**Improvements In the Control of Robotic**

**Motion Simulations Using the ATB Model**

by

Daniel W. Barineau

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Engineering Mechanics

APPROVED:

Dr. Daniel J. Schneck, Chairman

Dr. Daniel Frederick

Dr. John W. Grant

December 9, 1988

Blacksburg, Virginia

**Improvements In the Control of Robotic**

**Motion Simulations Using the ATB Model**

by

Daniel W. Barineau

Dr. Daniel J. Schneck, Chairman

Engineering Mechanics

(ABSTRACT)

Modifications were made to the control model for torque generation in the Air Force Articulated Total Body (ATB) simulation computer program. Limb motion stability was improved by introducing integral control in the existing feedback control equation. Motion studies were performed using a Merlin robot model to determine control equation gains for single and multi-joint rotations up to 180 degrees. The robotic motion was made to resemble coordinated angular motion profiles that had previously been determined for similar human arm motion. The control equation gains for the six joints examined were added to the input description as a tabular set of data, which the program could access depending on the joint target angles prescribed by the user. Simultaneous multi-joint rotations were also studied using the same controlling values as were used for single joint rotations. These numbers produced accurate results for all joint rotations, as long as either the shoulder or elbow joints were held at their initial angular positions. The errors produced when the target angles for both the shoulder and elbow were non-zero were less than two degrees of arc.

# Acknowledgements

I offer my most sincere thanks to my guide and counsel throughout my graduate career, Dr. Daniel J. Schneck. Without him, I would not have had the enthusiasm I now have for the field of biomedical engineering, and most likely would not have continued my education past the bachelors level.

I must also acknowledge the contributions of the Air Force Office of Scientific Research and the Air Force Systems command in giving me the invaluable research opportunity at Wright-Patterson AFB during the summer. The people I have interacted with in the Modelling and Analysis Branch of the H.G. Armstrong Medical Research Laboratory helped my program in every way possible, and without them this project would be far from completion.

Finally, I would like to thank my family for their support of my academic career. My parents, more than any, have always been a source of strength and love that many times kept my head above water. To them, and my sisters, my brother-in-law, my grandparents, aunts, uncles and cousins, I thank you from the bottom of my heart.

# Table of Contents

# List of Tables

# List of Figures

# Introduction and Historical Background

The Articulated Total Body (ATB) model is a computer simulation program used to analyze human body responses in dynamic environments. It was originally developed by Calspan Corporation (1) in the 1970's for the National Highway Traffic Safety Administration (NHTSA) to simulate three dimensional physiologic dynamics in automobile crashes. That initial program, called CVS for Crash Vehicle Simulator, was modified in 1975 to include aerodynamic force application and a harness belt capability (2), thus creating the first version of the ATB. In 1980, further improvements were made in the area of restraining system modelling, along with the addition of elements from the then three dimensional CVS program to form ATB-II (3). With the incorporation of effects from windblast, ATB-III was generated in 1983 (4). The latest version, ATB-IV, was documented in 1988 (5) and included many changes over previous models. Some of these include:

1. A new wind force option allowing segment contact ellipsoids to block wind;
2. Corrections to prevent angular drift in joints;
3. Improvements allowing the prescription of multi-axis angular displacements; and

4. A hyperellipsoid option for modelling surfaces in simulations such as corners and edges.

Modifications to the ATB program which have not yet been documented include such diverse situations as active (i.e. due to muscle contraction) torque generation at the joints, human interactions with water, and the modelling and control of robotic motion.

Aerospace operations including satellite retrieval, hazardous waste disposal, and the movements of heavy objects provide opportunities for the use of remotely controlled robots. Information on what forces and motions the equipment will be subjected to should logically precede any equipment development. With these objectives in mind, the ATB can provide kinematic and dynamic data by simulating the desired robotic system and environment. It could also provide useful information in simulating the interaction between a human subject wearing an exoskeletal device, and a remotely controlled slave robot.

Along the same line of reasoning, active torque generation at the joints of ejection or crash models could provide a more accurate picture of dynamic human response to high-speed ejections. The mannikin and dummy computer models currently being used can only respond passively to changes in their environment, while humans respond both passively and actively to the same situations. This represents a significant limitation of the current ATB model, and is the problem addressed in this thesis. An initial step in the direction of including active physiologic responses to dynamic environments is to include in the ATB the generation and accurate control of joint torques in a robotic arm system. This could then be expanded to include all of the joints in a given system, and ultimately modified to

simulate the forces and torques generated by in vivo muscle contraction in the human body.

It is the intent of this thesis to improve upon the current ability of the ATB model to generate precise angular motions in a robotic simulation. In particular, the work described herein has contributed significantly to removing steady state errors in the predicted angular displacement, and thus improving the stability of the ATB simulations. Moreover, the detailed analysis undertaken below of the variation in the proportional, integral, and derivative (PID) control variables will allow future input data files to be modified so that in simulating a given motion, the user need only specify the desired relative angular displacements of the joints in the system.

# Operation Background of the ATB

## *A. Kinetic Analysis*

In its current form, the ATB model is based on the rigid body dynamics of coupled systems with Lagrange type constraint equations. This allows a given system to be described as a set of rigid elements, coupled by joints at which torques are applied as functions of joint orientations and rates of change of orientations (6). The program assumes the presence of only one joint between any 2 segments of the model. The input data file used for the robotic simulations provides all information concerning moments of inertia, segment mass and geometry, material properties and orientations of all segments in the system, and the characteristics of each joint and actuator. All of the segmental equations of motion are computed using the centers of gravity of the segments in the inertial reference frame.

## B. Kinematic Analysis

As a basis for performing calculations in the ATB program, matrix and vector mathematics are used relative to right handed orthonormal reference systems (6). In this regard, a 3x3 directional cosine matrix, $\vec{D}$ , is used to define the orientation of each of the N segments in the model. As the program proceeds, the directional cosine matrices are updated by the use of quaternions, $q$,which are defined (1,7,8,9) as consisting of both a vector part $\vec{m}$ and a scalar part s and are written as:

$$q = (s + \vec{m}). \qquad [1]$$

Then rotation of vector $\vec{n}$ an angular amount $\theta$ about the axis $\vec{t}$ to vector $\vec{n}'$ can be expressed by:

$$\vec{n}' = q\vec{n}q' \qquad [2]$$

where

$$q = \cos\frac{\theta}{2} + (\sin\frac{\theta}{2})\vec{t}\,; \text{ and} \qquad [3]$$

$$q' = \cos\frac{\theta}{2} - (\sin\frac{\theta}{2})\vec{t}\,. \qquad [4]$$

For a null rotation, the scalar parts of $q$ and $q'$ would be equal to one while the vector terms would be equal to zero. In the ATB program, these quaternions are used to update the cosine matrices relating the changes in segment orientations.

For input and output purposes, the terms yaw, pitch and roll were chosen over spin, nutation and precession for signifying the directions of the cosine matrices. The

three terms signify yaw about the z-axis, pitch about the y-axis, and roll about the x-axis (see Figure 1).

## C. Integration

The integrator being used by the program to integrate the equations of motion is called the Vector Exponential Integrator, and is based on a fourth order Runge-Kutta technique with an exponential term for each variable. The integration of a first order differential equation proceeds as follows (1):

$$\dot{x} = f(x,t) \qquad [5]$$

For which a solution of the following form can be written:

$$x(t) = x(0) + \int_0^t e^{\alpha(t-\tau)} [\, f(x(\tau), \tau) - \alpha(x(\tau) - x(0))] \, d\tau \qquad [6]$$

where $\alpha$ is a coefficient to be determined. Assuming an approximation for $\dot{x}$ as:

$$\dot{x}(t) = f(x(t), t) \simeq \alpha x(t) + a_0 + a_1 t + a_2 t^2 \qquad [7]$$

where $\alpha$, $a_0$, $a_1$ *and* $a_2$ are parameters to be determined. The equation for $x(t)$ can then be solved to yield:

$$x(t) = x(0) + (\alpha x(0) + a_0)\, te_0(t) + a_1 t^2 e_1(t) + a_2 t^3 e_2(t) \qquad [8]$$

where

$$e_0(t) = \frac{(e^{\alpha t} - 1)}{(\alpha t)} \quad \rightarrow 1 \quad as\ \alpha t \quad \rightarrow 0$$

$$e_1(t) = \frac{(e_0^{\alpha t} - 1)}{(\alpha t)} \quad \rightarrow \frac{1}{2} \quad as\ \alpha t \quad \rightarrow 0$$

$$e_2(t) = \frac{(2e_1^{\alpha t} - 1)}{(\alpha t)} \quad \rightarrow \frac{1}{3} \quad as\ \alpha t \quad \rightarrow 0$$

These exponential terms are responsible for the integrator's name.

The four parameters $\alpha$, $a_0$, $a_1$ and $a_2$ determine the behavior of the integrator. They are always chosen so that the equation fits the computed derivatives at the beginning of each integration interval. For a successfully integrated time interval $\tau$, the value of (t + $\tau$) is substituted for t into equation [7], rewritten as follows:

$$\dot{x}(t) = \alpha(x(t) - x(0)) + \dot{x}(0) + a_1 t + a_2 t^2 \qquad [9]$$

which yields:

$$\begin{aligned} \dot{x}(t+\tau) &= \alpha(x(t+\tau) - x(\tau)) + (a_1 + 2a_2\tau)t \\ &+ a_2 t^2 + \alpha(x(\tau) - x(0)) + \dot{x}(0) + a_1\tau + a_2\tau^2. \end{aligned} \qquad [10]$$

The functions are then redefined to preserve the form of the above equation, i.e.:

$$a_1 = a_1 + 2a_2\tau,\ a_2 = a_2\ ,$$

$$\dot{x}(0) = \dot{x}(\tau)\ ,\ x(t) = x(t+\tau).$$

These terms are then used to estimate the value of x(t) at the first half step of the next interval, i.e., when $t = \frac{\tau}{2}$ (See reference 1 for more complete details).

## D. Segment and Joint Properties

The equations of motion used to describe the model are those for a set of connected rigid bodies. The joints between the segments define the "connectivity" of the model through a joint vector $\vec{JNT}(j)$. In the case of a null joint, i.e. $\vec{JNT}(j) = 0$, the limb segments articulating at that joint are considered to be disjointed segments. In this case, two segments are considered to be positioned as if they were connected, but no force exists to hold them in that position during the simulation. The solution of the system also depends on various constraint equations which are affected by joint type. Some of the possible constraints that the ATB model can accommodate include linear position (inseparable or "fused" joints), angular position (1-2-or-3-axis type of joint), zero distance (common points shared by more than one segment), fixed distance and rolling/sliding motion.

The six joints of the robotic model are not exactly equivalent to the joints found in the human body. The names of the six joints (waist, shoulder, elbow, forearm, wrist and palm) are more closely related to the positions of these joints in the human body than their corresponding function in the human body. The waist joint can create a rotation (yaw) around the z-axis of the upper torso relative to the lower torso as seen in Figure 1. The shoulder joint allows flexion and extension (pitch about the y-axis) of the robotic arm. The elbow joint comes closest to representing its namesake in human anatomy, and provides flexion and extension of the lower arm relative to the upper arm. The forearm joint in the robotic model allows supination and pronation (roll about the x-axis) of the hand. In the human system, this rotational effect is produced by the rotation of the radius about the ulna in the lower arm. In both cases the result is the same, this being the turning of the palm of the hand

anteriorly. The wrist joint in the model provides flexion and extension of the wrist, similar to one of the functions of the same joint in the human body. The palm joint allows abduction and adduction of the hand, which is anatomically another function of the human wrist.

Each body segment is approximated in the model by either an ellipsoid or a hyperellipsoid. This defines the segment shape for purposes of external force application and contact surface definition. Data that the program requires in order to describe each segment include inertial and material properties, environmental conditions, and various joint parameters. Segment resiliency during contact is described through the use of force deflection constitutive characteristics, which provide for energy losses, permanent offset, and impulsive forces which are experienced during the simulation.

## *E. Descriptive Elements*

The simulation of material properties is accomplished by the use of tension elements (passive longitudinal muscle), flexible elements (neck, torso, trunk), and actuator elements (contractile muscle). It can be seen that the objective of these elements is to deliver a gross description of body motion, and therefore they have little physiological discretion.

The tension elements are designed to behave statically in a manner similar to linear springs, without the corresponding stiffness, when subjected to a compression force. The ATB model represents the tension elements as a discrete system of N particles connected by N-1 springs. The tension element is subject to constraints

which insure that all particles lie on a straight line, and that the strains and relative motions within the element are uniquely determined by the positions and motions of the two end elements (1). The equations of motion for these elements are dependent on the positions, velocities, and accelerations of these same two end-points.

The flexible elements are composed of a chain of N joined rigid elements. Each joint has three degrees of freedom with three corresponding stiffness constants. In addition, each of the N-2 interior segments of the flexible element is constrained so that its orientation is uniquely determined by the orientations of the end (or outer) segments. These constraints have been introduced to approximate the effects of body muscles which are connected, so that rather than acting on individual joints, they determine the overall flexural characteristics of the represented body member (1).

In order to determine angular motion, the actuator elements subroutine requires inputs of position, velocity, and acceleration at the end points of the involved segments. Unlike the tension elements, which offer passive resistance, the actuators can produce an active force across a joint. Similar to human muscle, one of the segments attached to the joint is considered to be stationary (the origin) while the other segment (the insertion) is being moved through some angle $\theta$. The torque generated at this junction is determined and controlled by a feedback control equation that can be defined by the program user. It is this equation that is addressed in the work which follows.

## *F. Program Operation*

The total ATB program consists of over 130 subprograms totalling over 16,000 lines of Fortran source code. The flow of operation of the ATB program is shown in Figures 2-8. Each of the figures breaks the total program down into working units that perform the following functions:

The MAIN program (Figure 2) controls such activities as program initialization and execution, the use of required input and start-up routines, the restart procedure, optional output and post-processing operations. Using a given input file, the MAIN file reads the following:

1. Units of distance, force and time
2. The components of the gravity vector
3. Control parameters for the integration routine DINT
4. Various parameters affecting the output of data.

The flow pattern shown in Figure 3 controls various inputs and initializations required for subsequent calculations. Some of the subprograms involved are:

**BINPUT** processes the physical characteristics of the body segments and joints;

**SINPUT** reads and prints data involving constraints and body segment options;

**VINPUT** processes the prescribed motion of specified segments; and

**INITAL** performs the input and computation of the body segments initial positioning.

Other subroutines, most notably OUTPUT and POSTPR also process input data when they are initially called (1).

Subroutine DINT (Figure 4) is the fourth order Runge-Kutta integrator of the program, and advances a variable amount, dt, for each successful step. Subroutine OUTPUT is called at the end of each successful step, and reproduces the current time point in the tabular time histories.

The flowsheet shown in Figure 5 prepares the program between integrations for the next time step.

The DAUX flowsheet (Figure 6) involves control of the computational part of the program, primarily in the evaluation of the derivatives for use in subroutine DINT. Some of the major subprograms include:

**SETUP1** computes the parameters required for determining the forces and torques acting on the body segments and joints;

**SETUP2** sets up the equations for constraint forces specified by the program input;

**VEHPOS** computes linear and angular accelerations for those segments having a prescribed motion; and

**FLXSEG** sets up the equations to control the flexible elements specified by the input file.

Auxiliary DAUX programs reduce the matrix of equations representing the system. This set is then solved by the subroutine FSMSOL (1).

The CONTCT flow-pattern (Figure 7) computes the sum of the forces and torques acting on body segments as a result of contact with objects or internal generation. The main segment contact subroutines are:

**PLELP** computes the effects of plane-ellipsoid contacts;

**SEGSEG** computes the effects of ellipsoid-ellipsoid contacts; and

**SPDAMP** computes the spring and viscous forces of spring dampers between specified points on selected segments.

Force producing subroutines, such as those used in the actuator elements, are the most recent introductions into the ATB program. As seen in Figure 7, they primarily involve the following subprograms:

**CONTCT** controls the calling of subroutines required to compute those external and internal forces and torques acting on the body segments;

**APPLY** applies driving joint torques to the adjacent segments;

**JNTFNC** calculates the applied driving joint function torque; and

**USER** user supplied subroutine that calculates actuator torques. This is the subroutine addressed in this thesis.

The new subroutines that create the hyperellipsoids for modelling body segments (Figure 8) are included in version ATB-IV. Some of the subprograms involved are:

**HYVAL** computes the point on a hyperellipsoid that lies on a particular line

**HYBOX** computes the intersection of a plane with the edges of a rectangular box

**HYLIM** calculates the boundaries of the figure formed by the intersection of a hyperellipsoid and a plane; and

**HYREA** computes the approximate area and centroid for the figure formed by the intersection of a hyperellipsoid and a plane.

It should be noted that many subprograms quite frequently go unused in any given simulation. For instance, those subroutines involving seat belt harnesses, air bags and wind generation would most likely not be used in a robotic arm simulation. On the other hand, if a situation developed in which it became necessary to use these subroutines, the uniformity throughout the ATB's programming would allow it.

## *G. Program Output*

The ATB program provides for a wide variety of outputs with respect to motion, force generation, and damage due to interaction with the environment. Time histories for the entire simulation can be put together and projected as three-dimensional images, or in the form of tabular data. Every model segment and joint can be monitored for values of displacement, velocity and acceleration as well as applied and developed torques and forces at the joints. Furthermore, force-deflection characteristics between two or more segments can be analyzed to determine whether or not any damage has occurred during the simulation run. Damage can result from both subject-subject and subject-environment collisions.

# Problem Statement

## *A. Robot Description*

The basis for the current ATB model is an American Cimflex MR6500 Merlin Robot with six articulations including the waist, shoulder, elbow, forearm, wrist and palm as described previously in section D, and shown in Figure 1. This figure also shows the orientation of the Inertial Cartesian coordinate system with respect to the model. As noted earlier, each of the joints involved has one axis about which it can rotate, these being:

**Waist** yaw about the z-axis

**Shoulder** pitch about the y-axis

**Elbow** pitch about the y-axis

**Forearm** roll about the x-axis

**Wrist** roll about the x-axis

**Palm** pitch about the y-axis

In addition, each simulation is started in the same position for the sake of continuity, although different initial configurations could be used. The picture shown in Figure 1 is for illustrative purposes only. The actual segments would be simulated in the ATB program by elliptical elements. This particular model was chosen primarily because one was available, and the necessary physical data could easily be obtained.

## *B. System Control*

The control of angular displacement created by the driving joint torque is contained in subroutine USER (see Figures 7, 9 and 10), which can be modified to represent any controlling equation the operator desires. In previous simulations, the torque controlling equation was originally written as:

$$T = f_2(\theta - \theta_0) - f_3(\dot{\theta}) \qquad [11]$$

where

$T$ is the torque applied by the actuator, which may, in vivo, be due to the contraction of the musculature spanning any given joint;

$\theta_0$ represents the desired or target angle measured relative to the initial joint position and is some function of time, $f_1(t)$;

$\theta$ is the current joint angle measured relative to the initial joint position;

$\dot{\theta}$ is the joint angular velocity; and

$f_i$ is the input function with $i = 1, 2, 3, \ldots$

In the case to be considered here,

$$f_1(t) = constant, \text{ i.e. } \theta_0 = constant \qquad [12]$$

$$f_2(\theta - \theta_0) = K_1(\theta - \theta_0) \qquad [13]$$

$$f_3(\dot{\theta}) = K_2\dot{\theta} \qquad [14]$$

The coefficients $K_1$ and $K_2$ are transfer functions that scale the corresponding proportional and derivative control variables. The type of control shown in equation [11], is typically referred to as PD control.

Rotating the robot segments from an initial state of rest is considered by the ATB program to be a set-point change in the desired configuration. In the absence of a reset or integral term in the controlling equation, the system experiences some steady-state error in the desired angular displacement. This thesis addresses the addition of integral control to the existing control equation [11], which modifies it to create a more appropriate equation for the torque feedback:

$$T = f_2(\theta - \theta_0) - f_3(\dot{\theta}) - f_4(\int_0^t (\theta - \theta_0)dt) \qquad [15]$$

All terms are the same for those appearing in equation [11], plus;

$$f_4(\int_0^t (\theta - \theta_0)dt) = K_3(\int_0^t (\theta - \theta_0)dt) \qquad [16]$$

Where $K_3$ is another transfer function that scales the integral term.

Each of the terms signifies a different type of correction to the angular displacement profile in transitioning from one model configuration to another. The block diagram of a system that uses a controlling equation similar to equation [15] is shown in Figure 11. The influences of different types of control on a set-point disturbance are illustrated in Figures 12-14.

## C. Active Motion Simulations

The control of joint actuators is further complicated when arm configurations undergo changes relative to certain joints during multiple segment motions. The most predominant effects involve coordinating the activity between the shoulder and elbow joints during complex maneuvers that involve simultaneous movements of the upper and lower arms. This makes sense when one considers that the moments of inertia being controlled by the actuators at the proximal joints are much larger than those of the more distal joints. Moreover, it was also observed in earlier ATB simulations (6) that the wrist joint experienced some unstable oscillations during multi-joint motions. This could result from the inability of the ATB model to compensate for acceleration produced at this joint by rotations of the shoulder and elbow. Thus, it was further suggested that the additional control term shown in equation [15] might help to alleviate these errors, and so this thesis addresses the effects of such integral control on coordinated body movements.

In order to use the ATB program to model robotic movement, $K$ values had to be determined by trial and error to simulate the desired joint displacement profile. This profile is a description of how the joint angle changes with respect to time during a

simulation. Therefore, a study of the functional variations of the $K$ parameters, over the entire range of segment motion was undertaken. The values of $K_1$, $K_2$ and $K_3$ were varied until they produced a similar angular motion profile, which represents a good approximation of coordinated angular displacement patterns measured for a human arm (10, 11, 12, 13). The inability to create simulations of human motion without some experimental basis is one of the drawbacks of the ATB program. To overcome this problem, experimental motion data was used as an example of the how the robotic system should move. The conditions imposed are similar to those discussed by Schneck (14) in relation to human motion analysis, i.e.:

1. Less than 0.5 percent angular overshoot;
2. An ultimate angular displacement (i.e. steady-state) error of less than 0.03 degrees of arc; and
3. A rise or build-up time to the target angle somewhere in the range of 0.25 to 0.45 seconds.

If useful relationships were found, both the input data file, and subroutine USER were to be modified to incorporate the $K$ value data and thus simplify the use of the ATB program. That is, if successful, all that an operator would have to do to simulate joint motion under the previously mentioned conditions, would be to specify the desired target angles for each joint in the simulation. This is the basis for the work that follows.

# Solution Methods

## *A. Control Equation Changes*

The modifications to the torque controlling equation [11] of the ATB program were initially completed on the Perkin-Elmer computer system at the H.G. Armstrong Medical Research Lab at Wright-Patterson AFB. The controlling equation was altered from equation [11] to equation [15] by the addition of the following steps in subroutine USER:

1. Calculation of the differential time step dt, via subroutine DINT in Figure 2, for the proposed integration step;
2. Calculation of the product of the change in angular error and the differential time step, i.e. $(\theta_{target} - \theta_{current})dt$;
3. Addition of this differential product to the sum of all the products that have occurred to this point in the simulation. This creates a sum that, for small time steps, approximates the integral term in equation [15];

4. Multiplication of this sum and a coefficient $K_3$ (see Section B below); and
5. Use of the resulting value from step 4 in the torque controlling equation [15] if the integration of the equations of motion for the system are successful for that time interval. Otherwise, begin at step 1 again with the values for the summation produced at the last successful integration step, and proceed using the new differential time step provided by subroutine DINT.

The changes described above can be seen in Figures 9 and 10, with Figure 9 being the version of subroutine USER with only PD control, and Figure 10 being the updated version with PID control.

Verification of the integral term properly operating in the controlling equation [15] was limited to the observation and analysis of resulting simulation data. If the added term worked correctly, steady-state errors would be significantly decreased in single joint rotations, along with the introduction of oscillations about the desired target angle.

## *B Determination of the Controlling Equation K's*

$K_1$, $K_2$ and $K_3$ were initially determined for single joint rotations of the six joints in the model as functions of the relative angular displacement between the initial and desired final joint angles. In order to get an initial approximation for the behavior of these terms, the Newton-Euler and Statics equations describing a two link manipulator were examined (15). Any relevant insights thus derived could then be applied to the somewhat more complicated six link manipulator being used in the

actual ATB simulation. Proceeding in a manner similar to Brady (15), for the system in Figure 15, the equations describing the dynamics relative to joints 1 and 2 of the system are:

$$\begin{aligned} I_{11}\ddot{\theta}_1 + I_{12}\ddot{\theta}_2 = {} & \frac{1}{2} m_2 l_1 l_2 \sin\theta_2(\dot{\theta}_1 + \dot{\theta}_2)^2 - (\frac{1}{2} m_1 + m_2) l_1 g_2 \cos\theta_1 \\ & + n_{0,1} - n_{1,2} - l_1 f_{2,3y} \cos\theta_1 + l_1 f_{2,3x} \sin\theta_1 \end{aligned} \qquad [17]$$

where

$$I_{11} = I_1 + (\frac{1}{4} m_1 + m_2) l_1^2 + \frac{1}{2} m_2 l_1 l_2 \cos\theta_2$$

$$I_{12} = \frac{1}{2} m_2 l_1 l_2 \cos\theta_2$$

$$I_1 = \frac{m_1 l_1^2}{12} + \frac{m_1 R_1^2}{4}$$

and

$$\begin{aligned} I_{21}\ddot{\theta}_1 + I_{22}\ddot{\theta}_2 = {} & -\frac{1}{2} m_2 l_2 g_2 \cos(\theta_1 + \theta_2) + n_{1,2} - n_{2,3} \\ & - l_2 f_{2,3y} \cos(\theta_1 + \theta_2) + l_2 f_{2,3x} \sin(\theta_1 + \theta_2) \end{aligned} \qquad [18]$$

where

$$I_{21} = I_2 + \frac{1}{4} m_2 l_2^2 + \frac{1}{2} m_2 l_1 l_2 \cos\theta_2$$

$$I_{22} = I_2 + \frac{1}{4} m_2 l_2^2$$

$$I_2 = \frac{m_2 l_2^2}{12} + \frac{m_2 R_2^2}{4}$$

The terms used in these equations are defined as;

| | |
|---|---|
| $I_1$ , $I_2$ | are the rotary inertias through the link 1 and 2 centers of mass |
| $n_{0,1}$ , $n_{1,2}$ | are the total torques (active + passive) needed at joints 1 and 2 to produce a given motion |
| $l_1$ , $l_2$ | the lengths of segments 1 and 2 as shown in Figure 15 |
| $m_1$ , $m_2$ | the masses of segments 1 and 2 as shown in Figure 15 |
| $\theta_1$ , $\theta_2$ | are the joint angles as described by Figure 15 |
| $f_{2,3y}$ , $f_{2,3x}$ *and* $n_{2,3}$ | are unknown disturbance forces and torques applied at the end of segment 2; |
| $R_1$ , $R_2$ | are the distances from the base of the x-y axis in Figure 15 to the center of mass of segments 1 and 2 respectively; and |
| $g_2$ | is the component of the gravity vector in the vertical ( $-y$) direction. |

Using the previously described PID controlling equation [15], a steady state analysis can be performed to determine some of the conditions for stability. Taking equation [15] for T, it can be substituted into equations [17] and [18] for the $n_{0,1}$ , $n_{1,2}$ and $n_{2,3}$ terms. These equations can then be solved to get relationships between the controller gains and the terms in the equations of motion. Using a steady state analysis of these equations ( a steady state for the system occuring when all time derivatives in equations [15], [17] and [18] approach zero, the disturbances are constant, and the control system is stable) the following observations can be made;

1. All three coefficients ($K_1$ , $K_2$ and $K_3$) will be some functions of cos $\theta_0$ and sin $\theta_0$ (where $\theta_0$ is defined in the ATB as the amount of angle between the initial and desired final joint angles) or system parameters including the orientation, geometry, and segment masses;

2. The coefficients will be directly proportional to the mass and length of the segments being moved;

3. Gravity correction (i.e. integral) terms, will be important for actuators supporting heavier segments in the gravitational field;

4. The gain terms for properly controlling the motion of the most distal of two joints should be directly related to both its own rotation angle, and that of the joint proximal to it. In other words, instead of a dependance on $\theta_2$ as shown in Figure 15, it will exhibit a dependance on $(\theta_1 + \theta_2)$; and

5. The torque controlling equations for any joint will be affected by the torques and forces applied at the other joints of the model, i.e., the solutions to equations [17] and [18] obtained at joint $i$ will be coupled to simultaneous solutions of these same sets of equations applied to all other joints $j$ ($j = 1 - 5$) in the ATB model.

These points were also noted by Brady (15), who went on to state that a stability analysis on this system shows that in general, positive values for $K_1$ and $K_2$, and either positive or negative values for $K_3$ as described in equation [15] should result in stable model motion. Work by Golla, Garg and Hughes (16) produced similar positive values for the controlling coefficients. Hollerbach (17), and Smith and Corripio (18) come to the conclusion that for the same system shown in Figure 15, there are trajectories for which all three coefficients in the torque controlling equation would play significant roles.

All of the above information was used to determine the control equation gains and their variations with respect to changes in the joint target angle. The shoulder and elbow joints were examined first because of the relatively large moments of

inertia associated with the limbs that articulate there, and because of the large resulting steady state displacements. Subsequent simulations involved determining controller gains for the robotic joints referred to as the waist, forearm, wrist and palm. In all cases, angular rotations from initial positions covered a range of rotation from -180 to 180 degrees with respect to initial joint angles. Contact between segments was not taken into consideration in determining the external forces on the robotic model.

The simulation of multi-joint motion would initially be attempted using the same controller gains that were found to work for single joint rotations. From equations [17] and [18] this can be seen not to be exact, because of changes in the end effector forces and torques ($f_{2,3y}$ , $f_{2,3x}$ *and* $n_{2,3}$) introduced by the multi-level coupling discussed earlier. If the magnitudes of these coupling terms are small for a certain joint, this could be a good approximation. Such was the case initially assumed for the four robotic joints (waist, forearm, wrist and palm) that had not previously developed any significant displacement errors. Subsequent simulations included one of the two remaining joints, the shoulder and the elbow. Finally, simulations were attempted involving all six joints rotating simultaneously to different relative target angles.

## *C. Program and Input Deck Modifications*

After developing functional relationships for the controller gains in the torque controlling equation [15], modifications to the USER subroutine (Figures 9 and 10) and the ATB program input deck were made. Because of the program quality that allows input functions to only be described by simple polynomials, the new coefficient data

was incorporated as a tabulated set of values. This set of "look-up" tables define the relationships between $\theta_0$ and the PID coefficients for each of the six joints in the model (see Figure 16 for examples). Similar schemes for dynamics tabulation have been used in the past by Raibert (19) and Albus (20) to decrease the number of computations required by the computer system. Verification of correct program operation could be accomplished by comparing results of tabular and non-tabular simulations using similar joint target angle and controller gain conditions.

# Results and Discussion

## *A. Control Equation Changes*

Before the present work was accomplished, simulations of shoulder and elbow rotations had experienced the most significant displacement errors at steady state. As shown in Table 1, after modifying the controller equation as described above, these joints could be controlled with very small amounts of angular error for single joint rotations. The numbers in Table 1 represent the changes in rotational accuracy before and after the addition of integral control. Values for the desired and actual resulting joint angles are shown for these two joints because of the large effect the integral term had on these values. All angles are rounded to the nearest tenth of a degree, and the values obtained using integral control had a steady state oscillation of plus or minus 0.02 degrees. The summation of the differential changes in angular error as an approximation for the integral term worked correctly. If it had not been properly programmed, i.e. the summation was computed for both successful and

unsuccessful time steps, the integral term would have become excessively large in a short amount of time.

The characteristic oscillations associated with the addition of an integral term into the controlling equation appeared as expected, but were limited to a tolerable range of plus or minus 0.02 degrees about the target angle. This is the drawback to control solutions using integral control, i.e., by removing most of the steady state displacement that proportional and derivative terms cannot account for, an instability in the system is created. If this instability, as in this case, is within a small enough region about the target angle, then the equation with the integral term can be used.

## *B. Determination of the Controlling Equation K's*

Values of the control equation coefficients were determined using the desired angular motion profile as described on page 19 of this thesis, and the stability analysis performed on the two link system equations of motion [17] and [18]. The resulting values for the controller gains $K_1$, $K_2$ and $K_3$ for the six joints in the model are shown in Tables 2-7. These tables show the variation of the $K$'s with respect to the user prescribed target angles for each of the joints. The simulations were performed with rotations of the joint in question while holding the other five joints in the model at their initial angles. Examples of the three gains are plotted in Figures 17-19 as functions of $\theta_0$ for single joint rotations. These figures show more clearly the periodic relationship of the gains with respect to the joint target angles. With respect to this data, the following observations were made;

1. Nearly all of the $K$ values are either periodic (implying a cosine or sine relationship ), or constant (implying negligible effect of the periodic functions ) with varying $\theta_0$. This was predicted by the analysis of the Newtonian equations of motion [17] and [18]. The only exception is $K_1$ (proportional control ) for the shoulder joint which exhibits an inverse relationship with varying $\theta_0$. The cause of this difference is most likely due to the amount of weight this joint must manipulate within the gravity field, compared to the other joints in the model. If the total mass of the arm is set equal to some normalized value M, then the amount of mass being moved by each of the joints in the arm are as follows;

   - shoulder 1.00M
   - elbow 0.45M
   - forearm 0.04M
   - wrist 0.03M
   - palm 0.02M

   The waist joint was not included because of its rotation perpendicular to the gravity field.

2. Magnitudes of the $K$ values varied in proportion to the mass and lengths of the segments being moved by the joint. Once again, this was predicted by the analysis of the Newtonian equations of motion.

3. Stable $K$ values were found to occur with the predicted signs, i.e. $K_1$ and $K_2$ were positive while $K_3$ was both positive and negative.

4. $K_3$, the integral control term, plays a major role in those joints that must support large relative masses against the pull of gravity, these joints being the shoulder

and elbow. On the other hand, integral control appeared to be negligible at the other four joints due to the small moments of inertia of the segments involved, or the negligible effects of gravity, or both.

5. Assuming that $\alpha$-motoneurons in the human body act in a PID fashion when controlling human muscle contractions, then the tabulated values of the controller gains represent a first approximation of human myoelectric syntax. These values are relevant only in a gross anatomical sense, due to the complexity of the human musculature about any given joint. They do however give an idea of the variation in signal values and magnitudes necessary at different joints to produce similar angular motion.

Examples of the resulting controlled angular displacement profiles, can be seen in Figures 20-22 for typical joint rotations. Referring to the desired angular motion profile described on page 19 of this text, these figures can be seen to have all of the characteristics originally specified. All of these simulations are for single joint motions, with all other joints held at initial angles.

For multi-joint rotations, the same controlling coefficients (see Tables 2-7) were used as in single joint rotations. From Figures 23-25, it can be seen that the initial tests of four and five moving joints were almost as well coordinated as the single joint rotations had been. The one notable exception was for the wrist pitch, which experienced significant initial oscillations that eventually died out with no displacement error. This was most likely caused by the inability of the program to compensate for the acceleration at the end of the robot arm when the more axially located joints are simultaneously rotated.

In attempting to simulate coordinated rotation of all six joints simultaneously, end-effector forces and applied torques at the segment ends, as shown in equations [17] and [18], yielded noticeable errors in the desired angular displacement for the shoulder and elbow joints. The controller gains for the elbow joint did not remain the same as for single joint rotations. Instead, they were equal to the gains for a target angle determined by the sum of the shoulder and elbow rotation angles. This response was predicted by the previous analysis of equation [18]. Overall, the inaccuracies encountered in the shoulder and elbow joints for simultaneous rotations were approximately one percent of the relative angular displacement.

## *C. Program and Input Deck Modification*

Using the controlling equation coefficients derived for single joint rotations, the input deck was modified to include the newly acquired data. This created a need to modify the way in which subroutine USER obtained data from the input data file. The comparison of tabulated simulations and single value simulations resulted in the same output values for rotational angles included in the data set. *K* values for angles not included in the data set, were interpolated from gains that were in the table.

From the standpoint of a program user, the addition of the look-up tabular gain descriptions provides an increase in the flexibility of the ATB program. Prior to the modifications, the user had to define the three control equation gains for each joint in the model, and use a trial and error method of tuning these parameters to produce the desired response. For even a simple six jointed robotic model such as this one, that comes to 18 controller parameters that must be determined just for one set of

joint rotations. With the tabulation developed here, the user now needs only to specify the desired target angles for the joints in the simulation in order to generate fairly accurate angular rotations along the guidelines proposed in part C of the Problem Statement.

# Summary and Conclusions

## *A. ATB Program Changes*

The present study involved the addition of integral control to the joint torque controlling equations used in the ATB model. This modification eliminated most cases of steady state displacements (offset errors) in the robotic motion simulations considered. The primary joints that this modification affected were the shoulder and elbow, due to the large inertial masses being manipulated by these joints in the gravity field. Steady state oscillations of the joint about the desired target angle were introduced into the solution by the integral term as predicted, but were found to be in an acceptably small (less than 0.02 degrees) range.

After completing an analysis of the controller gains with respect to a range of desired angular joint displacements, further modifications were made to the computer program to incorporate this new data. This consisted of expressing the gains for each joint in a tabular form that the computer program could access during a simulation. This worked as predicted, and improved the ease with which a well

controlled coordinated robotic simulation could be produced by any user of the system.

## B. Control Equation Gains

Values were determined for the $K$ parameters found in equation [15] as functions of the joint rotational angles during single joint motions. These relationships were found to be either periodic or constant values with respect to a variation in $\theta_0$, the desired angular joint rotation. The one exception to this was the proportional control gain for the shoulder joint, which displayed a relationship proportional to the inverse of $\theta_0$. The $K$ terms also varied in accordance with a Newtonian analysis of a similar linkage system. The analysis of the equations of motion revealed information concerning magnitudes, functional properties and relative effects of the control equation gains, and made the prediction of those gains much easier. The results as seen in Tables 2-7 and Figures 20-25 show that the controlling equation can be made to produce well behaved angular motion profiles for up to five of the six joints in the model. Kinematic and dynamic data for these cases should prove useful in robotic motion modelling and analysis. The exception to this is found in simulations containing simultaneous rotation of the shoulder and elbow joints, which creates displacement errors that the current set of equations cannot correct. Possible ways of correcting this problem include solving the equations of motion exactly, including end-effector forces for each time step in the simulation, although this would significantly increase the number of calculations performed by the program. Another possibility is the use of successive approximation, in which an initial estimate for the

control equation gains is modified and updated during the simulation. This is a form of adaptive control, and is an area in which research should be focused in the future.

## C. Direction of Future Studies

The way in which the present system controls the simulated robotic arm is sufficient for all cases excluding the one in which both the shoulder and elbow are simultaneously moved. Even then, the error is still only about one percent of the desired angular displacement, which might not be unacceptable in the case that the user is considering. Following the previously mentioned addition of adaptive control, the next logical step is the validation of simulation results with the actual motion studies of a similar robot. If the results of the simulation are indeed valid, then the expansion of the model to include other joints and segments of the body (legs, neck, etc.) could be used to expand the model. This would in turn lead to the incorporation of active torque generation in crash and ejection dummy models. Simulations could then be run in which active human motion is attempted in a dynamic environment. This would create a much more realistic example of the actual dynamics in these situations.

If the adaptive control schemes prove to be successful, then some type of signal analysis could be performed on the controller gains in an attempt to better understand the way in which the human central nervous system controls active motion. This would be a step towards understanding myoelectric syntax, or the way in which the CNS speaks to the muscles to generate forces at joints.

One other possibility for future improvements in the ATB model is the incorporation of the concept of least energy with respect to human motion discussed by Schneck (14), and Nubar and Contini (21). This could remove the need to fit the ATB motion data to any experimental description of human motion, and instead allow the model to decide what motion would most likely occur in any given situation.

# Bibliography


1. Fleck, J.T., Butler, F.E., and N.J. Deleys, "Validation of the Crash Victim Simulator," Report Nos. DOT-HS-806-279 thru 282, Vols. 1-4, 1982.

2. Fleck, J.T. and F.E. Butler, "Development of an Improved Computer Model of the Human Body and Extremity Dynamics," Report No. AMRL-TR-75-14 , April 1975.

3. Butler, F.E. and J.T. Fleck, "Advanced Restraint System Modeling," Report No. AFAMRL-TR-80-14, May 1980 (NTIS No. AD-A088 029).

4. Butler, F.E., Fleck, J.T., and D.A. Difranco, "Modeling of Whole-Body Response to Windblast," Report No. AFAMRL-TR-83-073, October 1983 (NTIS No. AD-B079 184)

5. Obergefell, L.A., Kaleps, I., Gardner, T.R., and J.T. Fleck, "Articulated Total Body Model Enhancements," Report No. AAMRL-TR-88-007 thru 009, Vols. 1-3, 1988.

6. Obergefell, L.A., Avula, X., and I. Kaleps, "The uses of the Articulated Total Body Model as a Robot Dynamics Simulation Tool," *Proceedings of the 1988 SOAR Conference* , Wright State University, Dayton, Ohio, July 1988.

7. Taylor, R.H., "Planning and Execution of Straight Line Manipulator Trajectories," *IBM Journal of Research and Development* , 23(1979), 424-436.

8. Beeler, M., Gosper, R.W., and R. Schroeppel, "Hakmem," Artificial Intelligence Laboratory, MIT, AI Memo 239, February 1972.

9. Hamilton, W.R. **Elements of Quaternions.** Chelsea Publishing Co. New York, N.Y. 1969.

10. Flash, T. and N. Hogan, "The Coordination of Arm Movements: An Experimentally Confirmed Mathematical Model," *The Journal of Neuroscience,* 5(1985), 1688-1703.

11. Lacquanti, F. and J.F. Soechting, "Coordination of Arm and Wrist Motion During a Reaching Task," *The Journal of Neuroscience,* 2(1982), 399-408.

12. Winters, J.M. and L. Stark, "Simulation of Fundamental Movements: I. "Systems" Analysis," *Proceedings Of the Seventh Anual IEEE Conference of the Engineering in Medicine and Biology Society,* 1985, 39-50.

13. Van Dijk, J.M.N., "Simulation of Human Arm Movements Controlled by Peripheral Feedback," *Biological Cybernetics,* 29(1978), 175-185.

14. Schneck, D.J. **Mechanics of Muscle.** Santa Barbara, Ca. Kinko's Publishing Group. 1985.

15. Brady, M., et. al. Introductory section in **Robot Motion: Planning and Control.** Cambridge, Ma. The MIT Press. 1982.

16. Golla, D.F., Garg, S.C., and P.C. Hughes, "Linear State-Feedback Control of Manipulators," *Mech. Machine Theory* , 16(1981), 93-103.

17. Hollerbach, J.M. Dynamics section in **Robot Motion: Planning and Control.** Cambridge, Ma. The MIT Press. 1982.

18. Smith, C.A. and Corripio, A.B. **Principles and Practices of Automatic Process Controls.** New York, N.Y. John Wiley and Sons. 1985.

19. Raibert, M.H., "Analytical Equations vs. Table Look-Up for Manipulation: A Unifying Concept," *Proc. IEEE Conf. Decision and Control,* New Orleans, La., December 1977, 576-579.

20. Albus, J.S., "Data Storage in the Cerebellar Model Articulation Controller (CMAC)," *J. Dynamic Systems, Measurement, Control,* 97(1975), 228-233.

21. Nubar, Y. and R. Contini, "A Minimal Principle in Biomechanics," *Bull. Math. Biophysics,* 23(1961), 377-391.

22. Schneck, D.J., "Feedback Control and the Concept of Homeostasis," *Mathematical Modelling*, 9(1987), 889-900.

23. Benati, M., Gaglio, S., Morasso, P., Tagliasco, V., and R. Zaccaria, "Anthropomorphic Robotics," *Biological Cybernetics*, 38(1980), 141-150.

# TABLES

**Table 1.** **Differences between desired and actual angular rotations for two joints, before and after the addition of integral control**

| Elbow before | | Elbow after | |
|---|---|---|---|
| target | actual | target | actual |
| 30° | 28.9° | 30° | 30.0° |
| 60° | 59.2° | 60° | 60.0° |
| 90° | 91.7° | 90° | 90.0° |
| 120° | 123.1° | 120° | 120.0° |
| 150° | 154.2° | 150° | 150.0° |

| Shoulder before | | Shoulder after | |
|---|---|---|---|
| target | actual | target | actual |
| 30° | 28.1° | 30° | 30.0° |
| 60° | 59.3° | 60° | 60.0° |
| 90° | 92.0° | 90° | 90.0° |
| 120° | 123.8° | 120° | 120.0° |
| 150° | 155.5° | 150° | 150.0° |

**Table 2. Variations in the torque controlling equation gains for angular rotations of the Waist joint**

| $\theta_0$ | $K_1$ | $K_2$ | $K_3$ |
|---|---|---|---|
| -180 | 5500 | 600 | negligible |
| -160 | 4000 | 450 | negligible |
| -140 | 3500 | 380 | negligible |
| -120 | 3000 | 330 | negligible |
| -100 | 2500 | 320 | negligible |
| -90 | 2450 | 305 | negligible |
| -80 | 2500 | 320 | negligible |
| -60 | 3000 | 330 | negligible |
| -40 | 3500 | 380 | negligible |
| -20 | 4000 | 450 | negligible |
| 0 | 5500 | 600 | negligible |
| 20 | 4000 | 450 | negligible |
| 40 | 3500 | 380 | negligible |
| 60 | 3000 | 330 | negligible |
| 80 | 2500 | 320 | negligible |
| 90 | 2450 | 305 | negligible |
| 100 | 2500 | 320 | negligible |
| 120 | 3000 | 330 | negligible |
| 140 | 3500 | 380 | negligible |
| 160 | 4000 | 450 | negligible |
| 180 | 5500 | 600 | negligible |

**Table 3. Variations in the torque controlling equation gains for angular rotations of the Shoulder joint**

| $\theta_0$ | $K_1$ | $K_2$ | $K_3$ |
|---|---|---|---|
| -180 | 365 | 330 | -120 |
| -160 | 580 | 326 | -106 |
| -140 | 880 | 309 | -85 |
| -120 | 1200 | 280 | -55 |
| -100 | 1650 | 225 | -22 |
| -90 | 1800 | 150 | 0 |
| -80 | 2500 | 225 | -22 |
| -60 | 3700 | 280 | -55 |
| -40 | 6100 | 309 | -85 |
| -20 | 11200 | 326 | -106 |
| 0 | 400000 | 330 | +/-120 |
| 20 | 11200 | 326 | 106 |
| 40 | 6100 | 309 | 85 |
| 60 | 3700 | 280 | 55 |
| 80 | 2500 | 225 | 22 |
| 90 | 1800 | 150 | 0 |
| 100 | 1650 | 225 | 22 |
| 120 | 1200 | 280 | 55 |
| 140 | 880 | 309 | 85 |
| 160 | 580 | 326 | 106 |
| 180 | 365 | 330 | 120 |

**Table 4. Variations in the torque controlling equation gains for angular rotations of the Elbow joint**

| $\theta_0$ | $K_1$ | $K_2$ | $K_3$ |
|---|---|---|---|
| -180 | 20000 | 260 | -10.0 |
| -160 | 5200 | 220 | -9.5 |
| -140 | 2700 | 150 | -7.3 |
| -120 | 1600 | 110 | -4.8 |
| -100 | 1150 | 85 | -1.6 |
| -90 | 1000 | 70 | 0.0 |
| -80 | 1150 | 85 | 1.6 |
| -60 | 1600 | 110 | 4.8 |
| -40 | 2700 | 150 | 7.3 |
| -20 | 5200 | 220 | 9.5 |
| 0 | 20000 | 260 | 10.0 |
| 20 | 5200 | 220 | 9.5 |
| 40 | 2700 | 150 | 7.3 |
| 60 | 1600 | 110 | 4.8 |
| 80 | 1150 | 85 | 1.6 |
| 90 | 1000 | 70 | 0.0 |
| 100 | 1150 | 85 | -1.6 |
| 120 | 1600 | 110 | -4.8 |
| 140 | 2700 | 150 | -7.3 |
| 160 | 5200 | 220 | -9.5 |
| 180 | 20000 | 260 | -10.0 |

**Table 5. Variations in the torque controlling equation gains for angular rotations of the Forearm joint**

| $\theta_0$ | $K_1$ | $K_2$ | $K_3$ |
|---|---|---|---|
| -180 | 22 | 2.5 | negligible |
| -160 | 21 | 2.3 | negligible |
| -140 | 21 | 2.2 | negligible |
| -120 | 21 | 2.1 | negligible |
| -100 | 21 | 2.0 | negligible |
| -90 | 20 | 2.0 | negligible |
| -80 | 21 | 2.0 | negligible |
| -60 | 21 | 2.1 | negligible |
| -40 | 21 | 2.2 | negligible |
| -20 | 21 | 2.3 | negligible |
| 0 | 22 | 2.5 | negligible |
| 20 | 21 | 2.3 | negligible |
| 40 | 21 | 2.2 | negligible |
| 60 | 21 | 2.1 | negligible |
| 80 | 21 | 2.0 | negligible |
| 90 | 20 | 2.0 | negligible |
| 100 | 21 | 2.0 | negligible |
| 120 | 21 | 2.1 | negligible |
| 140 | 21 | 2.2 | negligible |
| 160 | 21 | 2.3 | negligible |
| 180 | 22 | 2.5 | negligible |

Table 6. Variations in the torque controlling equation gains for angular rotations of the Wrist joint

| $\theta_0$ | $K_1$ | $K_2$ | $K_3$ |
|---|---|---|---|
| -180 | 20 | 2.0 | negligible |
| -160 | 14 | 1.1 | negligible |
| -140 | 13 | 1.0 | negligible |
| -120 | 12 | 0.9 | negligible |
| -100 | 10 | 0.8 | negligible |
| -90 | 9 | 0.8 | negligible |
| -80 | 10 | 0.8 | negligible |
| -60 | 12 | 0.9 | negligible |
| -40 | 13 | 1.0 | negligible |
| -20 | 14 | 1.1 | negligible |
| 0 | 20 | 2.0 | negligible |
| 20 | 14 | 1.1 | negligible |
| 40 | 13 | 1.0 | negligible |
| 60 | 12 | 0.9 | negligible |
| 80 | 10 | 0.8 | negligible |
| 90 | 9 | 0.8 | negligible |
| 100 | 10 | 0.8 | negligible |
| 120 | 12 | 0.9 | negligible |
| 140 | 13 | 1.0 | negligible |
| 160 | 14 | 1.1 | negligible |
| 180 | 20 | 2.0 | negligible |

**Table 7. Variations in the torque controlling equation gains for angular rotations of the Palm joint**

| $\theta_0$ | $K_1$ | $K_2$ | $K_3$ |
|---|---|---|---|
| -180 | 3 | 0.75 | negligible |
| -135 | 3 | 0.75 | negligible |
| -90 | 3 | 0.75 | negligible |
| -45 | 3 | 0.75 | negligible |
| 0 | 3 | 0.75 | negligible |
| 45 | 3 | 0.75 | negligible |
| 90 | 3 | 0.75 | negligible |
| 135 | 3 | 0.75 | negligible |
| 180 | 3 | 0.75 | negligible |

# FIGURES

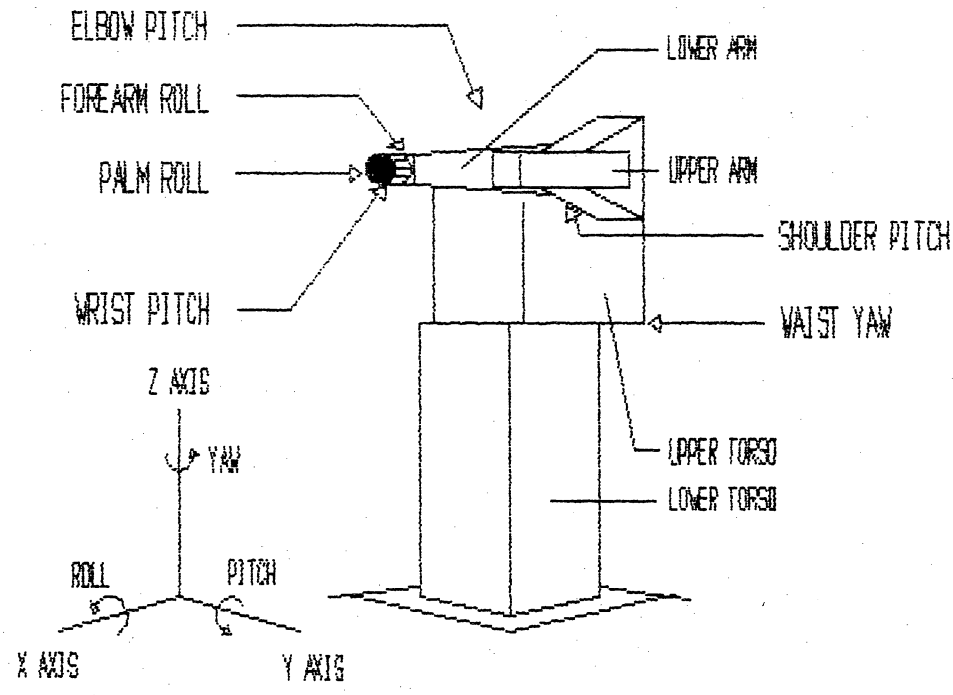


FIGURE 1 . ROBOT SIMULATION WITH SIX ARTICULATING JOINTS

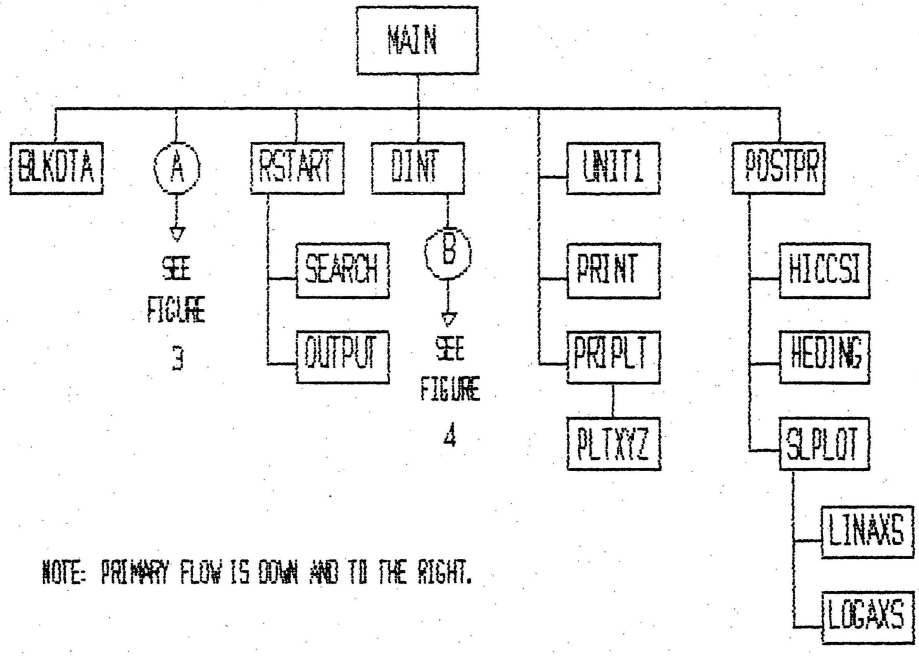


FIGURE 2 . SUBROUTINE FLOW DIAGRAM FOR THE ATB PROGRAM

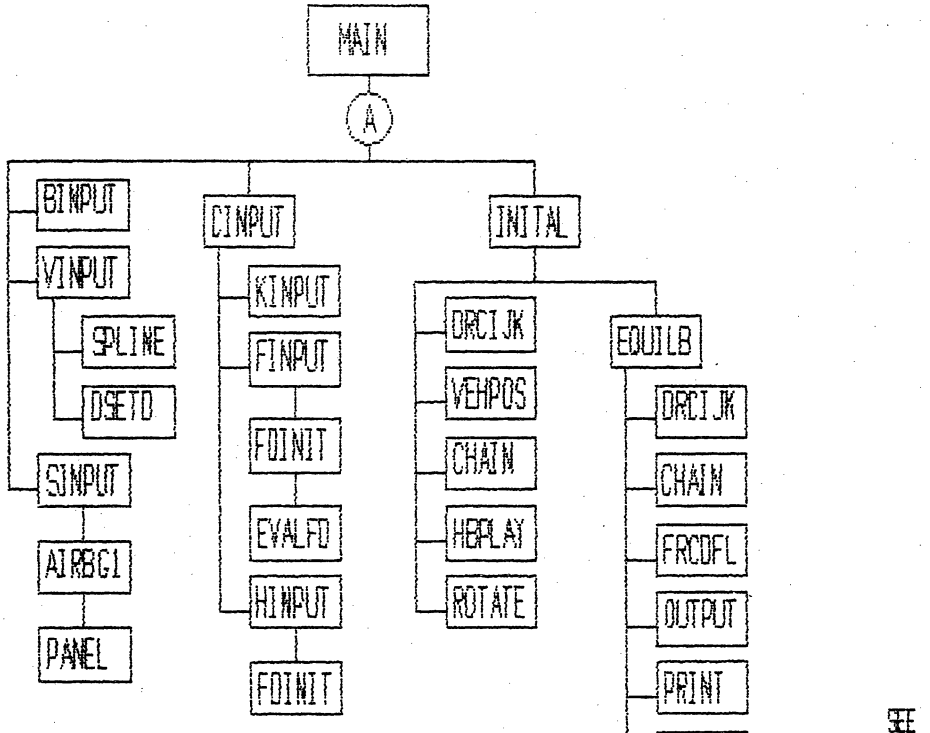


FIGURE 3. SUBROUTINE FLOW DIAGRAM FOR THE ATB PROGRAM

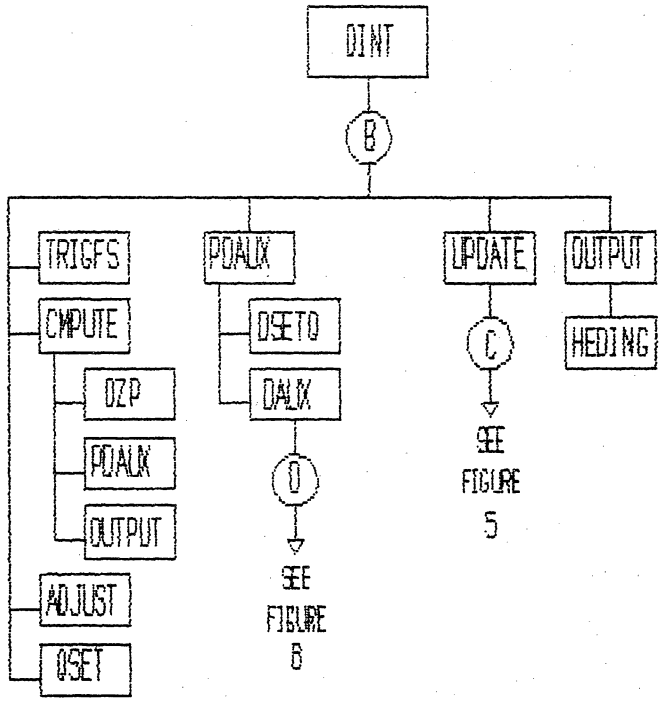


FIGURE 4. SUBROUTINE FLOW DIAGRAM FOR THE ATB PROGRAM

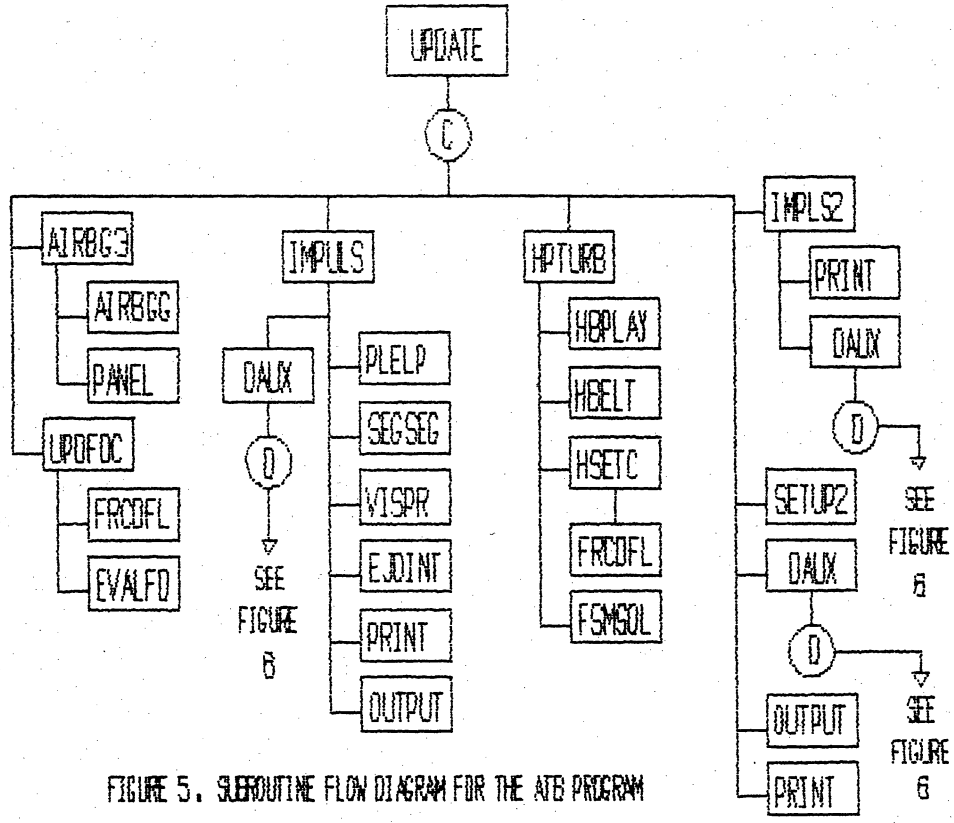


FIGURE 5. SUBROUTINE FLOW DIAGRAM FOR THE ATB PROGRAM

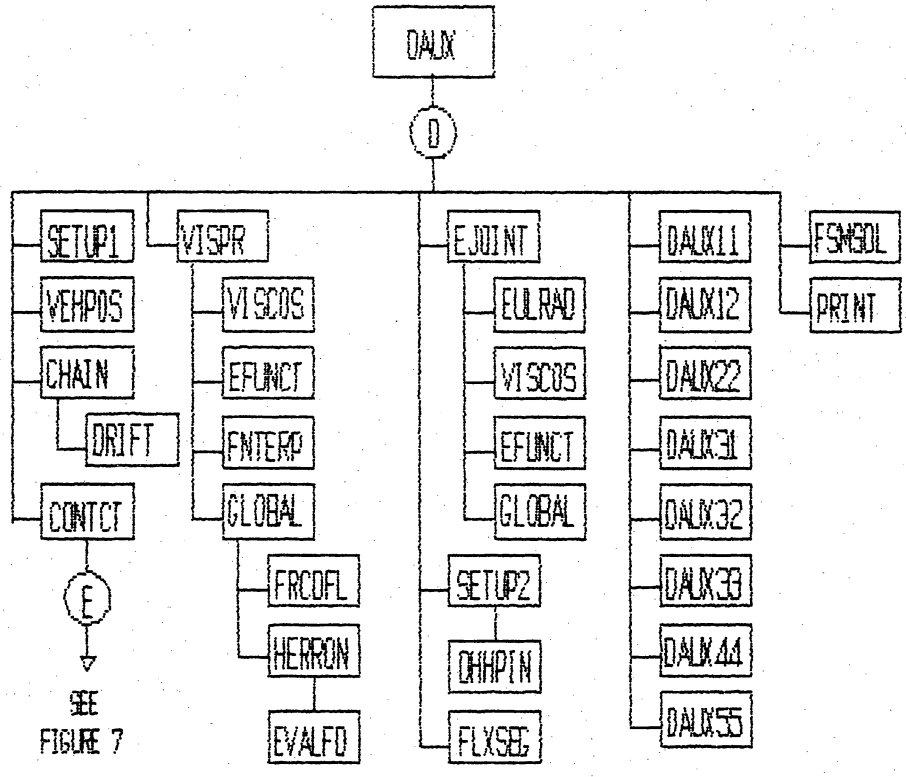


FIGURE 6. SUBROUTINE FLOW DIAGRAM FOR THE ATB PROGRAM

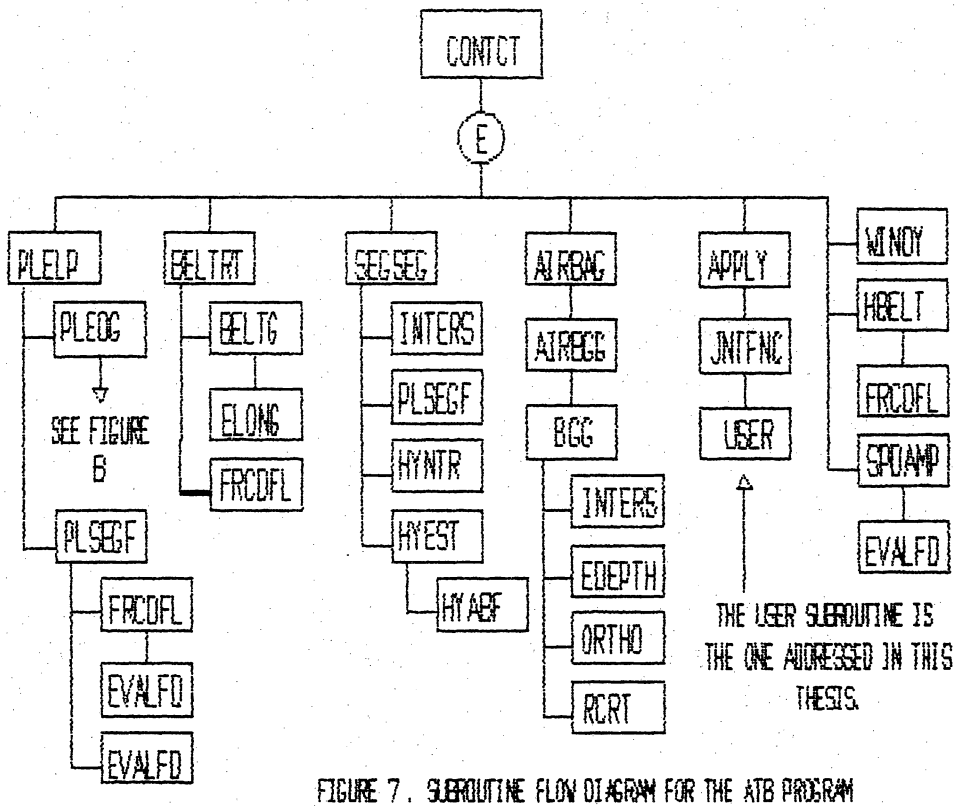


FIGURE 7. SUBROUTINE FLOW DIAGRAM FOR THE ATB PROGRAM

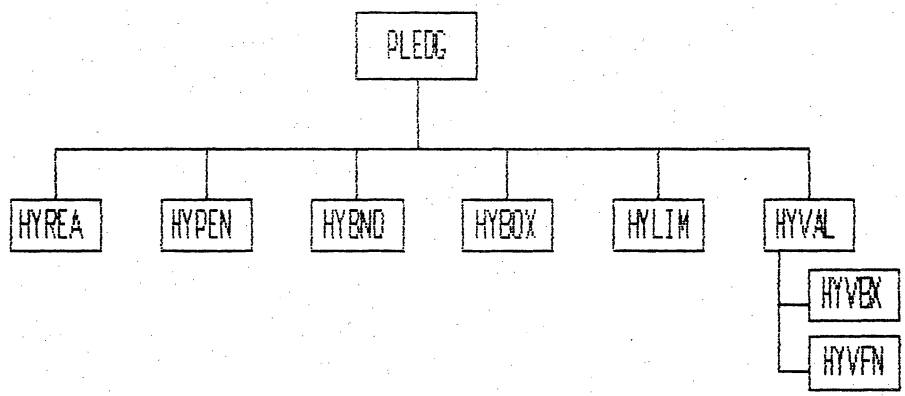


FIGURE B. SUBROUTINE FLOW DIAGRAM FOR THE ATB PROGRAM

# Legend for Figures 2-8

- ADJUST — Adjusts the current time step within subroutine DINT
- AIRBAG — Computes interactions in the system involving airbags
- AIRBGG — Computes the volumes of intersection between airbags and panels and segments
- AIRBG1 — Reads the physical properties of the airbag restraints and performs some initialization for the airbag routine
- AIRBG3 — Determines if the airbag has been fully inflated
- APPLY — Applies driving joint torques to the adjacent segments
- BELTG — Computes information involving belt restraint systems
- BELTRT — Performs additional harness belt calculations
- BGG — Performs additional airbag calculations
- BINPUT — Reads the input cards containing the physical dimensions and characteristics of the simulated body segments and joints
- BLKDTA — Initializes common/cncnts statements for the program
- CHAIN — Computes the linear position and velocity of the body segments in the inertial reference frame
- CINPUT — Input cards for the force deflection characteristics of the model
- CMPUTE — Contols the calculation of certain terms used in the integrator
- CONTCT — Controls the calling of subroutines required to compute those external forces and torques acting on body segments and joints
- DAUX — Computes derivatives used in the integrator subroutine DINT
- DAUX11 — Called by DAUX to comput a matrix multiplication
- DAUX12 — Called by DAUX to comput a matrix multiplication

- DAUX22 — Called by DAUX to comput a matrix multiplication
- DAUX31 — Called by DAUX to comput a matrix multiplication
- DAUX32 — Called by DAUX to comput a matrix multiplication
- DAUX33 — Called by DAUX to comput a matrix multiplication
- DAUX44 — Called by DAUX to comput a matrix multiplication
- DAUX55 — Called by DAUX to comput a matrix multiplication
- DHHPIN — Determines certain variables if a joint is or is not pinned
- DINT — Integrates a function supplied by the program and determines the time step length
- DRCIJK — Calls other subroutines to calculate a cosine matrix in terms of i,j and k coordinates
- DRIFT — Corrects for drift in constrained joints
- DSETD — Updates a direction cosine matrix
- DSETQ — Computes new direction matrix given the old matrix and the incremental motion expressed in quaternion form
- DZP — Computes the state variables from the parametric form assumed in DINT, along with the calculation of the exponential weights
- EDEPTH — Determines the points of maximum penetration for two intersecting ellipsoids
- EFUNCT — Computes the nonlinear spring torque for euler joints
- EJOINT — Computes the torques acting on an euler joint
- ELONG — Computes the arc length of an ellipse
- EQUILB — Adjusts initial position parameters so that normal contact forces equal the supplied values or constraint forces
- EULRAD — Conputes the euler angles in radians and places them into an array for a given direction cosine matrix
- EVALFD — Evaluates some function that is defined at a certain point in the input deck, either function, derivative or integral values
- FDINIT — Sets up certain arrays and counters for the program
- FINPUT — Reads data concerning allowable contact between body segments and the surrounding system
- FLXSEG — Sets up the equations to control the flexible elements as specified by the input file

- FNTERP — Computes the restoring torque of a joint as a function of the flexure and azimuth angles
- FRCDFL — Evaluates the force deflection function at a point D
- FSMSOL — Solves a set of simultaneous equations
- GLOBAL — Computes the effect of joint stoppage
- HBELT — Calculates functions involving the harness belt system
- HBPLAY — Performs calculations and updates functions involving the harness belt system
- HEDING — Creates the headings for time history files
- HERRON — Computes the angle of joint stoppage, and angular velocity
- HICCSI — Computes segment damage parameters for a simulation
- HINPUT — Reads the data concerning the setup and control of the harness belt system
- HPTURB — Records harness belt output values for each time step
- HYABF — Computes the hyperellipsoid function and its derivatives
- HYBND — Computes a point on a polygon, determined from the intersection of a plane and a box
- HYBOX — Computes the intersection of a plane with the edges of a rectangular box
- HYEST — Makes a preliminary approximation of the intersection of two hyperellipsoids
- HYLIM — Calculates the boundaries of the figure formed by the intersection of a hyperellipsoid and a plane
- HYNTR — Determines the points on intersecting hyperellipsoids that are used to determine peretration of these figures
- HYVAL — Computes the point on a hyperellipsoid that lies on a
- HYREA — Computes the approximate area and centroid for the figure formed by the intersection of a hyperellipsoid and a plane particular line
- HYVBX — Determines the intersection of a vector and a box
- IMPLS2 — Applies an impulse when joint J locks up
- IMPULS — Creates an impulse by contacting segments, or plane
- INITAL — Performs input and computations for initial positioning of the body segments in the simulation

- INTERS — Determines the intersection of ellipsoids
- JNTFNC — Calculates the applied driving joint function torque
- KINPUT — Processes the input for wind force and joint restoring force functions
- LINAXS — Prepares a linear axis on a plot
- LOGAXS — Prepares a logarithmic axis on a plot
- MAIN — Executable subroutine of the program
- ORTHO — Generates a right handed set of orthonormal vectors given one of the vectors
- OUTPUT — Controls tabulated output of selected optional segment accelerations, velocities and displacements, joint parameters and force data concerning segment contact
- PANEL — Computes airbag parameters during inflation of the bag
- PDAUX — Acts as an interface between subroutines DINT and DAUX to accomodate all of the variables being integrated
- PLEDG — Calculates functions when an ellipsoid and plane intersect
- PLELP — Computes the point of maximum penetration of an ellipsoid associated with a certain segment intersecting a certain plane
- PLREA — Computes the common area of an intersecting ellipsoid and plane, and its centroid
- PLSEGF — Calculates functions and interactions of environment and the model segments
- PLTXYZ — Stores plotting data in arrays for point P given in the inertial reference frame
- POSTPR — Controls the generation of printed tabular time histories
- PRINT — Prints segment linear and angular positions, velocities and accelerations for a given time
- PRIPLT — Produces printer plot of y-z and x-z plane views of body segment, and other selected points of vehicle components
- QSET — Performs internal calculations within DINT
- RCRT — Computes the radius of curvature at point Z of ellipsoid A in a certain plane
- ROTATE — Transforms variables supplied in local geometric coordinates to principal axis coordinates
- RSTART — Read input and initialization data from old restart tape, and write the new input and initialization dtat onto the new restart tape

- SEARCH — Computes certain values for use in RSTART
- SEGSEG — Computes the segment segment interactions during the simulation
- SETUP1 — Prepares certain arrays to be used for joint functions at the current time point
- SETUP2 — Sets up certain arrays for use in the DAUX subroutine
- SINPUT — Reads input data concerning vehicle geometry and restraint belts along with some body symmetry options and spring damper functions
- SLPLOT — Plotting subroutine
- SPDAMP — Computes the spring and viscous force of a spring damper between specified points of selected segments
- SPLINE — Routine to fit a set of polynamials of degree L to a set of given data points
- TRIGFS — Calculates terms used in arrays for subroutine DINT
- UNIT1 — Modifies data for use in certain AAMRL plotting programs
- UPDATE — Updates variables at the end of each successful integration step and sets up any new conditions at the beginning of a new step
- UPDFDC — Update the force deflection cureve definition assuming a successful integration step has just been completed
- USER — User defined subroutine containing the active joint torque controlling equation
- VEHPOS — Computes components of vehicle acceleration only as a function of time using data from subroutine VINPUT
- VINPUT — Reads input data concerning crash vehicle motion
- VISCOS — Computes the sum of viscous and coulomb torques at the joints
- VISPR — Computes viscous and spring torques at the joints
- WINDY — Computes forces and torques involved in windbalst effects on the model body

All definitions in this legend where obtained from reference 5, volumes 1 and 3.

```
      SUBROUTINE USER(J,THETA,THETAD,TORQ,NF)                          USER
C                                                        REV IVR   10/21/87USER
C     USER SUPPLIED SUBROUTINE THAT CALCULATES ACTUATOR TORQUES          USER
C                                                                        USER
C     INPUT PARAMETERS:                                                  USER
C                                                                        USER
C        J          ACTUATOR NUMBER                                     USER
C        THETA      JOINT ANGLE                                         USER
C        THETAD     JOINT ANGULAR VELOCITY                              USER
C        NF(I)      ARRAY OF ACTUATOR FUNCTION NUMBERS                  USER
C                                                                        USER
C     OUTPUT PARAMETER:                                                  USER
C                                                                        USER
C        TORQ(1)    ACTUATOR TORQUE                                     USER
C        TORQ(2-5) USER DEFINED PARAMETERS FOR OUTPUT IN ACTUATOR       USER
C                   TIME HISTORY                                         USER
C                                                                        USER
C                                                                        USER
      IMPLICIT REAL*8 (A-H,O-Z)                                          USER
      COMMON/CONTRL/ TIME,NSEG,NJNT,NPL,NBLT,NBAG,NVEH,NGRND,            USER
     *               NS,NQ,NSD,NFLX,NHRNSS,NWINDF,NJNTF,NPRT(36),NPG     USER
      COMMON/TABLES/MXNTI,MXNTB,MXTB1,MXTB2,NTI(50),NTAB(1250),TAB(4500)USER
      DIMENSION NF(5),TORQ(6)                                            USER
C                                                                        USER
C     THIS SUBROUTINE CALCULATES THE ACTUATOR TORQUE AS FOLLOWS          USER
C                                                                        USER
C       TORQ = F2(THETA-THETA0) - F3(THETAD)                             USER
C              WHERE THETA0 = F1(TIME)                                   USER
C                                                                        USER
C                                                                        USER
C     CALCULATE JOINT TARGET ANGLE FROM FIRST FUNCTION                   USER
C                                                                        USER
      KFT=NTI(NF(1))                                                     USER
      THETA0=EVALFD(TIME,KFT,1)                                          USER
      DTHETA=THETA-THETA0                                                USER
C                                                                        USER
C     CALCULATE FIRST TERM OF TORQUE FUNCTION FROM SECOND FUNCTION       USER
C                                                                        USER
      KFT=NTI(NF(2))                                                     USER
      TORQ(1)=EVALFD(DTHETA,KFT,1)                                       USER
C                                                                        USER
C     CALCULATE SECOND TERM OF TORQUE FUNCTION FROM THIRD FUNCTION       USER
C                                                                        USER
      KFT=NTI(NF(3))                                                     USER
      DTORQ=EVALFD(THETAD,KFT,1)                                         USER
      TORQ(1)=TORQ(1)-DTORQ                                              USER
C                                                                        USER
C     OUTPUT PARAMETERS ARE:                                             USER
C              THETA0,THETA,DTHETA,THETAD,DTORQ                          USER
C                                                                        USER
      TORQ(2)=THETA0                                                     USER
      TORQ(3)=THETA                                                      USER
      TORQ(4)=DTHETA                                                     USER
      TORQ(5)=THETAD                                                     USER
      TORQ(6)=DTORQ                                                      USER
      RETURN                                                             USER
      END                                                                USER
```

FIGURE 9. OLD VERSION OF SUBROUTINE USER

```
      SUBROUTINE USER(J,THETA,THETAD,TORQ,NF)                          USER
C                                                          REV IVR   07/25/88USER
C     USER SUPPLIED SUBROUTINE THAT CALCULATES ACTUATOR TORQUES         USER
C                                                                       USER
C     INPUT PARAMETERS:                                                 USER
C                                                                       USER
C        J          ACTUATOR NUMBER                                    USER
C        THETA      JOINT ANGLE                                        USER
C        THETAD     JOINT ANGULAR VELOCITY                             USER
C        NF(I)      ARRAY OF ACTUATOR FUNCTION NUMBERS                 USER
C                                                                       USER
C     OUTPUT PARAMETER:                                                 USER
C                                                                       USER
C        TORQ(1)    ACTUATOR TORQUE                                    USER
C        TORQ(2-5)  USER DEFINED PARAMETERS FOR OUTPUT IN ACTUATOR     USER
C                   TIME HISTORY                                        USER
C                                                                       USER
C                                                                       USER
      IMPLICIT REAL*8 (A-H,O-Z)                                         USER
      COMMON/CONTRL/ TIME,NSEG,NJNT,NPL,NBLT,NBAG,NVEH,NGRND,           USER
     *               NS,NQ,NSD,NFLX,NHRNSS,NWINDF,NJNTF,NPRT(36),NPG    USER
      COMMON/TABLES/MXNTI,MXNTB,MXTB1,MXTB2,NTI(50),NTAB(1250),TAB(4500)USER
      COMMOM/WINDFR/ WTIME(30),QFU(3,5),QFV(3,5),WF(3,30),QRU(3,30),    USER
     *               TORQUE(6,30),IWIND(30),MWSEG(7,30),NFVSEG(6),      USER
     *               NFVNT(5),MOWSEG(30,30),NRTORQ,NRJNT(30),NRS(30),   USER
     *               NRF(5,30),TIMEI,THETAI(2)                          USER
      DIMENSION NF(5),TORQ(8)                                           USER
C                                                                       USER
C     THIS SUBROUTINE CALCULATES THE ACTUATOR TORQUE AS FOLLOWS         USER
C                                                                       USER
C       TORQ = F2(THETA-THETA0) - F3(THETAD) - F4(THETAI)               USER
C              WHERE THETA0 = F1(TIME)                                  USER
C                                                                       USER
C                                                                       USER
C     CALCULATE JOINT TARGET ANGLE FROM FIRST FUNCTION                  USER
C                                                                       USER
      KFT=NTI(NF(1))                                                    USER
      THETA0=EVALFD(TIME,KFT,1)                                         USER
      DTHETA=THETA-THETA0                                               USER
C                                                                       USER
C     CALCULATE FIRST TERM OF TORQUE FUNCTION FROM SECOND FUNCTION      USER
C                                                                       USER
      KFT=NTI(NF(2))                                                    USER
      PCONT = EVALFD(THETA0,KFT,1)                                      USER
      TORQ(1) = PCONT*(DTHETA)                                          USER
C                                                                       USER
C     CALCULATE SECOND TERM OF TORQUE FUNCTION FROM THIRD FUNCTION      USER
C                                                                       USER
      KFT=NTI(NF(3))                                                    USER
      DCONT = EVALFD(THETA0,KFT,1)                                      USER
      DTORQ = DCONT*THETAD                                              USER
```

FIGURE 10. NEW VERSION OF SUBROUTINE USER

```
C                                                                       USER
C     CALCULATE THIRD TERM OF TORQUE FUNCTION FROM THE FOURTH FUNCTION  USER
C                                                                       USER
      DTIME = TIME - TIMEI                                              USER
      THETAIN = DTIME*DTHETA                                            USER
      KFT = NTI(NF(4))                                                  USER
      THETAI(2) = THETAI(1) + THETAIN                                   USER
      SCONT = EVALFD(THETAO,KFT,1)                                      USER
      TORQI = THETAI(2)*SCONT                                           USER
C                                                                       USER
C     CALCULATE THE ACTUATOR TORQUE                                     USER
C                                                                       USER
      TORQ(1) = TORQ(1) - DTORQ - TORQI                                 USER
C                                                                       USER
C     OUTPUT PARAMETERS ARE:                                            USER
C              THETAO,THETA,DTHETA,THETAD,DTORQ,THETAI,TORQI            USER
C                                                                       USER
      TORQ(2)=THETAO                                                    USER
      TORQ(3)=THETA                                                     USER
      TORQ(4)=DTHETA                                                    USER
      TORQ(5)=THETAD                                                    USER
      TORQ(6)=DTORQ                                                     USER
      TORQ(7)=THETAI(2)                                                 USER
      TORQ(8)=TORQI                                                     USER
      RETURN                                                            USER
      END                                                               USER
```

FIGURE 10. (CONTINUED)

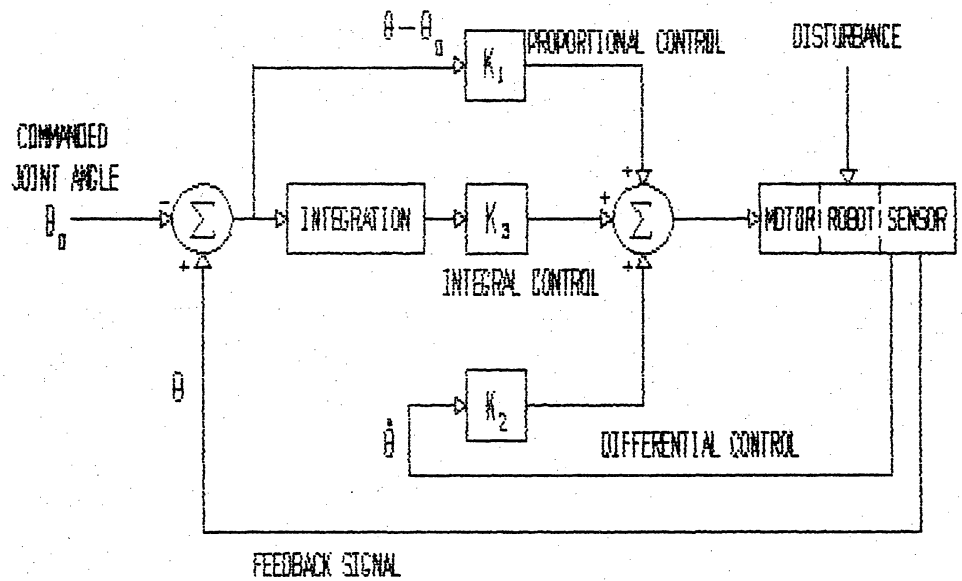


FIGURE 11 . BLOCK DIAGRAM FOR THE TORQUE CONTROLLING EQUATION OF A ROBOT

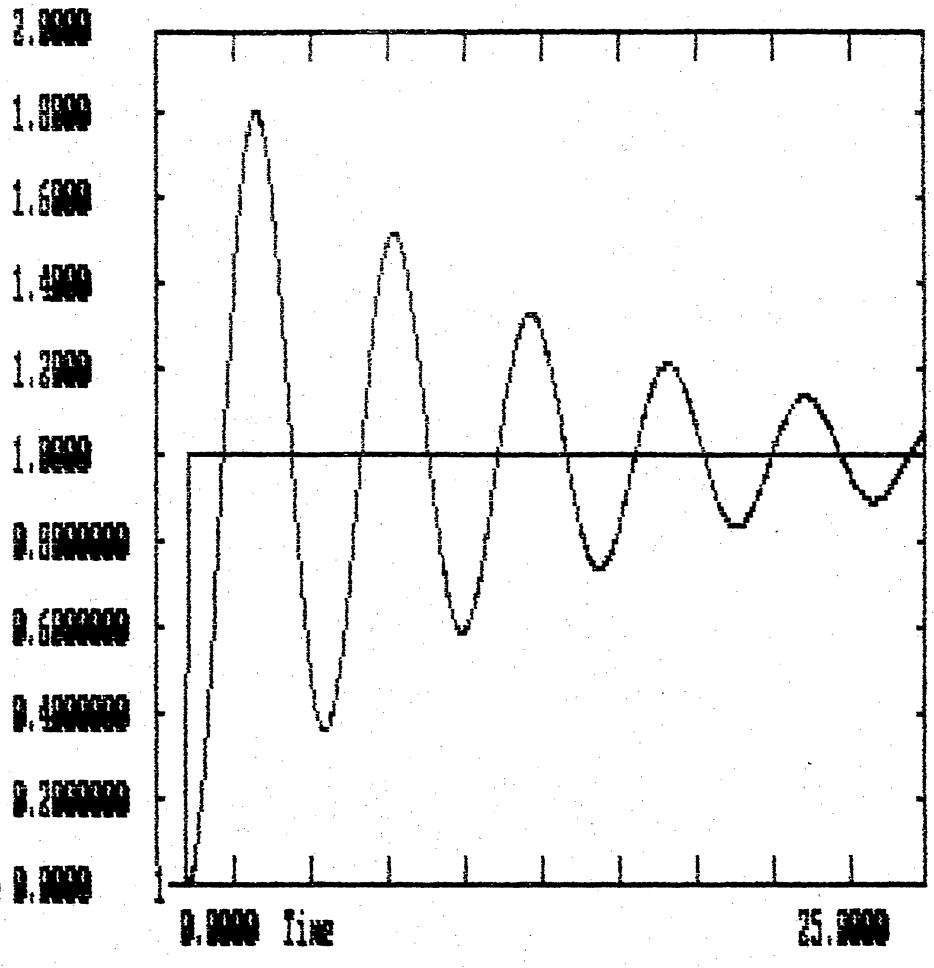


FIGURE 12. PI CONTROL OF A FIRST ORDER SYSTEM

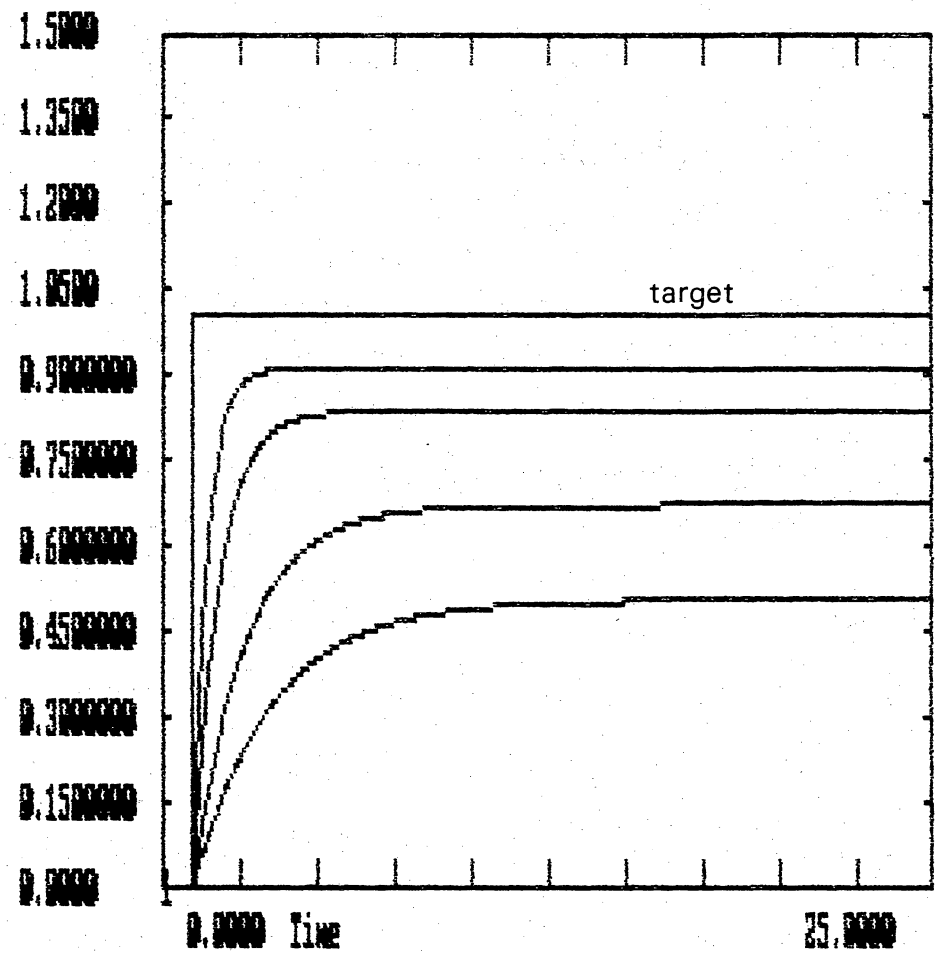


FIGURE 13. PD CONTROL OF A FIRST ORDER SYSTEM

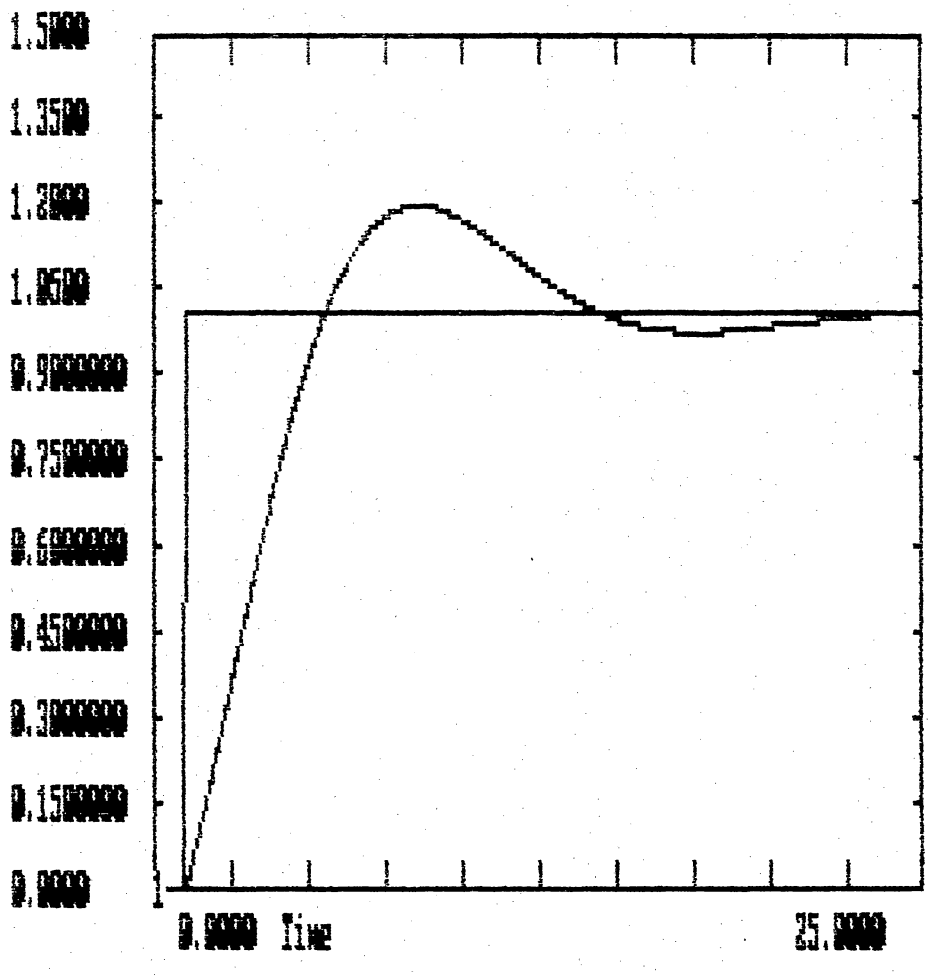


FIGURE 14. PID CONTROL OF A FIRST ORDER SYSTEM

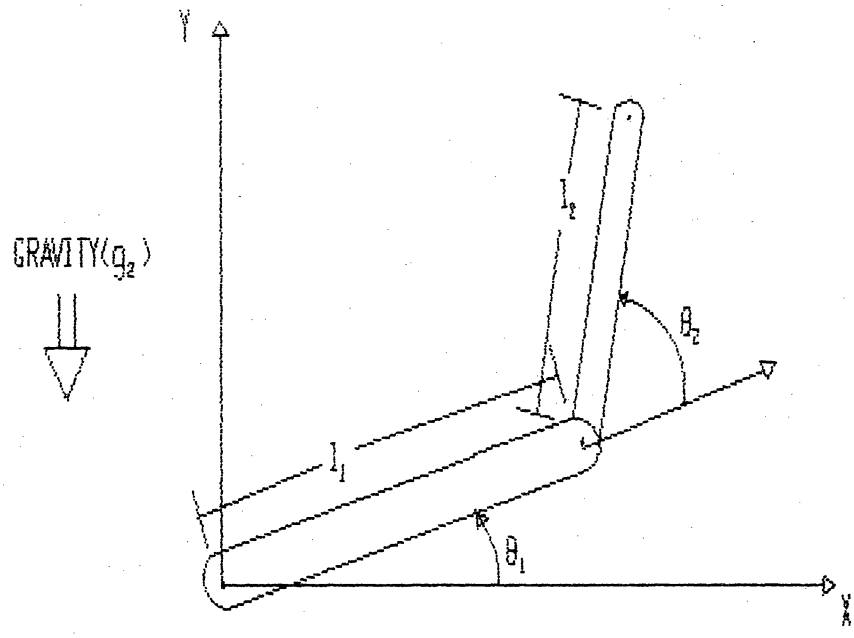


FIGURE 15. TWO LINK MANIPULATOR SYSTEM

```
3    ELBOW JOINT'S FUNCTION MODIFIER, (PRO. CONT.)
  -1000.0    -1000.0          0.0
 51
  -1000.0   10000.0    -180.0   20000.0    -170.0    9000.0
   -165.0    6500.0    -160.0    5200.0    -150.0    3700.0
   -140.0    2700.0    -135.0    2600.0    -130.0    2100.0
   -120.0    1600.0    -110.0    1300.0    -105.0    1200.0
   -100.0    1150.0     -90.0    1000.0     -80.0    1150.0
    -75.0    1200.0     -70.0    1300.0     -60.0    1600.0
    -50.0    2100.0     -45.0    2600.0     -40.0    2700.0
    -30.0    3700.0     -20.0    5200.0     -15.0    6500.0
    -10.0    9000.0       0.0   20000.0      10.0    9000.0
     15.0    6500.0      20.0    5200.0      30.0    3700.0
     40.0    2700.0      45.0    2600.0      50.0    2100.0
     60.0    1600.0      70.0    1300.0      75.0    1200.0
     80.0    1150.0      90.0    1000.0     100.0    1150.0
    105.0    1200.0     110.0    1300.0     120.0    1600.0
    130.0    2100.0     135.0    2600.0     140.0    2700.0
    150.0    3700.0     160.0    5200.0     165.0    6500.0
    170.0    9000.0     180.0   20000.0    1000.0   10000.0
```

```
23    ELBOW JOINT'S FUNCTION MODIFIER, (DERIV. CONT.)
  -1000.0    -1000.0
 51
  -1000.0     220.0    -180.0     260.0    -170.0     245.0
   -165.0     230.0    -160.0     220.0    -150.0     180.0
   -140.0     150.0    -135.0     130.0    -130.0     123.0
   -120.0     110.0    -110.0     100.0    -105.0      90.0
   -100.0      85.0     -90.0      70.0     -80.0      85.0
    -75.0      90.0     -70.0     100.0     -60.0     110.0
    -50.0     123.0     -45.0     130.0     -40.0     150.0
    -30.0     180.0     -20.0     220.0     -15.0     230.0
    -10.0     245.0       0.0     260.0      10.0     245.0
     15.0     230.0      20.0     220.0      30.0     180.0
     40.0     150.0      45.0     130.0      50.0     123.0
     60.0     110.0      70.0     100.0      75.0      90.0
     80.0      85.0      90.0      70.0     100.0      85.0
    105.0      90.0     110.0     100.0     120.0     110.0
    130.0     123.0     135.0     130.0     140.0     150.0
    150.0     180.0     160.0     220.0     165.0     230.0
    170.0     245.0     180.0     260.0    1000.0     220.0
```

FIGURE 16. EXAMPLES OF TWO "LOOK-UP" TABLES

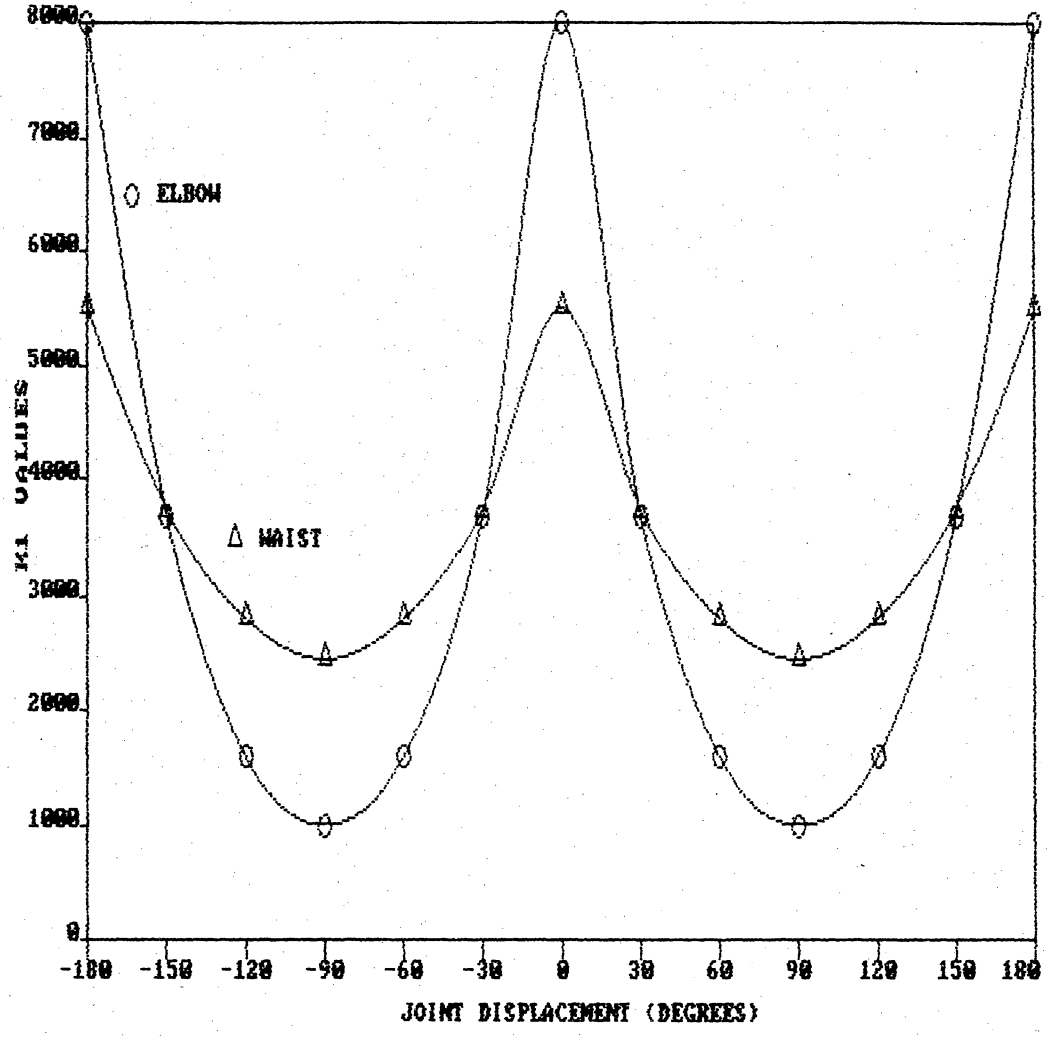


FIGURE 17. VARIATIONS IN $K_1$ FOR THE WAIST AND ELBOW JOINTS

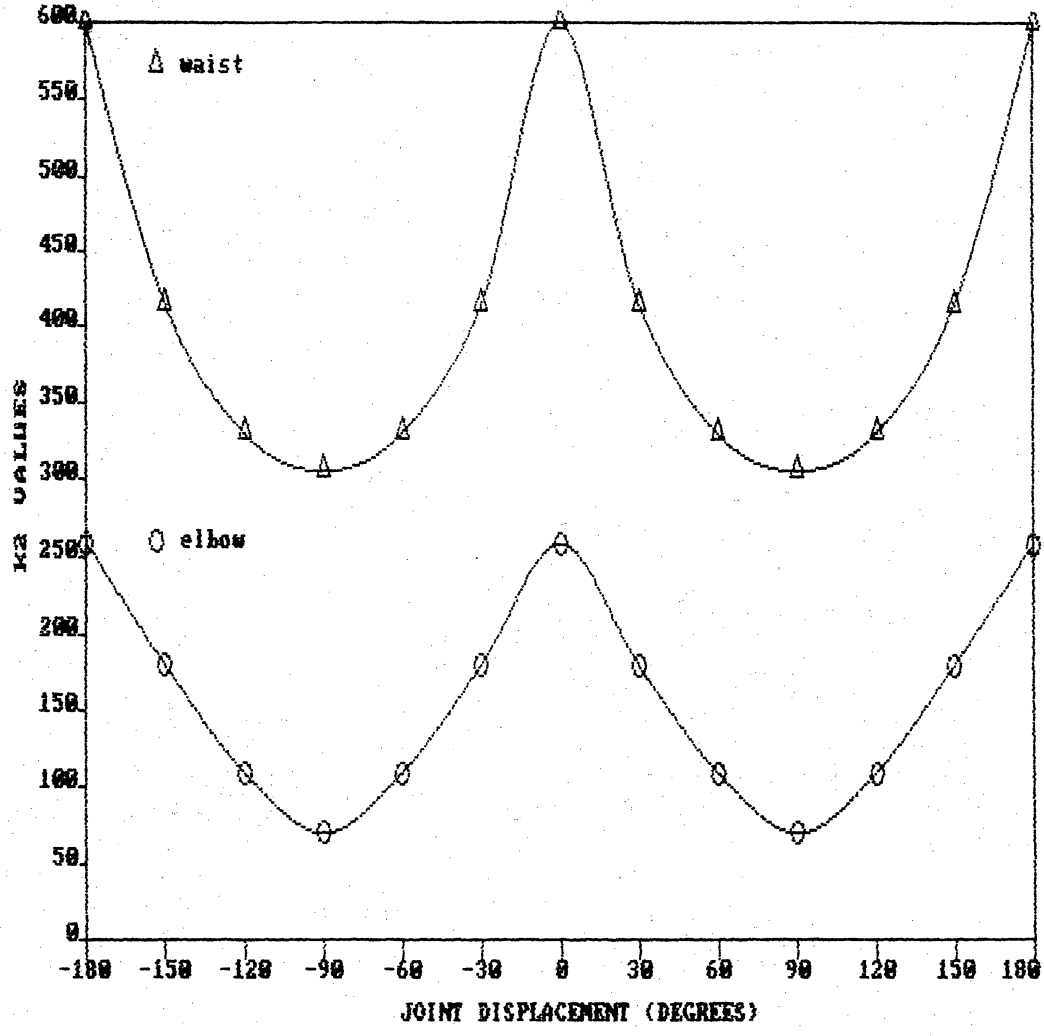


FIGURE 18. VARIATIONS IN $K_2$ FOR THE WAIST AND ELBOW JOINTS

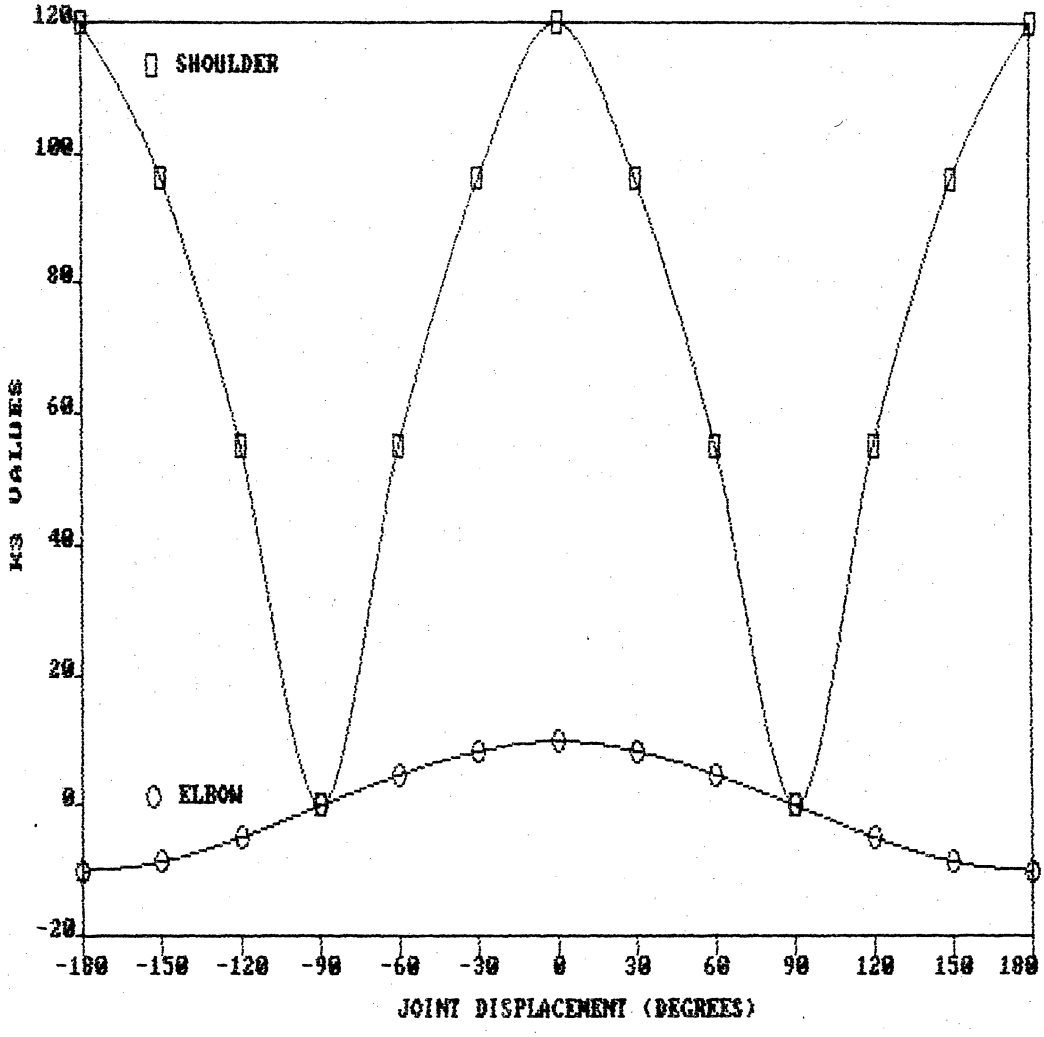


FIGURE 19. VARIATIONS IN $K_3$ FOR THE ELBOW AND SHOULDER JOINTS

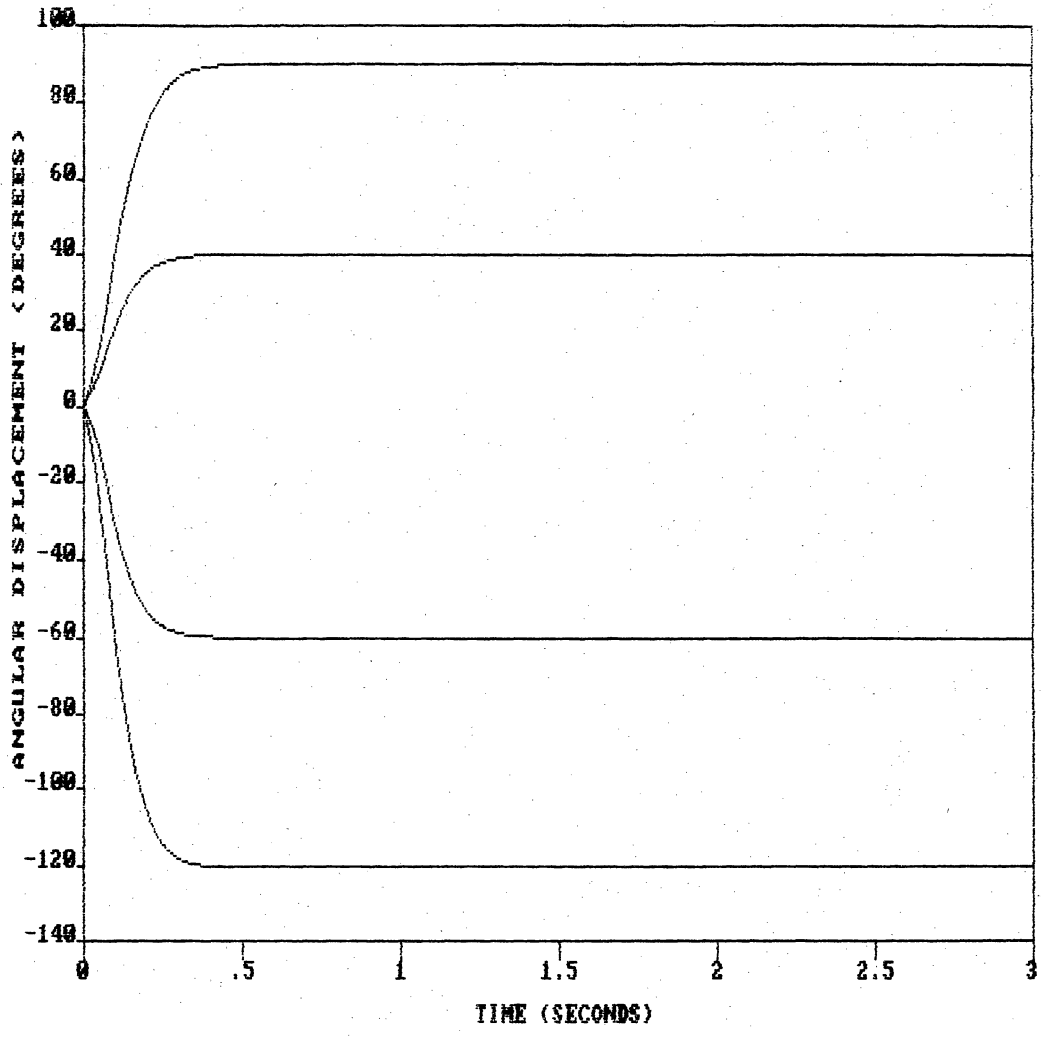


FIGURE 20. MOTION PROFILES INVOLVING SEPARATE WAIST JOINT ROTATIONS

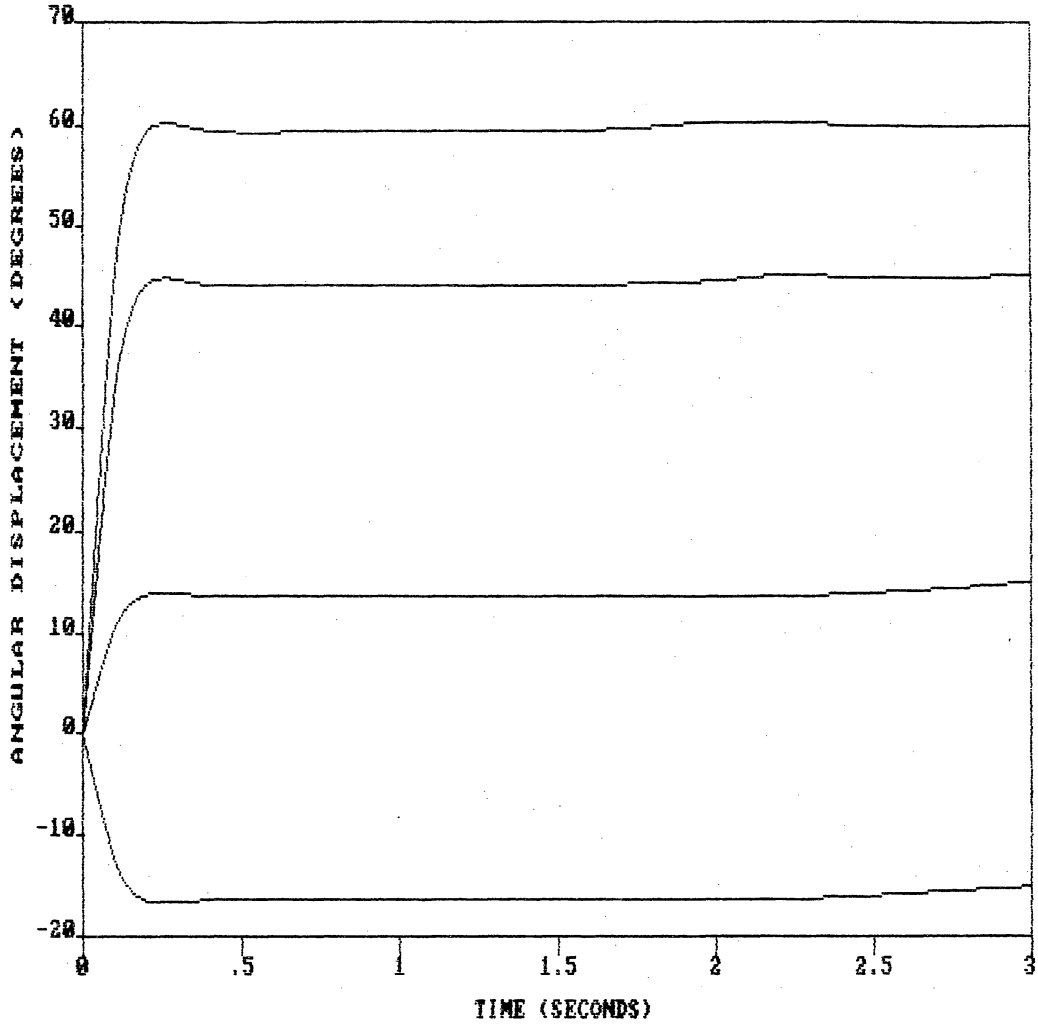


FIGURE 21. MOTION PROFILES INVOLVING SEPARATE SHOULDER JOINT ROTATIONS

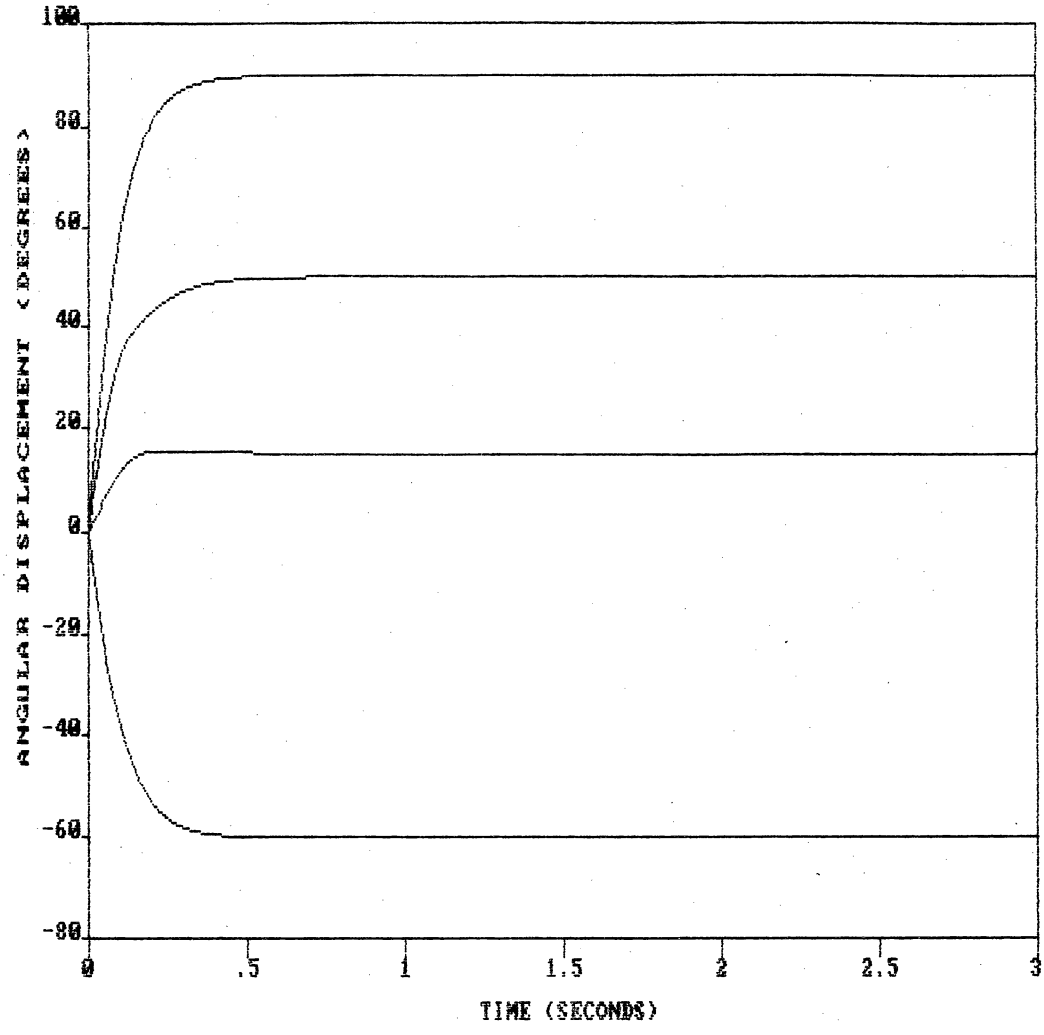


FIGURE 22. MOTION PROFILES INVOLVING SEPARATE FOREARM JOINT ROTATIONS

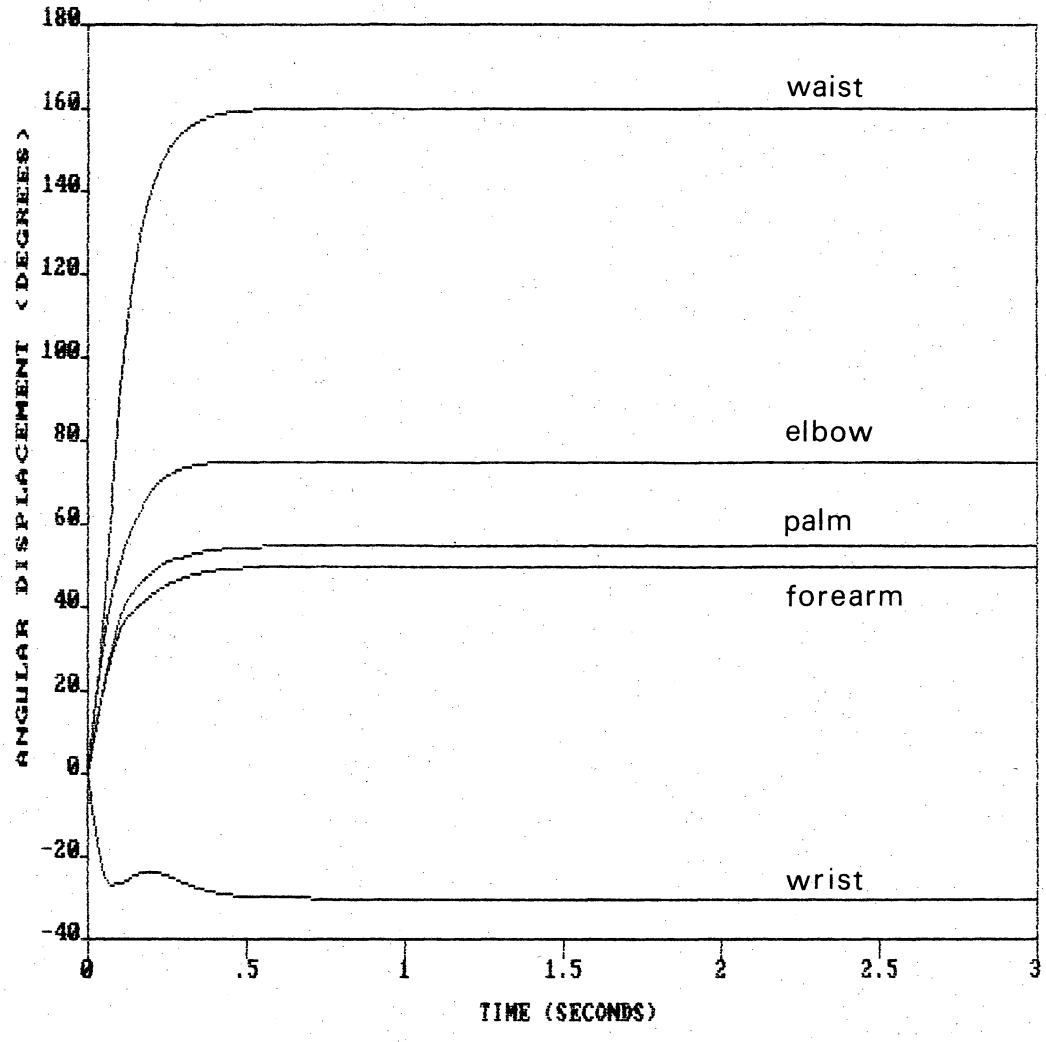


FIGURE 23. MULTI-JOINT ROTATIONS EXCLUDING THE SHOULDER JOINT

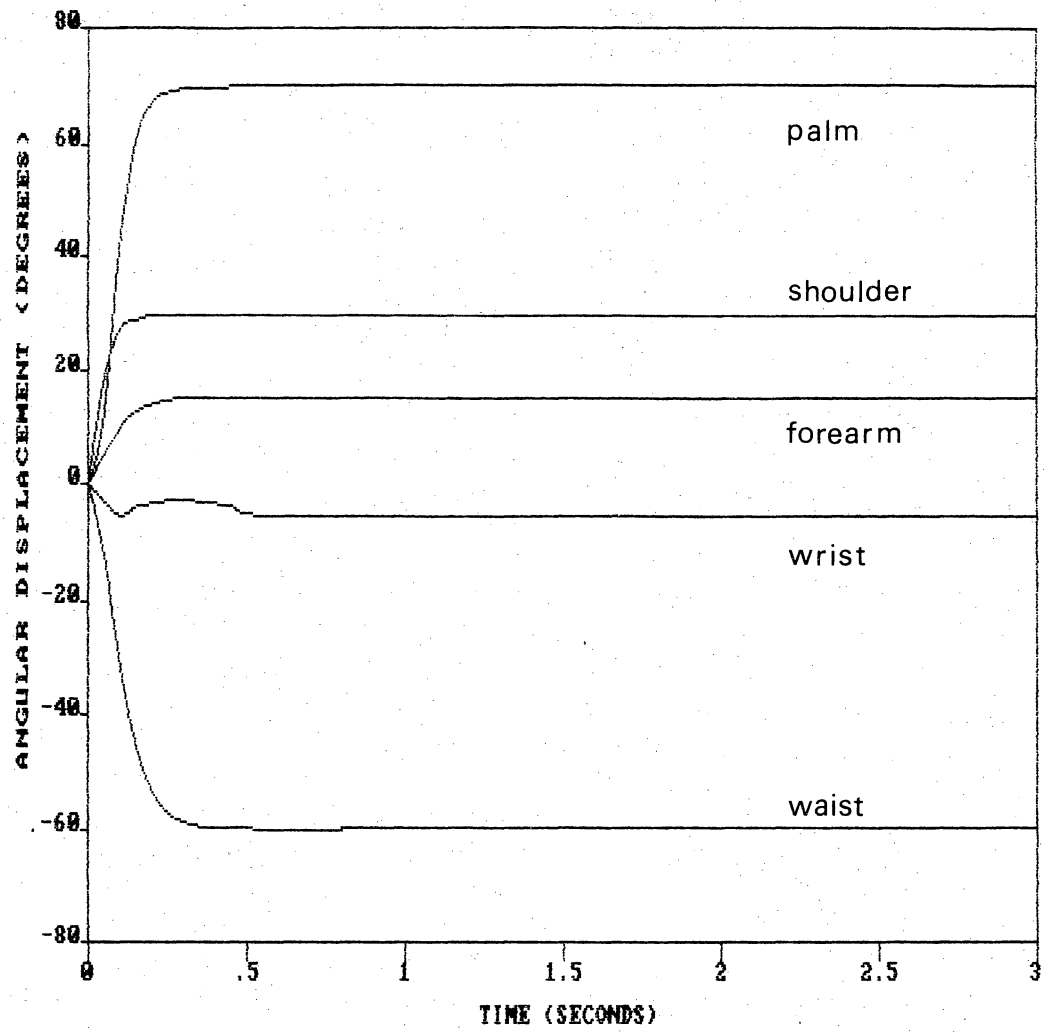


FIGURE 24. MULTI-JOINT ROTATIONS EXCLUDING THE ELBOW JOINT

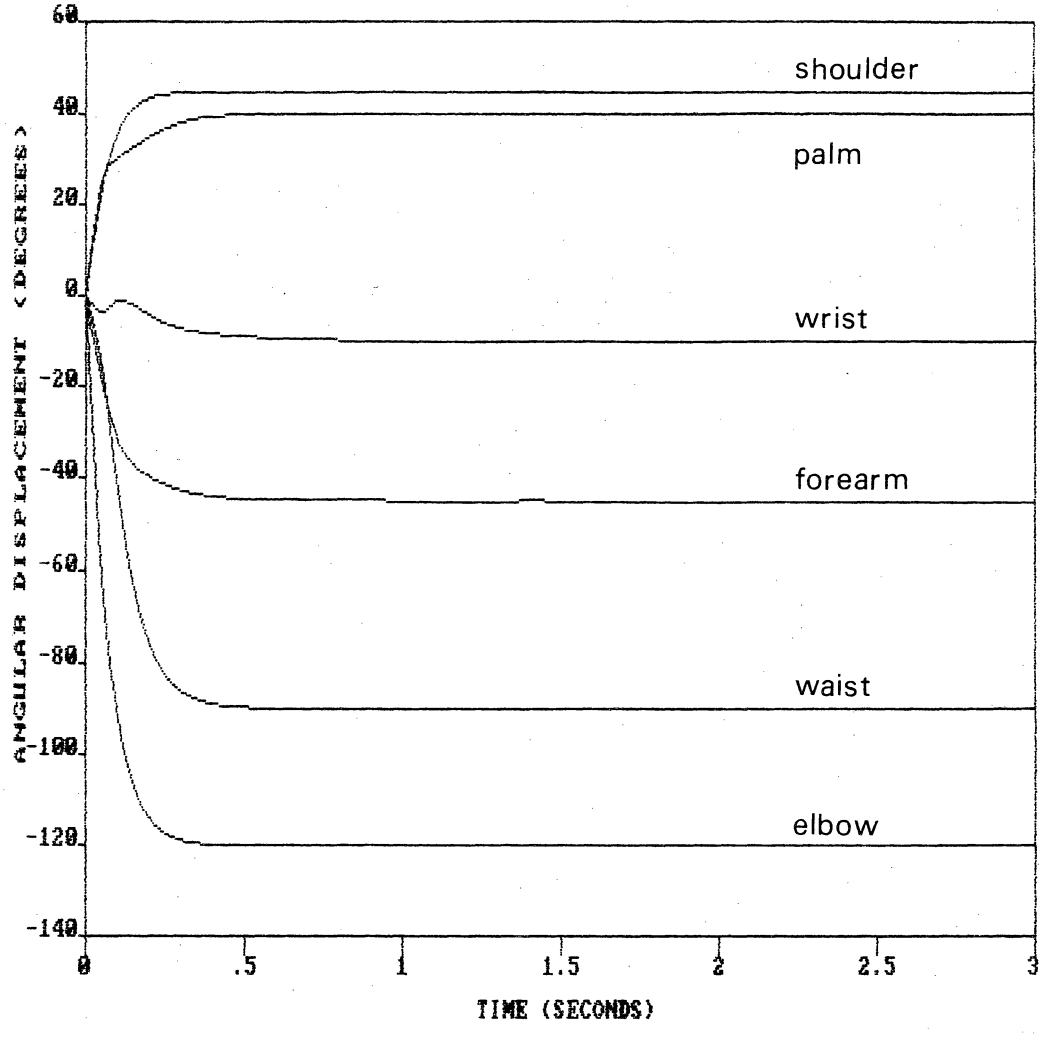


FIGURE 25. MULTI-JOINT ROTATIONS OF ALL SIX MODEL JOINTS

The vita has been removed from the scanned document