	10.0	2	10.0	1
2	-10.0	3	-10.0	21

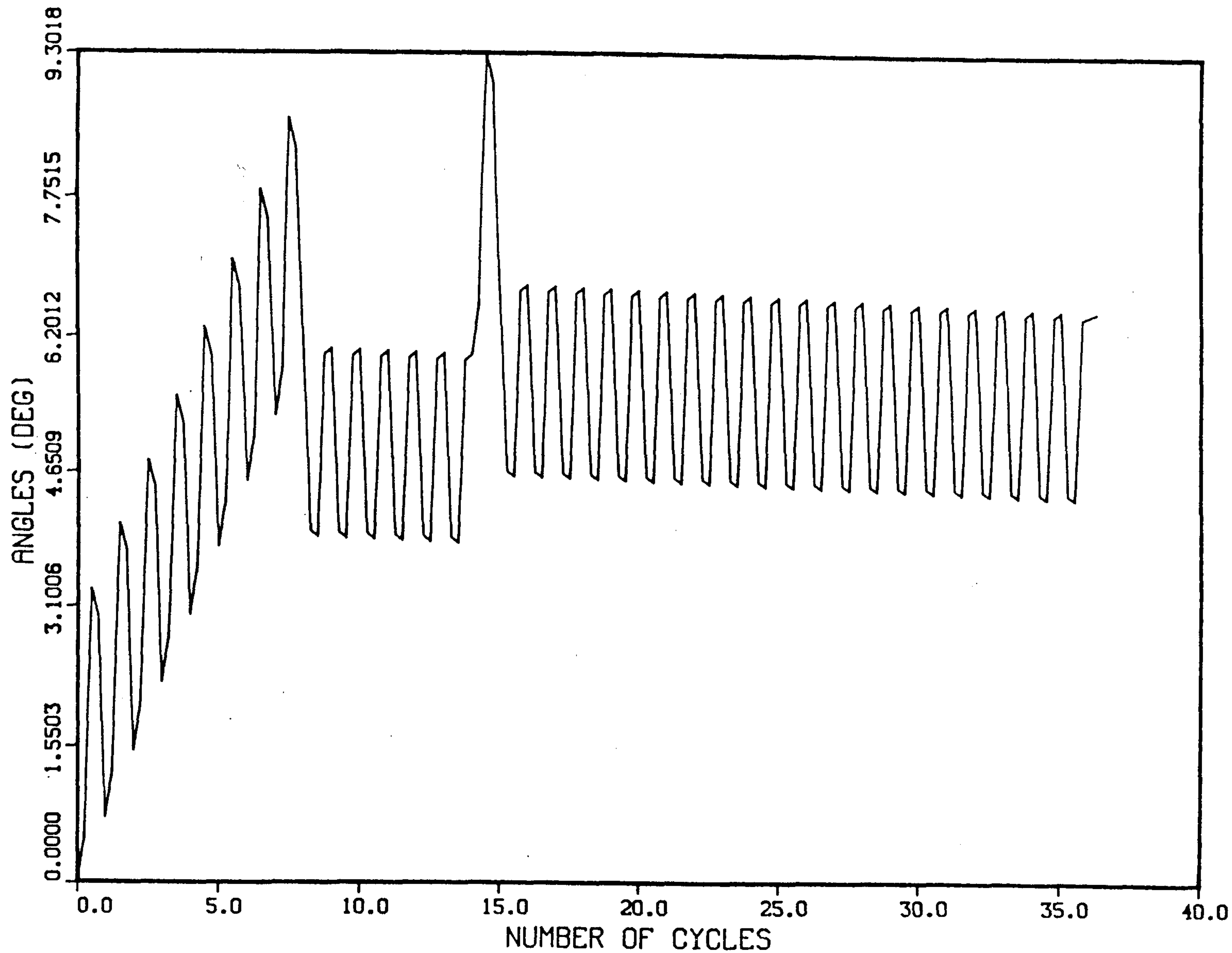

Figure 9. Change in yaw (τ) as a result of manipulator motions.

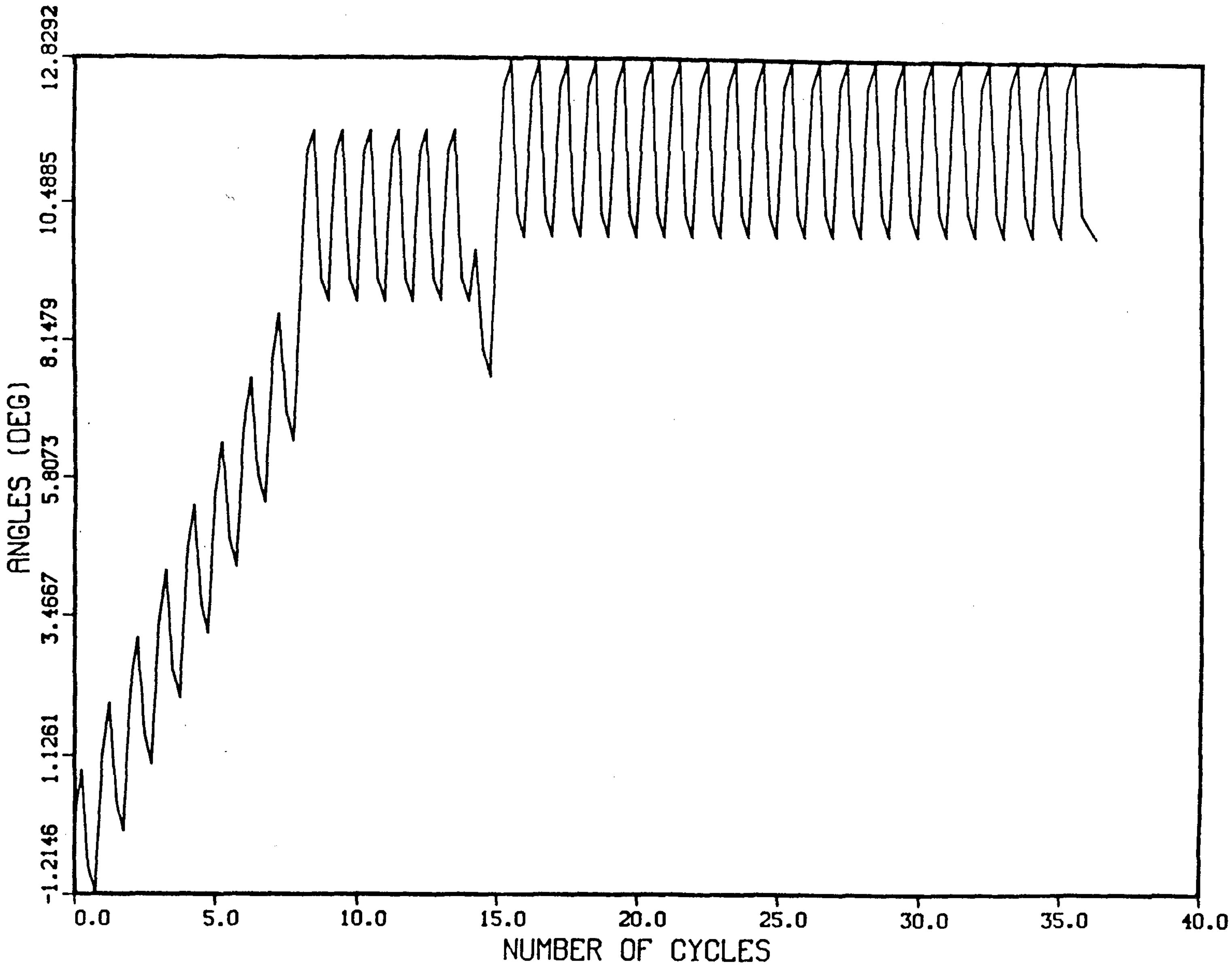

Figure 10. Change in pitch (β) as a result of manipulator motions.

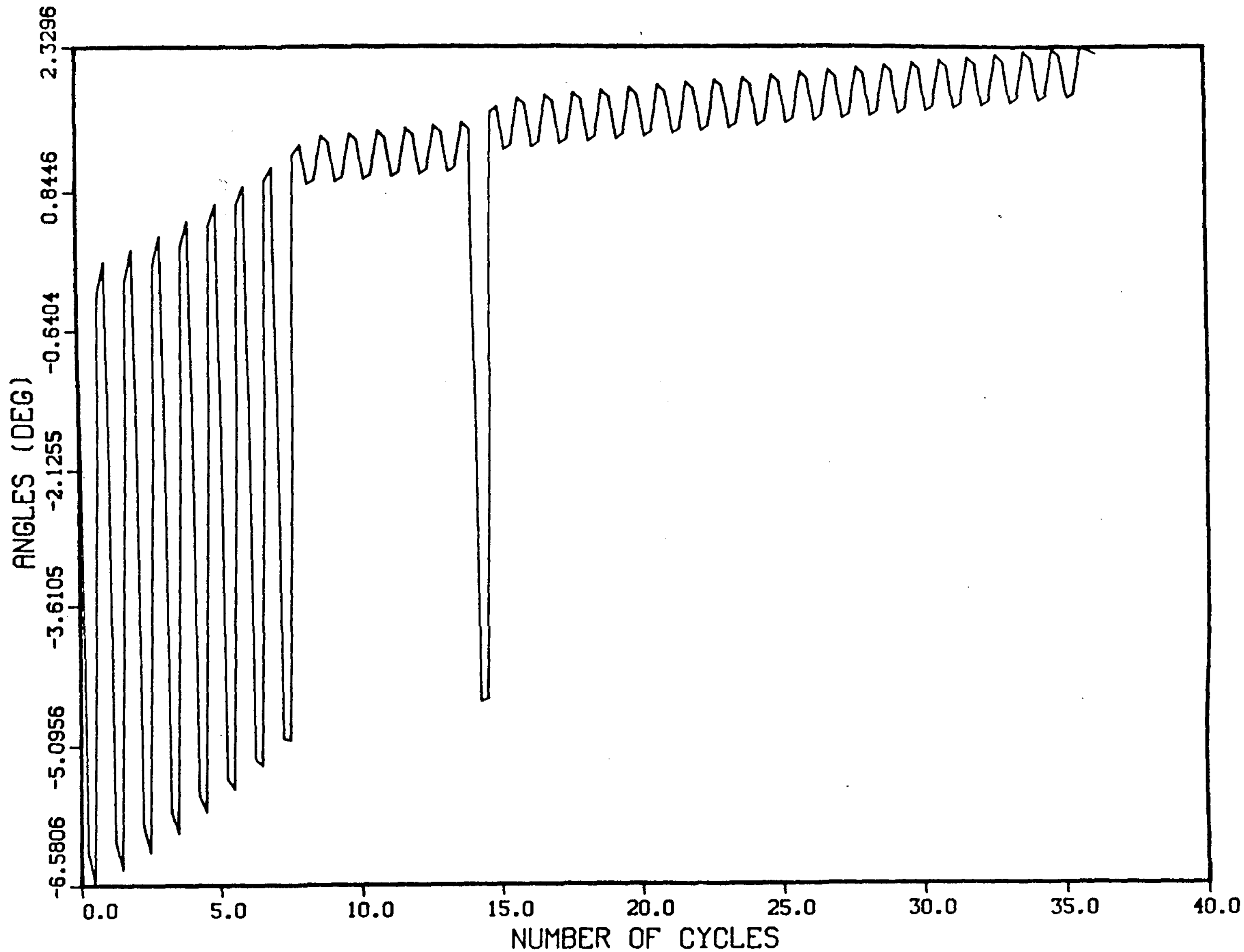

Figure 11. Change in roll (ψ) as a result of manipulator motions.

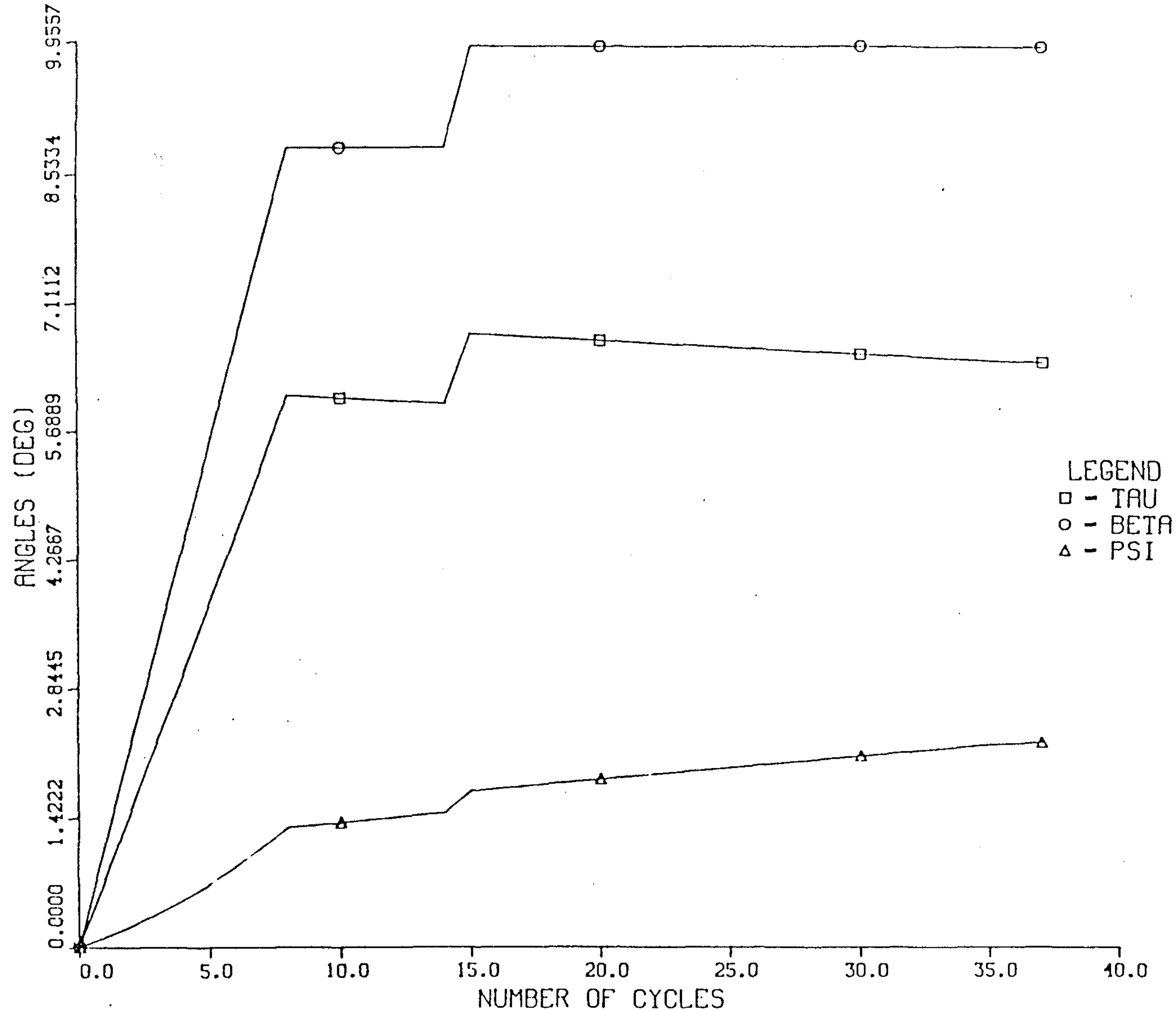

Figure 12. Changes in the attitude angles of the OMV after each cycle of the manipulator motion.

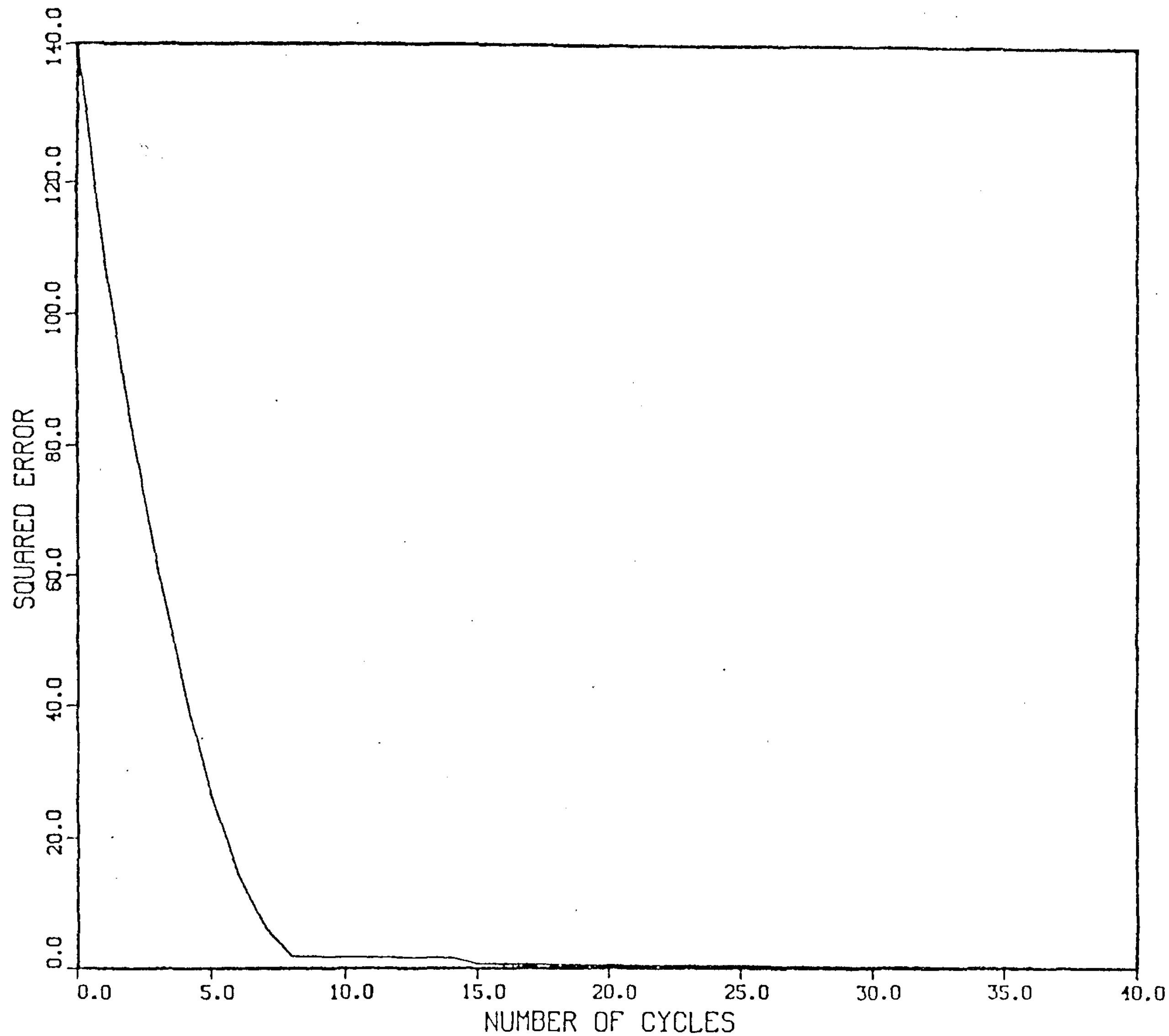

Figure 13. Squared error of the attitude angles of the OMV after each cycle of manipulator motion.

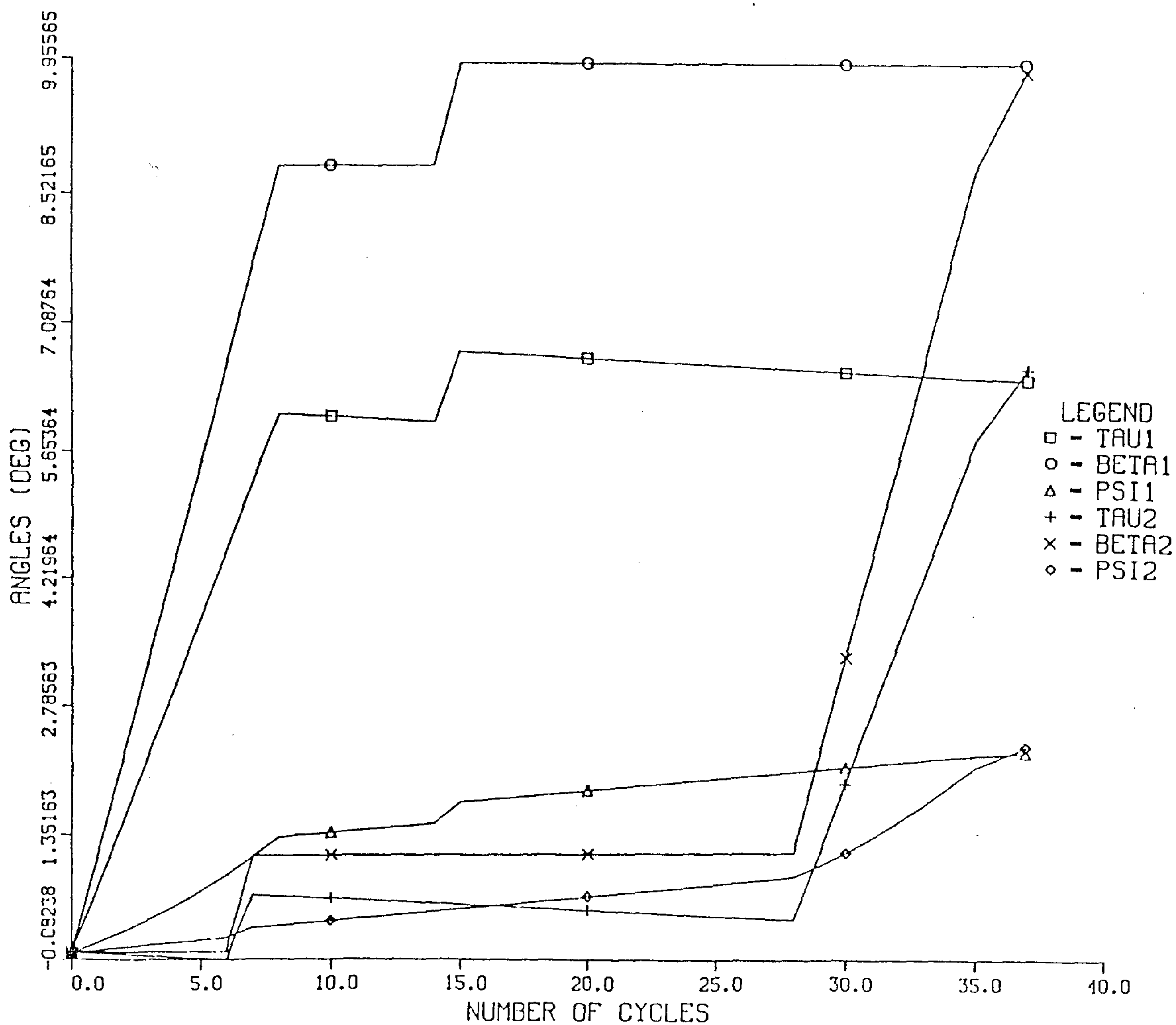

Figure 14. Changes in the attitude angles of the OMV when order of manipulator link motions are different.

CHAPTER III

COMPUTER GRAPHICS SIMULATION

Introduction

The design of robot manipulators requires the study of a number of complicated multilink mechanical systems as a controlled object. The major manipulator design task is to set up a workspace and see if the arm can accomplish the desired task. Often this leads to multiple trials of custom designed grippers and some of the links. A company cannot afford to buy several robots to compare performance characteristics or move these systems within their plants in order to find a suitable application. A number of computer graphic simulation programs have been developed to solve such problems [32], [5]. Simulation provides a way of visualizing robot kinematics without the use of real robots. The answers to questions arising in the preliminary design stages, in the development, and during the tests of robot manipulators and also during their use, can be obtained by simulation.

Another area in which computer graphic simulation programs can be useful is in testing robot control algorithms. Using real robots to test a control algorithm may result in an undesired situation, for example, robots may collide accidentally with obstacles within the workcell. This type of problems can be prevented by first testing the algorithm via a computer graphic simulation program.

In this chapter, results of the control algorithm developed in Chapter II will be animated using a computer graphics simulation program, namely ROBOSIM. First, an overview of ROBOSIM will be given, then simulation of the algorithm will be presented.

An Overview of ROBOSIM

Robosim was initially implemented in FORTRAN on a DEC VAX 11/780 computer. The display terminals were a TEKTRONIX 4014 printer or a dynamic display system, such as Evans & Sutherland PS330 graphics terminal. The software structure of ROBOSIM is divided into three levels. At the lowest level there are subroutines which do the simulation tasks such as vector and matrix computations and display control. The routines at the first level are inflexible, and data must be input in a specific format. At the second level there are routines about display management, subroutines to perform view-point and perspective transformations, and robot control subroutines, such as computation of the Jacobian matrix. At the third level, robots and other workcell components are modelled, programmed dynamically, simulated and viewed by the use of ROBOSIM instructions.

Recently, ROBOSIM was ported to an HP350SRX graphics workstation [33]. The version on the HP350SRX system is fully compatible with the first implementation, but it is written in C. Using the HP350SRX graphics terminal increased the speed and allowed three dimensional graphics with shading, perspective views and colors.

In order to simulate a robot system, the user should write a program consisting of third level ROBOSIM instructions [34]. Each link of the robot can be modelled by a combination of simple geometric objects i. e., box, cylinder, cone etc.. Then these links are translated and/or rotated, and combined with other links to form the robot system. The type (fixed, prismatic or rotational) and configuration of the joints should be supplied by the user. Each link of the robot is stored in a file. This file contains Denavit-Hartenberg parameters, joint type, homogeneous transformation matrix $\mathbf{A}_i$ of the link, generalized inertia matrix of the link and visual representation of the link's geometry. After creating the links, robot motion can be animated using either the VIEW-ROBOT or CONTROL-ROBOT command.

Animation of the Results of Fine Attitude Control Algorithm

To animate the results of the fine attitude control algorithm, a space manipulator system was designed using third level ROBOSIM commands, and motion of the system was monitored on HP350SRX graphics terminal. ROBOSIM simulation commands used in construction of the space manipulator system is given in Appendix C. Satellite main body is modelled by a cylinder, with an imbedded coordinate frame at the center of mass of the cylinder. The manipulator has 6 links. Each link is rotational, and coordinate frames are assigned such that rotation occurs about the z axis. ROBOSIM simulation of the space manipulator system is shown in Figure 15 with satellite attitude angles and manipulator joint angles all equal to 0.0^0. Figures 16-19 show a cycle of motion of the system. In Figure 16, the first joint moves from 0.0^0 to 10.0^0 and the satellite main body rotates about the **z** axis in the opposite direction. This is an expected result of conservation of angular momentum. In a multibody system, if some part of the body rotates in one direction, the rest of the body rotates in the opposite direction. The rotations of the parts of the body are inversely proportional to the moment of inertia ratios of the two parts of the body. In Figure 17, the second joint angle of the manipulator changes from 0.0^0 to 10.0^0. In the following two figures, the first and second joint angles return to their original values, respectively. When the manipulator returns to the initial position in Figure 19, the orientation of the satellite main body is different than the orientation when the cycle started. Figure 20 shows the final position of the system when the cycle of motion is applied several times. This shows that, even though the manipulator returns to its initial position, orientation of the satellite main body has changed.

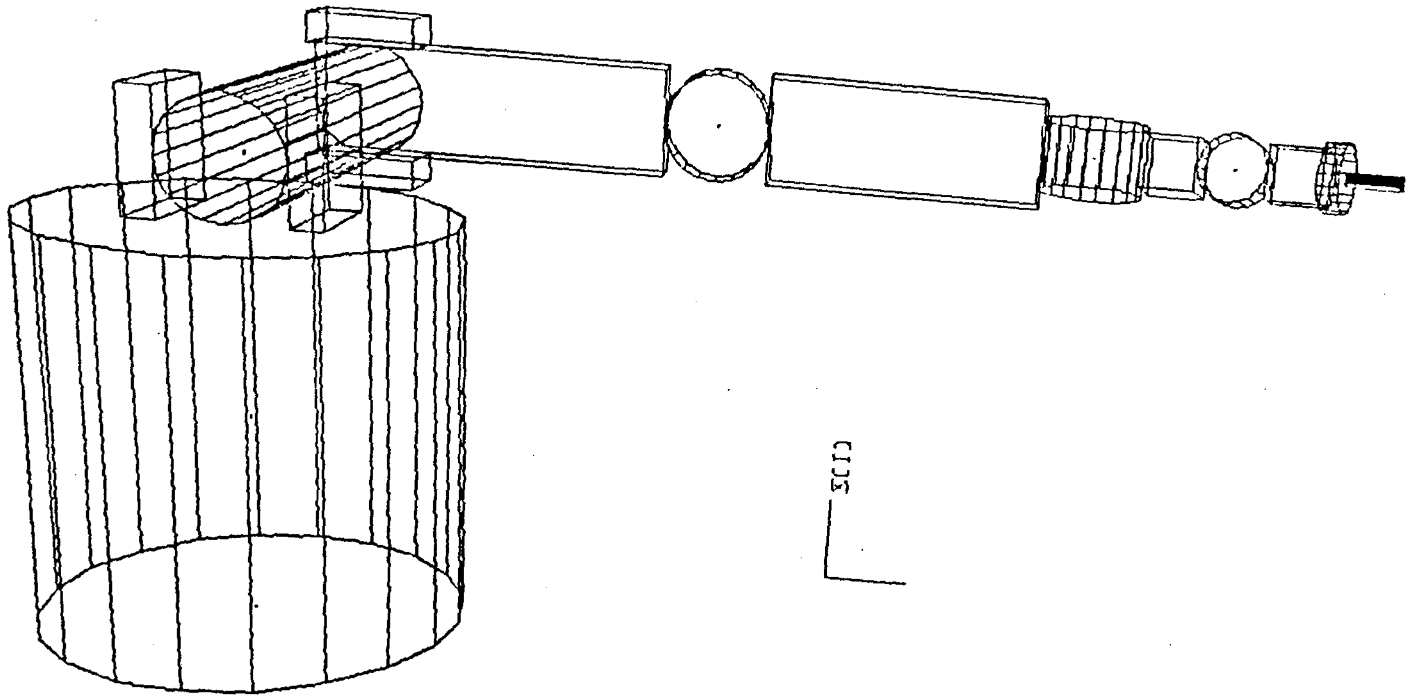

Figure 15. Space manipulator system when all angles equal to 0.0^0.

ROBOSIM
NASA-MSFC

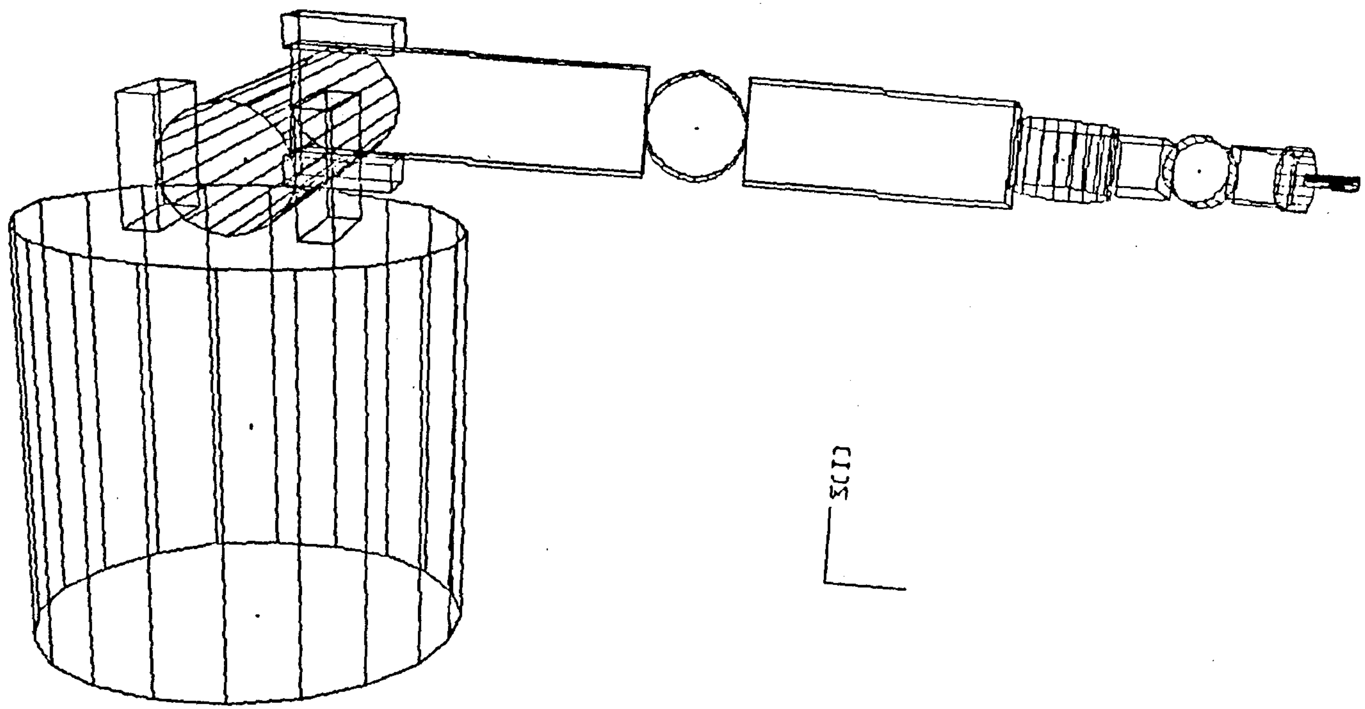

Figure 16. Space manipulator system after first joint rotated from 0.0^0 to 10.0^0.

ROBOSIM
NASA-MSFC

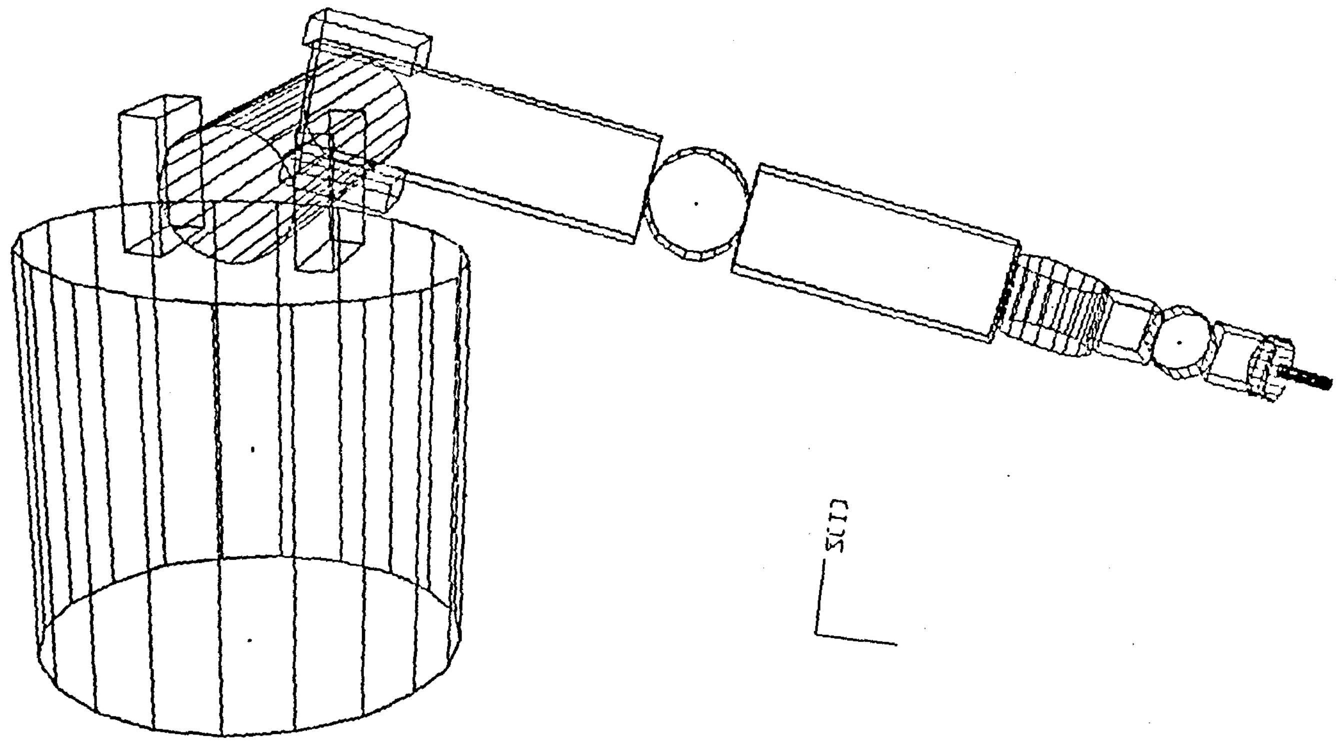

Figure 17. Space manipulator system after second joint rotated from 0.0° to 10.0°, first joint angle is 10.0°.

ROBOSIM
NASA-MSFC

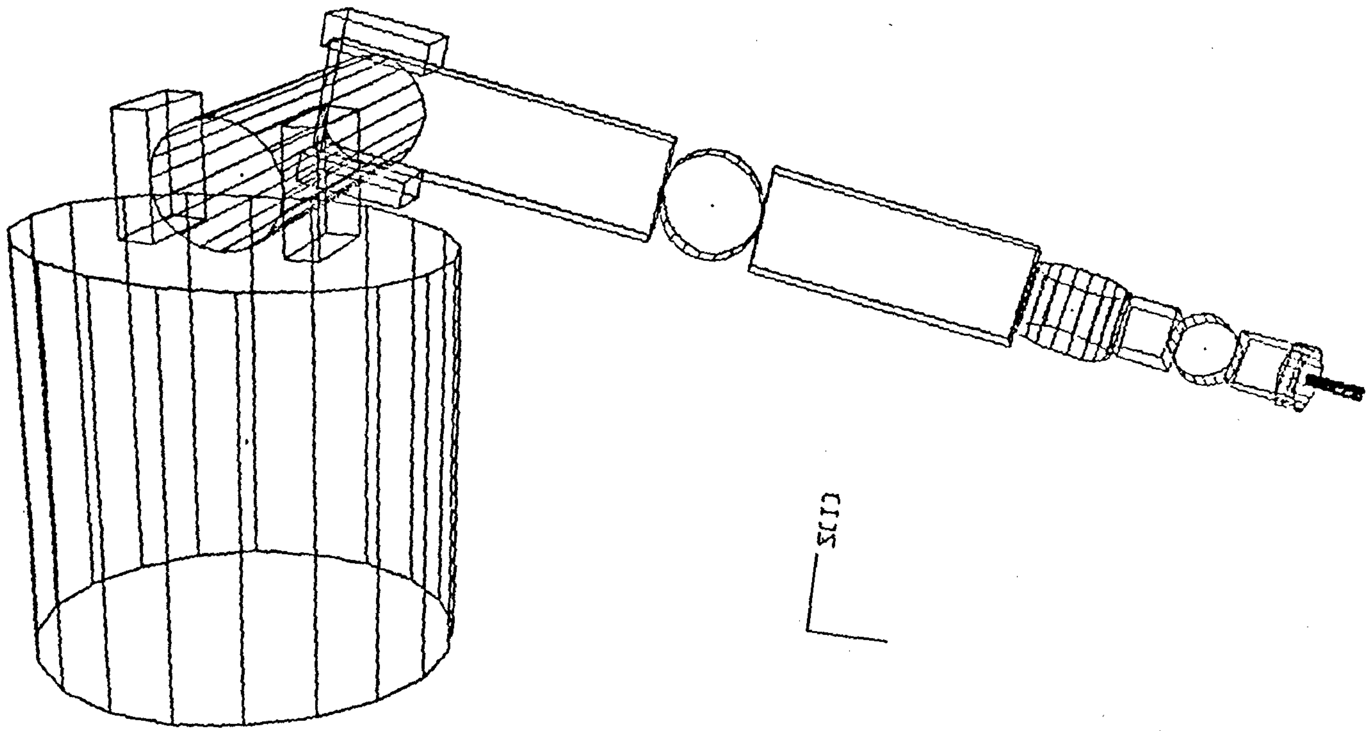

Figure 18. Space manipulator system after first joint angle returned to 0.0^0, second joint angle is 10.0^0

ROBOSIM
NASA-MSFC

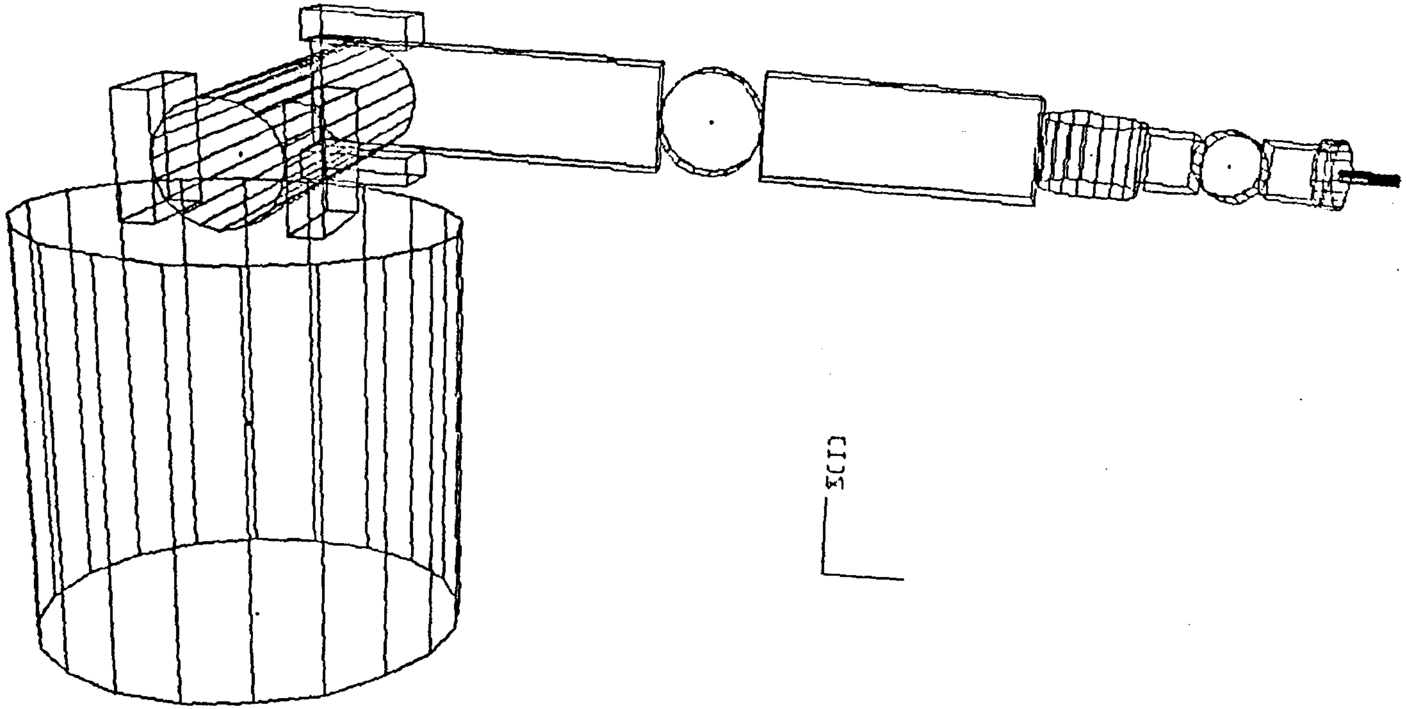

Figure 19. Space manipulator system after second joint angle returned to 0.0°

ROBOSIM
NASA-MSFC

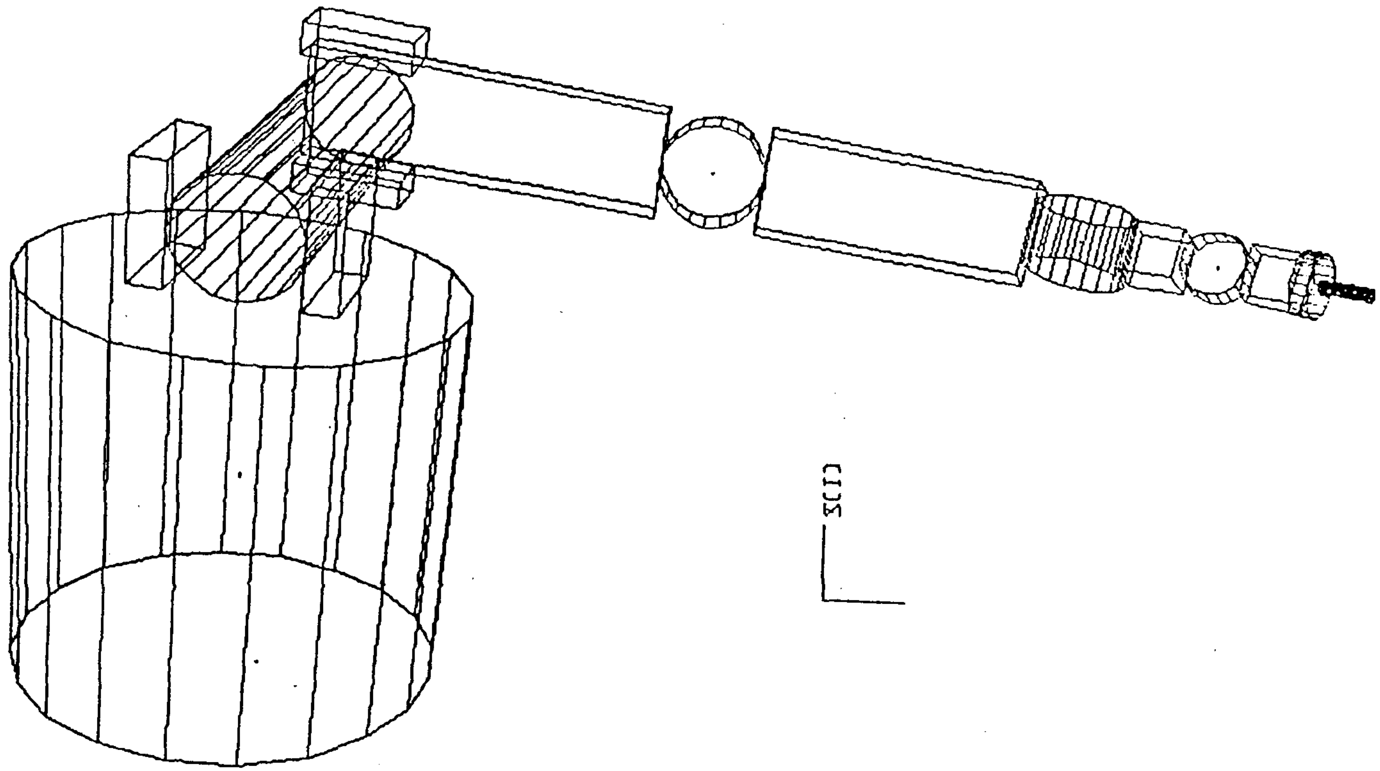

Figure 20. Space manipulator system after manipulator motion applied 8 times

ROBOSIM
NASA-MSFC

CHAPTER IV

DISCUSSION AND FURTHER RESEARCH

Discussion

In the near future space manipulators may replace astronauts in doing some of the jobs which astronauts do today. Before space manipulators can be utilized effectively, the control algorithms to control the end effector of the manipulator must be developed. Since the base of the manipulator moves as a result of manipulator motions, control algorithms which were developed for fixed base manipulators can not be used and new algorithms are required.

In this thesis, a modified resolved motion rate control algorithm was developed to control the end effector of the space manipulator. The assumptions made were that no external forces were applied to the system and the angular momentum conservation law holds. A generalized Jacobian matrix of a space manipulator system was derived and it was shown that the manipulator end effector moves in the desired direction in spite of the rotations and translations of the satellite main body.

The satellite main body rotates when the manipulator moves. If the satellite main body orientation becomes misaligned while docking with another satellite, its orientation must be corrected to accomplish the docking process. Finding a way to rotate the main body without using thrusters will reduce the cost of operations and increase the life of the satellite system.

In this thesis, an algorithm was developed to rotate the main body using manipulator motions. It was assumed that the law of angular momentum conservation holds. It was also assumed that the attitude angle rotations due to manipulator motion were small, so they could be added vectorially. A computer simulation program was written and it was shown that it is possible to find a sequence of manipulator motions which rotate the satellite main body in the desired direction. Motion of the satellite main body was also animated using the ROBOSIM computer graphic simulation program [34].

Further Research

For further study, the following suggestions are presented for the reader's consideration.

1) In calculating the kinematics the system was considered to be a free-flying multibody system with no attitude control of the satellite main body. Sometimes it is necessary to control the satellite main body around some axes and allow the main body to rotate around the remaining axes. The algorithm developed in this thesis can be modified to keep the main body fixed around some axes while allowing it to rotate around the remaining axes.

2) The kinematics algorithm was developed for a system with one manipulator arm. The algorithm can be extended to include a space manipulator system with

two or more manipulator arms.

3) A space experiment may be ruined if there is a change in the orientation of the space vehicle. Motion of a manipulator arm (perhaps used in performing the experiment) may cause rotation in the main body. In this case, motion of a second arm can be used to cancel out the effect of the "working" arm. Thus, the working arm will be able to do its task without causing problems to its own or other experiments. The algorithm developed in this thesis can be extended to accomplish cooperative work of two or more manipulator arms.

4) The fine attitude control algorithm was developed without consideration of any type of minimization criterion. The algorithm can be modified to determine the manipulator motions which minimize power consumption while rotating the satellite main body towards the desired orientation. This would reduce the cost of the operation. Other criteria might be considered include minimum time or minimum motion.

APPENDIX A

SPACE MANIPULATOR KINEMATICS PROGRAM

```
C      MAIN PROGRAM
C      THIS PROGRAM CALCULATES THE KINEMATICS OF A SPACE
C      MANIPULATOR USING A MODIFIED RESOLVED MOTION RATE
C      CONTROL ALGORITHM
       DOUBLEPRECISION AM(7),R0(4),R1(4),R2(4),R3(4),R4(4),R5(4)
       DOUBLEPRECISION RPY(4,4),A1(4,4),A2(4,4),A3(4,4),A4(4,4)
       DOUBLEPRECISION RPYB(4,4),RPYT(4,4),X,Y,Z,A6(4,4),R6(4)
       COMMON AM,R0,R1,R2,R3,R4,R5,R6,X,Y,Z,RPY,A1,A2,A3,A4,A5,A6
       COMMON RPYB,RPYT,PI
       DOUBLEPRECISION TA(6),ALP1,BET1,TAU1,RT(4),PN(4),ALP3,BET3
       DOUBLEPRECISION A5(4,4),T6(4,4),DJ(6,6),BIR,PI,ZEROM(4,4)
       DOUBLEPRECISION T1(4,4),T2(4,4),T3(4,4),T4(4,4),T5(4,4)
       DOUBLEPRECISION Q0X(4,4),Q0Y(4,4),Q0Z(4,4),RPYA(4,4)
       DOUBLEPRECISION QA(4,4),A56(4,4),T5TA(4,4),T65A(4,4),AI40(4,4)
       DOUBLEPRECISION A12(4,4),A13(4,4),A14(4,4),A15(4,4),A16(4,4)
       DOUBLEPRECISION T61A(4,4),A23(4,4),A24(4,4),A25(4,4),A26(4,4)
       DOUBLEPRECISION A34(4,4),A35(4,4),A36(4,4),A45(4,4),A46(4,4)
       DOUBLEPRECISION C(3,9),QX(4),QY(4),QZ(4),QAL(4),QB(4),QTA(4)
       DOUBLEPRECISION Q1(4),Q2(4),Q3(4),Q4(4),Q5(4),Q6(4),SO1(3,3)
       DOUBLEPRECISION C33(3,3),C39(3,9),C33I(3,3),HS(6,3),HM(6,6)
       DOUBLEPRECISION AIS(4,4),AISI(4,4),AIM(3,6),T(6),SO(3,3)
       DOUBLEPRECISION AISM(3,6),HSIM(6,6),AJ(6,6),TD(4,6),TF(6)
       DOUBLEPRECISION T1X(4,4),T2X(4,4),T3X(4,4),T4X(4,4),T5X(4,4)
       DOUBLEPRECISION T1Y(4,4),T2Y(4,4),T3Y(4,4),T4Y(4,4),T5Y(4,4)
       DOUBLEPRECISION T1Z(4,4),T2Z(4,4),T3Z(4,4),T4Z(4,4),T5Z(4,4)
       DOUBLEPRECISION T1A(4,4),T2A(4,4),T3A(4,4),T4A(4,4),T5A(4,4)
       DOUBLEPRECISION T1B(4,4),T2B(4,4),T3B(4,4),T4B(4,4),T5B(4,4)
       DOUBLEPRECISION T6X(4,4),T6Y(4,4),T6Z(4,4),T6A(4,4),T6B(4,4)
       DOUBLEPRECISION T1TA(4,4),T2TA(4,4),T3TA(4,4),T4TA(4,4)
       DOUBLEPRECISION T6TA(4,4),T62A(4,4),T63A(4,4),T64A(4,4)
       DOUBLEPRECISION T11(4,4),T21(4,4),T31(4,4),T41(4,4),T51(4,4)
       DOUBLEPRECISION T22(4,4),T32(4,4),T42(4,4),T52(4,4),T33(4,4)
       DOUBLEPRECISION T43(4,4),T53(4,4),T54(4,4),T63(4,4),T64(4,4)
       DOUBLEPRECISION T61(4,4),T62(4,4),T65(4,4),AI6(4,4)
       DOUBLEPRECISION AI0(4,4),T44(4,4),T55(4,4),T66(4,4),T66A(4,4)
       DOUBLEPRECISION AI1(4,4),AI2(4,4),AI3(4,4),AI4(4,4),AI5(4,4)
```

```
      DOUBLEPRECISION AI00(4,4),AI10(4,4),AI20(4,4),AI30(4,4)
      DOUBLEPRECISION AI50(4,4),AI60(4,4),C1(4),C2(4),C3(4),C4(4)
      DOUBLEPRECISION C6(4),C7(4),C8(4),C9(4),AI0A(3),AI1K(3)
      DOUBLEPRECISION C5(4),AI2K(3),AI0T(3),R60(4),A6C(4)
      DOUBLEPRECISION AI3K(3),AI4K(3),AI5K(3),AI6K(3),AI0B(3)
      DOUBLEPRECISION R00(4),R10(4),R20(4),R30(4),R40(4),R50(4)
      DOUBLEPRECISION S1A(4),S2A(4),S3A(4),S4A(4),S5A(4),S6A(4)
      DOUBLEPRECISION S1B(4),S2B(4),S3B(4),S4B(4),S5B(4),S6B(4)
      DOUBLEPRECISION S1T(4),S2T(4),S3T(4),S4T(4),S5T(4),S6T(4)
      DOUBLEPRECISION S11(4),S21(4),S31(4),S41(4),S51(4),S61(4)
      DOUBLEPRECISION S12(4),S22(4),S32(4),S42(4),S52(4),S62(4)
      DOUBLEPRECISION S13(4),S23(4),S33(4),S43(4),S53(4),S63(4)
      DOUBLEPRECISION S14(4),S24(4),S34(4),S44(4),S54(4),S64(4)
      DOUBLEPRECISION S15(4),S25(4),S35(4),S45(4),S55(4),S65(4)
      DOUBLEPRECISION S16(4),S26(4),S36(4),S46(4),S56(4),S66(4)
      DOUBLEPRECISION A0C(4),A1C(4),A2C(4),A3C(4),A4C(4),A5C(4)
      DOUBLEPRECISION B6C(4),T6C(4),CP60(4),CP61(4),CP62(4)
      DOUBLEPRECISION CP63(4),CP64(4),CP65(4),CP66(4)
      DOUBLEPRECISION B0C(4),B1C(4),B2C(4),B3C(4),B4C(4),B5C(4)
      DOUBLEPRECISION T0C(4),T1C(4),T2C(4),T3C(4),T4C(4),T5C(4)
      DOUBLEPRECISION CP10(4),CP20(4),CP30(4),CP40(4),CP50(4)
      DOUBLEPRECISION CP11(4),CP21(4),CP31(4),CP41(4),CP51(4)
      DOUBLEPRECISION CP12(4),CP22(4),CP32(4),CP42(4),CP52(4)
      DOUBLEPRECISION CP13(4),CP23(4),CP33(4),CP43(4),CP53(4)
      DOUBLEPRECISION CP14(4),CP24(4),CP34(4),CP44(4),CP54(4)
      DOUBLEPRECISION CP15(4),CP25(4),CP35(4),CP45(4),CP55(4)
      DOUBLEPRECISION CP16(4),CP26(4),CP36(4),CP46(4),CP56(4)
      DOUBLEPRECISION P0D(6),P1D(6),ABT(4,3),ABT1(6),XYZ(4,3)
      DOUBLEPRECISION ALP2,BET2,TAU2,W0Z(3),W0X(3),W0Y(3),DEL,TIM
      DATA P0D,P1D/12*0.0/
      DATA R00,R10,R20,R30,R40,R50,R60,ZEROM /44*0.0/
      DATA Q0X,Q0Y,Q0Z,QA /64*0.0/
      Q0X(1,4)=1
      BIR=-1
      PI=ACOS(BIR)
      Q0Y(2,4)=1
      Q0Z(3,4)=1
      QA(1,2)=-1
      QA(2,1)=1
      DEL=0.1
      TIM=0.0
C     READ INITIAL JOINT ANGLES, ATTITUDE ANGLES OF SATELLITE
```

```
C       POSITION OF SATELLITE, POSITION OF CENTER OF MASS OF
C       EACH LINK, INERTIA MATRIX OF EACH LINK
        READ (20,1) (TA(I),I=1,6),ALP1,BET1,TAU1,X,Y,Z
1       FORMAT (3F12.4)
        READ (20,3) (R0(I),I=1,4),(R1(I),I=1,4),(R2(I),I=1,4)
        READ (20,3) (R3(I),I=1,4),(R4(I),I=1,4),(R5(I),I=1,4)
        READ (20,3) (R6(I),I=1,4),(RT(I),I=1,4)
3       FORMAT (4F12.4)
        READ (20,10) (AM(I),I=1,7)
10      FORMAT (7F12.4)
        READ (20,345)((AI0(I,J),J=1,4),I=1,4),((AI1(I,J),J=1,4),I=1,4)
        READ (20,345)((AI2(I,J),J=1,4),I=1,4),((AI3(I,J),J=1,4),I=1,4)
        READ (20,345)((AI4(I,J),J=1,4),I=1,4),((AI5(I,J),J=1,4),I=1,4)
        READ (20,345)((AI6(I,J),J=1,4),I=1,4)
345     FORMAT(4F14.6)
76      IF(TIM.GT.40.)STOP
        X1=X
        Y1=Y
        Z1=Z
        ALP2=ALP1
        BET2=BET1
        TAU2=TAU1
        DO 67 I=1,6
67      T(I)=TA(I)*PI/180
C       CALCULATE HSMOGENEOUS TRANSFORMATION MATRICES
        CALL AMAT (T,ALP2,BET2,TAU2)
        CALL MMUL (RPY,A1,T1)
        CALL MMUL (T1,A2,T2)
        CALL MMUL (T2,A3,T3)
        CALL MMUL (T3,A4,T4)
        CALL MMUL (T4,A5,T5)
        CALL MMUL (T5,A6,T6)
C       CALCULATE POSITION OF CENTER OF MASS OF EACH LINK
C       WRT INERTIAL FRAME. PN=POSITION OF END EFFECTOR
        CALL MVMUL (RPY,R0,R00)
        CALL MVMUL (T1,R1,R10)
        CALL MVMUL (T2,R2,R20)
        CALL MVMUL (T3,R3,R30)
        CALL MVMUL (T4,R4,R40)
        CALL MVMUL (T5,R5,R50)
        CALL MVMUL (T6,R6,R60)
        CALL MVMUL (T6,RT,PN)
```

```
      WRITE (17,46) TAU1,BET1,ALP1
      WRITE (18,46) (RPY(I,4),I=1,3)
      WRITE (16,45)(TA(I),I=1,6)
      WRITE (19,46) (PN(I),I=1,3)
      DO 70 IJK=1,4
C     VELOCITY ALONG TRAJECTORY
      IF (TIM.LT.10.) P1D(1)=TIM
      IF (TIM.GT.30.) P1D(1)=40.-TIM
      IF (TIM.GE.10.AND.TIM.lE.30.) P1D(1)=10.
C     CALCULATE DERIVATIVES OF HOMOGENEOUS MATRICES
      CALL MMUL (QA,RPY,RPYA)
      CALL MMUL (A1,A2,A12)
      CALL MMUL (A12,A3,A13)
      CALL MMUL (A13,A4,A14)
      CALL MMUL (A14,A5,A15)
      CALL MMUL (A15,A6,A16)
      CALL MMUL (RPY,QA,T61A)
      CALL MMUL (T61A,A16,T61)
      CALL MMUL (A2,A3,A23)
      CALL MMUL (A23,A4,A24)
      CALL MMUL (A24,A5,A25)
      CALL MMUL (A25,A6,A26)
      CALL MMUL (T1,QA,T62A)
      CALL MMUL (T62A,A26,T62)
      CALL MMUL (A3,A4,A34)
      CALL MMUL (A34,A5,A35)
      CALL MMUL (A35,A6,A36)
      CALL MMUL (T2,QA,T63A)
      CALL MMUL (T63A,A36,T63)
      CALL MMUL (A4,A5,A45)
      CALL MMUL (A45,A6,A46)
      CALL MMUL (T3,QA,T64A)
      CALL MMUL (T64A,A46,T64)
      CALL MMUL (A5,A6,A56)
      CALL MMUL (T4,QA,T65A)
      CALL MMUL (T65A,A56,T65)
      CALL MMUL (T5,QA,T66A)
      CALL MMUL (T66A,A6,T66)
      CALL MMUL (Q0X,A1,T1X)
      CALL MMUL (Q0Y,A1,T1Y)
      CALL MMUL (Q0Z,A1,T1Z)
      CALL MMUL (RPYT,A1,T1TA)
```

```
CALL MMUL (RPYB,A1,T1B)
CALL MMUL (RPYA,A1,T1A)
CALL MMUL (Q0X,A12,T2X)
CALL MMUL (Q0Y,A12,T2Y)
CALL MMUL (Q0Z,A12,T2Z)
CALL MMUL (RPYB,A12,T2B)
CALL MMUL (RPYT,A12,T2TA)
CALL MMUL (RPYA,A12,T2A)
CALL MMUL (Q0X,A13,T3X)
CALL MMUL (Q0Y,A13,T3Y)
CALL MMUL (Q0Z,A13,T3Z)
CALL MMUL (RPYB,A13,T3B)
CALL MMUL (RPYT,A13,T3TA)
CALL MMUL (RPYA,A13,T3A)
CALL MMUL (Q0X,A14,T4X)
CALL MMUL (Q0Y,A14,T4Y)
CALL MMUL (Q0Z,A14,T4Z)
CALL MMUL (RPYB,A14,T4B)
CALL MMUL (RPYT,A14,T4TA)
CALL MMUL (RPYA,A14,T4A)
CALL MMUL (Q0X,A15,T5X)
CALL MMUL (Q0Y,A15,T5Y)
CALL MMUL (Q0Z,A15,T5Z)
CALL MMUL (RPYB,A15,T5B)
CALL MMUL (RPYT,A15,T5TA)
CALL MMUL (RPYA,A15,T5A)
CALL MMUL (Q0X,A16,T6X)
CALL MMUL (Q0Y,A16,T6Y)
CALL MMUL (Q0Z,A16,T6Z)
CALL MMUL (RPYB,A16,T6B)
CALL MMUL (RPYA,A16,T6A)
CALL MMUL (RPYT,A16,RPYT)
CALL MMUL (T61A,A1,T11)
CALL MMUL (T61A,A12,T21)
CALL MMUL (T62A,A2,T22)
CALL MMUL (T61A,A13,T31)
CALL MMUL (T62A,A23,T32)
CALL MMUL (T63A,A3,T33)
CALL MMUL (T61A,A14,T41)
CALL MMUL (T62A,A24,T42)
CALL MMUL (T63A,A34,T43)
CALL MMUL (T64A,A4,T44)
```

```
      CALL MMUL (T61A,A15,T51)
      CALL MMUL (T62A,A25,T52)
      CALL MMUL (T63A,A35,T53)
      CALL MMUL (T64A,A45,T54)
      CALL MMUL (T65A,A5,T55)
C     CALCULATE Q MATRICES GIVEN BY EQNS. 15-21
      CALL JAC (QX,T1X,T2X,T3X,T4X,T5X,T6X)
      CALL JAC (QY,T1Y,T2Y,T3Y,T4Y,T5Y,T6Y)
      CALL JAC (QZ,T1Z,T2Z,T3Z,T4Z,T5Z,T6Z)
      CALL JAC (QAL,T1A,T2A,T3A,T4A,T5A,T6A)
      CALL JAC (QB,T1B,T2B,T3B,T4B,T5B,T6B)
      CALL JAC (QTA,T1TA,T2TA,T3TA,T4TA,T5TA,T6TA)
      DO 653 I=1,3
      Q1(I)=0
      Q2(I)=0
      Q3(I)=0
      Q4(I)=0
      Q5(I)=0
      Q6(I)=0
      DO 653 J=1,4
      Q1(I)=Q1(I)+AM(2)*T11(I,J)*R1(J)+AM(3)*T21(I,J)*R2(J)+
     *AM(4)*T31(I,J)*R3(J)+AM(5)*T41(I,J)*R4(J)+AM(6)*T51(I,J)
     **R5(J)+AM(7)*T61(I,J)*R6(J)
      Q2(I)=Q2(I)+AM(3)*T22(I,J)*R2(J)+AM(4)*T32(I,J)*R3(J)+AM(5)*
     *T42(I,J)*R4(J)+AM(6)*T52(I,J)*R5(J)+AM(7)*T62(I,J)*R6(J)
      Q3(I)=Q3(I)+AM(4)*T33(I,J)*R3(J)+AM(5)*T43(I,J)*R4(J)+
     *AM(6)*T53(I,J)*R5(J)+AM(7)*T63(I,J)*R6(J)
      Q4(I)=Q4(I)+AM(5)*T44(I,J)*R4(J)+AM(6)*T54(I,J)*R5(J)+
     *AM(7)*T64(I,J)*R6(J)
      Q5(I)=Q5(I)+AM(6)*T55(I,J)*R5(J)+AM(7)*T65(I,J)*R6(J)
      Q6(I)=Q6(I)+AM(7)*T66(I,J)*R6(J)
653   CONTINUE
C     CALCULATE C MATRIX GIVEN BY EQN 22.
      DO 500 I=1,3
      C33(I,1)=QX(I)
      C33(I,2)=QY(I)
      C33(I,3)=QZ(I)
      C39(I,1)=QTA(I)
      C39(I,2)=QB(I)
      C39(I,3)=QAL(I)
      C39(I,4)=Q1(I)
      C39(I,5)=Q2(I)
```

```
      C39(I,6)=Q3(I)
      C39(I,7)=Q4(I)
      C39(I,8)=Q5(I)
      C39(I,9)=Q6(I)
500   CONTINUE
      C33(1,1)=C33(1,1)+AM(1)
      C33(2,2)=C33(2,2)+AM(1)
      C33(3,3)=C33(3,3)+AM(1)
      CALL DLINRG (3,C33,3,C33I,3)
      DO 501 I=1,3
      DO 501 J=1,9
      C(I,J)=0
      DO 501 K=1,3
      C(I,J)=C(I,J)-C33I(I,K)*C39(K,J)
501   CONTINUE
      DO 504 I=1,3
      C1(I)=C(I,1)
      C2(I)=C(I,2)
      C3(I)=C(I,3)
      C4(I)=C(I,4)
      C5(I)=C(I,5)
      C6(I)=C(I,6)
      C7(I)=C(I,7)
      C8(I)=C(I,8)
      C9(I)=C(I,9)
504   CONTINUE
      ALP3=ALP2*PI/180
      BET3=BET2*PI/180
C     CALCULATE W0 MATRIX GIVEN BY EQN 10
      W0X(1)=HS(4,1)
      W0X(2)=HS(5,1)
      W0X(3)=HS(6,1)
      W0Y(1)=HS(4,2)
      W0Y(2)=HS(5,2)
      W0Y(3)=HS(5,2)
      W0Z(1)=HS(4,3)
      W0Z(2)=HS(5,3)
      W0Z(3)=HS(6,3)
C     CALCULATE HS AND HM MATRICES GIVEN BY EQNS 33-36
      HS(4,1)=COS(BET3)*COS(ALP3)
      HS(4,2)=-SIN(ALP3)
      HS(4,3)=0
```

```
      HS(5,1)=SIN(ALP3)*COS(BET3)
      HS(5,2)=COS(ALP3)
      HS(5,3)=0
      HS(6,1)=-SIN(BET3)
      HS(6,2)=0
      HS(6,3)=1
      DO 600 I=1,3
      HS(I,1)=0
      HS(I,2)=0
      HS(I,3)=0
      HM(I,1)=0
      HM(I,2)=0
      HM(I,3)=0
      HM(I,4)=0
      HM(I,5)=0
      HM(I,6)=0
      DO 601 J=1,4
      HS(I,1)=HS(I,1)+(T6TA(I,J)+T6X(I,J)*C(1,1)+T6Y(I,J)*
     *C(2,1)+T6Z(I,J)*C(3,1))*RT(J)
      HS(I,2)=HS(I,2)+(T6B(I,J)+T6X(I,J)*C(1,2)+T6Y(I,J)*
     *C(2,2)+T6Z(I,J)*C(3,2))*RT(J)
      HS(I,3)=HS(I,3)+(T6A(I,J)+T6X(I,J)*C(1,3)+T6Y(I,J)*
     *C(2,3)+T6Z(I,J)*C(3,3))*RT(J)
      HM(I,1)=HM(I,1)+(T61(I,J)+T6X(I,J)*C(1,4)+T6Y(I,J)*
     *C(2,4)+T6Z(I,J)*C(3,4))*RT(J)
      HM(I,2)=HM(I,2)+(T62(I,J)+T6X(I,J)*C(1,5)+T6Y(I,J)*
     *C(2,5)+T6Z(I,J)*C(3,5))*RT(J)
      HM(I,3)=HM(I,3)+(T63(I,J)+T6X(I,J)*C(1,6)+T6Y(I,J)*
     *C(2,6)+T6Z(I,J)*C(3,6))*RT(J)
      HM(I,4)=HM(I,4)+(T64(I,J)+T6X(I,J)*C(1,7)+T6Y(I,J)*
     *C(2,7)+T6Z(I,J)*C(3,7))*RT(J)
      HM(I,5)=HM(I,5)+(T65(I,J)+T6X(I,J)*C(1,8)+T6Y(I,J)*
     *C(2,8)+T6Z(I,J)*C(3,8))*RT(J)
      HM(I,6)=HM(I,6)+(T66(I,J)+T6X(I,J)*C(1,9)+T6Y(I,J)*
     *C(2,9)+T6Z(I,J)*C(3,9))*RT(J)
601   CONTINUE
      HM(I+3,1)=T1(I,3)
      HM(I+3,2)=T2(I,3)
      HM(I+3,3)=T3(I,3)
      HM(I+3,4)=T4(I,3)
      HM(I+3,5)=T5(I,3)
      HM(I+3,6)=T6(I,3)
```

```
600   CONTINUE
C     CALCULATE INERTIA MATRICES IN FIRST TERMS OF EQNS 41-44
      CALL INERT2 (RPY,AI0,AI00)
      CALL INERT2 (T1,AI1,AI10)
      CALL INERT2 (T2,AI2,AI20)
      CALL INERT2 (T3,AI3,AI30)
      CALL INERT2 (T4,AI4,AI40)
      CALL INERT2 (T5,AI5,AI50)
      CALL INERT2 (T6,AI6,AI60)
      DO 506 I=1,4
      DO 506 J=1,4
      AI50(I,J)=AI50(I,J)+AI60(I,J)
      AI40(I,J)=AI40(I,J)+AI50(I,J)
      AI30(I,J)=AI30(I,J)+AI40(I,J)
      AI20(I,J)=AI20(I,J)+AI30(I,J)
      AI10(I,J)=AI10(I,J)+AI20(I,J)
      AI00(I,J)=AI00(I,J)+AI10(I,J)
506   CONTINUE
C     CALCULATE THIRD TERMS IN EQNS 41-44
      CALL SC (T1A,C(1,3),T1X,C(2,3),T1Y,C(3,3),T1Z,R1,S1A)
      CALL SC(T2A,C(1,3),T2X,C(2,3),T2Y,C(3,3),T2Z,R2,S2A)
      CALL SC(T3A,C(1,3),T3X,C(2,3),T3Y,C(3,3),T3Z,R3,S3A)
      CALL SC(T4A,C(1,3),T4X,C(2,3),T4Y,C(3,3),T4Z,R4,S4A)
      CALL SC(T5A,C(1,3),T5X,C(2,3),T5Y,C(3,3),T5Z,R5,S5A)
      CALL SC(T6A,C(1,3),T6X,C(2,3),T6Y,C(3,3),T6Z,R6,S6A)
      CALL SC (T1B,C(1,2),T1X,C(2,2),T1Y,C(3,2),T1Z,R1,S1B)
      CALL SC(T2B,C(1,2),T2X,C(2,2),T2Y,C(3,2),T2Z,R2,S2B)
      CALL SC(T3B,C(1,2),T3X,C(2,2),T3Y,C(3,2),T3Z,R3,S3B)
      CALL SC(T4B,C(1,2),T4X,C(2,2),T4Y,C(3,2),T4Z,R4,S4B)
      CALL SC(T5B,C(1,2),T5X,C(2,2),T5Y,C(3,2),T5Z,R5,S5B)
      CALL SC(T6B,C(1,2),T6X,C(2,2),T6Y,C(3,2),T6Z,R6,S6B)
      CALL SC (T1TA,C(1,1),T1X,C(2,1),T1Y,C(3,1),T1Z,R1,S1A)
      CALL SC(T2TA,C(1,1),T2X,C(2,1),T2Y,C(3,1),T2Z,R2,S2A)
      CALL SC(T3TA,C(1,1),T3X,C(2,1),T3Y,C(3,1),T3Z,R3,S3A)
      CALL SC(T4TA,C(1,1),T4X,C(2,1),T4Y,C(3,1),T4Z,R4,S4A)
      CALL SC(T5TA,C(1,1),T5X,C(2,1),T5Y,C(3,1),T5Z,R5,S5A)
      CALL SC(T6TA,C(1,1),T6X,C(2,1),T6Y,C(3,1),T6Z,R6,S6A)
      CALL SC (T11,C(1,4),T1X,C(2,4),T1Y,C(3,4),T1Z,R1,S11)
      CALL SC(T21,C(1,4),T2X,C(2,4),T2Y,C(3,4),T2Z,R2,S21)
      CALL SC(T31,C(1,4),T3X,C(2,4),T3Y,C(3,4),T3Z,R3,S31)
      CALL SC(T41,C(1,4),T4X,C(2,4),T4Y,C(3,4),T4Z,R4,S41)
      CALL SC(T51,C(1,4),T5X,C(2,4),T5Y,C(3,4);T5Z,R5,S51)
```

```
      CALL SC(T61,C(1,4),T6X,C(2,4),T6Y,C(3,4),T6Z,R6,S61)
      CALL SC(ZEROM,C(1,5),T1X,C(2,5),T1Y,C(3,5),T1Z,R1,S12)
      CALL SC(T22,C(1,5),T2X,C(2,5),T2Y,C(3,5),T2Z,R2,S22)
      CALL SC(T32,C(1,5),T3X,C(2,5),T3Y,C(3,5),T3Z,R3,S32)
      CALL SC(T42,C(1,5),T4X,C(2,5),T4Y,C(3,5),T4Z,R4,S42)
      CALL SC(T52,C(1,5),T5X,C(2,5),T5Y,C(3,5),T5Z,R5,S52)
      CALL SC(T62,C(1,5),T6X,C(2,5),T6Y,C(3,5),T6Z,R6,S62)
      CALL SC(ZEROM,C(1,6),T1X,C(2,6),T1Y,C(3,6),T1Z,R1,S13)
      CALL SC(ZEROM,C(1,6),T2X,C(2,6),T2Y,C(3,6),T2Z,R2,S23)
      CALL SC(T33,C(1,6),T3X,C(2,6),T3Y,C(3,6),T3Z,R3,S33)
      CALL SC(T43,C(1,6),T4X,C(2,6),T4Y,C(3,6),T4Z,R4,S43)
      CALL SC(T53,C(1,6),T5X,C(2,6),T5Y,C(3,6),T5Z,R5,S53)
      CALL SC(T63,C(1,6),T6X,C(2,6),T6Y,C(3,6),T6Z,R6,S63)
      CALL SC(ZEROM,C(1,7),T1X,C(2,7),T1Y,C(4,7),T1Z,R1,S14)
      CALL SC(ZEROM,C(1,7),T2X,C(2,7),T2Y,C(4,7),T2Z,R2,S24)
      CALL SC(ZEROM,C(1,7),T3X,C(2,7),T3Y,C(4,7),T3Z,R4,S34)
      CALL SC(T44,C(1,7),T4X,C(2,7),T4Y,C(4,7),T4Z,R4,S44)
      CALL SC(T54,C(1,7),T5X,C(2,7),T5Y,C(4,7),T5Z,R5,S54)
      CALL SC(T64,C(1,7),T6X,C(2,7),T6Y,C(4,7),T6Z,R6,S64)
      CALL SC(ZEROM,C(1,8),T1X,C(2,8),T1Y,C(4,8),T1Z,R1,S15)
      CALL SC(ZEROM,C(1,8),T2X,C(2,8),T2Y,C(4,8),T2Z,R2,S25)
      CALL SC(ZEROM,C(1,8),T3X,C(2,8),T3Y,C(4,8),T3Z,R4,S35)
      CALL SC(ZEROM,C(1,8),T4X,C(2,8),T4Y,C(4,8),T4Z,R4,S45)
      CALL SC(T55,C(1,8),T5X,C(2,8),T5Y,C(4,8),T5Z,R5,S55)
      CALL SC(T65,C(1,8),T6X,C(2,8),T6Y,C(4,8),T6Z,R6,S65)
      CALL SC(ZEROM,C(1,9),T1X,C(2,9),T1Y,C(4,9),T1Z,R1,S16)
      CALL SC(ZEROM,C(1,9),T2X,C(2,9),T2Y,C(4,9),T2Z,R2,S26)
      CALL SC(ZEROM,C(1,9),T3X,C(2,9),T3Y,C(4,9),T3Z,R4,S36)
      CALL SC(ZEROM,C(1,9),T4X,C(2,9),T4Y,C(4,9),T4Z,R4,S46)
      CALL SC(ZEROM,C(1,9),T5X,C(2,9),T5Y,C(4,9),T5Z,R5,S56)
      CALL SC(T66,C(1,9),T6X,C(2,9),T6Y,C(4,9),T6Z,R6,S66)
C     CALCULATE CROSS PRODUCT TERMS IN EQNS 41-44.
      CALL CROPRO (R00,C1,T0C)
      CALL CROPRO (R10,S1A,A1C)
      CALL CROPRO (R20,S2A,A2C)
      CALL CROPRO (R30,S3A,A3C)
      CALL CROPRO (R40,S4A,A4C)
      CALL CROPRO (R50,S5A,A5C)
      CALL CROPRO (R60,S6A,A6C)
      CALL CROPRO (R00,C2,B0C)
      CALL CROPRO (R10,S1B,B1C)
      CALL CROPRO (R20,S2B,B2C)
```

```
CALL CROPRO (R30,S3B,B3C)
CALL CROPRO (R40,S4B,B4C)
CALL CROPRO (R50,S5B,B5C)
CALL CROPRO (R60,S6B,B6C)
CALL CROPRO (R00,C3,A0C)
CALL CROPRO (R10,S1T,T1C)
CALL CROPRO (R20,S2T,T2C)
CALL CROPRO (R30,S3T,T3C)
CALL CROPRO (R40,S4T,T4C)
CALL CROPRO (R50,S5T,T5C)
CALL CROPRO (R60,S6T,T6C)
CALL CROPRO (R00,C4,CP10)
CALL CROPRO (R00,C5,CP20)
CALL CROPRO (R00,C6,CP30)
CALL CROPRO (R00,C7,CP40)
CALL CROPRO (R00,C8,CP50)
CALL CROPRO (R00,C9,CP60)
CALL CROPRO (R10,S11,CP11)
CALL CROPRO (R20,S21,CP21)
CALL CROPRO (R30,S31,CP31)
CALL CROPRO (R40,S41,CP41)
CALL CROPRO (R50,S51,CP51)
CALL CROPRO (R60,S61,CP61)
CALL CROPRO (R10,S12,CP12)
CALL CROPRO (R20,S22,CP22)
CALL CROPRO (R30,S32,CP32)
CALL CROPRO (R40,S42,CP42)
CALL CROPRO (R50,S52,CP52)
CALL CROPRO (R60,S62,CP62)
CALL CROPRO (R10,S13,CP13)
CALL CROPRO (R20,S23,CP23)
CALL CROPRO (R30,S33,CP33)
CALL CROPRO (R40,S43,CP43)
CALL CROPRO (R50,S53,CP53)
CALL CROPRO (R60,S63,CP63)
CALL CROPRO (R10,S14,CP14)
CALL CROPRO (R20,S24,CP24)
CALL CROPRO (R30,S34,CP34)
CALL CROPRO (R40,S44,CP44)
CALL CROPRO (R50,S54,CP54)
CALL CROPRO (R60,S64,CP64)
CALL CROPRO (R10,S15,CP15)
```

```
      CALL CROPRO (R20,S25,CP25)
      CALL CROPRO (R30,S35,CP35)
      CALL CROPRO (R40,S45,CP45)
      CALL CROPRO (R50,S55,CP55)
      CALL CROPRO (R60,S65,CP65)
      CALL CROPRO (R10,S16,CP16)
      CALL CROPRO (R20,S26,CP26)
      CALL CROPRO (R30,S36,CP36)
      CALL CROPRO (R40,S46,CP46)
      CALL CROPRO (R50,S56,CP56)
      CALL CROPRO (R60,S66,CP66)
C     CALCULATE EQNS 41-44
      DO 583 I=1,3
      AI0A(I)=0
      AI0B(I)=0
      AI0T(I)=0
      AI1K(I)=0
      AI2K(I)=0
      AI3K(I)=0
      AI4K(I)=0
      AI5K(I)=0
      AI6K(I)=0
      DO 583 J=1,3
      AI0T(I)=AI0T(I)+AI00(I,J)*HS(J+3,1)
      AI0B(I)=AI0B(I)+AI00(I,J)*HS(J+3,2)
      AI0A(I)=AI0A(I)+AI00(I,J)*HS(J+3,3)
      AI1K(I)=AI1K(I)+AI10(I,J)*T1(J,3)
      AI2K(I)=AI2K(I)+AI20(I,J)*T2(J,3)
      AI3K(I)=AI3K(I)+AI30(I,J)*T3(J,3)
      AI4K(I)=AI4K(I)+AI40(I,J)*T4(J,3)
      AI5K(I)=AI5K(I)+AI50(I,J)*T5(J,3)
      AI6K(I)=AI6K(I)+AI60(I,J)*T6(J,3)
583   CONTINUE
C     CALCULATE IS AND IM MATRICES GIVEN BY EQN 46
      DO 510 I=1,3
      AIS(I,3)=AI0A(I)+AM(1)*A0C(I)+AM(2)*A1C(I)+AM(3)*A2C(I)+
      *AM(4)*A3C(I)+AM(5)*A4C(I)+AM(6)*A5C(I)+AM(7)*A6C(I)
      AIS(I,2)=AI0B(I)+AM(1)*B0C(I)+AM(2)*B1C(I)+AM(3)*B2C(I)+
      *AM(4)*B3C(I)+AM(5)*B4C(I)+AM(6)*B5C(I)+AM(7)*B6C(I)
      AIS(I,1)=AI0T(I)+AM(1)*T0C(I)+AM(2)*T1C(I)+AM(3)*T2C(I)+
      *AM(4)*T3C(I)+AM(5)*T4C(I)+AM(6)*T5C(I)+AM(7)*T6C(I)
      AIM(I,1)=AI1K(I)+AM(1)*CP10(I)+AM(2)*CP11(I)+AM(3)*CP21(I)+
```

```
     *AM(4)*CP31(I)+AM(5)*CP41(I)+AM(6)*CP51(I)+AM(7)*CP61(I)
      AIM(I,2)=AI2K(I)+AM(1)*CP20(I)+AM(2)*CP12(I)+AM(3)*CP22(I)+
     *AM(4)*CP32(I)+AM(5)*CP42(I)+AM(6)*CP52(I)+AM(7)*CP62(I)
      AIM(I,3)=AI3K(I)+AM(1)*CP30(I)+AM(2)*CP13(I)+AM(3)*CP23(I)+
     *AM(4)*CP33(I)+AM(5)*CP43(I)+AM(6)*CP53(I)+AM(7)*CP63(I)
      AIM(I,4)=AI4K(I)+AM(1)*CP40(I)+AM(2)*CP14(I)+AM(3)*CP24(I)+
     *AM(4)*CP34(I)+AM(5)*CP44(I)+AM(6)*CP54(I)+AM(7)*CP64(I)
      AIM(I,5)=AI5K(I)+AM(1)*CP50(I)+AM(2)*CP15(I)+AM(3)*CP25(I)+
     *AM(4)*CP35(I)+AM(5)*CP45(I)+AM(6)*CP55(I)+AM(7)*CP65(I)
      AIM(I,6)=AI6K(I)+AM(1)*CP60(I)+AM(2)*CP16(I)+AM(3)*CP26(I)+
     *AM(4)*CP36(I)+AM(5)*CP46(I)+AM(6)*CP56(I)+AM(7)*CP66(I)
510   CONTINUE
C     CALCULATE GENERALIZED INERTIA MATRIX GIVEN BY EQNS
C     47 AND 48. AJ=GENERALIZED INERTIA MATRIX
C     DJ=INVERSE OF GENERALIZED INERTIA MATRIX
      CALL DLINRG(3,AIS,3,AISI,3)
      DO 30 I=1,3
      DO 30 J=1,6
      AISM(I,J)=0
      DO 30 K=1,3
      AISM(I,J)=AISM(I,J)+AISI(I,K)*AIM(K,J)
30    CONTINUE
      DO 32 I=1,6
      DO 32 J=1,6
      HSIM(I,J)=0
      DO 31 K=1,3
      HSIM(I,J)=HSIM(I,J)+HS(I,K)*AISM(K,J)
31    CONTINUE
      AJ(I,J)=HM(I,J)-HSIM(I,J)
32    CONTINUE
C     DLINRG IS A SUBROUTINE IN IMSL LIBRARY WHICH SOLVES
C     SYSTEMS OF LINEAR EQUATIONS
      CALL DLINRG(6,AJ,6,DJ,6)
C     USE RUNGE-KUTTA METHOD TO INTEGRATE EQN 49
      DO 61 I=1,6
      TD(IJK,I)=0
      DO 61 J=1,6
      TD(IJK,I)=DJ(I,J)*(P1D(J)-P0D(J))*DEL+TD(IJK,I)
61    CONTINUE
      DO 62 I=1,6
      ABT1(I)=0.
      DO 62 J=1,6
```

```
62    ABT1(I)=DJ(I,J)*(P1D(J)-P0D(J))*DEL+ABT1(I)
      DO 63 I=1,3
      ABT(IJK,I)=0.
      DO 63 J=1,6
63    ABT(IJK,I)=ABT(IJK,I)-AISM(I,J)*ABT1(J)
      DO 64 I=1,3
      XYZ(IJK,I)=0.
      DO 64 J=1,3
64    XYZ(IJK,I)=C(I,J)*ABT(IJK,J)+XYZ(IJK,I)
      DO 65 I=1,3
      DO 65 J=4,9
65    XYZ(IJK,I)=XYZ(IJK,I)+C(I,J)*tD(IJK,(J-3))
      IF (IJK.EQ.4) GOtO 72
      IF (IJK.EQ.2) GOtO 75
      IF (IJK.EQ.3) GOtO 73
      TIM=TIM+DEL/2
75    DO 60 I=1,6
60    TF(I)=TA(I)+TD(IJK,I)/2
      X=X1+XYZ(IJK,1)/2
      Y=Y1+XYZ(IJK,2)/2
      Z=Z1+XYZ(IJK,3)/2
      ALP2=ALP1+ABT(IJK,3)/2
      BET2=BET1+ABT(IJK,2)/2
      TAU2=TAU1+ABT(IJK,1)/2
      GOtO 74
73    DO 80 I=1,6
80    TF(I)=TA(I)+TD(IJK,I)
      X=X1+XYZ(IJK,1)
      Y=Y1+XYZ(IJK,2)
      Z=Z1+XYZ(IJK,3)
      ALP2=ALP1+ABT(IJK,3)
      BET2=BET1+ABT(IJK,2)
      TAU2=TAU1+ABT(IJK,1)
      TIM=TIM+DEL/2
74    DO 68 I=1,6
68    TF(I)=TF(I)*PI/180
      CALL AMAT (TF,ALP2,BET2,TAU2)
      CALL MMUL (RPY,A1,T1)
      CALL MMUL (T1,A2,T2)
      CALL MMUL (T2,A3,T3)
      CALL MMUL (T3,A4,T4)
      CALL MMUL (T4,A5,T5)
```

```
      CALL MMUL (T5,A6,T6)
      CALL MVMUL (RPY,R0,R00)
      CALL MVMUL (T1,R1,R10)
      CALL MVMUL (T2,R2,R20)
      CALL MVMUL (T3,R3,R30)
      CALL MVMUL (T4,R4,R40)
      CALL MVMUL (T5,R5,R50)
      CALL MVMUL (T6,R6,R60)
      CALL MVMUL (T6,RT,PN)
70    CONTINUE
72    DO 71 I=1,6
71    TA(I)=TA(I)+(TD(1,I)+TD(2,I)*2+TD(3,I)*2+TD(4,I))/6
      X=X1+(XYZ(1,1)+XYZ(2,1)*2+XYZ(3,1)*2+XYZ(4,1))/6
      Y=Y1+(XYZ(1,2)+XYZ(2,2)*2+XYZ(3,2)*2+XYZ(4,2))/6
      Z=Z1+(XYZ(1,3)+XYZ(2,3)*2+XYZ(3,3)*2+XYZ(4,3))/6
      ALP1=ALP1+(ABT(1,3)+ABT(2,3)*2+ABT(3,3)*2+ABT(4,3))/6
      BET1=BET1+(ABT(1,2)+ABT(2,2)*2+ABT(3,2)*2+ABT(4,2))/6
      TAU1=TAU1+(ABT(1,1)+ABT(2,1)*2+ABT(3,1)*2+ABT(4,1))/6
      GOtO 76
45    FORMAT(6(2X,F14.7))
46    FORMAT(3(2X,F14.7))
      END
      SUBROUTINE JAC(QR0,TX1,TX2,TX3,TX4,TX5,TX6)
      DOUBLEPRECISION AM(7),R0(4),R1(4),R2(4),R3(4),R4(4),R5(4),Z
      DOUBLEPRECISION RPY(4,4),A1(4,4),A2(4,4),A3(4,4),A4(4,4)
      DOUBLEPRECISION QR0(4),TX1(4,4),TX2(4,4),TX3(4,4),TX4(4,4)
      DOUBLEPRECISION TX5(4,4),TX6(4,4),A5(4,4),A6(4,4),R6(4),X,Y
      COMMON AM,R0,R1,R2,R3,R4,R5,R6,X,Y,Z,RPY,A1,A2,A3,A4,A5,A6
      DO 14 I=1,4
      QR0(I)=0
      DO 15 J=1,4
      QR0(I)=QR0(I)+AM(2)*TX1(I,J)*R1(J)+
      *AM(3)*TX2(I,J)*R2(J)+AM(4)*TX3(I,J)*R3(J)+AM(5)*TX4(I,J)*R4(J)+
      *AM(6)*TX5(I,J)*R5(J)+AM(7)*TX6(I,J)*R6(J)
15    CONTINUE
14    CONTINUE
      RETURN
      END
      SUBROUTINE MATTR4(A,D)
      DOUBLEPRECISION A(4,4),D(4,4)
      DO 7 I=1,4
      DO 8 J=1,4
```

```
D(I,J)=A(J,I)
CONTINUE
CONTINUE
RETURN
END
SUBROUTINE AMAT (T,ALP1,BET1,TAU1)
DOUBLEPRECISION AM(7),R0(4),R1(4),R2(4),R3(4),R4(4),R5(4),Z
DOUBLEPRECISION RPY(4,4),A1(4,4),A2(4,4),A3(4,4),A4(4,4)
DOUBLEPRECISION RPYB(4,4),RPYT(4,4),A5(4,4),A6(4,4),R6(4),X,Y
COMMON AM,R0,R1,R2,R3,R4,R5,R6,X,Y,Z,RPY,A1,A2,A3,A4,A5,A6
COMMON RPYB,RPYT,PI
DOUBLEPRECISION T(6),ALP,BET,TAU,ALP1,BET1,TAU1,PI
DATA A1,A2,A3,A4,A5,A6 /96*0.0/
ALP=ALP1*PI/180
BET=BET1*PI/180
TAU=TAU1*PI/180
A1(1,1)=COS(T(1))
A1(1,3)=-SIN(T(1))
A1(2,1)=SIN(T(1))
A1(2,3)=COS(T(1))
A1(3,2)=-1
A1(3,4)=39.02
A1(4,4)=1
A2(1,1)=COS(T(2))
A2(1,2)=-SIN(T(2))
A2(2,1)=SIN(T(2))
A2(2,2)=COS(T(2))
A2(3,3)=1
A2(4,4)=1
A2(1,4)=45*COS(T(2))
A2(2,4)=45*SIN(T(2))
A2(3,4)=25.0
A3(1,1)=COS(T(3))
A3(1,3)=SIN(T(3))
A3(1,4)=-2.0*COS(T(3))
A3(2,1)=SIN(T(3))
A3(2,3)=-COS(T(3))
A3(2,4)=-2.0*SIN(T(3))
A3(3,2)=1
A3(4,4)=1
A4(1,1)=COS(T(4))
A4(1,3)=-SIN(T(4))
```

```
A4(2,1)=SIN(T(4))
A4(3,4)=45.0
A4(2,3)=COS(T(4))
A4(3,2)=-1
A4(4,4)=1
A5(1,1)=COS(T(5))
A5(1,3)=SIN(T(5))
A5(2,1)=SIN(T(5))
A5(2,3)=-COS(T(5))
A5(3,2)=1
A5(4,4)=1
A6(1,1)=COS(T(6))
A6(1,2)=-SIN(T(6))
A6(2,1)=SIN(T(6))
A6(2,2)=COS(T(6))
A6(3,3)=1
A6(3,4)=5.625
A6(4,4)=1
RPY(1,1)=COS(ALP)*COS(BET)
RPY(1,2)=COS(ALP)*SIN(BET)*SIN(TAU)-COS(TAU)*SIN(ALP)
RPY(1,3)=COS(ALP)*SIN(BET)*COS(TAU)+SIN(ALP)*SIN(TAU)
RPY(2,1)=SIN(ALP)*COS(BET)
RPY(2,2)=SIN(ALP)*SIN(BET)*SIN(TAU)+COS(ALP)*COS(TAU)
RPY(2,3)=SIN(ALP)*SIN(BET)*COS(TAU)-SIN(TAU)*COS(ALP)
RPY(3,1)=-SIN(BET)
RPY(3,2)=COS(BET)*SIN(TAU)
RPY(3,3)=COS(BET)*COS(TAU)
RPY(1,4)=X
RPY(2,4)=Y
RPY(3,4)=Z
RPY(4,4)=1
RPYB(1,1)=-COS(ALP)*SIN(BET)
RPYB(1,2)=COS(ALP)*COS(BET)*SIN(TAU)
RPYB(1,3)=COS(ALP)*COS(BET)*COS(TAU)
RPYB(2,1)=-SIN(ALP)*SIN(BET)
RPYB(2,2)=SIN(ALP)*COS(BET)*SIN(TAU)
RPYB(2,3)=SIN(ALP)*COS(BET)*COS(TAU)
RPYB(3,1)=-COS(BET)
RPYB(3,2)=-SIN(BET)*SIN(TAU)
RPYB(3,3)=-SIN(BET)*COS(TAU)
RPYT(1,2)=COS(ALP)*SIN(BET)*COS(TAU)+SIN(TAU)*SIN(ALP)
RPYT(1,3)=-COS(ALP)*SIN(BET)*SIN(TAU)+SIN(ALP)*COS(TAU)
```

```
      RPYT(2,2)=SIN(ALP)*SIN(BET)*COS(TAU)-COS(ALP)*SIN(TAU)
      RPYT(2,3)=-SIN(ALP)*SIN(BET)*SIN(TAU)-COS(TAU)*COS(ALP)
      RPYT(3,2)=COS(BET)*COS(TAU)
      RPYT(3,3)=-COS(BET)*SIN(TAU)
      RETURN
      END
      SUBROUTINE CROPRO (RC,S,CP)
      DOUBLEPRECISION RC(4),S(4),CP(4)
      CP(1)=RC(2)*S(3)-S(2)*RC(3)
      CP(2)=RC(3)*S(1)-RC(1)*S(3)
      CP(3)=RC(1)*S(2)-RC(2)*S(1)
      RETURN
      END
      SUBROUTINE INERT2(A,B,C)
      DOUBLEPRECISION A(4,4),B(4,4),C(4,4),D(4,4),E(4,4)
      CALL MATTR4(A,D)
      CALL MMUL(B,D,E)
      CALL MMUL(A,E,C)
      RETURN
      END
      SUBROUTINE MMUL (B,A,F)
      DOUBLEPRECISION A(4,4),B(4,4),F(4,4)
      DO 1 I=1,4
      DO 1 J=1,4
      F(I,J)=0
      DO 1 K=1,4
1     F(I,J)=F(I,J)+B(I,K)*A(K,J)
      RETURN
      END
      SUBROUTINE MMUL3 (B,A,C)
      DOUBLEPRECISION A(4,4),B(4,4),C(4,4)
      DO 1 I=1,3
      DO 1 J=1,3
      C(I,J)=0
      DO 1 K=1,3
1     C(I,J)=C(I,J)+B(I,K)*A(K,J)
      RETURN
      END
      SUBROUTINE MVMUL (A,B,C)
      DOUBLEPRECISION A(4,4),B(4),C(4)
      DO 1 I=1,4
      C(I)=0.
```

```
      DO 1 J=1,4
1     C(I)=C(I)+A(I,J)*B(J)
      RETURN
      END
      SUBROUTINE SC (A,B,C,D,E,F,G,R,S)
      DOUBLEPRECISION A(4,4),B,C(4,4),D,E(4,4),F,G(4,4)
      DOUBLEPRECISION R(4),S(4)
      DO 1 I=1,3
      S(I)=0
      DO 1 J=1,4
1     S(I)=S(I)+(A(I,J)+B*C(I,J)+D*E(I,J)+F*G(I,J))*R(J)
      RETURN
      END
```

APPENDIX B

FINE ATTITUDE CONTROL PROGRAM

```
C       MAIN PROGRAM
        DOUBLEPRECISION AM(4),R1(4),R2(4),R3(4)
        DOUBLEPRECISION A1(4,4),A2(4,4),A3(4,4)
        DOUBLEPRECISION W0(3,3),DEL,T1D(3),R0(4)
        COMMON AM,R0,R1,R2,R3,A1,A2,A3,PI,W0
        COMMON AI0,AI1,AI2,AI3,TM,QA,DEL,T1D
        DOUBLEPRECISION T(3),AI0(4,4),TM
        DOUBLEPRECISION ALP1,BET1,TAU1,BIR,PI,QA(4,4)
        DOUBLEPRECISION AI1(4,4),AI2(4,4),AI3(4,4),TAU6
        DOUBLEPRECISION F0,FF,A,TF,BETD,ALPD,TAUD,BET6,ALP6
        DOUBLEPRECISION BETI,ALPI,TAUI,ALPF,BETF,TAUF
        DOUBLEPRECISION CHA(4,2),BETD1,ALPD1,TAUD1,TEMP2(4)
        DOUBLEPRECISION BETD2,ALPD2,TAUD2,ERRO,ERRO1,TEMP1(4)
        DOUBLEPRECISION ALP7,BET7,TAU7,ALP8,BET8,TAU8,ALP9
        DOUBLEPRECISION TAU9, TZ1(6),TZ2(6),TZ3(6),TZ4(6),BET9
        INTEGER ICHO(6,2)
        DATA ICHO /1,2,1,3,2,3,2,1,3,1,3,2/
        DATA TZ1,TZ2,TZ3,TZ4 /24*0.0/
        DATA CHA /10.0,10.0,-10.0,-10.0,10.0,-10.0,10.0,-10.0/
        TF=1.0
        DATA QA /16*0.0/
        BIR=-1
        PI=ACOS(BIR)
        QA(1,2)=-1
        QA(2,1)=1
        DEL=0.1
        READ (10,1) ALPI,BETI,TAUI,ALPF,BETF,TAUF,(T(I),I=1,3)
1       FORMAT (3F12.4)
        READ (10,3) (R0(I),I=1,4)
        READ (10,3) (R1(I),I=1,4),(R2(I),I=1,4),(R3(I),I=1,4)
3       FORMAT (4F12.4)
271     FORMAT (2X,F4.2)
        WRITE (26,48) TAUI,BETI,ALPI,TAUF,BETF,ALPF,(T(I),I=1,3)
48      FORMAT (3(2X,F15.7))
        WRITE (30,46) TAUI,BETI,ALPI,(T(I),I=1,3),(TZ1(JK),JK=4,6)
        READ (10,345) (AM(I),I=1,4)
```

```
      READ (10,345)((AI0(I,J),J=1,4),I=1,4),((AI1(I,J),J=1,4),I=1,4)
      READ (10,345)((AI2(I,J),J=1,4),I=1,4),((AI3(I,J),J=1,4),I=1,4)
345   FORMAT(4F14.6)
      ERRO1=(TAUF-TAUI)**2+(BETF-BETI)**2+(ALPF-ALPI)**2
      TM=0.0
      DO 433 I=1,4
433   TM=TM+AM(I)
      IJK=0
925   DO 926 JKT=1,4
      TEMP1(JKT)=cHA(JKT,1)
926   TEMP2(JKT)=cHA(JKT,2)
7     DO 2 I=1,6
      IK=ICHO(I,1)
      IL=ICHO(I,2)
      DO 4 J=1,4
      F0=T(IK)
      FF=T(IK)+CHA(J,1)
      ALP1=ALPI
      BET1=BETI
      TAU1=TAUI
      CALL INTEG (F0,FF,TF,IK,T,ALP1,BET1,TAU1,1)
      F0=T(IL)
      T(IK)=FF
      FF=T(IL)+CHA(J,2)
      CALL INTEG (F0,FF,TF,IL,T,ALP1,BET1,TAU1,1)
      F0=T(IK)
      T(IL)=FF
      FF=T(IK)-CHA(J,1)
      CALL INTEG (F0,FF,TF,IK,T,ALP1,BET1,TAU1,1)
      F0=T(IL)
      T(IK)=FF
      FF=T(IL)-CHA(J,2)
      CALL INTEG (F0,FF,TF,IL,T,ALP1,BET1,TAU1,1)
      T(IL)=FF
      ALPD1=ABS(ALPF-ALP1)
      ALPD2=ABS(ALPF-ALPI)
      BETD1=ABS(BETF-BET1)
      BETD2=ABS(BETF-BETI)
      TAUD1=ABS(TAUF-TAU1)
      TAUD2=ABS(TAUF-TAUI)
      ERRO=ALPD1**2+BETD1**2+TAUD1**2
      IF (ERRO1.LT.ERRO) GOTO 4
```

```
      IF(TAUD2.LT.0.01.AND.BETD2.LT.0.01.AND.ALPD1.LT.ALPD2)
      *GOTO 20
      IF(TAUD2.LT.0.01.AND.ALPD2.LT.0.01.AND.BETD1.LT.BETD2)
      *GOTO 21
      IF(BETD2.LT.0.01.AND.ALPD2.LT.0.01.AND.TAUD1.LT.TAUD2)
      * GOTO 22
      IF(TAUD2.LT.0.01)GOTO 14
      IF(BETD2.LT.0.01)GOTO 16
      IF(ALPD2.LT.0.01)GOTO 18
      IF(ALPD1.LT.ALPD2.AND.BETD1.LT.BETD2.AND.TAUD1.
      LT.TAUD2) GOTO 5
      GOTO 4
14    IF(BETD1.LT.BETD2.AND.ALPD1.LT.ALPD2) GOTO 13
      GOTO 4
16    IF(TAUD1.LT.TAUD2.AND.ALPD1.LT.ALPD2) GOTO 17
      GOTO 4
18    IF(TAUD1.LT.TAUD2.AND.BETD1.LT.BETD2) GOTO 19
4     CONTINUE
2     CONTINUE
      JKI=JKI+1
      IF(JKI.Eq.3) GOTO 93
      DO 921 JKT=1,4
921   cHA(JKT,JKI)=cHA(JKT,JKI)/2
      IF (JKI.Eq.1) GOTO 7
      DO 922 JKT=1,4
922   cHA(JKT,1)=TEMP1(JKT)
      GOTO 7
93    DO 923 JKT=1,4
923   cHA(JKT,2)=TEMP2(JKT)
      DO 6 I=1,4
      DO 6 J=1,2
6     CHA(I,J)=CHA(I,J)/2.0
      JKI=0
      GOTO 925
5     ALPD=(ALPF-ALPI)/(ALP1-ALPI)
      BETD=(BETF-BETI)/(BET1-BETI)
      TAUD=(TAUF-TAUI)/(TAU1-TAUI)
      AMINK=ALPD
      IF (BETD.LT.AMINK) AMINK=BETD
      IF (TAUD.LT.AMINK) AMINK=TAUD
      MINK=IFIX(AMINK)+1
      GOTO 15
```

```
13      ALPD=(ALPF-ALPI)/(ALP1-ALPI)
        BETD=(BETF-BETI)/(BET1-BETI)
        AMINK=ALPD
        IF(BETD.LT.AMINK) AMINK=BETD
        MINK=IFIX(AMINK)+1
        GOTO 15
17      TAUD=(TAUF-TAUI)/(TAU1-TAUI)
        ALPD=(ALPF-ALPI)/(ALP1-ALPI)
        AMINK=ALPD
        IF(TAUD.LT.AMINK) AMINK=TAUD
        MINK=IFIX(AMINK)+1
        GOTO 15
19      TAUD=(TAUF-TAUI)/(TAU1-TAUI)
        BETD=(BETF-BETI)/(BET1-BETI)
        AMINK=TAUD
        IF(BETD.LT.AMINK) AMINK=BETD
        MINK=IFIX(AMINK)+1
        GOTO 15
20      AMINK=(ALPF-ALPI)/(ALP1-ALPI)
        MINK=IFIX(AMINK)+1
        GOTO 15
21      AMINK=(BETF-BETI)/(BET1-BETI)
        MINK=IFIX(AMINK)+1
        GOTO 15
22      AMINK=(TAUF-TAUI)/(TAU1-TAUI)
        MINK=IFIX(AMINK)+1
15      ALP1=ALPI
        BET1=BETI
        TAU1=TAUI
266     FORMAT (2X,17HNUMBER OF CYCLES=,I5)
        DO 8 I=1,MINK
        ALP6=ALP1
        BET6=BET1
        TAU6=TAU1
        F0=T(IK)
        FF=T(IK)+CHA(J,1)
        DO 267 JK=1,3
267     TZ1(JK)=T(JK)
        TZ1(IK)=TZ1(IK)+CHA(J,1)
        CALL INTEG (F0,FF,TF,IK,T,ALP1,BET1,TAU1,2)
        F0=T(IL)
        T(IK)=FF
```

```
      FF=T(IL)+CHA(J,2)
      ALP7=ALP1
      BET7=BET1
      TAU7=TAU1
      DO 268 JK=1,3
268   TZ2(JK)=TZ1(JK)
      TZ2(IL)=TZ2(IL)+CHA(J,2)
      CALL INTEG (F0,FF,TF,IL,T,ALP1,BET1,TAU1,2)
      F0=T(IK)
      T(IL)=FF
      FF=T(IK)-CHA(J,1)
      ALP8=ALP1
      BET8=BET1
      TAU8=TAU1
      DO 269 JK=1,3
269   TZ3(JK)=TZ2(JK)
      TZ3(IK)=TZ3(IK)-CHA(J,1)
      CALL INTEG (F0,FF,TF,IK,T,ALP1,BET1,TAU1,2)
      F0=T(IL)
      T(IK)=FF
      FF=T(IL)-CHA(J,2)
      ALP9=ALP1
      BET9=BET1
      TAU9=TAU1
      DO 270 JK=1,3
270   TZ4(JK)=TZ3(JK)
      TZ4(IL)=TZ4(IL)-CHA(J,2)
      CALL INTEG (F0,FF,TF,IL,T,ALP1,BET1,TAU1,2)
      T(IL)=FF
46    FORMAT (9(1X,F13.6))
      ALPD=(ALPF-ALP1)**2
      BETD=(BETF-BET1)**2
      TAUD=(TAUF-TAU1)**2
      ERRO=ALPD+BETD+TAUD
      IF (ERRO1.LT.ERRO) GOTO 77
      ERRO1=ERRO
      WRITE (30,46) TAU7,BET7,ALP7,(TZ1(JK),JK=1,6)
      WRITE (30,46) TAU8,BET8,ALP8,(TZ2(JK),JK=1,6)
      WRITE (30,46) TAU9,BET9,ALP9,(TZ3(JK),JK=1,6)
      WRITE (30,46) TAU1,BET1,ALP1,(TZ4(JK),JK=1,6)
      IF((ALPD+BETD+TAUD).LT.0.001) GOTO 94
49    FORMAT (2X,25HSUM OF THE ERROR SQUARES=,F20.7)
```

```
8       CONTINUE
        IJK=I-1
        BETI=BET1
        ALPI=ALP1
        TAUI=TAU1
        GOTO 7
77      IJK=I-1
        BETI=BET6
        TAUI=TAU6
        ALPI=ALP6
        GOTO 7
94      WRITE (27,46) TAU1,BET1,ALP1
        STOP
        END
        SUBROUTINE INTEG (F0,FF,TF,IJ,T,ALP1,BET1,TAU1,IN)
        DOUBLEPRECISION AM(4),R1(4),R2(4),R3(4)
        DOUBLEPRECISION A1(4,4),A2(4,4),A3(4,4)
        DOUBLEPRECISION W0(3,3),TIM,DEL,T1D(3),R0(4)
        COMMON AM,R0,R1,R2,R3,A1,A2,A3,PI,W0
        COMMON AI0,AI1,AI2,AI3,TM,QA,DEL,T1D
        DOUBLEPRECISION AK0,AK1,AK2,AK3,AM0,AM1,AM2,AM3,AN0,AN1
        DOUBLEPRECISION AN2,AN3,T(3),AI0(4,4),TM
        DOUBLEPRECISION ALP1,BET1,TAU1,PI,QA(4,4)
        DOUBLEPRECISION T2(3),AI1(4,4),AI2(4,4),AI3(4,4)
        DOUBLEPRECISION BET3,TAU3,ALP3,ALP4,BET4,TAU4
        DOUBLEPRECISION ALP2,BET2,TAU2,F0,FF,A,TF,FUN1,FUN2
        DATA T1D,T2 /6*0.0/
        FUN1(F0,FF,A,TF,PI)=F0+(FF-F0)*(A/TF-1/(2*PI)*
        *SIN(2*PI*A/TF))
        FUN2(F0,FF,A,TF,PI)=(FF-F0)*(1/TF*(1-COS(2*PI*A/TF)))
        TIM=0.0
        IC=TF/DEL
        DO 4 ICOUNT=1,IC
        T(IJ)=FUN1(F0,FF,TIM,TF,PI)
        T1D(IJ)=FUN2(F0,FF,TIM,TF,PI)
        CALL AMAT (T,ALP1,BET1,TAU1)
        CALL AK01 (AK0,AM0,AN0)
        ALP2=ALP1+AN0/2
        BET2=BET1+AM0/2
        TAU2=TAU1+AK0/2
        T(IJ)=FUN1(F0,FF,TIM+DEL/2,TF,PI)
        T1D(IJ)=FUN2(F0,FF,TIM+DEL/2,TF,PI)
```

```
      CALL AMAT (T,ALP2,BET2,TAU2)
      CALL AK01 (AK1,AM1,AN1)
      ALP3=ALP1+AN1/2
      BET3=BET1+AM1/2
      TAU3=TAU1+AK1/2
      CALL AMAT (T,ALP3,BET3,TAU3)
      CALL AK01 (AK2,AM2,AN2)
      ALP4=ALP1+AN2
      BET4=BET1+AM2
      TAU4=TAU1+AK2
      T(IJ)=FUN1(F0,FF,TIM+DEL,TF,PI)
      T1D(IJ)=FUN2(F0,FF,TIM+DEL,TF,PI)
      CALL AMAT(T,ALP4,BET4,TAU4)
      CALL AK01 (AK3,AM3,AN3)
      ALP1=ALP1+(AN0+2*AN1+2*AN2+AN3)/6
      BET1=BET1+(AM0+2*AM1+2*AM2+AM3)/6
      TAU1=TAU1+(AK0+2*AK1+2*AK2+AK3)/6
      TIM=TIM+DEL
4     CONTINUE
      RETURN
      END
      SUBROUTINE AK01(AK0,AM0,AN0)
      DOUBLEPRECISION AM(4),R1(4),R2(4),R3(4)
      DOUBLEPRECISION A1(4,4),A2(4,4),A3(4,4)
      DOUBLEPRECISION W0(3,3),W0C(3,3),W0CI(3,3),R0(4)
      COMMON AM,R0,R1,R2,R3,A1,A2,A3,PI,W0
      COMMON AI0,AI1,AI2,AI3,TM,QA,DEL,T1D
      DOUBLEPRECISION AK0,AM0,AN0,AKAT(3,3),DEL,T1D(3)
      DOUBLEPRECISION PI,QA(4,4)
      DOUBLEPRECISION T2(4,4),T3(4,4)
      DOUBLEPRECISION A23(4,4),Q1(4),Q2(4),Q3(4)
      DOUBLEPRECISION T11(4,4),T21(4,4),T31(4,4)
      DOUBLEPRECISION T22(4,4),T32(4,4),AI0(4,4)
      DOUBLEPRECISION AI1(4,4),AI2(4,4),AI3(4,4)
      DOUBLEPRECISION AI1K(3),AI2K(3),AI3K(3),T33(4,4)
      DOUBLEPRECISION R10(4),R20(4),R30(4)
      DOUBLEPRECISION T1R1(4),T1R2(4),T1R3(4),T3R3(4)
      DOUBLEPRECISION T2R2(4),T2R3(4),TM,RG(4)
      DOUBLEPRECISION Q11(4),Q12(4),Q13(4)
      DOUBLEPRECISION Q22(4),Q23(4),Q33(4),T11A(4,4),T22A(4,4)
      DOUBLEPRECISION AJ1(4,4),AJ2(4,4),AJ3(4,4),AK(4,4)
      DOUBLEPRECISION AI10(4,4),AI20(4,4),AI30(4,4)
```

```
      CALL MMUL (A1,A2,T2)
      CALL MMUL (T2,A3,T3)
      CALL MMUL (A2,A3,A23)
      CALL MMUL (QA,A1,T11)
      CALL MMUL (T11,A2,T21)
      CALL MMUL (T21,A3,T31)
      CALL MMUL (A1,QA,T11A)
      CALL MMUL (T11A,A2,T22)
      CALL MMUL (T11A,A23,T32)
      CALL MMUL (T2,QA,T22A)
      CALL MMUL (T22A,A3,T33)
      CALL MVMUL (A1,R1,R10)
      CALL MVMUL (T2,R2,R20)
      CALL MVMUL (T3,R3,R30)
      DO 434 I=1,3
434   RG(I)=(AM(2)*R10(I)+AM(3)*R20(I)+AM(4)*R30(I))/TM
      RG(4)=1.0
      CALL MVMUL (T11,R1,T1R1)
      CALL MVMUL (T21,R2,T1R2)
      CALL MVMUL (T31,R3,T1R3)
      CALL MVMUL (T22,R2,T2R2)
      CALL MVMUL (T32,R3,T2R3)
      CALL MVMUL (T33,R3,T3R3)
      DO 327 I=1,3
      DO 328 J=1,3
      AK(I,J)=-RG(I)*RG(J)*TM
      AJ1(I,J)=-R10(I)*R10(J)*AM(2)
      AJ2(I,J)=-R20(I)*R20(J)*AM(3)
      AJ3(I,J)=-R30(I)*R30(J)*AM(4)
328   CONTINUE
      AK(I,I)=TM*(RG(1)**2+RG(2)**2+RG(3)**2)+AK(I,I)
      AJ1(I,I)=AM(2)*(R10(1)**2+R10(2)**2+R10(3)**2)+AJ1(I,I)
      AJ2(I,I)=AM(3)*(R20(1)**2+R20(2)**2+R20(3)**2)+AJ2(I,I)
      AJ3(I,I)=AM(4)*(R30(1)**2+R30(2)**2+R30(3)**2)+AJ3(I,I)
327   CONTINUE
      CALL CROPRO (R10,RG,T1R1,Q11)
      CALL CROPRO (R20,RG,T1R2,Q12)
      CALL CROPRO (R30,RG,T1R3,Q13)
      CALL CROPRO (R20,RG,T2R2,Q22)
      CALL CROPRO (R30,RG,T2R3,Q23)
      CALL CROPRO (R30,RG,T3R3,Q33)
      DO 653 I=1,3
```

```
      Q1(I)=AM(2)*Q11(I)+AM(3)*Q12(I)+AM(4)*Q13(I)
      Q2(I)=AM(3)*Q22(I)+AM(4)*Q23(I)
      Q3(I)=AM(4)*Q33(I)
653   CONTINUE
      CALL INERT2 (A1,AI1,AI10)
      CALL INERT2 (T2,AI2,AI20)
      CALL INERT2 (T3,AI3,AI30)
      DO 506 I=1,3
      DO 507 J=1,3
      AI20(I,J)=AI20(I,J)+AI30(I,J)
      AI10(I,J)=AI10(I,J)+AI20(I,J)
507   CONTINUE
506   CONTINUE
      DO 654 I=1,3
      AI1K(I)=0
      AI2K(I)=0
      AI3K(I)=0
      DO 655 J=1,3
      AI1K(I)=AI1K(I)+AI10(I,J)*A1(J,3)
      AI2K(I)=AI2K(I)+AI20(I,J)*T2(J,3)
      AI3K(I)=AI3K(I)+AI30(I,J)*T3(J,3)
655   CONTINUE
      Q1(I)=AI1K(I)+Q1(I)
      Q2(I)=AI2K(I)+Q2(I)
      Q3(I)=AI3K(I)+Q3(I)
654   CONTINUE
      DO 656 I=1,3
      DO 656 J=1,3
      W0C(I,J)=0
      DO 657 K=1,3
      W0C(I,J)=W0C(I,J)+(AI0(I,K)+AI10(I,K)+AJ1(I,K)+AJ2(I,K)+
      *AJ3(I,K)-AK(I,K))*W0(K,J)
656   CONTINUE
      CALL DLINRG (3,W0C,3,W0CI,3)
      DO 658 I=1,3
      AKAT(I,1)=0
      AKAT(I,2)=0
      AKAT(I,3)=0
      DO 659 J=1,3
      AKAT(I,1)=AKAT(I,1)-W0CI(I,J)*Q1(J)
      AKAT(I,2)=AKAT(I,2)-W0CI(I,J)*Q2(J)
      AKAT(I,3)=AKAT(I,3)-W0CI(I,J)*Q3(J)
```

```
659     CONTINUE
658     CONTINUE
        AN0=0
        AM0=0
        AK0=0
        DO 660 I=1,3
        AN0=AN0+AKAT(3,I)*T1D(I)
        AM0=AM0+AKAT(2,I)*T1D(I)
        AK0=AK0+AKAT(1,I)*T1D(I)
660     CONTINUE
        AN0=AN0*DEL
        AM0=AM0*DEL
        AK0=AK0*DEL
        RETURN
        END
        SUBROUTINE MATTR4(A,D)
        DOUBLEPRECISION A(4,4),D(4,4)
        DO 7 I=1,4
        DO 8 J=1,4
        D(I,J)=A(J,I)
8       CONTINUE
7       CONTINUE
        RETURN
        END
        SUBROUTINE AMAT (T,ALP5,BET5,TAU5)
        DOUBLEPRECISION AM(4),R1(4),R2(4),R3(4)
        DOUBLEPRECISION A1(4,4),A2(4,4),A3(4,4)
        DOUBLEPRECISION W0(3,3),R0(4),T1(3)
        COMMON AM,R0,R1,R2,R3,A1,A2,A3,PI,W0
        DOUBLEPRECISION T(3),ALP,BET,TAU,PI
        DOUBLEPRECISION ALP5,BET5,TAU5
        DATA A1,A2,A3 /48*0.0/
        ALP=ALP5*PI/180
        BET=BET5*PI/180
        TAU=TAU5*PI/180
        DO 1 I=1,3
1       T1(I)=T(I)*PI/180
        W0(1,1)=COS(ALP)*COS(BET)
        W0(1,2)=-SIN(ALP)
        W0(1,3)=0
        W0(2,1)=SIN(ALP)*COS(BET)
        W0(2,2)=COS(ALP)
```

```
W0(2,3)=0
W0(3,1)=-SIN(BET)
W0(3,2)=0
W0(3,3)=1
A1(1,1)=COS(T1(1))
A1(1,3)=-SIN(T1(1))
A1(2,1)=SIN(T1(1))
A1(2,3)=COS(T1(1))
A1(3,2)=-1
A1(3,4)=32.5
A1(4,4)=1
A2(1,1)=COS(T1(2))
A2(1,2)=-SIN(T1(2))
A2(2,1)=SIN(T1(2))
A2(2,2)=COS(T1(2))
A2(3,3)=1
A2(4,4)=1
A2(1,4)=50*COS(T1(2))
A2(2,4)=50*SIN(T1(2))
A2(3,4)=55.
A3(1,1)=COS(T1(3))
A3(1,3)=SIN(T1(3))
A3(2,1)=SIN(T1(3))
A3(2,3)=-COS(T1(3))
A3(3,2)=1
A3(4,4)=1
RETURN
END
SUBROUTINE CROPRO (RC,RG,S,CP)
DOUBLEPRECISION RC(4),S(4),CP(4),RG(4)
CP(1)=(RC(2)-RG(2))*S(3)-S(2)*(RC(3)-RG(3))
CP(2)=(RC(3)-RG(3))*S(1)-(RC(1)-RG(1))*S(3)
CP(3)=(RC(1)-RG(1))*S(2)-(RC(2)-RG(2))*S(1)
RETURN
END
SUBROUTINE INERT2(A,B,C)
DOUBLEPRECISION A(4,4),B(4,4),C(4,4),D(4,4),E(4,4)
CALL MATTR4(A,D)
CALL MMUL(B,D,E)
CALL MMUL(A,E,C)
RETURN
END
```

```
      SUBROUTINE MMUL (B,A,F)
      DOUBLEPRECISION A(4,4),B(4,4),F(4,4)
      DO 1 I=1,4
      DO 1 J=1,4
      F(I,J)=0
      DO 1 K=1,4
1     F(I,J)=F(I,J)+B(I,K)*A(K,J)
      RETURN
      END
      SUBROUTINE MVMUL (A,B,C)
      DOUBLEPRECISION A(4,4),B(4),C(4)
      DO 1 I=1,4
      C(I)=0
      DO 1 J=1,4
1     C(I)=A(I,J)*B(J)+C(I)
      RETURN
      END
```

APPENDIX C

ROBOSIM SIMULATION FILE

```
LOOK-FROM X=0.,Y=-200.,Z=80.
LOOK-AT X=0.,Y=0.,Z=20.
CLEAR
;***********************
; DEFINE THE LOCATION
;***********************
F-JOINT-I
EXECUTE-FILE MARKI.DAT
STORE B
CLEAR
F-JOINT-I+1
TRANSLATE X=-70.,Y=0.,Z=-10.
ADD B
STORE-LINK OSMAN4.LOC
CLEAR
STORE B
;**************************
;BUILT 0TH COORDINATE FRAME
;ROTATION ABOUT X-AXIS (YAW)
;**************************
F-JOINT-I
STORE B
CLEAR
R-JOINT-I+1
TRANSLATE X=0.,Y=-25.,Z=0.
ROTATE X=-90.
ROTATE Z=-90.
ADD B
STORE-LINK OSMAN4.L0
CLEAR
STORE B
;***************************
;ROTATION ABOUT Y-AXIS (PITCH)
```

```
;****************************
R-JOINT-I
STORE B
CLEAR
R-JOINT-I+1
ROTATE X=90.
ROTATE Z=-90.
ADD B
STORE-LINK OSMAN4.L1
CLEAR
STORE B
;**************************
;ROTATION ABOUT Z-AXIS (ROLL)
;**************************
R-JOINT-I
STORE B
CLEAR
R-JOINT-I+1
ROTATE X=90.
ROTATE Z=90.
ADD B
STORE-LINK OSMAN4.L2
CLEAR
STORE B
;*******************************
;SATELLITE MAIN BODY (A CYLINDER)
;*******************************
CYLINDER R=25.,H=50.
STORE C
CLEAR
R-JOINT-I
ADD C
STORE C
CLEAR
R-JOINT-I+1
TRANSLATE X=0.0,Y=0.,Z=32.5
ADD B
ADD C
STORE-LINK OSMAN4.L3
```

```
CLEAR
STORE B
STORE C
STORE D
;********************************
;BUILD LINK 1 OF THE MANIPULATOR
;*********************************
R-JOINT-I
STORE B
CLEAR
BOX X=4.,Y=15.,Z=15.
TRANSLATE X=9.5,Y=0.,Z=0.
STORE D
TRANSLATE X=-19.,Y=0.,Z=0.
ADD D
ADD B
STORE B
CLEAR
CYLINDER R=7.5,H=60.
TRANSLATE X=0.,Y=0.,Z=22.5
ROTATE X=-90.
STORE C
CLEAR
R-JOINT-I+1
ROTATE X=-90.
ADD B
ADD C
STORE-LINK OSMAN4.L4
CLEAR
STORE B
STORE C
;*************************
;LINK 2 OF THE MANIPULATOR
;*************************
R-JOINT-I
STORE B
CLEAR
BOX X=15.,Y=4.,Z=10.
TRANSLATE X=0.,Y=9.5,Z=52.5
```

```
STORE D
TRANSLATE X=0.,Y=-19.,Z=0.
ADD D
STORE D
CLEAR
BOX X=50.,Y=15.,Z=5.
TRANSLATE X=17.5,Y=0.,Z=55.
STORE C
CLEAR
R-JOINT-I+1
TRANSLATE X=50.,Y=0.,Z=55.
ADD B
ADD C
ADD D
STORE-LINK OSMAN4.L5
CLEAR
STORE B
STORE C
STORE D
;************************
;LINK 3 OF THE MANIPULATOR
;************************
CYLINDER R=7.5,H=5.
R-JOINT-I
STORE B
CLEAR
BOX X=40.,Y=15.,Z=5.
TRANSLATE X=27.5,Y=0.,Z=0.
STORE C
CLEAR
R-JOINT-I+1
TRANSLATE X=55.,Y=0.,Z=0.
ROTATE X=90.
ADD B
ADD C
STORE-LINK OSMAN4.L6
CLEAR
STORE B
STORE C
```

```
;************************
;LINK 4 OF THE MANIPULATOR
;************************
CYLINDER R=7.5,H=10.
R-JOINT-I
STORE B
CLEAR
R-JOINT-I+1
TRANSLATE X=20.5,Y=0.,Z=0.
ROTATE X=-90.
STORE C
CLEAR
BOX X=8.,Y=8.,Z=8.
TRANSLATE X=11.5,Y=0.,Z=0.
ADD B
ADD C
STORE-LINK OSMAN4.L7
CLEAR
STORE B
STORE C
;************************
;LINK 5 OF THE MANIPULATOR
;************************
CYLINDER R=5.,H=5.
R-JOINT-I
STORE B
CLEAR
R-JOINT-I+1
ROTATE X=90.
ROTATE Z=90.
STORE C
CLEAR
BOX X=8.,Y=8.,Z=5.
TRANSLATE X=9.,Y=0.,Z=0.
ADD B
ADD C
STORE-LINK OSMAN4.L8
CLEAR
STORE B
```

```
STORE C
;************************
;LINK 6 OF THE MANIPULTOR
;************************
R-JOINT-I
STORE B
CLEAR
R-JOINT-I+1
ROTATE Z=90.
ADD B
STORE B
CLEAR
CYLINDER R=5.,H=3.
TRANSLATE X=0.,Y=0.,Z=14.5
ADD B
STORE-LINK OSMAN4.L9
CLEAR
STORE B
;*****************************
;END EFFECTOR (A CYLINDER ROD)
;*****************************
F-JOINT-I+1
TRANSLATE X=0.,Y=0.,Z=22.
STORE B
CLEAR
CYLINDER R=1.,H=8.
TRANSLATE X=0.,Y=0.,Z=20.
F-JOINT-I
ADD B
STORE-LINK OSMAN4.L10
;*************************
;VIEW THE SYSTEM
;*************************
VIEW-ROBOT OSMAN4
END
```

REFERENCES

[1] "Advancing Automation and Robotics Technology for the Space Station and for the U. S. Economy," NASA TR 87566, 1985.

[2] Bronez, M.A., Clarke, M. M., and Quinn, A., "Requirements for a Free Flying Robot- The 'ROBIN'," Proceedings, IEEE International Conference on Robotics and Automation, 1986, San Francisco, CA.

[3] "Automation and Robotics for the National Space Program," NASA-CR-175881, University of California, 1985.

[4] Fernandez, K., "The Use of Computer Graphics Simulation in the Development of Robotic Systems," Acta Astronautica, vol. 17, no. 1, 1988, pp. 115-122.

[5] ——— "Robotic Simulation and a Method for Jacobian Control of a Redundant Mechanism with Imbedded Constraints," Ph. D. Dissertation, Vanderbilt University, 1988.

[6] Paul, R. P., Robot Manipulators, Mathematics, Programming and Control, MIT Press, Cambridge, Mass., 1981.

[7] Brady, M., Hollerbach, J. M.,Johnson, T. L., Lozano-Perez, T., Mason, M. T., Robot Motion, Planning and Control, MIT Press, Cambridge, Mass., 1982.

[8] Yamada, K., Tsuchiya, K., and Tadakawa, S., "Modeling and Control of a Space Manipulator," in Proceedings of 13th International Symposium on Space Technology and Science, pp. 993-998, 1982.

[9] Luh, J. Y. S., Walker, M. W., and Paul, R. P. "On-Line Computational Scheme for Mechanical Manipulators," Transaction of ASME Journal of Dynamics, Measurement and Control, June 1980, vol. 120, pp. 69-76.

[10] Longman, R. W., Lindberg, R. E., and Zedd, M. F., "Satellite-Mounted Robot Manipulators- New Kinematics and Reaction Moment Compensation," International Journal of Robotics Research, vol. 6, no. 3, pp. 87-103, 1987.

[11] Vafa, Z., and Dubowsky, S., "Kinematics, Dynamics and Control of Space Manipulators: The Virtual Manipulator Concept," Ph. D. Dissertation, MIT, 1987.

[12] Umetani, Y., and Yoshida, K., "Continuous Path Control of Space Manipulators Mounted on OMV," (IAF Paper 86-13), Acta Astronautica, vol. 15, no. 12, pp. 981-986, 1987.

[13] ——— "Resolved Motion Rate Control of Space Manipulators with Generalized Jacobian Matrix," IEEE Transactions on Robotics and Automation, vol. 5, no. 3, pp. 303-314, June 1989.

[14] Whitney, D. E., "Resolved Motion Rate Control of Manipulators and Human Prostheses," IEEE Transactions, Man-Machine Systems, vol. MMS-10, no. 2, pp. 47-53, 1969.

[15] Denavit, J., and Hartenberg, R. S., "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices," ASME Journal of Applied Mechanics, vol. 22, pp. 215-221, 1955.

[16] Asada, H., nad Slotine, J. J. E., Robot Analysis and Control, John Wiley & Sons, New York, N. Y., 1986.

[17] Torby, B. J., Advanced Dynamics for Engineers, Holt,Rinehart and Winston, New York, N. Y., 1984.

[18] McDonald, Donald, "How Does a Cat Fall on Its Feet?" New Scientist, 30 June 1960, pp. 1647-1649.

[19] ——— "How Does a Man Twist in the Air?" New Scientist, 1 June 1961, pp. 501-503.

[20] Kulwicki, P. V., Schlei, E. J., and Vergamini, P. L., "Weightless Man. Self-orientation Techniques," 6570th Aerospace Medical Research Lab., Wright-Patterson Air Force Base, Ohio. Technical Report No TDR 62-129, 1962.

[21] McCrank, J. M., "Torque Free Rotational Dynamics of a Variable Configuration Body (Application to Weightless Man)," M. S. Thesis, Air Force Institute of Technology, May 1964.

[22] Stephantsov, V. I., Yeremin, A., and Alekperov, S., "Maneuvering in Free Space," NASA TT F-9883, 1966.

[23] Stephantsov, V. I., and Yeremin, A., "Fundamentals of Human Biomechanics in an Unsupported Situation," NASA TT F-10411, 1967.

[24] Smith, P. G., and Kane, T. R., "The Reorientation of a Human Body in Free Fall," Technical Report, no. 171, Stanford University, 1967.

[25] Kane, T. R., and Scher, M. P., "Human Self-Rotation by Means of Limb Movements," Journal of Biomechanics, vol. 3, pp. 39-49, 1970.

[26] Passerello, C. E., and Huston, R. L., "Human Attitude Control," Journal of Biomechanics, vol. 4, pp. 95-102, 1971.

[27] Vafa, Z., and Dubovsky, S., "On the Dynamics of Manipulators in Space Using Virtual Manipulator Approach," Proceedings of IEEE International Conference on Robotics and Automation, pp. 579-585, 1979.

[28] Symon, K. R. Mechanics 3rd ed., Addison-Wesley, Reading, Mass., 1971.

[29] Crandall, H., Dynamics of Mechanical and Electromechanical Systems, McGraw Hill Inc., New York, 1968.

[30] Borg, S. F., Matrix-Tensor Methods in Continuum Mechanics, Van Nostrand Co. Inc., New Jersey, 1963.

[31] Hibbeler, R. C., Engineering Mechanics, Dynamics, Macmillan Publishing Co. Inc., New York, 2nd Ed., 1978.

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

[33] Springfield, J. F. "ROBOSIM Workstation Extensions," M. S. Thesis, Vanderbilt University, 1988.

[34] Fernandez, K., "ROBOSIM User's Guide," Internal Publication of Electrical Engineering Department, Vanderbilt University, July 1986.