

THE AUTOMATIC OFF-LINE GENERATION OF WELDING ROBOT TRAJECTORIES
WITH EMPHASIS ON KINEMATIC FEASIBILITY AND
COLLISION DETECTION

by

RALPH O. BUCHAL

B.A.Sc., University of British Columbia, 1980

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

in

FACULTY OF GRADUATE STUDIES
Department of Mechanical Engineering

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
September 1984

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at THE UNIVERSITY OF BRITISH COLUMBIA, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the Head of my Department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Mechanical Engineering

THE UNIVERSITY OF BRITISH COLUMBIA
2075 Wesbrook Place
Vancouver, Canada
V6T 1W5

Date: October 10, 1984

ABSTRACT

This thesis defines and discusses the problems involved in automatic off-line programming of a robot welding workstation with a minimum of human intervention, and proposes some solutions. The work is motivated by the desire for faster and more powerful programming capabilities combined with reduced robot down-time for programming, resulting in increased flexibility, efficiency and productivity. The system proposed in the thesis uses data generated by commercial Computer Aided Design (CAD) systems. An "Expert Welder" software module selects the correct welder settings based on workpiece and workstation characteristics. Provision is made for the eventual incorporation of a real-time seam tracking system. The kinematics of a welding robot and workstation are considered in detail. The welding torch position and orientation relative to the robot are found from a series of known relative homogeneous coordinate transformations. Path feasibility is determined by the kinematic constraints of joint limits and the work envelope, as well as by physical interference between the arm and other objects in the workstation. The inverse kinematics solution is found and the solutions are checked against the joint limits and work envelope for each desired robot position. The sixth degree of freedom is redundant for welding, and this redundancy is used to search for feasible kinematic solutions.

A simplified interference detection algorithm is proposed. The workstation is modelled as a collection of solid polyhedra. The robot links sweep out volumes of space which are approximated by volumes bounded

by parametric ruled surfaces. A number of simple tests can be made to determine whether any intersections exist between the swept volumes and the stationary polyhedra, indicating interference.

Much of this fundamental work is incorporated in a preliminary interactive programming software package called AUTOP. The interference detection algorithm is demonstrated by a program called TESTIN.

ACKNOWLEDGEMENT

I would like to thank Professor Dale Cherchas for his enthusiastic help and guidance throughout my work. A large part of this thesis is based on work done for the Industrial Materials Research Institute, a division of the National Research Council under DSS Contract 12SD, 31155-2-5016.

TABLE OF CONTENTS

	Page
ABSTRACT	ii
ACKNOWLEDGEMENT	iv
LIST OF TABLES	x
LIST OF FIGURES	xii
NOMENCLATURE	xiv
LIST OF SYMBOLS	xv
1. INTRODUCTION	1
2. PROBLEM DEFINITION	6
3. DESCRIPTION OF WORKSTATION	9
3.1 Description of Workpiece Using CAD	9
3.2.1 Geometrical Description	10
3.1.2 Weld Seam Description	10
3.2 Description of Robot	11
3.2.1 Link Geometry and Dimensions	11
3.3 Description of Welding Machine	14
3.4 Description of Positioning Table	15
4. AUTOMATIC WELDING PARAMETER SELECTION USING AN "EXPERT WELDER" SOFTWARE MODULE	16
5. KINEMATICS	17
5.1 Review of Homogeneous Transformations	17
5.1.1 Homogeneous Coordinate Transformation	17
5.1.2 Serial Combination of Homogeneous Transformations	19
5.1.3 Serial Combination of Homogeneous Frames	19

	Page
5.2 Statement of Kinematics Problem	22
5.3 Intermediate Homogeneous Coordinate Frames	22
5.3.1 Table Frame Relative to Robot Base	22
5.3.2 Workpiece Frame Relative to Table	25
5.3.3 Weld Seam Frame Relative to Workpiece	25
5.3.4 Tool Frame Relative to Weld Seam	25
5.4 Calculation of the Tool Position and Orientation Relative to World Coordinates	28
6. PATH FEASIBILITY	29
6.1 Statement of Feasibility Problem	29
6.2 Kinematic Feasibility	29
6.2.1 Definition of Robot Joint Frames and Resulting Transformations	31
6.2.2 Inverse Kinematics Solution	32
6.2.2.1 General Approach to Inverse Kinematics Problem	32
6.2.2.2 PUMA 560 Solution	34
6.2.2.3 Discussion of PUMA Solutions	44
6.2.3 Comparison of Five Degree of Freedom and Six Degree of Freedom Robots	44
6.2.3.1 Solution for a Five Degree of Freedom Robot	44
6.2.3.2 ASEA IRB60 Solution	57
6.2.3.3 Redundancy of Sixth Joint for Welding .	59

	Page
6.2.4 Specification of Robot Coordinates	60
6.2.4.1 Calculation of PUMA Tool Location and Rotation Coordinates	62
6.3 Collision Avoidance	66
6.3.1 World Model	68
6.3.2 Development of Swept Volume	71
6.3.3 Interference Conditions	75
6.3.3.1 Condition 1 - Intersection of an Edge and a Parametric Surface	77
6.3.3.2 Condition 2 - Intersection of an Edge and a Cylinder	80
6.3.3.3 Condition 3 - Intersection of an Edge and a Polyhedral Face	83
6.3.3.4 Condition 4 - Intersection of a Cylinder and a Polyhedral Face	85
6.3.3.5 Condition 5 - Any Polyhedron is Totally Enclosed by a Swept Volume Bounded by Parametric Surfaces	88
6.4 Interference Detection Demonstration Program TESTIN ...	89
6.4.1 Functional Description of TESTIN	90
6.4.2 List and Description of all Subroutines used by TESTIN	90
6.4.3 Demonstration of TESTIN	96
6.5 Possible Improvements to the Algorithm	102

	Page
7. SOFTWARE DESIGN AND DEVELOPMENT	103
7.1 Design Approach	103
7.2 Features and Capabilities of AUTOP	104
7.3 General Functional Description of AUTOP	105
7.3.1 List and Description of all Subroutines called from AUTOP	107
7.4 Data Structures	116
7.4.1 Description of all I/O Files and their Formats	117
7.5 Graphical Depiction of Solids Using Panels	132
7.6 User's Guide to Running Program AUTOP	135
7.7 Demonstration of AUTOP and Results	137
7.8 Future Capabilities	150
7.8.1 Real-time Seam Tracking	150
7.8.2 CAD System Interface	150
7.8.3 Generation of High-Level Descriptive Program ..	150
8. CONCLUSIONS	154
8.1 Analysis and Evaluation of Work Completed	154
8.2 Suggestions for Further Work	155
REFERENCES	157

	Page
APPENDIX I SOFTWARE DOCUMENTATION	160
I.1 Implementation Specific Software	161
I.2 AUTOP Program Listing	162
I.3 CAD File Generating Program CADGEN Listing	266
I.4 TESTIN Program Listing	270

LIST OF TABLES

	Page
3.1 Manipulator Link Geometry File Format	14
6.1 PUMA Link Parameters	36
6.2 Five Degree of Freedom PUMA Link Parameters	45
7.1 File Name List FILIST	118
7.2 CAD File List CADLST	119
7.3 Processed Seam File List WPCLST	119
7.4 Robot Location Data File List ROBLST	119
7.5 Object Geometry Data File List OBJLST	120
7.6 Seam CAD File	120
7.7 Processed Seam Data File	121
7.8 Robot Location Data File	121
7.9 Object Geometry Data Files	122
7.10 System Configuration File WKSTAT	123
7.11 Example of File WKSTAT	124
7.12 Positioning Table Specification File TABLE	125
7.13 Example of File TABLE	126
7.14 Graphic System Initialization File GRAPH	127
7.15 Example of GRAPH	127
7.16 Workstation Setup File for Current Seam LOCSET	128
7.17 Example of LOCSET for seams 5,1,3	128
7.18 Current Table Position Data STOVAR	129
7.19 Example of STOVAR for 3 Degree of Freedom Table	129
7.20 Temporary Weld Parameter File WPARAM	130
7.21 Example of WPARAM for a Seam with 10 Locations	130

	Page
7.22 Fortran File Formats	131
7.23 Subroutines Accessing Data Files	132
7.24 Contents of Seam CAD File	142
7.25 Contents of Processed Seam Data File	143
7.26 Joint Angles Calculated during Kinematic Feasibility Search	144
7.27 Generated Robot Location Data	149
7.28 Task Precedence Table	153

LIST OF FIGURES

	Page
1.1 Simplified Hardware Schematic for an Automatically Programmed Robot Welding Workstation	5
3.1 Link Geometry and Frames for a Revolute Joint	12
3.2 Link Geometry and Frames for a Prismatic Joint	13
5.1 Coordinate Transformation	18
5.2 Serial Combination of Homogeneous Transformations	20
5.3 Homogeneous Coordinate Frame Defined by $\underline{n}$, $\underline{o}$, $\underline{a}$, $\underline{p}$	21
5.4 Series of Homogeneous Coordinate Frames	23
5.5 Weld Frame Relative to Workpiece	26
5.6 Tool Frame Relative to Weld Seam	27
6.1 PUMA Joint Frames and Link Parameters	35
6.2 Relative Geometry of PUMA Link Frames F_3 , F_4 , F_5 , F_6	38
6.3 Tool Frame Relative to Robot Tool Mounting Flange	46
6.4 Rotation of Tool Frame about Tool Axis	61
6.5 Definition of α , β , γ angles for the PUMA Arm	63
6.6 World Model	70
6.7 Cylindrical Approximation of a Manipulator Link	72
6.8 Parametric Approximation of Swept Volume	74
6.9 Interference Conditions	76
6.10 Algorithm for Interference Detection	78
6.11 Intersection of an Edge and a Parametric Surface	79
6.12 Intersection of an Edge and a Cylinder	81
6.13 Intersection of an Edge and a Polyhedral Face	84

	Page
6.14 Intersection of Cylinder End Face and Polyhedral Face	86
6.15 Intersection Condition 1 between Face and Swept Volume	98
6.16 Intersection Condition 3 between Face and Swept Volume	99
6.17 Intersection Condition 2 between Face and Swept Volume	100
6.18 Intersection Condition 1 between Face and Swept Volume Generated by Pure Translation	101
7.1 Software Structure and Data Flow of AUTOP	106
7.2 Simplified Flowchart of AUTOP	108
7.3 Relationship of Two Lines to View Point	134
7.4 Graphical Depiction of the Workpiece by AUTOP	139
7.5 New Workpiece Position after Interactively Changing Table Position	140
7.6 Workpiece Shown from a Different View Point	141
7.7 Simplified High-Level Task Description	152

NOMENCLATURE

- AUTOP: the automatic off-line robot programming software package.
- CAD: Computer-Aided Design.
- IGL: Tektronix Interactive Graphics Library software package.
- PUMA: the Unimation PUMA 560 industrial robot.
- TESTIN: an elementary interference testing demonstration program.

LIST OF SYMBOLS

$\underline{a}$	third basis vector of a frame relative to the robot.
a_i	length of the common normal between the joint i and joint $i-1$ axes of a robot.
$\underline{a_i}$	the common normal between the joint i and joint $i-1$ axes of a robot.
d_i	the distance between links i and $i-1$ of a robot.
F_B	the workpiece coordinate frame.
F_C	the instantaneous weld seam coordinate frame.
F_G	the tool or torch coordinate frame.
F_i	the i^{th} robot link coordinate frame.
F_R	the robot base coordinate frame.
F_T	the positioning table frame.
$\underline{H_{i-1,i}}$	a homogeneous coordinate transformation from frame F_i to frame F_{i-1} .
$\underline{J_i}$	the i^{th} joint axis of a robot.
$\underline{n}$	the first basis vector of a frame relative to the robot.
$\underline{o}$	the second basis vector of a frame relative to the robot.
$\underline{p}$	the position of a frame origin relative to the robot.
$\underline{q}$	the position of the robot wrist center relative to the robot.
$\underline{r_b}$	a point lying on a bounding parametric surface.
$\underline{r_p}$	a point on a parent parametric surface.
$\underline{T_n}$	transformation matrix giving torch position as a function of n joint variables.

$\underline{u}_0$	one of the vectors defining the boundaries of a parametric surface patch.
$\underline{u}_1$	one of the vectors defining the boundaries of a parametric surface patch.
$\underline{v}_0$	one of the vectors defining the boundaries of a parametric surface patch.
$\underline{x}_i$	x-axis of a frame F_i .
$\underline{y}_i$	y-axis of a frame F_i .
$\underline{z}_i$	z-axis of a frame F_i .
α	longitudinal angular offset of torch from seam.
α_i	the twist angle of the i th robot link
β	lateral angular offset of torch from seam.
θ_i	the i th joint variable of a robot.
λ_u	a parameter of a parametric surface.
λ_v	a parameter of a parametric surface.
λ_w	a parameter of a parametric volume.
ρ	torch stick-out gap.

1. INTRODUCTION

As technology progresses, robots are becoming an increasingly important tool in industry. Continual progress in programming and control methods combined with the incorporation of sensory feedback promise to vastly increase the scope of applications for robots in the factory. A great deal of fundamental research remains to be done, however, before the full potential of robots can be realized.

Although robots have already had a tremendous impact in manufacturing, they are still very limited in their capabilities. Programming of a robot can be time consuming and difficult for tasks of even moderate complexity. Typically, a robot is programmed by physically guiding it through its motions under manual control. Positions are recorded along the way so that the motions can be repeated or played-back as often as desired. The user has little or no capability to edit or modify the resulting program. The program is completely developed using the robot to define locations, and existing Computer-Aided Design (CAD) data is not utilized. The programming is done on-line, removing the robot from production for possibly unreasonable lengths of time. These shortcomings are particularly acute for small batch production where the robot must be reprogrammed frequently.

Since most commercial robots have no sensory capabilities, they cannot adapt to random variations in the tasks they perform. This necessitates very strict structuring and control of the workstation to insure a high degree of accuracy and repeatability for the process. Process variations can occur due to variations in workpiece positioning,

and variations between different samples of the workpiece. The requirement for exact repeatability could be greatly relaxed if the robot could adapt to process variations using real-time sensory feedback.

We can formulate two main objectives for the improvement of robot performance. First, we would like to simplify and speed-up the robot programming process, and perform as much of the programming as possible off-line. The method must be easy to use, fast and flexible. Second, we must incorporate sensory feedback and real-time correction of process variations. This thesis considers the first problem only.

The problem of off-line programming has been studied by a number of investigators, but a commercially feasible system has not yet been demonstrated. Two basic approaches have so far been demonstrated for generation of a robot program, interactive simulation and explicit task description using high-level languages.

In the first approach, simulation of the process is done using interactive graphics. The robot and workstation are depicted graphically, and the operator interactively generates a program which is simulated on a graphics display. It is difficult to specify locations exactly with this method, and there may be problems visualizing the scene. The demonstrations described in the literature [Sata, Fumihiko and Akio 1981; Ambler, Popplestone and Kempf 1982; Sjolund and Donath 1983] provide only limited software tools to aid the user.

A second and more promising approach to off-line programming is offered by high-level robot languages. These languages allow the user to fully describe the robot task in a descriptive, natural way. Locations can be specified explicitly, or can be referred to a mathematical world

model. Many languages have been devised by robotics researchers, but off-line programming has been discussed more than demonstrated. Ambler, Popplestone and Kempf [1982] generated an explicit program in the VAL language for a PUMA robot, including all of the location data, using the high-level language RAPT to describe a task. Their experience showed a need for graphical simulation of the program for verification and debugging. Other existing high-level robot languages are summarized in a number of recent papers [Gruver et al 1983; Kempf 1982; Tarvin 1980].

The off-line programming problem encompasses a number of difficult sub-problems. Two problems which are considered in detail in this thesis concern kinematic feasibility and collision detection.

The kinematic feasibility of a proposed robot trajectory must be automatically checked to ensure that a valid set of joint angle solutions exists. A robot position is kinematically feasible if an inverse kinematic solution exists for the joint angles, and if all the joint angles are within their physical limits. We have found by experience that even when programming a robot on-line it is difficult to avoid paths which lead to a joint angle limit being exceeded, particularly when moving in straight lines.

To verify kinematic feasibility, we must solve the inverse kinematic problems to find the joint angles at each point along a proposed trajectory. The inverse kinematic problem has been well studied by many investigators (see Section 6.2 for a review). An explicit solution for most real robots can be derived by matrix algebra methods described by Paul [1981]. In order to increase the chance of finding a feasible solution for a given trajectory we will exploit the redundancy of the

sixth degree of freedom in welding to search for a feasible set of joint solutions at each point on the trajectory.

The collision detection problem is one part of the far more difficult problem of path planning to avoid obstacles. Many algorithms have been proposed for finding intersections between stationary geometric solids and between stationary and moving solids. Previous work in geometric modeling and interference detection is reviewed in Section 6.3. The more difficult path planning problem has not yet been solved in a satisfactory way, although some promising strategies have been proposed for simple non-redundant systems. No work has yet emerged in the literature dealing with path planning for systems with redundancy.

This thesis discusses the problems associated with developing an integrated off-line welding robot programming system, and offers some solutions. The solutions are incorporated into a functional software package called AUTOP. Particular emphasis is placed on the kinematic feasibility problem, and on elementary methods for collision detection. Many other vital issues are beyond the scope of this thesis and are discussed only briefly. Fig. 1.1 illustrates the required hardware structure for an automatically programmed welding workstation.

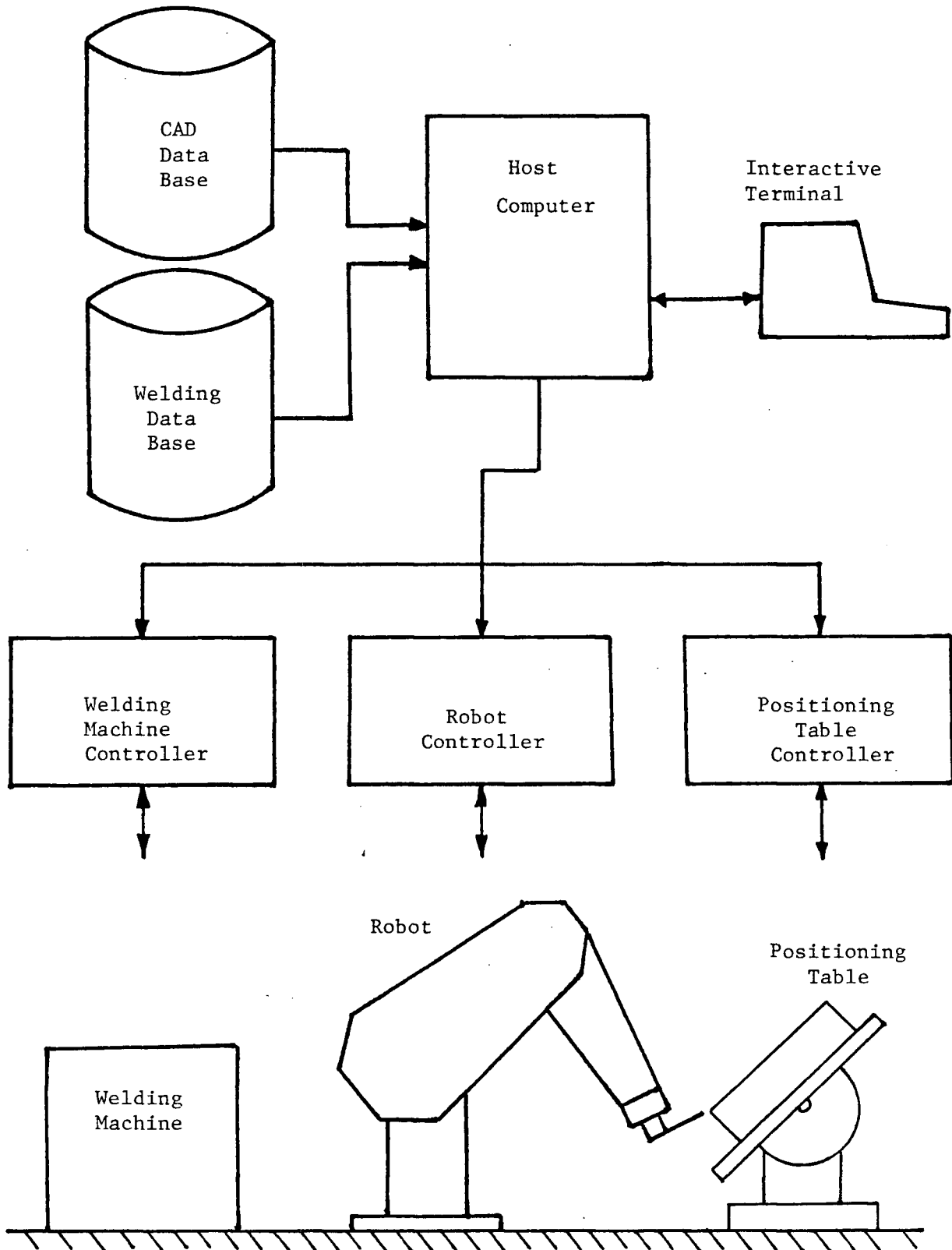


Figure 1.1 Simplified Hardware Schematic for an Automatically Programmed Robot Welding Workstation.

2. PROBLEM DEFINITION

The ultimate objective of the work described in this thesis is the development of a system for automatically programming a welding robot using data derived from CAD files. An expert welder data base will provide the correct welding parameter values, and a real-time seam tracking system will correct for real-world variations. The structure of the resulting program will be hierarchical, with the highest level controlling the entire workstation as an integrated unit.

We can isolate a number of sub-problems which must be solved before an automatic programming system can be realized:

1. Choice of World Model and Description

We must choose a way to represent the workpiece explicitly, without ambiguity. Important information includes seam location data, workpiece geometry and welding parameters. We must also represent the workstation, including the robot, positioning table and welding machine.

2. Generation of a Welding Procedure

An acceptable and feasible welding procedure must be generated. The procedure includes the order in which seams are welded, workpiece repositioning, inter-weld paths, etc.

3. Automatic Generation of Welding Parameter Values

Welding output parameters such as current and speed can be

determined automatically if all relevant workpiece and welding machine characteristics are known.

4. Generation of Welding Torch Trajectories

We must generate torch trajectory data including torch position and orientation along each seam in terms of robot coordinates.

5. Path Feasibility Testing

The calculated torch trajectory may not be feasible due to robot work envelope and joint limit constraints. Also, interference may occur between the robot and other objects in the workstation.

6. Generation of High-Level Program to Direct and Coordinate the Welding Process

We need a descriptive, high-level method for controlling and coordinating the different devices in the workstation. Welding requires a carefully controlled interaction between the robot, positioning table and welding machine.

7. Program Modification in Response to Sensory Feedback

Due to real-world variations and uncertainties in positioning and welding, a seam tracking system based on real-time sensory feedback is an important capability.

8. Hardware and Software Interfaces

Hardware and software interfaces must be designed between the

various components in the workstation. A host computer must be able to communicate with the robot controller, positioning table controller and welding machine.

These problems will be discussed and some solutions will be proposed.

3. DESCRIPTION OF WORKSTATION

We must describe all the important characteristics of the workstation in a concise way. The workstation is composed of the workpiece, positioning table, welding machine, robot and any other peripheral equipment.

3.1 Description of Workpiece Using CAD

A complete description of the workpiece is generated by an external CAD system. The CAD description must contain the following information:

1. Geometry

- a) dimensions and shape of object
- b) weld seam trajectories and surface normals adjacent to seams in digitized form.

2. Welding Information

- a) material
- b) thickness
- c) weld joint types
- d) weld cross sectional area.

The CAD file must be translated from the CAD output format into the specified input format for the software system, which we will call AUTOP. This translation is required so that any external CAD system may be used. Details of the CAD system need not be known at this point, as long as the specified information is generated in some form.

3.1.1 Geometrical Description

We wish to represent the geometry of the workpiece by an approximating polyhedron, whose vertices, edges and faces are stored in a hierarchical, file-based data structure. At the top of the hierarchy is a list of the object faces, including their surface normals, distance from origin and number of vertices. Each face record contains the coordinates of the face vertices in any sequential order around the face. Each vertex of the polyhedron is shared by at least three faces. A polyhedral representation is well suited to graphical depiction and interference detection.

3.1.2 Weld Seam Description

Seam data is also stored in a hierarchical, file-based structure. A list of seam identifiers is stored in a file. Each entry points to a corresponding seam data file. The head of each seam data file contains welding data relevant to that seam, including material thicknesses, seam types and weld cross sectional area. The rest of the file contains digitized coordinates of the seam trajectory and adjacent surface normals relative to a reference frame fixed on the workpiece.

Each seam defined in the CAD file is treated as a single, indivisible entity by AUTOP. Long seams should be broken down into shorter segments which can be treated separately. For example, a circular weld around the circumference of a cylinder should be divided into two or four segments so that the workpiece can be repositioned before welding each segment.

3.2 Description of Robot

AUTOP requires a careful and complete specification of the robot in use. Data which are required are:

1. Number of degrees of freedom.
2. Complete link geometry and dimensions
3. Joint limits.
4. Tool geometry relative to tool mounting flange.
5. Default robot configuration.
6. Position of robot base relative to the world coordinate frame, stored as a homogeneous coordinate transformation.

3.2.1 Link Geometry and Dimensions

We will follow the conventions of Denavit and Hartenberg [1955] in describing link geometry and dimensions. We can define an n degree of freedom robot as a series of n rigid links connected by n joints (Figs. 3.1, 3.2). The i^{th} link is characterized by a length a_i and a twist angle α_i . a_i is the length of the common normal $\underline{a}_i$ between the two joint axes, and α_i is the angle between the axes in a plane perpendicular to $\underline{a}_i$.

The relative displacement between $\underline{a}_{i-1}$ and $\underline{a}_i$ at the joint i axis is d_i , the distance between the links. θ_i is the angular displacement of $\underline{a}_i$ and $\underline{a}_{i-1}$ about the joint i axis and is called the angle between the links.

Parameters a_i and α_i are fixed for a given link. For a prismatic joint where motion is along the joint axis, d_i varies and θ_i is fixed. For a revolute joint, d_i is fixed and θ_i varies.

Given these parameters, we can completely define the kinematics of

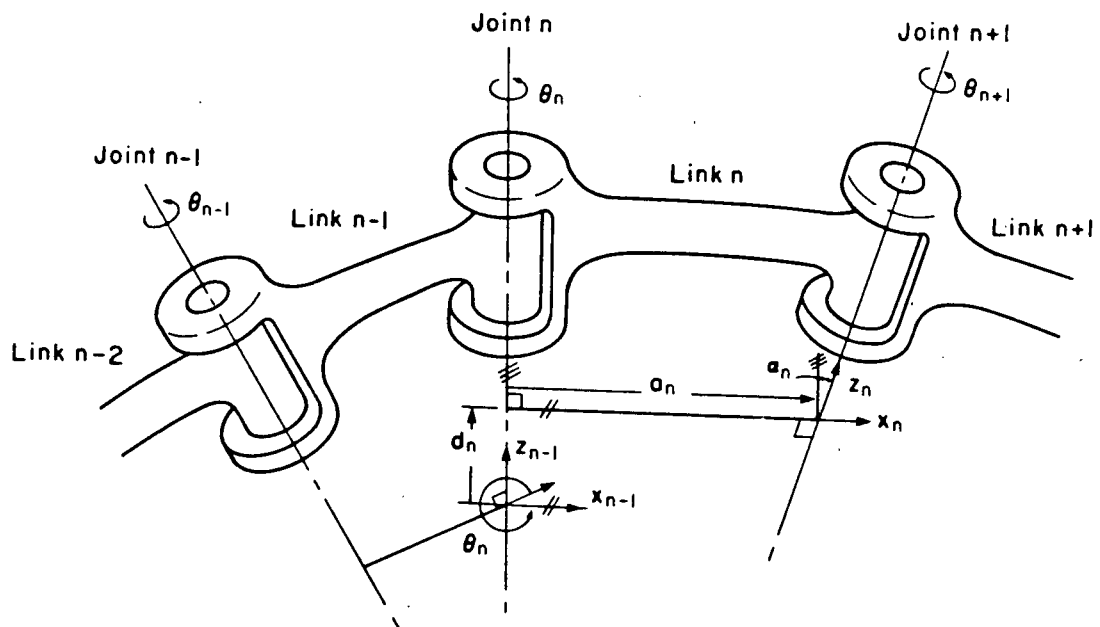


Figure 3.1 Link geometry and frames for a Revolute Joint.

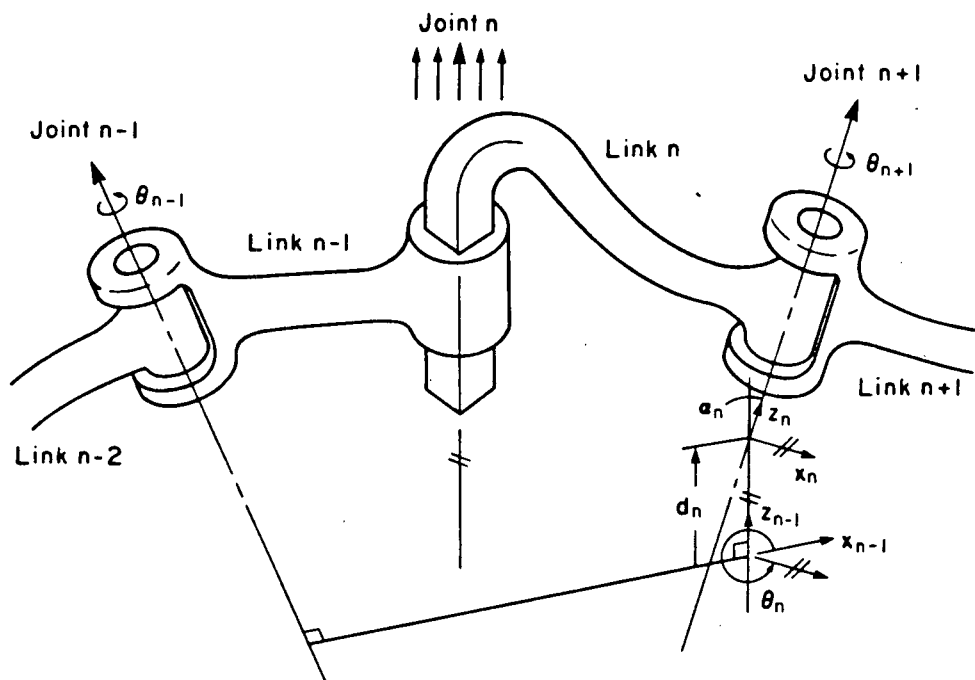


Figure 3.2 Link geometry and frames for a Prismatic Joint

the robot. The parameters are stored in a file with the format of Table 3.1.

Table 3.1 - Manipulator Link Geometry File Format

link #	id	α	a	d	θ
1					
2					
:					
n					

id = 1 revolute

id = 2 prismatic

3.3 Description of Welding Machine

The welder characteristics and specifications are stored in a data file where they are easily referenced by the Expert Welder module. The file contains all fixed settings and characteristics of the welder which are relevant to the welding process. The file might include some or all of the following:

- a) voltage setting
- b) wire diameter
- c) wire type
- d) wire feed rate

- e) gas type, flow rate
- f) polarity
- g) minimum, maximum current (if under program control), or current setting.

The contents of the file depend on the particular welder in use.

The format has not yet been completely specified.

3.4 Description of Positioning Table

AUTOP requires a complete description of the positioning table, which is stored in a data file. The following information is stored:

- a) number of degrees of freedom
- b) link geometry and dimensions
- c) joint types (revolute or prismatic)
- d) joint motion (continuous or discrete steps)
- e) joint limits
- f) first joint location relative to world coordinates
- g) table coordinate frame location relative to last joint frame.

The table is treated as analogous to an n degree of freedom manipulator, so the link geometry and dimension parameters have the same definitions as for a robot.

We can easily accommodate a totally different type of positioning table simply by revising the entries in this data file.

4. AUTOMATIC WELDING PARAMETER SELECTION USING AN
"EXPERT WELDER" SOFTWARE MODULE

A welding data base and interactive software module have been developed by Dr. F. Sassani of the Department of Mechanical Engineering, The University of British Columbia. A detailed description of this system, henceforth referred to as the Expert Welder, can be found in a previous report [Buchal et al 1984]. Essentially, the Expert Welder module uses a data base derived from standard welding practice to choose appropriate welding parameter settings as a function of given workpiece and fixed welder parameters.

5. KINEMATICS

5.1 Review of Homogeneous Transformations

5.1.1 Homogeneous Coordinate Transformation

It is often useful to transform the coordinates of a point in space from one coordinate frame to another (Fig. 5.1). A rotational transformation is required if corresponding axes of two coordinate frames F_{i-1} , F_i are not parallel. A translational transformation is required if the frame origins are not coincident. In general, both a rotation and a translation are required.

A homogeneous coordinate transformation allows us to perform both rotation and translation in a single operation using a 4×4 homogeneous transformation matrix. For example, a vector $\underline{r}$ expressed relative to and projected onto a coordinate frame F_i can be transformed into a vector $\underline{R}$ relative to and projected onto a frame F_{i-1} by the following matrix multiplication:

$$\underline{R} = \underline{H}_{i-1,i} \underline{r} , \quad (5.1)$$

where $\underline{H}_{i-1,i}$ is a 4×4 homogeneous transformation matrix relating frame F_i to Frame F_{i-1} . The vectors $\underline{R}$ and $\underline{r}$ are homogeneous 4×1 column vectors, where the first three elements are the cartesian coordinates and the last element is a scaling factor. For our purposes this scaling factor is always set to 1.

$\underline{H}_{i-1,i}$ has the following form:

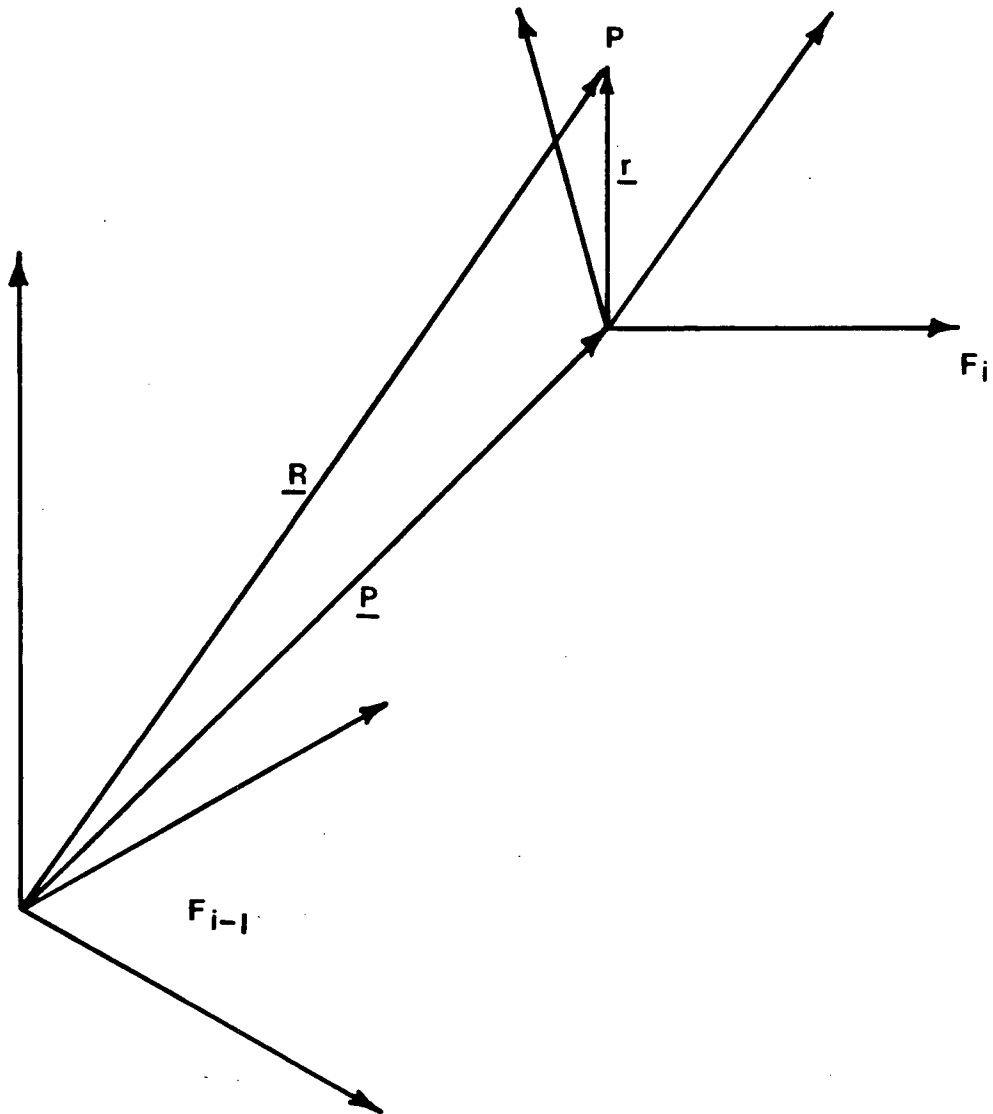


Figure 5.1 Coordinate Transformation.

$$\underline{H}_{i-1,i} = \begin{array}{c|c} \begin{array}{c} \text{rotation} \\ 3 \times 3 \\ \text{rotation matrix} \end{array} & \begin{array}{c} \text{translation} \\ P_x \\ P_y \\ P_z \end{array} \\ \hline \begin{array}{ccc} 0 & 0 & 0 \end{array} & 1 \end{array} \quad (5.2)$$

A more detailed description of homogeneous transformations can be found in other references [Paul 1981].

5.1.2 Serial Combination of Homogeneous Transformations

We can combine homogeneous transforms in series as shown in Fig. 5.2.

Suppose we want to find $\underline{R}$ given $\underline{r}$. If we know $\underline{H}_{1,2}$ and $\underline{H}_{2,3}$ we can find $\underline{H}_{1,3}$ as the product of $\underline{H}_{1,2}$ and $\underline{H}_{2,3}$:

$$\underline{H}_{1,3} = \underline{H}_{1,2} \underline{H}_{2,3} \quad (5.3)$$

$$\text{Thus,} \quad \underline{R} = \underline{H}_{1,3} \underline{r} \quad (5.4)$$

5.1.3 Serial Combination of Homogeneous Frames

We can show that the homogeneous transformation $\underline{H}_{i-1,i}$ defines a coordinate frame F_i relative to F_{i-1} .

$\underline{H}_{i-1,i}$ can be written as

$$\underline{H}_{i-1,i} = [\underline{n}, \underline{o}, \underline{a}, \underline{p}] \quad (5.5)$$

where $\underline{n}, \underline{o}, \underline{a}, \underline{p}$ are 4×1 column vectors. $\underline{n}, \underline{o}, \underline{a}$ define the basis vectors of F_i relative to F_{i-1} , and $\underline{p}$ is the origin of F_i relative to F_{i-1} , as shown in Fig. 5.3.

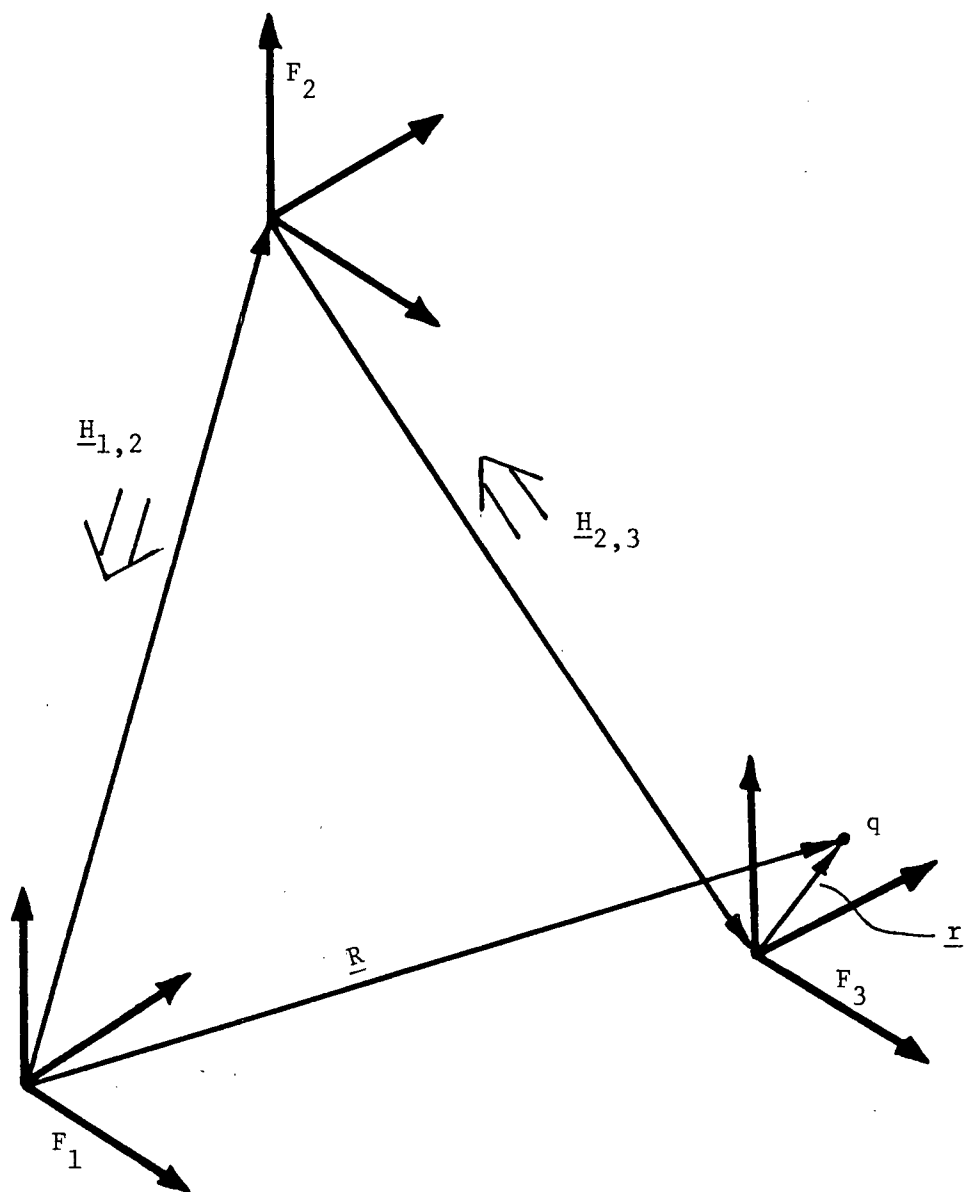


Figure 5.2 Serial Combination of Homogeneous Transformations.

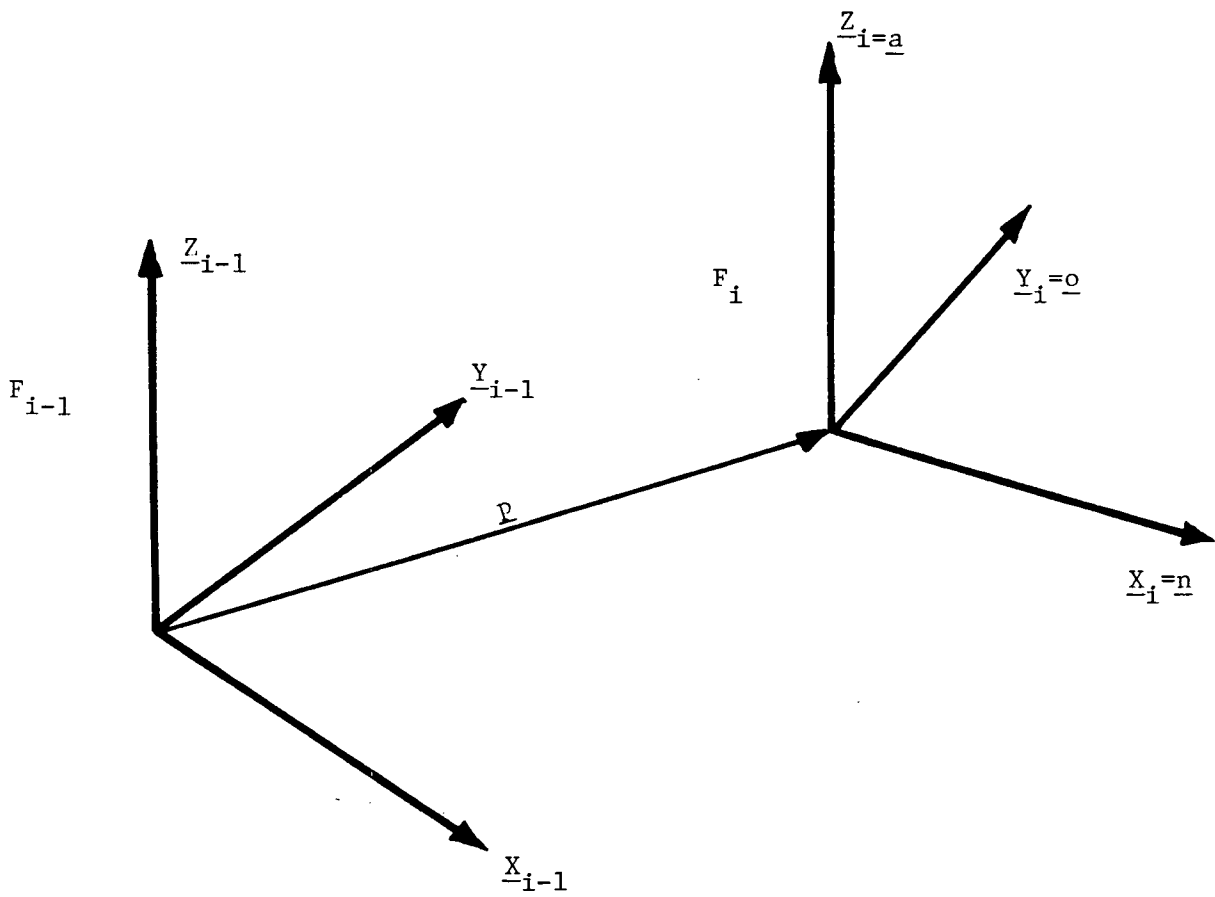


Figure 5.3 Homogeneous Coordinate Frame defined by $\underline{n}$, $\underline{o}$, $\underline{a}$, $\underline{p}$.

Note that if we know the position and orientation of Frame F_i relative to F_{i-1} , $\underline{H}_{i-1,i}$ is also known.

5.2 Statement of Kinematics Problem

To generate a robot program, we must find the tool position and orientation at each point along the weld seam trajectory. Mathematically stated, we wish to find the homogeneous coordinate frame F_G of the tool relative to and projected onto the robot base coordinate frame F_R . F_G is defined by the homogeneous coordinate transformation $\underline{H}_{R,G}$ which takes us from F_G to F_R . Thus, we wish to find $\underline{H}_{R,G}$. We can define a string of intermediate coordinate frames F_i for which $\underline{H}_{i-1,i}$ can be calculated, as shown in Fig. 5.4.

We define the following homogeneous coordinate frames:

- a) F_R - robot base frame.
- b) F_T - positioning table frame.
- c) F_B - workpiece frame.
- d) F_C - weld seam frame.
- e) F_G - tool frame.

We can solve for the homogeneous transformations $\underline{H}_{R,T}$, $\underline{H}_{T,B}$, $\underline{H}_{B,C}$, $\underline{H}_{C,G}$. We can then find $\underline{H}_{R,G}$.

$$\underline{H}_{R,G} = \underline{H}_{R,T} \underline{H}_{T,B} \underline{H}_{B,C} \underline{H}_{C,G} \quad (5.6)$$

5.3 Intermediate Homogeneous Coordinate Frames

5.3.1 Table Frame Relative to Robot Base

We wish to define the position of the table frame F_T relative to

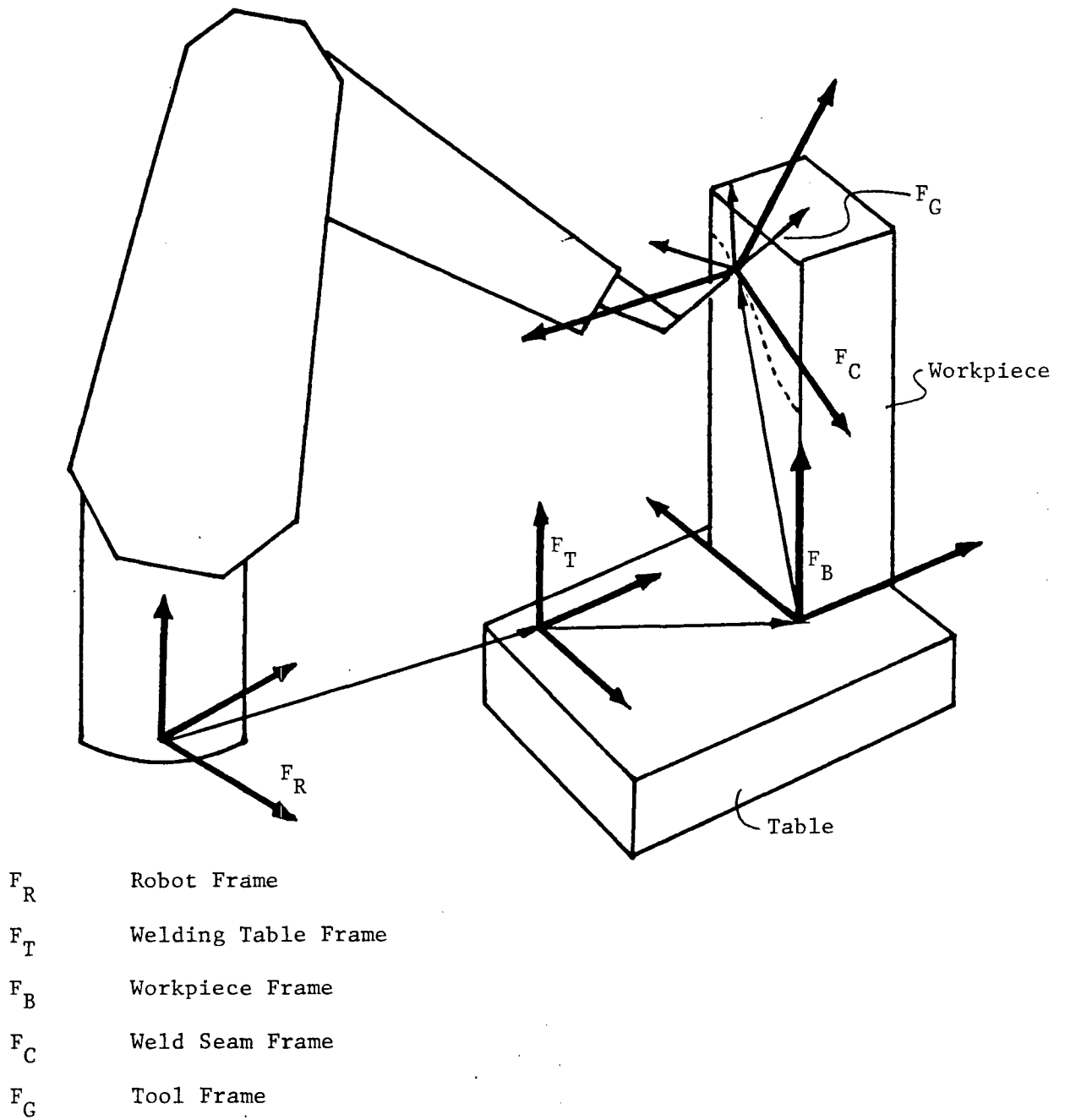


Figure 5.4 Series of Homogeneous Coordinate Frames.

the robot base coordinate frame F_R . The problem is to find the transformation $H_{R,T}$.

Suppose we analyse the table as an n-link system, similar to a robot. We can then define the transformation $H_{i-1,i}$ from link i-1 to link i by the Denavit-Hartenberg [1955] equations (see Section 6.2.2.1 for derivation):

$$H_{i-1,i} = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cos \alpha_i & \sin \theta_i \sin \alpha_i & a_i \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cos \alpha_i & -\cos \theta_i \sin \alpha_i & a_i \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.7)$$

where θ_i is the i^{th} joint angle and α_i and a_i are fixed link geometry parameters. Either θ_i (revolute joint) or d_i (prismatic joint) is variable.

We can combine the transformation for all n links as

$$H_{0,n} = H_{0,1} H_{1,2} \cdots H_{n-1,n} \quad (5.8)$$

This gives us the n^{th} link coordinate frame F_n relative to the first link coordinate frame F_0 .

We also wish to specify the location of the table frame F_T relative to F_n , and the first link frame F_0 relative to the robot base frame F_R . These two transformations $H_{R,0}$, $H_{n,T}$ are fixed and known. Thus,

$$H_{R,T} = H_{R,0} H_{0,n} H_{n,T} \quad (5.9)$$

5.3.2 Workpiece Frame Relative to Table

We can safely assume that the workpiece is firmly attached to the table. Thus, $\underline{H}_{T,B}$ is some fixed rotation and translation relative to F_T which needs to be specified only once during initialization.

5.3.3 Weld Seam Frame Relative to Workpiece

We define the weld seam frame F_C as shown in Figure 5.5, where $\underline{T}$ is tangent to the seam trajectory and $\underline{A}$ bisects the surface normals $\underline{N}_1, \underline{N}_2$. (i.e., $\underline{A}$ is normal to weld face). The transformation $\underline{H}_{B,C}$ is then

$$\underline{H}_{B,C} = [\underline{T}, \underline{B}, \underline{A}, \underline{R}] \quad . \quad (5.10)$$

5.3.4 Tool Frame Relative to Weld Seam

The welding rod is positioned relative to the weld as required by standard welding practice. The position and orientation of the tool relative to the weld are characterized by angular offsets from the weld surface normal, α and β , and a tool gap ρ (Fig. 5.6). ρ, α, β are welding parameters determined by the Expert Welder module.

We know that the elements of $\underline{H}_{C,G}$ can be found from

$$\underline{H}_{C,G} = [\underline{n}, \underline{o}, \underline{a}, \underline{p}] \quad . \quad (5.11)$$

From trigonometry:

$$\underline{a} = \begin{bmatrix} -\sin \alpha \\ \sin \beta \\ -\cos \alpha \cos \beta \\ 0 \end{bmatrix} \quad , \quad (5.12)$$

$$\text{and } \underline{p} = -\rho \underline{a} \quad . \quad (5.13)$$

$\underline{n}, \underline{o}$ are arbitrary -- let us define them as

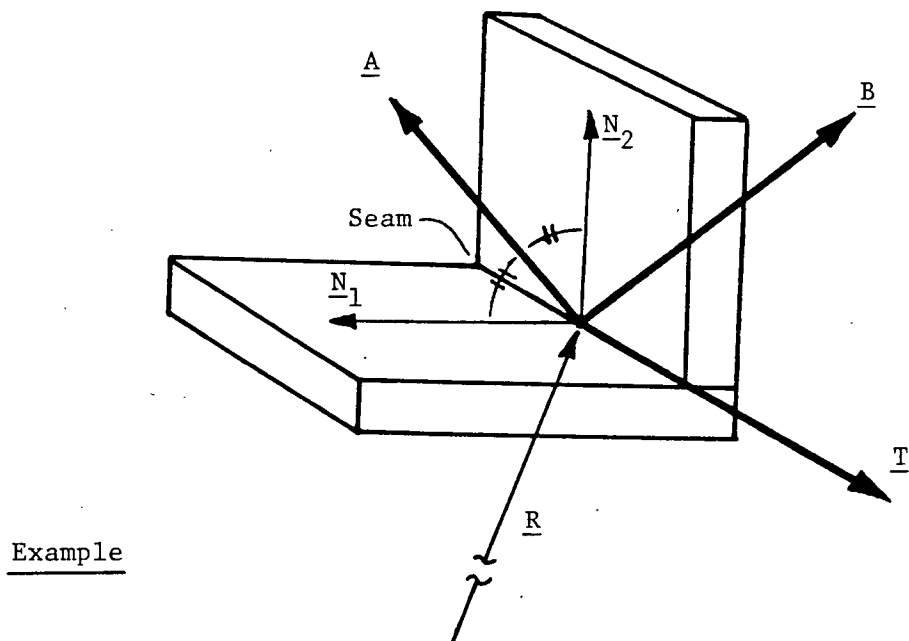
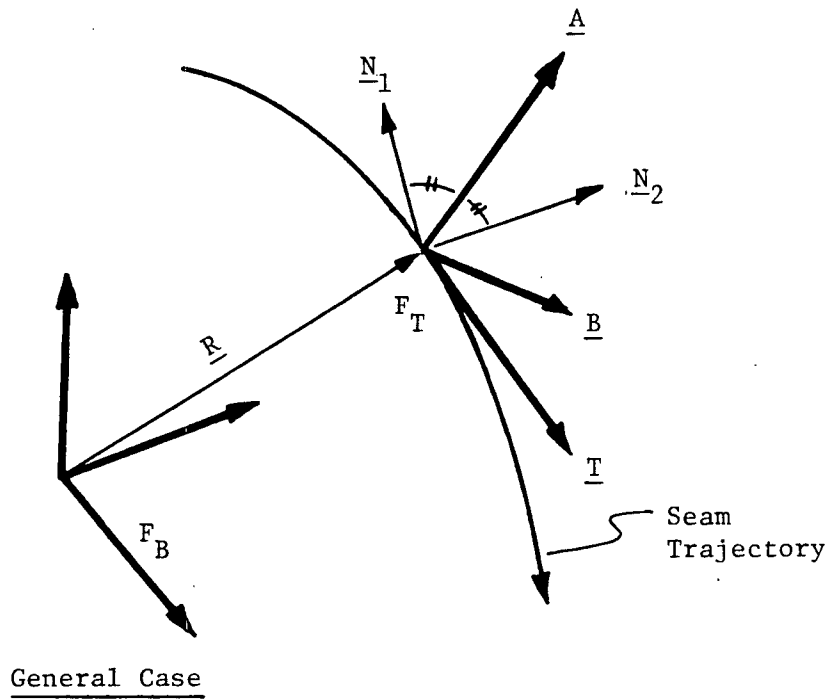


Figure 5.5 Weld Frame Relative to Workpiece.

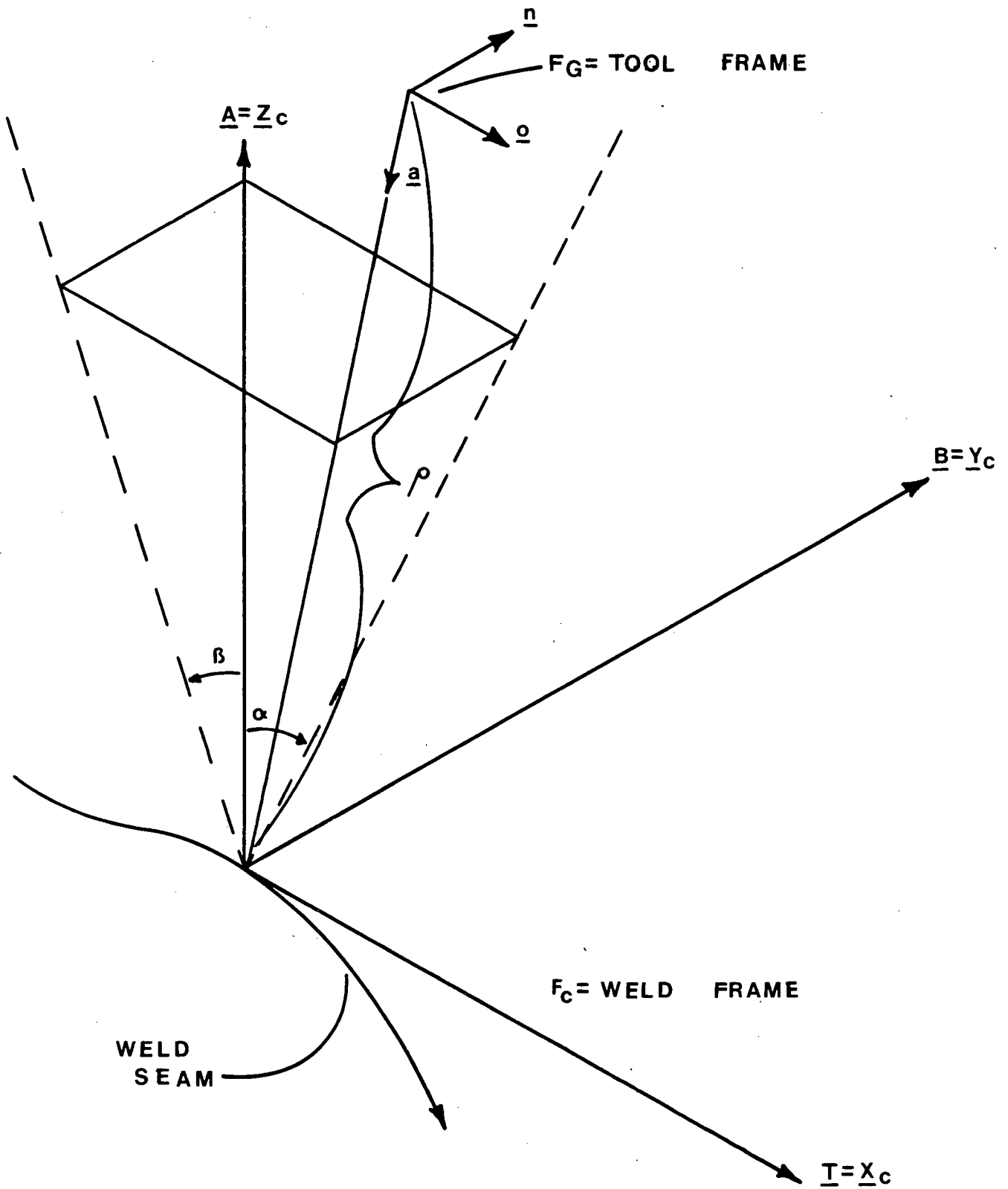


Figure 5.6 Tool Frame Relative to Weld Seam.

$$\underline{n} = \begin{bmatrix} 0 \\ \cos \beta \cos \alpha \\ \sin \beta \\ 0 \end{bmatrix} \quad (5.14)$$

$$\text{and } \underline{o} = \underline{a} \times \underline{n} . \quad (5.15)$$

We can now find elements of $\underline{H}_{C,G} = [\underline{n}, \underline{o}, \underline{a}, \underline{P}]$.

5.4 Calculation of the Tool Position and Orientation Relative to World Coordinates

Once we have specified all the intermediate homogeneous transformations, we can calculate the tool position and orientation relative to F_R . This information is given by the homogeneous transformation $\underline{H}_{R,G}$ which takes us from the tool coordinates to robot coordinates. By combination of homogeneous transforms, we can calculate $\underline{H}_{R,G}$:

$$\underline{H}_{R,G} = \underline{H}_{R,T} \underline{H}_{T,B} \underline{H}_{B,C} \underline{H}_{C,G} . \quad (5.16)$$

The elements of $\underline{H}_{R,G}$ have the form of four column vectors, $\underline{n}, \underline{o}, \underline{a}, \underline{P}$:

$$\underline{H}_{R,G} = [\underline{n}, \underline{o}, \underline{a}, \underline{P}] . \quad (5.17)$$

We know that the vectors $\underline{n}, \underline{o}$, and $\underline{a}$ define the basis vectors of the tool coordinate frame F_G projected onto the robot base frame F_R , and $\underline{P}$ is the displacement of F_G relative to F_R . If we wish to define a world coordinate frame which is different from the robot base frame, they can be simply related by a fixed transformation.

6. PATH FEASIBILITY

6.1 Statement of Feasibility Problem

When we calculate a nominal robot tool trajectory through space, we may find that the trajectory is not feasible. The feasibility issue can be broken into two fundamental problems:

1. Kinematic constraints of the robot.
2. Interference or collision with objects in the workstation.

We can define a trajectory as being feasible if the robot can follow the trajectory smoothly, without violating the kinematic constraints and without colliding with or contacting any of the objects in the workstation.

Of the two problems we have defined, the kinematics problem is the least difficult, and we will discuss it first.

6.2 Kinematic Feasibility

Mathematically the conditions which must be satisfied for kinematic feasibility are:

1. The work envelope must not be exceeded.
2. The robot joint angles must remain within their physical limits.

In order for us to test these conditions it is necessary to solve for the joint angles as a function of the desired tool location and orientation relative to the robot base (i.e., $\underline{H}_{R,G}$). This is commonly known as the inverse kinematics solution. There are two particular cases which are of practical interest. Suppose we define a robot gripper state vector $\underline{X}$ defining n independent state variables. Let us also define a joint

variable vector $\underline{\theta}$ defining m joint variables. If $n = m$, we can find a finite set of solutions of $\underline{\theta}$ for any set $\underline{X}$ in the work envelope. If $n < m$, an infinite set of solutions exists and we say the robot has redundant degrees of freedom.

The kinematic problem has been considered by many researchers over the years. Denavit and Hartenberg [1955] first formally specified the geometry of connected links with homogeneous matrices. This work has been fundamental to most later work.

Whitney [1969; 1972] developed the mathematics for the kinematics of multi-degree-of-freedom manipulators and prosthetic arms, and solved the inverse problem by finding the inverse Jacobian matrix, $\underline{J}^{-1}$, relating gripper motion to joint rate (i.e. $\dot{\underline{\theta}} = \underline{J}^{-1} \dot{\underline{X}}$). This method allows relative cartesian motion along world coordinates, based on real-time calculations of instantaneous $\underline{J}(\theta)$ and $\underline{J}^{-1}(\theta)$.

No general solution has been found for the inverse problem for absolute positions. An iterative solution based on finite displacement analysis has been proposed by Konstantinov [Konstantinov and Markov 1980; Konstantinov, Genova and Zahariev 1981; Konstantinov, Markov and Nechev 1981; Konstantinov and Patarinski 1982].

Several general recursive techniques have been proposed based on the approximate decoupling of the minor and major axis of motion [Gaglio et al 1981; Milenkov and Huang 1983; Benati et al 1982].

Explicit solutions can be found for most robot configurations of practical importance. Lee and Ziegler [1983] have derived an explicit solution for the PUMA robot based on geometry. Explicit solutions can

also be obtained by matrix algebra using the Denavit-Hartenberg matrices [Paul, Shimano and Mayer 1981; Paul 1981].

Redundant degrees of freedom are useful for avoiding joint constraints and allowing collision-free configurations to be found. The inverse problem can be solved by optimizing some criterion function. Redundant systems have been considered in more detail by a number of investigators [Fournier and Khalil 1977; Liégeois 1977; Konstantinov Markov and Nechev 1981; Benati, Morasso and Tagliasco 1982].

We will develop an explicit inverse kinematics solution based on the matrix method of Paul [Paul, Shimano and Mayer 1981; Paul 1981].

6.2.1 Definition of Robot Joint Frames and Resulting Transformations

A robot arm consists of a series of rigid links connected by revolute or prismatic joints. Coordinate frames are attached to each link according to the specification of Denavit and Hartenberg [1955]. The geometry is shown in Figs. 3.1 and 3.2. First let us state some definitions:

$\underline{J}_i$ = joint i axis of rotation or translation

$\underline{a}_i$ = common normal of axes $\underline{J}_i, \underline{J}_{i+1}$.

For a revolute joint, the origin of F_i is located at the intersection of $\underline{J}_{i+1}$ and $\underline{a}_i$. If $\underline{J}_i, \underline{J}_{i+1}$ intersect, the origin is at the intersection point. If $\underline{J}_i, \underline{J}_{i+1}$ are parallel, $\underline{a}_i$ is chosen to intersect $\underline{J}_{i+1}$ at the same place as $\underline{a}_{i+1}$. If joint $i+1$ is prismatic, replace $\underline{J}_{i+1}$ above with the axis of the next revolute joint. The origin of F_i is defined for

$$d_{i+1} = 0.$$

The z axis, $\underline{z}_i$, of F_i is along $\underline{J}_{i+1}$. The x-axis, $\underline{x}_i$, of F_i is along $\underline{a}_i$. If $\underline{J}_i$, $\underline{J}_{i+1}$ intersect, $\underline{x}_i$ is parallel or anti-parallel to $\underline{J}_i \times \underline{J}_{i+1}$.

θ_i is zero when $\underline{x}_{i-1} = \underline{x}_i$.

For a prismatic joint, d_i becomes the joint variable. In this case $\underline{a}_i = 0$. The origin of F_i is coincident with the origin of F_{i-1} for $d_i = 0$. $\underline{z}_i$ is along $\underline{J}_{i+1}$ and $\underline{x}_i$ is parallel or anti-parallel to $\underline{J}_i \times \underline{J}_{i+1}$.

6.2.2 Inverse Kinematics Solution

6.2.2.1 General Approach to Inverse Kinematics Problem

Once we have defined the joint frames as shown, we can calculate the homogeneous transformation relating frame F_i to frame F_{i-1} . The transformation is composed of the following sequence:

1. rotate about z axis of F_{i-1} by an angle θ_i ;
2. translate along z axis of F_{i-1} by distance d_i ;
3. translate along new x axis of F_{i-1} by length a_i ; and
4. rotate about new x axis of F_{i-1} by the twist angle α_i .

These four transformations can be combined as follows:

$$\underline{H}_{i-1,i} = \underline{\text{Rot}}(z, \theta_i) \underline{\text{Trans}}(z, d_i) \underline{\text{Trans}}(x, a_i) \underline{\text{Rot}}(x, \alpha_i).$$

$$\underline{H}_{i-1,i} = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.1)$$

where $C\theta_i \equiv \cos \theta_i$, $S\theta_i \equiv \sin \theta_i$, etc. Multiplying out, we get

$$\underline{H}_{i-1,i} = \begin{bmatrix} C\theta_i & -S\theta_i C\alpha_i & S\theta_i S\alpha_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\alpha_i & -C\theta_i S\alpha_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.2)$$

This transformation was first derived by Denavit and Hartenberg [1955].

Once we have defined $\underline{H}_{i-1,i}$ for the links $i=1, \dots, n$, we can combine them to find each link frame relative to the robot base frame F_R , i.e.,

$$\underline{H}_{R,i} = \underline{H}_{R,1} \underline{H}_{1,2} \cdots \underline{H}_{i-1,i} \quad (6.3)$$

for $i = 1, n$. We can also find the transformation

$$\underline{H}_{i,n} = \underline{H}_{i,i+1} \cdots \underline{H}_{n-1,n} \quad (6.4)$$

relating the last frame to the i^{th} frame.

We now wish to solve for the joint variables for a given tool position and orientation. We can find the homogeneous transformation $\underline{H}_{R,n}$ which specifies the tool orientation and position as a function of the joint variables $\theta_1, \theta_2, \dots, \theta_n$. We know the tool location and orientation relative to the robot base, $\underline{H}_{R,G}$. The tool frame is related to the last joint frame by a fixed known transformation $\underline{H}_{n,G}$, so we can easily find $\underline{H}_{R,n} = \underline{H}_{R,G} \underline{H}_{n,G}^{-1}$. The elements of $\underline{H}_{R,n}$ are now known: i.e.,

$$\underline{H}_{R,n} = \begin{bmatrix} \underline{n} & \underline{o} & \underline{a} & \underline{p} \end{bmatrix} , \quad (6.5)$$

where

$\underline{n}$, $\underline{o}$, $\underline{a}$ are basis vectors of link n frame relative to F_R ;

$\underline{p}$ is position vector of link n frame relative to F_R .

Let us call this matrix $\underline{T}_n$.

We can now equate $\underline{T}_n$ and $\underline{H}_{R,n}$ element by element, searching for explicit equations for the joint angles $\theta_1, \dots, \theta_n$. If necessary, we can also compare the elements of other equivalent matrices defined below.

$$\underline{T}_n = \underline{H}_{R,n} \quad , \quad (6.6)$$

$$\underline{H}_{R,1}^{-1} \underline{T}_n = \underline{H}_{1,n} \quad , \quad (6.7)$$

$$\underline{H}_{1,2}^{-1} \underline{H}_{R,1}^{-1} \underline{T}_n = \underline{H}_{2,n} \quad , \quad \text{etc.} \quad (6.8)$$

We continue until we have explicit solutions for all the joint variables.

The actual solution of a particular robot is generally based on intuition and simplification. The particular solution for a UNIMATION PUMA 560 follows.

6.2.2.2 PUMA 560 Solution

For the PUMA robot, we can define a set of link frames as shown in Fig. 6.1. We can also specify the appropriate link parameters in Table 6.1.

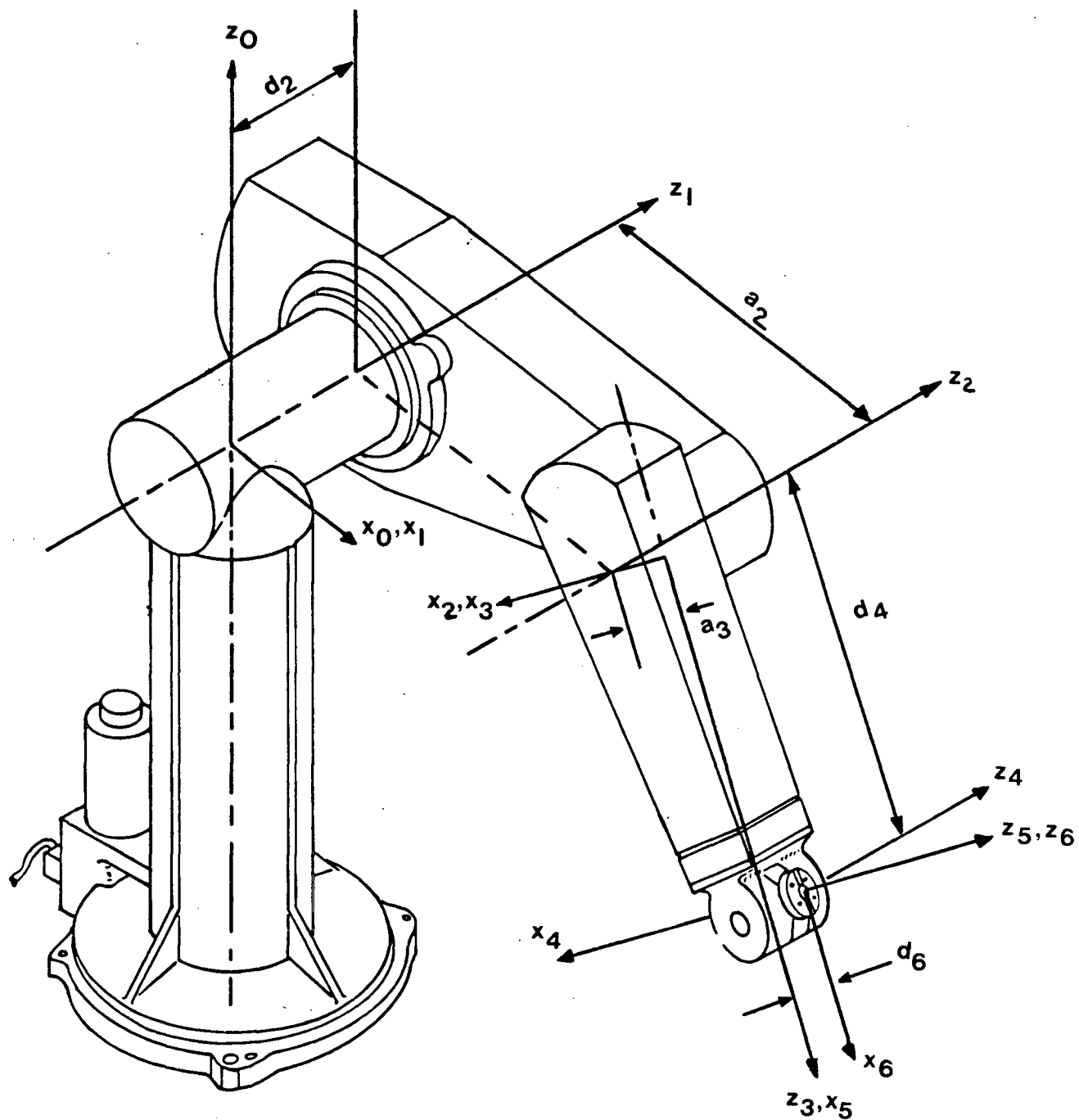


Figure 6.1 PUMA Joint Frames and link parameters.

Table 6.1 - PUMA Link Parameters

Link	Variable	α	a	d
1	θ_1	-90	0	0
2	θ_2	0	a_2	d_2
3	θ_3	90	$-a_3$	0
4	θ_4	-90	0	d_4
5	θ_5	90	0	0
6	θ_6	0	0	d_6

Note that the sign of a_3 is negative to conform to our definition of x_2 .

For the PUMA, the link parameters have the following values:

$$a_2 = 432 \text{ mm}$$

$$d_2 = 149.5 \text{ mm}$$

$$a_3 = 20.5 \text{ mm}$$

$$d_4 = 432 \text{ mm}$$

$$d_6 = 56.5 \text{ mm}$$

Note that the following analysis also applies to any other 6 degrees of freedom robot with the same geometry. The correct values of the link parameters a_2 , a_3 , d_2 , d_4 , and d_6 are simply substituted into the solutions.

We know the position and orientation of F_6 relative to F_0 from

$$H_{0,6} = [\underline{n}, \underline{o}, \underline{a}, \underline{p}] \quad , \quad (6.9)$$

where

$\underline{n}$, $\underline{o}$, $\underline{a}$ are the F_6 basis vectors projected onto F_0 ;

$\underline{P}$ is the F_6 origin relative to F_0 .

Note that F_0 is the frame attached to link 0 by our definition, and is equivalent to the robot base frame F_R . F_6 is attached to the tool mounting flange.

We have defined the inter-link transformations $\underline{H}_{i-1,i}$ for each link and we know that $\underline{H}_{0,i} = \underline{H}_{0,1} \underline{H}_{1,2} \cdots \underline{H}_{i-1,i}$. (6.10)

From the geometry of the arm (Figure 6.2) we note that the origin of F_4 is dependent on the variables $\theta_1, \theta_2, \theta_3$ only, and is coincident with the origin of F_5 .

Let $\underline{q} = \underline{P} - d_6 \underline{a}$ (origin of F_5) , (6.11)

and $\underline{q} = 4^{\text{th}}$ column of $\underline{H}_{0,4}$ (origin of F_4) .

We can find $\underline{q}$ by multiplying $\underline{H}_{0,1} \underline{H}_{1,2} \underline{H}_{2,3} \underline{H}_{3,4}$ (we need the 4^{th} column only).

$$\text{The 4th column of } \underline{H}_{2,4} \text{ is } \begin{bmatrix} s_3 d_4 + a_3 c_3 \\ -c_3 d_4 + a_3 s_3 \\ 0 \\ 1 \end{bmatrix} .$$

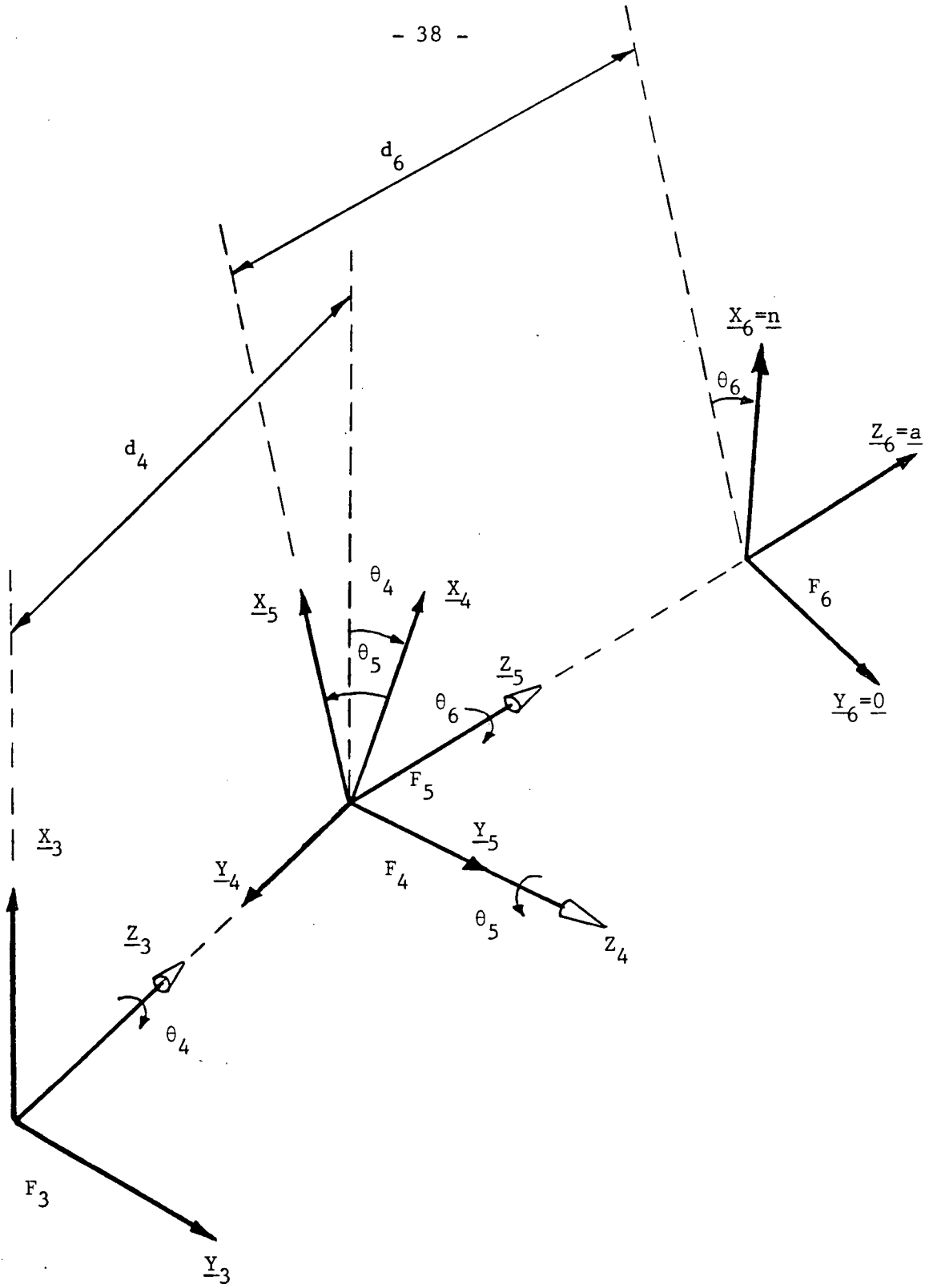


Figure 6.2 Relative geometry of PUMA link frames F_3 , F_4 , F_5 , F_6 .

The 4th column of $\underline{H}_{-1,4}$ is
$$\begin{bmatrix} C_2(S_3 d_4 + a_3 C_3) + S_2(C_3 d_4 - a_3 S_3) + a_2 C_2 \\ S_2(S_3 d_4 + a_3 C_3) - C_2(C_3 d_4 - a_3 S_3) + a_2 S_2 \\ d_2 \\ 1 \end{bmatrix} .$$

$\underline{q}$ = 4th column of $\underline{H}_{0,4}$.

Multiplying out, we get

$$\underline{q} = \begin{bmatrix} C_1 [C_2(S_3 d_4 + a_3 C_3) + S_2(C_3 d_4 - a_3 S_3) + a_2 C_2] - S_1 d_2 \\ S_1 [C_2(S_3 d_4 + a_3 C_3) + S_2(C_3 d_4 - a_3 S_3) + a_2 C_2] + C_1 d_2 \\ -S_2(S_3 d_4 + a_3 C_3) + C_2(C_3 d_4 - a_3 S_3) - a_2 S_2 \\ 1 \end{bmatrix} . \quad (6.12)$$

$\underline{q}$ can be simplified as follows:

$$\underline{q} = \begin{bmatrix} C_1 [d_4 S_{23} + a_3 C_{23} + a_2 C_2] - S_1 d_2 \\ S_1 [d_4 S_{23} + a_3 C_{23} + a_2 C_2] + C_1 d_2 \\ d_4 C_{23} - a_3 S_{23} - a_2 S_2 \\ 1 \end{bmatrix} , \quad (6.13)$$

where

$$\begin{aligned} S_{23} &= \sin(\theta_2 + \theta_3) \\ C_{23} &= \cos(\theta_2 + \theta_3) . \end{aligned}$$

Solution for θ_1

From the first two rows of (6.13) we can show that

$$\theta_1 = \tan^{-1} \left[\frac{\pm q_2(q_1^2 + q_2^2 - d_2^2)^{1/2} - d_2 q_1}{\pm q_1(q_1^2 + q_2^2 - d_2^2)^{1/2} + d_2 q_2} \right], \quad (6.14)$$

+ for left handed arm configuration,

- for right handed arm configuration.

Note: For $\theta = \tan^{-1} \left(\frac{x}{y} \right)$;

$0 < \theta < 90^\circ$ for $x > 0$ and $y > 0$;

$90^\circ < \theta < 180^\circ$ for $x < 0$ and $y > 0$;

$-180^\circ < \theta < -90^\circ$ for $x < 0$ and $y < 0$;

$-90^\circ < \theta < 0$ for $x > 0$ and $y < 0$.

Solution for θ_3

Let

$$A = \frac{q_1^2 + q_2^2 + q_3^2 - d_2^2 - d_4^2 - a_3^2 - a_2^2}{2 a_2}; \quad (6.15)$$

$$B = -d_4; \quad C = a_3. \quad (6.16), (6.17)$$

We can show that $A = -B \sin \theta_3 + C \cos \theta_3$. The solution has the following form:

$$\theta_3 = \tan^{-1} \left(\frac{C}{B} \right) - \tan^{-1} \left(\frac{A}{\pm \sqrt{r^2 - A^2}} \right) \quad (6.18)$$

where

$$r^2 = B^2 + C^2. \quad (6.19)$$

Substituting (6.16), (6.17) into (6.18) we get

$$\theta_3 = \tan^{-1}\left(\frac{a_3}{-d_4}\right) - \tan^{-1}\left(\frac{A}{\pm \sqrt{r^2 - A^2}}\right) \quad (6.20)$$

where $r^2 = a_3^2 + d_4^2$. (6.21)

Two different solutions result depending on the sign:

+ for elbow above configuration,

and - for elbow below configuration, (for left arm configuration);

or - for elbow above configuration,

and + for elbow below configuration, (for right arm configuration).

Solution for θ_2

From (6.13) we get

$$q_3 = d_4 C_{23} - a_3 S_{23} - a_2 S_2 \quad (6.22)$$

Expanding, we get

$$q_3 = d_4 [C_2 C_3 - S_2 S_3] - a_3 [S_2 C_3 + C_2 S_3] - a_2 S_2 \quad (6.23)$$

Rearranging we get

$$q_3 = [d_4 C_3 - a_3 S_3] C_2 - [d_4 S_3 + a_3 C_3 + a_2] S_2 \quad (6.24)$$

Let

$$A = d_4 C_3 - a_3 S_3 \quad (6.25)$$

$$B = d_4 S_3 + a_3 C_3 + a_2 \quad (6.26)$$

Then the solution is

$$\theta_2 = \tan^{-1}\left(\frac{A}{B}\right) - \tan^{-1}\left(\frac{q_3}{\pm \sqrt{r^2 - q_3^2}}\right) \quad (6.27)$$

Substituting (6.25), (6.26) into (6.27) we get

$$\theta_2 = \tan^{-1} \left(\frac{d_4 C_3 - a_3 S_3}{d_4 S_3 + a_3 C_3 + a_2} \right) - \tan^{-1} \left(\frac{q_3}{\pm \sqrt{r^2 - q_3^2}} \right) \quad (6.28)$$

where

$$r^2 = A^2 + B^2 \quad (6.29)$$

Expanding (6.29), we get

$$r^2 = (d_4 C_3 - a_3 S_3)^2 + (d_4 S_3 + a_3 C_3 + a_2)^2 \quad (6.30)$$

Again, there are two solutions for (6.28);

+ is for the left handed arm configuration

and - is for the right handed arm configuration.

There are two solutions for θ_2 because the parameters q_1 and q_2 are not in the equation. It is intuitively obvious that for a given waist rotation θ_1 , there exist two possible shoulder rotations, θ_2 , which position the arm at the same height q_3 . The possible rotations position the arm in opposite quadrants of the horizontal plane. The correct solution can be determined by substituting θ_2 into the equation for the parameter q_1 or q_2 (equation 6.1) and comparing the sign of the result. It turns out, however, that the positive solution is always associated with the left arm configuration and the negative solution with the right arm configuration.

Solution for $\theta_4, \theta_5, \theta_6$

Set θ_4 such that $\underline{z}_4$ is normal to the plane formed by $\underline{z}_3$ and $\underline{a}$, i.e.,

$$\underline{z}_4 = \pm \frac{(\underline{z}_3 \cdot \underline{a})}{|\underline{z}_3 \cdot \underline{a}|} \quad (6.31)$$

and

$$\sin \theta_4 = \pm (\underline{x}_3 \cdot \underline{z}_4) , \quad (6.32)$$

$$\cos \theta_4 = \pm (\underline{y}_3 \cdot \underline{z}_4) . \quad (6.33)$$

Note that $\underline{x}_3, \underline{y}_3$ are the first two columns of $\underline{H}_{0,3}$. Expanding out $\underline{H}_{0,3}$ and substituting $\underline{x}_3, \underline{y}_3$ and $\underline{z}_4$ into (6.32), (6.33), we can show

$$\theta_4 = \tan^{-1} \left[\frac{\pm (C_1 a_y - S_1 a_x)}{\pm (C_1 C_{23} a_x + S_1 C_{23} a_y - S_{23} a_z)} \right] \quad (6.34)$$

where $+$ gives wrist not flipped configuration

and $-$ gives wrist flipped configuration.

$$\text{Set } \theta_5 \text{ so that } \underline{z}_5 = \underline{a} , \quad (6.35)$$

$$\text{i.e., } \sin \theta_5 = \underline{x}_4 \cdot \underline{a} , \quad (6.36)$$

$$\text{and } \cos \theta_5 = -\underline{y}_4 \cdot \underline{a} . \quad (6.37)$$

$\underline{x}_4, \underline{y}_4$ are the first two columns of $\underline{H}_{0,4}$. Expanding out $\underline{H}_{0,4}$ and substituting $\underline{x}_4, \underline{y}_4$ into (6.36), (6.37) we can show

$$\theta_5 = \tan^{-1} \left[\frac{(C_1 C_{23} C_4 - S_1 S_4) a_x + (S_1 C_{23} C_4 + C_1 S_4) a_y - C_4 S_{23} a_z}{C_1 S_{23} a_x + S_1 S_{23} a_y + C_{23} a_z} \right] . \quad (6.38)$$

$$\text{Set } \theta_6 \text{ so that } \underline{y}_6 = \underline{o} , \quad (6.39)$$

$$\text{i.e., } \sin \theta_6 = \underline{y}_5 \cdot \underline{n} , \quad (6.40)$$

$$\text{and } \cos \theta_6 = \underline{y}_5 \cdot \underline{o} . \quad (6.41)$$

$\underline{y}_5$ is the second column of $\underline{H}_{0,5}$.

Multiplying out $\underline{H}_{0,5}$ and substituting $\underline{y}_5$ into (6.40), (6.41) we can show

$$\theta_6 = \tan^{-1} \left[\frac{(-S_1 C_4 - C_1 C_{23} S_4) n_x + (C_1 C_4 - S_1 C_{23} S_4) n_y + S_4 S_{23} n_z}{(-S_1 C_4 - C_1 C_{23} S_4) o_x + (C_1 C_4 - S_1 C_{23} S_4) o_y + S_4 S_{23} o_z} \right] . \quad (6.42)$$

6.2.2.3 Discussion of PUMA Solutions

When we inspect the joint angle solutions, we note that joints 1, 3 and 4 have 2 solutions each. These solutions correspond to different robot configurations which yield the same tool position $H_{0,6}$. For instance, the arm may be in a left-handed or right-handed configuration, the elbow may be above or below the tool, and the wrist may be flipped. This suggests that different configurations might be tried when searching for a feasible configuration. In order to obtain smooth motion between points, however, we must insist that the same configuration be maintained throughout a given trajectory.

6.2.3 Comparison of Five Degree of Freedom and Six Degree of Freedom Robots

A 6 degree of freedom robot allows us to completely specify the tool position and orientation in space. In the case of welding, however, we note that the tool is symmetric about its axis. This means that any rotation about the tool axis does not change the effective tool orientation. Thus we see that we need only specify the tool position and a single orientation vector $\underline{a}$. Five degrees of freedom are sufficient and a 6th degree of freedom is redundant. For a 5 degree of freedom robot, the tool frame basis vectors $\underline{n}$, $\underline{o}$ are found to be functions of the orientation vector $\underline{a}$ and the position vector $\underline{p}$. The relationships are derived in the following analysis for a PUMA robot with the 4th joint omitted.

6.2.3.1 Solution for a Five Degree of Freedom Robot

Let us now define a 5 degree of freedom robot which is

geometrically and dimensionally identical to the 6 degree of freedom PUMA previously analysed except that the 4th joint (wrist) is now rigidly fixed at $\theta_4 = 0$.

We can redefine the link parameters as follows:

Table 6.2 - Five Degrees of Freedom PUMA Link Parameters

link	α	a	d
1	-90	0	0
2	0	a_2	d_2
3	0	a_2	0
4	90	0	0
5	90	0	d_5

Let us also define a tool relative to the last link mounting flange (Fig. 6.3). The transformation matrix $H_{5,6}$ is

$$H_{5,6} = \begin{bmatrix} 0 & 0 & 1 & d_x \\ 0 & 1 & 0 & d_y \\ -1 & 0 & 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6.43)$$

It can be shown that we can achieve any location and orientation of the tool axis $\underline{Z}_6$ if $\underline{Z}_6$ is perpendicular to $\underline{Z}_5$.

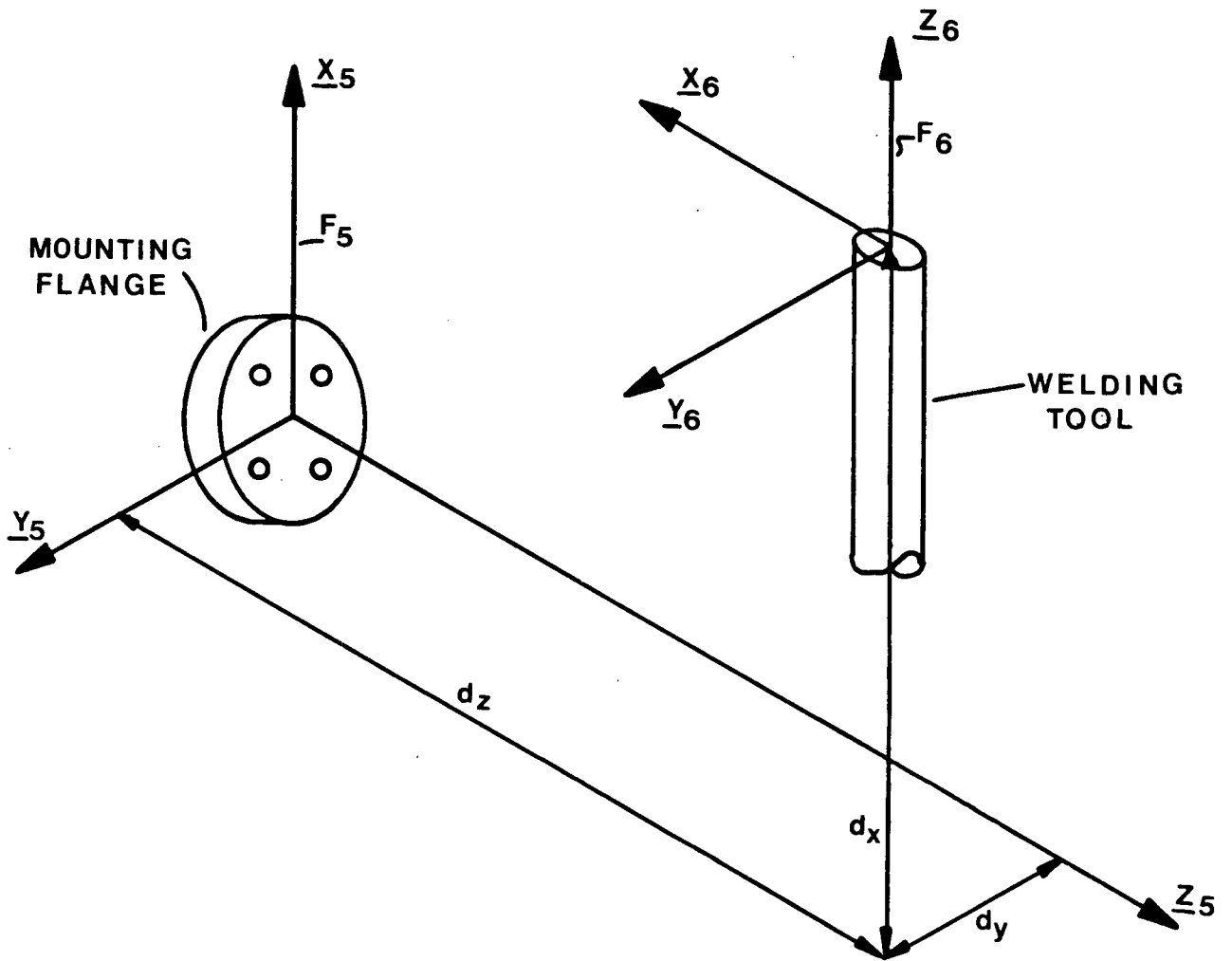


Figure 6.3 Tool Frame Relative to Robot Tool Mounting Flange.

Solve for tool frame F_6 relative to F_0 . The Denavit-Hartenberg matrices are:

$$\underline{H}_{0,1} = \begin{bmatrix} c\theta_1 & 0 & -s\theta_1 & 0 \\ s\theta_1 & 0 & c\theta_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.44)$$

$$\underline{H}_{1,2} = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & a_2 c\theta_2 \\ s\theta_2 & c\theta_2 & 0 & a_2 s\theta_2 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.45)$$

$$\underline{H}_{2,3} = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & a_3 c\theta_3 \\ s\theta_3 & c\theta_3 & 0 & a_3 s\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.46)$$

$$\underline{H}_{3,4} = \begin{bmatrix} c\theta_4 & 0 & s\theta_4 & 0 \\ s\theta_4 & 0 & -c\theta_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.47)$$

$$\underline{H}_{4,5} = \begin{bmatrix} c\theta_5 & -s\theta_5 & 0 & 0 \\ s\theta_5 & c\theta_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.48)$$

$$\underline{H}_{5,6} = \begin{bmatrix} 0 & 0 & 1 & d_x \\ 0 & 1 & 0 & d_y \\ -1 & 0 & 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.49)$$

The inverse matrices are:

$$\underline{H}_{0,1}^{-1} = \begin{bmatrix} c_1 & s_1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -s_1 & c_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.50)$$

$$\underline{H}_{1,2}^{-1} = \begin{bmatrix} c_2 & s_2 & 0 & -a_2 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 1 & -d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.51)$$

$$\underline{H}_{2,3}^{-1} = \begin{bmatrix} c_3 & s_3 & 0 & -a_3 \\ -s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.52)$$

$$\underline{H}_{3,4}^{-1} = \begin{bmatrix} c_4 & s_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ s_4 & -c_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.53)$$

$$\underline{H}_{4,5}^{-1} = \begin{bmatrix} c_5 & s_5 & 0 & 0 \\ -s_5 & c_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.54)$$

$$\underline{H}_{5,6}^{-1} = \begin{bmatrix} 0 & 0 & -1 & d_z \\ 0 & 1 & 0 & -d_y \\ 1 & 0 & 0 & -d_x \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.55)$$

The intermediate transformations are:

$$\underline{H}_{5,6} = \begin{bmatrix} 0 & 0 & 1 & d_x \\ 0 & 1 & 0 & d_y \\ -1 & 0 & 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.56)$$

$$\underline{H}_{4,6} = \begin{bmatrix} 0 & -s_5 & c_5 & c_5 d_x - s_5 d_y \\ 0 & c_5 & s_5 & s_5 d_x + c_5 d_y \\ -1 & 0 & 0 & d_z + d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.57)$$

$$\underline{H}_{3,6} = \begin{bmatrix} -s_4 & -c_4 s_5 & c_4 c_5 & c_4 (c_5 d_x - s_5 d_y) \\ & & & +s_4 (d_z + d_5) \\ c_4 & -s_4 s_5 & s_4 c_5 & s_4 (c_5 d_x - s_5 d_y) \\ & & & -c_4 (d_z + d_5) \\ 0 & c_5 & s_5 & s_5 d_x + c_5 d_y \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.58)$$

Let $d_s = d_5 + d_z$. (6.59)

$$\underline{H}_{2,6} = \begin{bmatrix} -c_3 s_4 - s_3 c_4 & | & -c_3 c_4 s_5 + s_3 s_4 s_5 & | \\ -s_3 s_4 + c_3 c_4 & | & -s_3 c_4 s_5 - c_3 s_4 s_5 & | \\ 0 & | & c_5 & | \\ 0 & | & 0 & | \end{bmatrix} . \quad (6.60)$$

$$\begin{bmatrix} | & c_3 c_4 c_5 - s_3 s_4 c_5 & | & c_3 [c_4 (c_5 d_x - s_5 d_y) + s_4 d_s] & | \\ | & & | & -s_3 [s_4 (c_5 d_x - s_5 d_y) - c_4 d_s] + a_3 c_3 & | \\ | & s_3 c_4 c_5 + c_3 s_4 c_5 & | & s_3 [c_4 (c_5 d_x - s_5 d_y) + s_4 d_s] & | \\ | & & | & +c_3 [s_4 (c_5 d_x - s_5 d_y) - c_4 d_x] + a_3 s_3 & | \\ | & s_5 & | & s_5 d_x + c_5 d_y & | \\ | & 0 & | & 1 & | \end{bmatrix}$$

$\underline{H}_{1,6}$ is broken down into columns.

$$\underline{H}_{1,6}^{(1)} = \begin{bmatrix} c_2[-c_3s_4 - s_3c_4] - s_2[-s_3s_4 + c_3c_4] \\ s_2[-c_3s_4 - s_3c_4] + c_2[-s_3s_4 + c_3c_4] \\ 0 \\ 0 \end{bmatrix} . \quad (6.61)$$

$$\underline{H}_{1,6}^{(2)} = \begin{bmatrix} c_2[-c_3c_4s_5 + s_3s_4s_5] - s_2[-s_3c_4s_5 - c_3s_4s_5] \\ s_2[-c_3c_4s_5 + s_3s_4s_5] + c_4[-s_3c_4c_5 - c_3s_4s_5] \\ s_5 \\ 0 \end{bmatrix} . \quad (6.62)$$

$$\underline{H}_{1,6}^{(3)} = \begin{bmatrix} c_2[c_3c_4c_5 - s_3s_4c_5] - s_2[s_3c_4c_5 + c_3s_4c_5] \\ s_2[c_3c_4c_5 - s_3s_4c_5] + c_2[s_3c_4c_5 + c_3s_4c_5] \\ s_5 \\ 0 \end{bmatrix} . \quad (6.63)$$

$$\underline{H}_{1,6}^{(4)} = \begin{bmatrix} c_2[\cdot \cdot \cdot] - s_2[\cdot \cdot \cdot] + a_2 c_2 \\ s_2[\cdot \cdot \cdot] + c_2[\cdot \cdot \cdot] + a_2 s_2 \\ s_5 d_x + c_5 d_y + d_2 \\ \dots \\ 1 \end{bmatrix} . \quad (6.64)$$

$$\underline{H}_{0,6} = [\underline{n}, \underline{o}, \underline{a}, \underline{P}] . \quad (6.65)$$

Let

$$\underline{H}_{0,1}^{-1} \underline{H}_{0,6} = \begin{bmatrix} f_{11}(n) & f_{11}(o) & f_{11}(a) & f_{11}(P) \\ f_{12}(n) & f_{12}(o) & f_{12}(a) & f_{12}(P) \\ f_{13}(n) & f_{13}(o) & f_{13}(a) & f_{13}(P) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.66)$$

where

$$f_{11}(A) = c_1 A_x + s_1 A_y , \quad (6.67)$$

$$f_{12}(A) = -A_z , \quad (6.68)$$

$$f_{13}(A) = -s_1 A_x + c_1 A_y , \quad (6.69)$$

and

A is n, o, a or P.

We also know $\underline{H}_{0,1}^{-1} \underline{H}_{0,6} = \underline{H}_{1,6}$, which we have found as a function of θ_2 ,

$\theta_3, \theta_4, \theta_5$. Equating elements we find

$$0 = -S_1 n_x + C_1 n_y, \quad (6.70)$$

$$C_5 = -S_1 o_x + C_1 o_y, \quad (6.71)$$

$$S_5 = -S_1 a_x + C_1 a_y, \quad (6.72)$$

$$d_2 + S_5 d_x + C_5 d_y = -S_1 P_x + C_1 P_y. \quad (6.73)$$

We know the position P and axis orientation a of the tool. We now solve for the remaining tool orientation vectors n, o, as a function of a, P.

Multiply (6.71) by d_y ,

multiply (6.72) by d_x

and subtract (6.71) & (6.72) from (6.73).

$$d_2 = -S_1 [P_x - d_y o_x - d_x a_x] + C_1 [P_y - d_y o_y - d_x a_y]. \quad (6.74)$$

Rewrite as

$$d_2 = S_1 \alpha + C_1 \beta \quad (6.75)$$

where

$$\alpha = P_x - d_y o_x - d_x a_x \quad (6.76)$$

$$\text{and } \beta = P_y - d_y o_y - d_x a_y. \quad (6.77)$$

Let

$$\alpha = r \cos \phi, \quad r = \sqrt{\alpha^2 + \beta^2}, \quad (6.78), (6.79)$$

$$\beta = r \sin \phi, \quad \phi = \tan^{-1} \left(\frac{\beta}{\alpha} \right). \quad (6.80), (6.81)$$

Substitute for α , β in (6.75).

$$\sin \phi \cos \theta_1 - \cos \phi \sin \theta_1 = \frac{d_2}{r} \quad (6.82)$$

$$\text{or } \sin(\phi - \theta_1) = \frac{d_2}{r} \quad (6.83)$$

and

$$\cos(\phi - \theta_1) = \pm \sqrt{\frac{d_2^2}{1 - (\frac{d_2}{r})^2}} . \quad (6.84)$$

We can then show

$$\theta_1 = \tan^{-1}\left(\frac{\beta}{\alpha}\right) - \tan^{-1}\left(\frac{d_2}{\pm \sqrt{r^2 - d_2^2}}\right) \quad (6.85)$$

where

$$\alpha = P_x - d_y o_x - d_x a_x , \quad (6.86)$$

$$\beta = P_y - d_y o_y - d_x a_y , \quad (6.87)$$

$$r^2 = \alpha^2 + \beta^2 . \quad (6.88)$$

Notice that θ_1 is a function of $\underline{o}$, which is what we are trying to find.

The solution is simplified if we insist that $d_y = 0$.

Now

$$\alpha = P_x - d_x a_x , \quad (6.89)$$

$$\beta = P_y - d_x a_y . \quad (6.90)$$

We can solve for θ_1 since we know $\underline{P}$, $\underline{a}$.

From (6.70) we know

$$n_y = n_x \tan^{-1} \theta_1 . \quad (6.91)$$

We also know that the vectors are unit vectors and that they are orthogonal. Thus,

$$n_x^2 + n_y^2 + n_z^2 = 0 \quad (6.92)$$

and

$$n_x a_x + n_y a_y + n_z a_z = 0 . \quad (6.93)$$

Combining and solving for n_y , we get

$$n_y = \frac{-2n_x a_x a_y \pm [(4n_x^2 a_x^2 a_y^2) - 4(a_y^2 + a_z^2)(n_x^2(a_x^2 + a_z^2) - a_z^2)]^{1/2}}{2(a_y^2 + a_z^2)} \quad (6.94)$$

Combining (6.91) and (6.94) we get

$$n_x = \pm \left[\frac{e a_z^2}{e^2 \tan^2 \theta_1 + 2e \tan \theta_1 a_x a_y + e(a_x^2 + a_z^2)} \right]^{1/2} \quad (6.95)$$

where

$$e = a_y^2 + a_z^2 \quad (6.96)$$

and

$$n_z = \pm \sqrt{1 - n_x^2 - n_y^2} \quad (6.97)$$

Choose the sign of n_z which gives $\underline{a} \cdot \underline{n} = 0$. Note that there are two antiparallel solutions for $\underline{n}$. We can now find $\underline{o}$ simply as

$$\underline{o} = \underline{a} \times \underline{n} \quad (6.98)$$

We could now find explicit solutions of the joint angles as functions of $\underline{P}$ and $\underline{a}$. However, having found $\underline{n}$, $\underline{o}$, we have defined two unique tool orientations for which the 6 degree of freedom PUMA solution is guaranteed to give $\theta_4 = 0$. If we post multiply $\underline{H}_{0,6}$ by the inverse tool transformation $\underline{H}_{5,6}^{-1}$ we get the tool mounting flange location and orientations, i.e.,

$$\underline{H}_{0,5} = \underline{H}_{0,6} \underline{H}_{5,6}^{-1} \quad (6.99)$$

This is equivalent to $\underline{H}_{0,6} = [\underline{n}, \underline{o}, \underline{a}, \underline{P}]$ for the 6 degree of freedom case. Thus the 6 degree of freedom solution for this $\underline{H}_{0,6}$ will yield the

5 degree of freedom joint angles.

6.2.3.2 ASEA IRB60 Solution

For a 5 degree of freedom robot such as the ASEA IRB60, we can specify the tool location $\underline{p}$ and just one tool frame basis vector, $\underline{a}$. The other basis vectors $\underline{n}$, $\underline{o}$ are determined by $\underline{p}$ and $\underline{a}$.

We can solve for $\underline{n}$, $\underline{o}$ as we did previously for the 5 degree of freedom PUMA solution. The complete tool frame is then known. We can modify the 6 degree of freedom PUMA solutions by substituting the ASEA link parameters and solving for the angles given

$$\underline{H}_{R,T} = [\underline{n}, \underline{o}, \underline{a}, \underline{p}] . \quad (6.100)$$

As we noted in Section 6.2.3.1, this $\underline{H}_{R,T}$ is guaranteed to give a wrist rotation $\theta_4 = 0$. Thus, if we neglect the θ_4 solution and reassign the other joint angles as $\theta_5 := \theta_4$, $\theta_6 := \theta_5$ we can generate the solutions for the ASEA IRB60.

The results of Section 6.2.3.1 can be simplified for an ASEA IRB60 by noting that $d_2 = 0$.

From (6.85) we get

$$\theta_1 = \tan^{-1} \left(\frac{P_y - d_x a_{xy}}{P_x - d_x a_{xx}} \right) \quad (6.101)$$

for the following tool transformation:

$$\underline{H}_{5,6} = \begin{bmatrix} 0 & 0 & 1 & d_x \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.102)$$

From (6.91), (6.95) and (6.97) we can calculate $\underline{n}$ as a function of $\underline{a}$, θ_1 .

$$n_x = \pm \left[\frac{(a_y^2 + a_z^2) a_z^2}{(a_y^2 + a_z^2) \tan^2 \theta_1 + 2(a_y^2 + a_z^2) a_x a_y \tan \theta_1 + (a_y^2 + a_z^2)(a_x^2 + a_z^2)} \right]^{1/2}, \quad (6.103)$$

$$n_y = n_x \tan \theta_1, \quad (6.104)$$

$$n_z = \pm \sqrt{1 - n_x^2 - n_y^2}. \quad (6.105)$$

The sign of n_z is chosen to give $\underline{a} \cdot \underline{n} = 0$. Note that there are two antiparallel solutions for $\underline{n}$. Generally only one of the solutions will lead to a feasible set of joint angles.

We can now find $\underline{o}$ from (6.98), i.e.

$$\underline{o} = \underline{a} \times \underline{n}.$$

The joint solutions can be found by plugging the appropriate components of $\underline{n}$, $\underline{o}$, $\underline{a}$, $\underline{p}$ into the joint angle equations from Section 6.2.2.2.

The equations can be simplified for the ASEA IRB60 by noting that the link parameters d_2 , a_3 and the wrist rotation θ_4 are zero.

Rename the joint angles, $\theta_4 := \theta_5$, $\theta_5 := \theta_6$.

The resulting solutions are as follows:

$$\theta_1 = \tan^{-1} \left(\frac{p_y - d_x a_y}{p_x - d_x a_x} \right), \quad (6.106)$$

$$\theta_3 = \tan^{-1} \left[\frac{q_1^2 + q_2^2 + q_3^2 - d_4^2 - a_2^2}{\pm (4d_4^2 a_2^2 - (q_1^2 + q_2^2 + q_3^2 - d_4^2 - a_2^2)^2)^{1/2}} \right], \quad (6.107)$$

$$\theta_2 = \tan^{-1} \left[\frac{d_4 c_3}{d_4 s_3 + a_2} \right] - \tan^{-1} \left[\frac{q_3}{\pm (d_4^2 + d_4 a_2 s_3 + a_2^2 - q_3^2)^{1/2}} \right], \quad (6.108)$$

$$\theta_4 = \tan^{-1} \left[\frac{c_1 c_{23}^a x + s_1 c_{23}^a y - s_{23}^a z}{c_1 s_{23}^a x + s_1 s_{23}^a y + c_{23}^a z} \right], \quad (6.109)$$

$$\theta_5 = \tan^{-1} \left[\frac{-s_1^n x + c_1^n y}{-s_1^o x + c_1^o y} \right], \quad (6.110)$$

where $c_i = \cos \theta_i$, $s_i = \sin \theta_i$, $c_{ij} = \cos(\theta_i + \theta_j)$, $s_{ij} = \sin(\theta_i + \theta_j)$.

The link parameter values for the ASEA IRB60 are:

$$\begin{aligned} a_2 &= 800 \text{ mm}, \\ a_3 &= 1150 \text{ mm}, \\ d_5 &= 200 \text{ mm} + z \text{ component of the tool transformation.} \end{aligned}$$

6.2.3.3 Redundancy of the Sixth Joint for Welding

It is important to note that the welding torch is axisymmetric. Thus, we need only specify the position of the torch tip and the orientation vector of the torch axis. Any arbitrary rotation of the torch frame about the torch axis does not change the effective torch orientation (Fig. 6.4). We find that only 5 degrees of freedom are needed to define a position $\underline{P}$ and a single orientation vector $\underline{a}$. We can show that the other two basis vectors $\underline{n}$, $\underline{o}$, of F_6 are then determined as a function of the joint angles and of $\underline{P}$ and $\underline{a}$.

If we have 6 degrees of freedom, we can independently define the position $\underline{P}$ and any 2 of the basis vectors $\underline{n}$, $\underline{o}$, $\underline{a}$ (the third basis vector is defined by the other 2). This means that we can explicitly rotate F_6 about the torch axis $\underline{a}$ without changing the tools effective orientation (Fig. 6.4). We will use this trick in searching for feasible solutions of a trajectory by specifying a rotation of F_6 about $\underline{a}$, calculating a new $\underline{H}_{0,6}$ and testing the resulting joint angles for feasibility.

6.2.4 Specification of Robot Coordinates

The position of a robot may be uniquely specified two ways:

1. all joint angles specified (explicit); and
2. position and orientation of tool frame specified.

The first case is trivial once we have solved for the joint angles. The second case requires transformation of the tool frame specification F_6 into position and orientation coordinates for the target robot.

We know

$$F_G = [\underline{n}, \underline{o}, \underline{a}, \underline{P}] \quad , \quad (6.111)$$

where

$\underline{n}$, $\underline{o}$, $\underline{a}$ are basis vectors of F_G projected onto F_R ,

and $\underline{P}$ is the position vector of the F_G origin relative to F_R .

For example, for the PUMA 560 robot we must specify coordinates X, Y, Z, O, A, T where X, Y, Z are the Cartesian coordinates of the tool frame F_G relative to the robot base F_R and O, A, T are three Euler rotation angles of F_G about the axes of F_R .

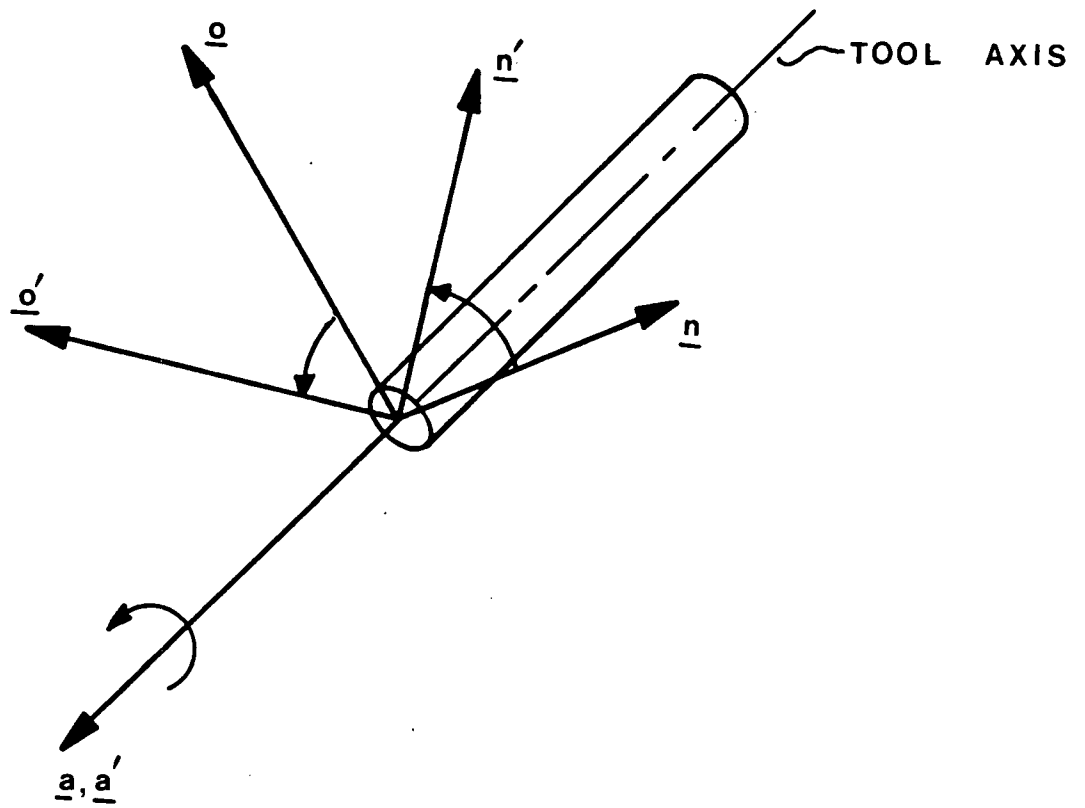


Figure 6.4 Rotation of Tool Frame About Tool Axis.

Calculations of the robot position and orientation specifications from the joint angles or from $\underline{F}_G$ depends on the robot being used and how it defines its locations.

6.2.4.1 Calculation of PUMA Tool Location and Rotation

Coordinates

The PUMA robot defines locations by the Cartesian coordinates of the tool center, X, Y, Z, and three Euler angles o, a, t, the orientation, altitude and tool angles (Fig. 6.5).

For o, a, t angles all equal to zero, the tool frame F_T is related to the robot base frame F_R by a transformation $\underline{H}_{R,T_0}$.

$$\underline{H}_{R,T_0} = \begin{bmatrix} 0 & 1 & 0 & X \\ 0 & 0 & -1 & Y \\ -1 & 0 & 0 & Z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (6.112)$$

where X,Y,Z are the position vectors of the origin of F_T relative to F_R . X, Y, Z do not change with angles o, a, t.

The rotated tool frame F_T is found by performing an Euler angle rotation sequence t, a, o to find the transformation $\underline{H}_{R,T}$.

$$\underline{H}_{R,T} = \underline{\text{rot}}(o,3) \underline{\text{rot}}(a,1) \underline{\text{rot}}(t,2) \underline{H}_{R,T_0}, \quad (6.113)$$

where $\underline{\text{rot}}(o,3)$ means rotate about z axis ($\underline{i}_3$) of F_R by an angle o, etc.

$$\underline{\text{rot}}(o,3) = \begin{bmatrix} \cos o & -\sin o & 0 & 0 \\ \sin o & \cos o & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (6.114)$$

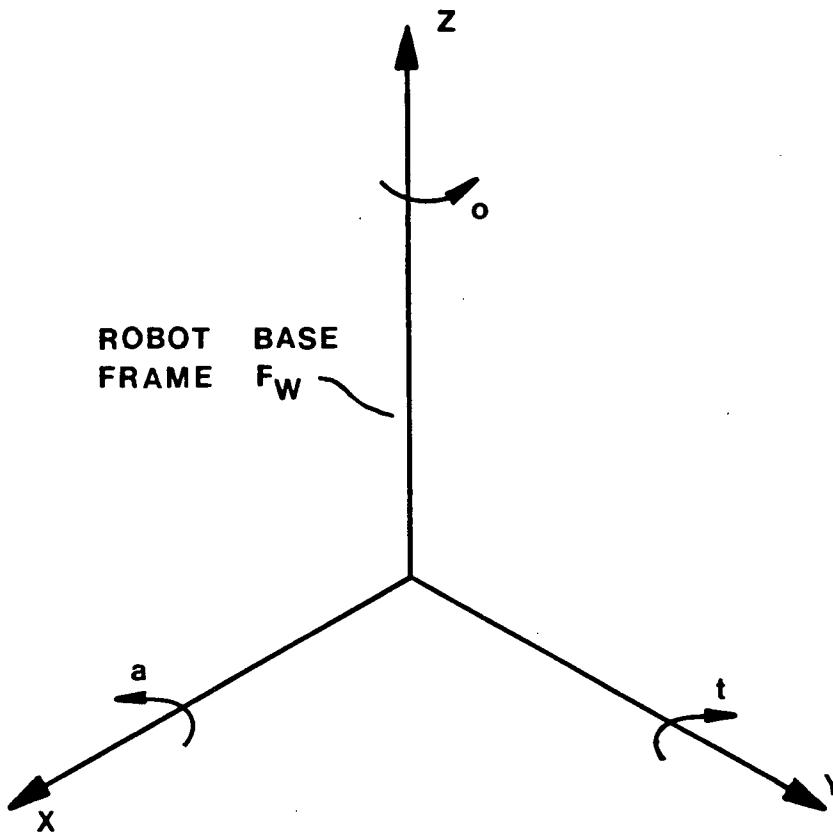


Figure 6.5 Definition of o , a , t angles for the PUMA Arm.

$$\underline{\text{rot}}(a,1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a & -\sin a & 0 \\ 0 & \sin a & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.115)$$

$$\underline{\text{rot}}(t,2) = \begin{bmatrix} \cos t & 0 & \sin t & 0 \\ 0 & 1 & 0 & 0 \\ -\sin t & 0 & \cos t & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad (6.116)$$

Multiplying out we get

$$\underline{H}_{R,T} = \begin{bmatrix} -C_o S_t - S_o S_a C_t & C_o C_t - S_o S_a S_t & S_o C_a & X \\ -S_o S_t + C_o S_a C_t & S_o C_t + C_o S_a S_t & -C_o C_a & Y \\ -C_a C_t & -C_a S_t & -S_a & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} , \quad (6.117)$$

where

$$\begin{aligned} C_o &\equiv \cos(o), \\ S_o &\equiv \sin(o), \text{ etc.} \end{aligned}$$

If $a = \pm 90^\circ$, rotations o, t are coaxial. In this case we will arbitrarily set $t = 0$.

To find the angles o, a, t from a known transformation $\underline{H}_{R,T}$ we proceed as follows:

$$\text{We know that } \underline{H}_{R,T} = [\underline{VN}, \underline{VO}, \underline{VA}, \underline{VP}] \quad (6.118)$$

where

$$\underline{VP} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \quad \underline{VN} = \begin{bmatrix} VN_1 \\ VN_2 \\ VN_3 \\ 0 \end{bmatrix}, \text{ etc.}$$

Premultiply both sides of (6.118) by $\underline{rot}(o,z)^{-1}$,

$$\underline{rot}(o,z)^{-1} \underline{H}_{R,T} = \underline{rot}(o,z)^{-1} [\underline{VN}, \underline{VO}, \underline{VA}, \underline{VP}] . \quad (6.119)$$

Multiplying out and equating elements we get

$$0 = -\sin(o) VA_1 - \cos(o) VA_1 , \quad (6.120)$$

$$\cos(t) = \cos(o) VO_1 + \sin(o) VO_2 , \quad (6.121)$$

$$\sin(t) = -\cos(o) VN_1 - \sin(o) VN_2 , \quad (6.122)$$

$$\sin(a) = -VA_3 , \quad (6.123)$$

$$\cos(a) = \sin(o) VA_1 - \cos(o) VA_2 . \quad (6.124)$$

From these equations we get

$$o = \tan^{-1} \left[\frac{-VA_1}{VA_2} \right] , \quad o = o + 180 , \quad (6.125)$$

$$a = \tan^{-1} \left[\frac{-VA_3}{\sin(o) VA_1 - \cos(o) VA_2} \right] , \quad (6.126)$$

$$t = \tan^{-1} \left[\frac{-\cos(o) VN_1 - \sin(o) VN_2}{\cos(o) VO_1 + \sin(o) VO_2} \right] . \quad (6.127)$$

For the PUMA, the angle t is defined as the negative of the usual Euler rotation convention. If we change the sign of t to conform to the PUMA definition we get:

$$t = \tan^{-1} \left[\frac{\cos(o) VN_1 + \sin(o) VN_2}{\cos(o) VO_1 + \sin(o) VO_2} \right] . \quad (6.128)$$

Also the following is true:

$$(o, a, t) \equiv (o + 180, 180 - a, t + 180).$$

6.3 Collision Avoidance

In order to successfully generate a robot program off-line, it is essential that we be able to reliably avoid collisions between the robot and objects in the workstation. This is a difficult problem which is the subject of much current research.

To handle this problem our world model must contain complete geometric descriptions of the robot and objects in the workstation. The geometric models can be provided by CAD systems in several forms. Solids are usually represented by a description of their surfaces, or as a composite of primitive solid elements. The most common surface representation is the polyhedral model which defines a solid polyhedron by specifying all of its vertices, edges and faces [Hosaka, Kimura and Kakishita 1974; Wesley et al 1980]. The resulting description is simple, uniform and unambiguous.

More advanced CAD systems use constructive solid geometry to generate complex solids by performing boolean operations on a set of simple solid primitives such as boxes and cylinders [Braid 1975; Requicha 1980]. The resulting representation must contain a record of the composite primitives as well as the sequence of operations. More complete surveys of solid representations can be found in the literature [Baer, Eastman and Henrion 1979; Requicha 1980].

The collision avoidance problem has been well summarized by Lozano-Pérez [Brady et al 1982]. The three classes of collision avoidance algorithm are: hypothesize and test, penalty functions and explicit free space.

The hypothesize and test algorithm generates a collision free path by trial and error. Each time a collision is detected an alternate path is formulated and tried. This method is simple but does not lead to an optimum solution and may not find a solution at all because the knowledge of the world is localized only and thus a global strategy is impossible.

The penalty function scheme attempts to define a penalty function whose value reflects the probability of collision as a function of position in space. The algorithm then generates a path following the local minima of the penalty function. This method has been successfully applied to extremely simple world models only, and the algorithm becomes very complex for more realistic situations.

The third class of algorithm generates a set of all robot positions which are free of collisions and calls this set the free space. The shortest path through the free space joining an initial and final position is then the optimum path. The main difficulty lies in generating the free space, and all algorithms to date incorporate simplifications and approximations which restrict their generality. Several free space path planning algorithms have been presented in the literature [Udupa 1977; Lozano-Pérez and Wesley 1979; Lozano-Pérez 1983; Brooks 1983].

The problem of simply detecting collisions between the robot and objects in the workstation is a simpler problem to tackle at this point. However, a feasible path must then be found iteratively or by trial and error. We will not consider the alternate path selection problem, other than to note that in the case of welding we must reposition the workpiece if a collision is detected.

The simplest interference detection problem is the problem of interference between fixed, stationary objects. Hosaka [1974] describes a method for completely specifying the intersection between two stationary convex polyhedra. Maruyama [1972] has proposed a simple method for determining intersections between polyhedral solids using a hierarchical set of tests.

The problem of interference between moving and stationary polyhedra was considered by Boyse [1979]. The tests which he devised apply only to the simple cases of pure translation and rotation and cannot be extended to the more general case of complicated motion. An algorithm for testing for interference between two moving convex polyhedra is suggested by Schwartz [1981]. His method finds the minimum distance between two convex polyhedra as a function of time if their individual trajectories are known functions of time. Schwartz demonstrates the algorithm for the two dimensional case only.

Another way of handling interference detection between moving objects is to define a swept volume generated by the moving object. The problem is then reduced to the simpler case of interference between stationary solids. This method is discussed by Requicha [1980] and Lozano-Pérez and Wesley [1979]. We will develop a detailed interference detection algorithm based on the swept volume method.

6.3.1 World Model

Any world model representation embodies a trade-off between accuracy and conciseness for solids of arbitrary shapes. If we use a polyhedral representation we can choose any number of surface facets to

approximate curved surfaces to any degree of accuracy. In practice, however, we will generally use a coarse approximation to reduce the size of the data structure and increase the speed of any processing we wish to do.

Unfortunately, any simplifying approximations that we make introduce uncertainty into our collision predictions. We find that we must trade off the simplicity of our mathematical model and algorithm against the reliability of the resulting collision predictions. We can bias our model for the worst case, however, to ensure that all collisions will be predicted. The price we pay is a higher probability of collisions being predicted where none occur. This can be an important consideration if we must move near the objects in the workstation, as in welding.

The world model which we will adopt will be a simple one for the purpose of demonstrating the basic concepts of interference detection. The general approach which will be described can be applied to a more sophisticated and accurate world model and interference detection scheme.

We will consider a world model which divides the workstation into two sets of objects, moving solids and stationary solids (Fig. 6.6). The moving solids include the robot manipulator and all parts attached to it. Stationary solids include the workpiece, positioning table, fixtures, etc. Although the table may be movable we will assume that its motion cannot cause interference.

Stationary objects in the workstation will be modelled as solid polyhedra. A polyhedral representation allows our model to have any

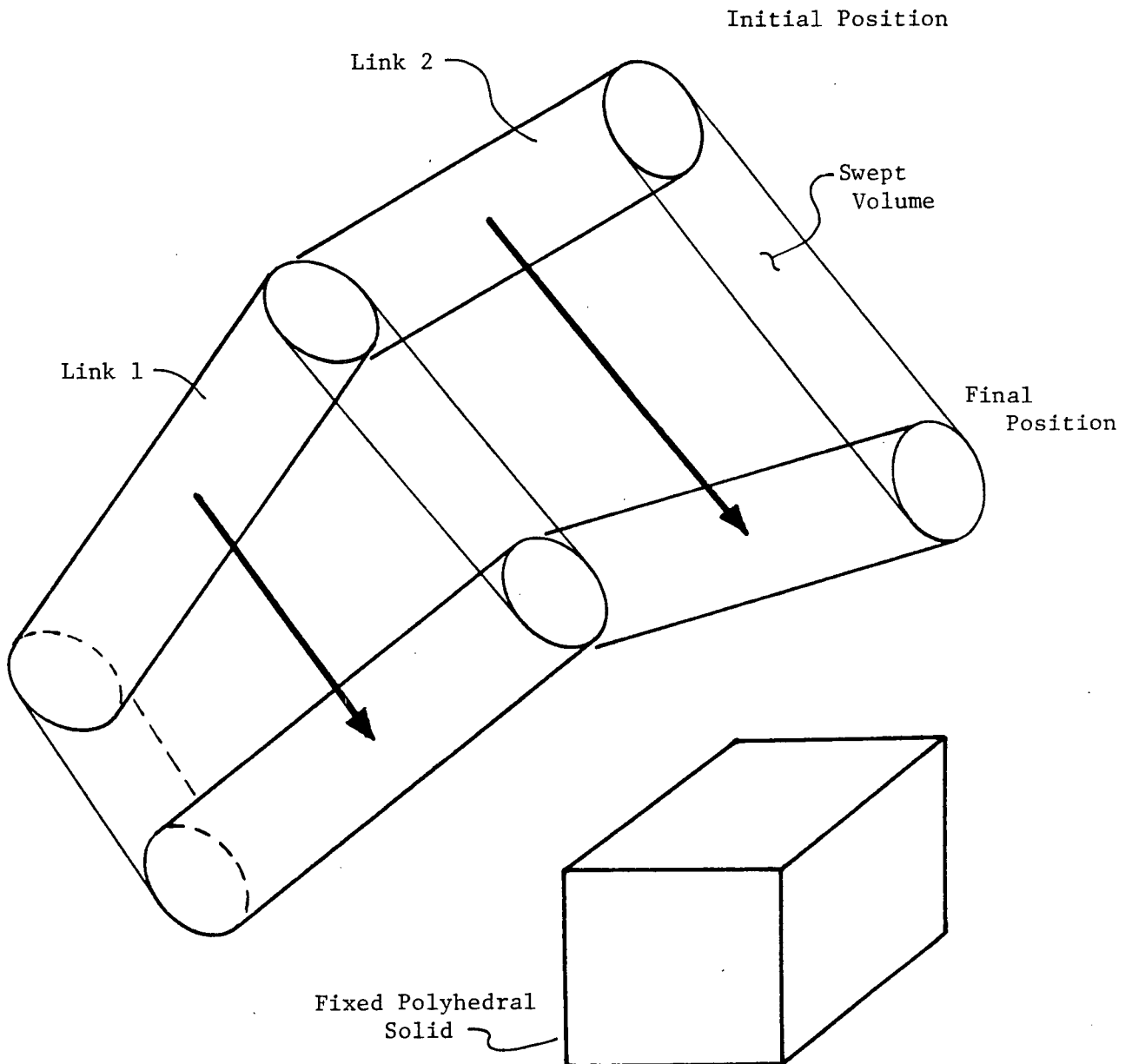


Figure 6.6 World Model

degree of accuracy we desire, and allows a simple data structure of vertices, plane faces and straight edges.

For simplicity, we will model the robot links as circular cylinders. The axes of the cylinders will coincide with the axes of the links and the cylinder radius and length will be chosen such that the cylinder fully contains the link (see Fig. 6.7). Each cylinder will be defined relative to the corresponding link coordinate frame.

6.3.2 Development of Swept Volume

As the robot links move they sweep out a volume of space, as shown in Fig. 6.6. If we can analytically describe that volume, we can test if the volume intersects any fixed solid in the workstation.

In order to simplify the algorithm and minimize the required calculations we will adopt a method for approximating the swept volume in a concise way. To illustrate the method, we will consider the motion of one cylindrical solid through space.

Suppose we have a cylinder of radius R and length L . It has a central axis or spine with endpoints A and B . If the cylinder is moved from an initial position P_1 to some other position P_2 , then some volume has been swept out by the cylinder. If we assume the motion was smooth and continuous between P_1 and P_2 , we can use the following approximation for the swept volume.

The initial and final endpoints of the spine are connected with straight lines A_1A_2 , B_1B_2 . These line segments, along with the initial and final spine positions, define a ruled, parametric surface. The surface has a simple, non-linear parametric equation of the form

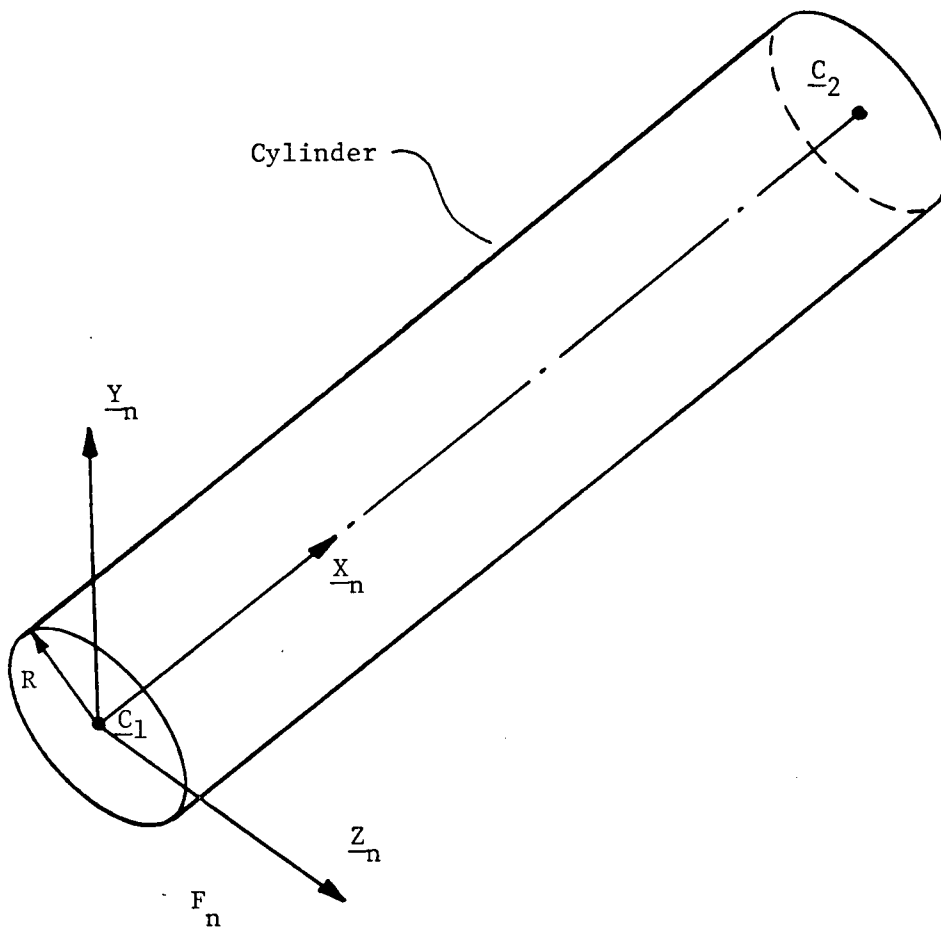


Figure 6.7 Cylindrical Approximation of a Manipulator Link.

$$\underline{r}_p(\lambda_u, \lambda_v) = \underline{r}_o + \lambda_u \underline{u}_o + \lambda_v \underline{v}_o + \lambda_u \lambda_v (\underline{u}_1 - \underline{u}_o) \quad (6.129)$$

where $\underline{r}_o$, $\underline{u}_o$, $\underline{v}_o$, $\underline{u}_1$ are defined as shown in Fig. 6.8.

If we restrict the ranges of the parameters to $0 \leq \lambda_u \leq 1$, $0 \leq \lambda_v \leq 1$, we obtain a surface patch or region bounded by the vectors $\underline{u}_o$, $\underline{u}_1$, $\underline{v}_o$, $\underline{v}_1$ as shown in Fig. 6.8. We can now define two bounding parametric surfaces offset from this parent surface by a distance R on each side. To do this, we define the corner points of the new surfaces by finding the surface normals of the parent surface at each corner, and translating along those normals by distances $\pm R$.

The two offset bounding surfaces can be expressed relative to the parent surface by the following equation:

$$\underline{r}_b(\lambda_u, \lambda_v) = \underline{r}_p(\lambda_u, \lambda_v) \pm R \underline{n}(\lambda_u, \lambda_v) \quad (6.130)$$

where R is the cylinder radius,

λ_u, λ_v are the surface parameters,

$\underline{n}$ is the unit surface normal of the parent surface,

$\underline{r}_p$ is any point on the parent surface

satisfying equation (6.129).

The unit surface normal is defined as follows:

$$\underline{n} = \frac{\underline{v}_u \times \underline{v}_v}{|\underline{v}_u \times \underline{v}_v|} \quad (6.131)$$

$$\text{where } \underline{v}_u = \frac{\partial \underline{r}_p}{\partial \lambda_u} = \underline{u}_o + \lambda_v (\underline{u}_1 - \underline{u}_o), \quad (6.132)$$

$$\underline{v}_v = \frac{\partial \underline{r}_p}{\partial \lambda_v} = \underline{v}_o + \lambda_u (\underline{u}_1 - \underline{u}_o). \quad (6.133)$$

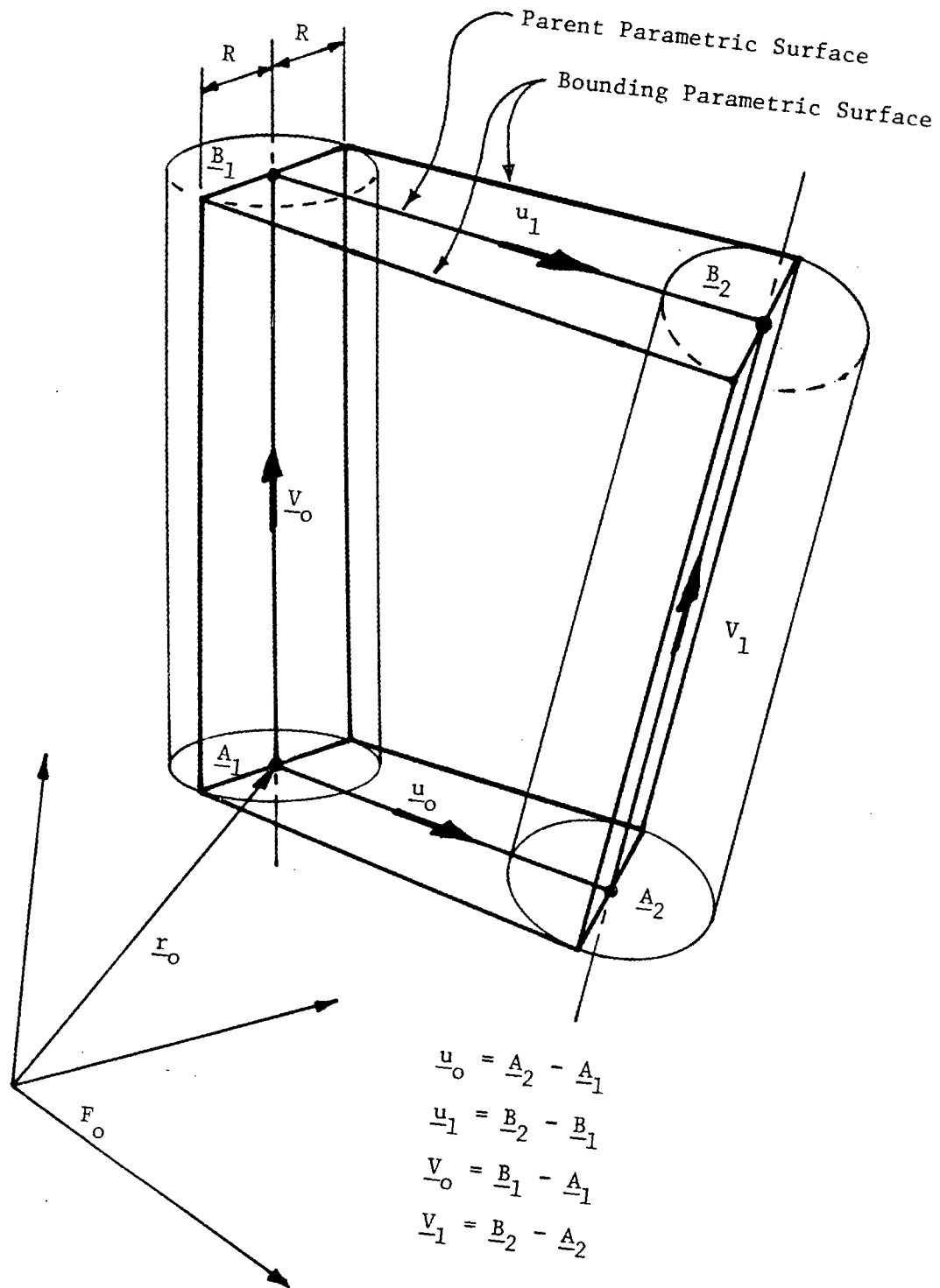


Figure 6.8 Parametric Approximation of Swept Volume.

If we expand these equations and collect the terms, we find that the bounding surfaces can be expressed by equations of the form

$$\underline{r}_b = \underline{A} + \lambda_u \underline{B} + \lambda_v \underline{C} + \lambda_u \lambda_v \underline{D} \quad (6.134)$$

where $\underline{A}$, $\underline{B}$, $\underline{C}$, $\underline{D}$ are known.

Thus, the bounding surfaces are also ruled, parametric surfaces.

Given these two bounding surfaces, we can easily derive similar surfaces over each of the four open faces of the volume between the bounding surfaces. Only two of these surfaces need to be found since the other two faces are bounded by the cylindrical surfaces of the link in its initial and final position.

The interference detection problem is now to find intersections between the polyhedra representing stationary objects and the parametric swept volumes generated by the moving objects. A series of simple tests can be performed which will determine if intersection occurs.

In practice, we would generate swept volumes for each moving link of interest. The motion of the manipulator would be divided into a series of intermediate motions, with each intermediate motion generating a swept volume.

6.3.3 Interference Conditions

We can formulate five simple intersection conditions (see Fig.6.9):

1. Any solid polyhedral edge intersects any swept volume bounding parametric surface.
2. Any polyhedral edge intersects the cylindrical solid of a link in its initial or final position.

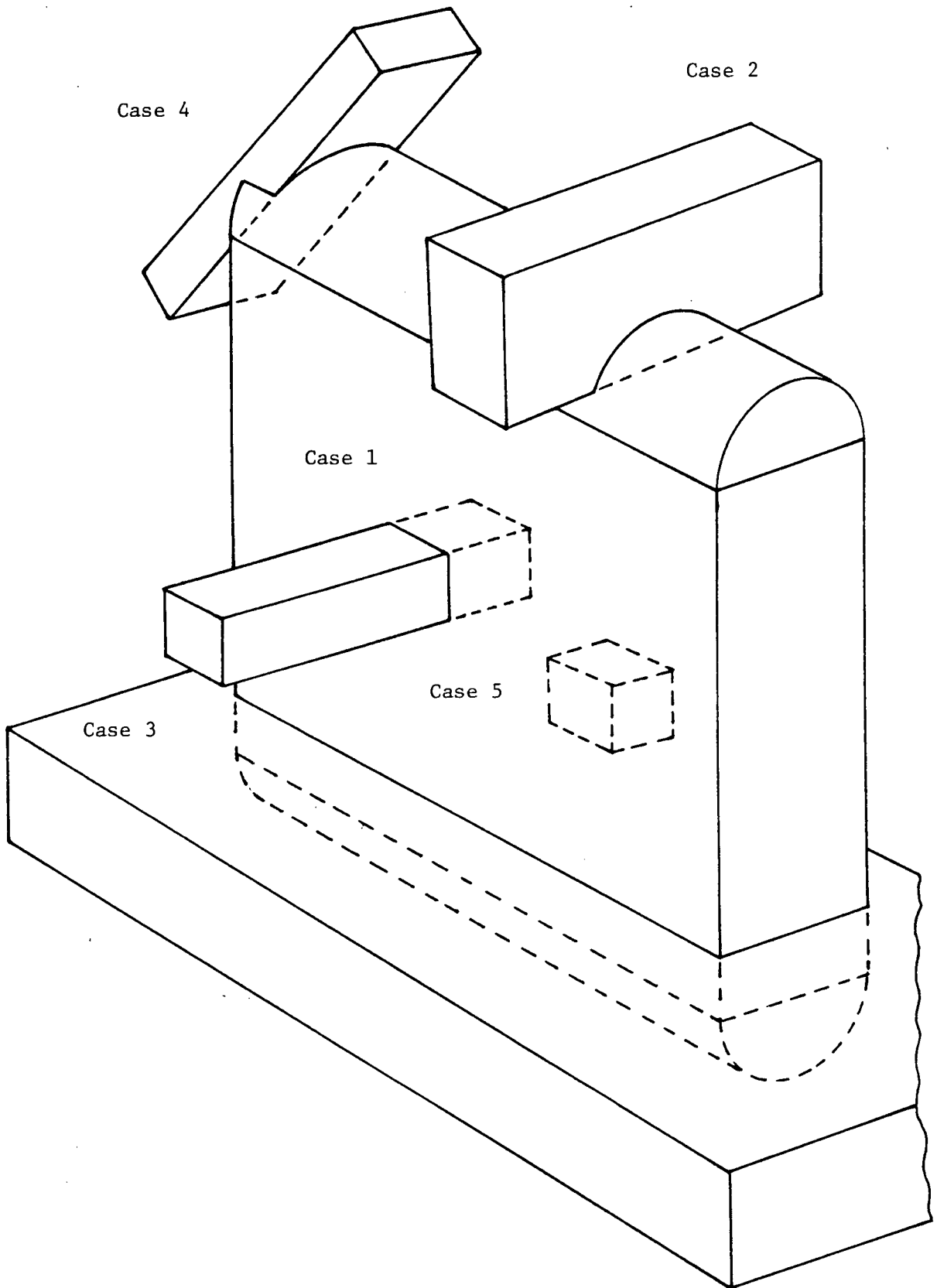


Figure 6.9 Interference Conditions.

3. Any swept surface edge intersects any solid polyhedral surface.
4. The cylindrical volume of any link in its initial or final position intersects any polyhedral face.
5. Any polyhedral solid is wholly enclosed by any swept volume.

The interference algorithm now consists of testing for each of these cases for every possible combination of solid polyhedra and swept volumes. A hierarchy of tests can be devised to determine if intersection will occur, as shown in Fig. 6.10. Each test will now be considered in detail.

6.3.3.1 Condition 1 - Intersection of an Edge and a Parametric Surface (Fig. 6.11)

If we are given 4 points in space, $\underline{P}_1, \underline{P}_2, \underline{P}_3, \underline{P}_4$, we can define a ruled, parametric surface with the following equation:

$$\underline{r} = \underline{P}_1 + \lambda_u \underline{u}_0 + \lambda_v \underline{v}_0 + \lambda_u \lambda_v (\underline{u}_1 - \underline{u}_0) , \quad (6.135)$$

where $\underline{u}_0 = \underline{P}_2 - \underline{P}_1$,

$$\underline{v}_0 = \underline{P}_4 - \underline{P}_1 ,$$

$$\underline{u}_1 = \underline{P}_3 - \underline{P}_4 ,$$

λ_u, λ_v are scalar parameters.

Also, if we are given the end points of a line segment, $\underline{E}_1, \underline{E}_2$, we can define a line with the following equation:

$$\underline{r} = \underline{E}_1 + \lambda_E \underline{u}_E \quad (6.136)$$

where $\underline{u}_E = \underline{E}_2 - \underline{E}_1$

Combining (6.135), (6.136) we get the following system of equations:

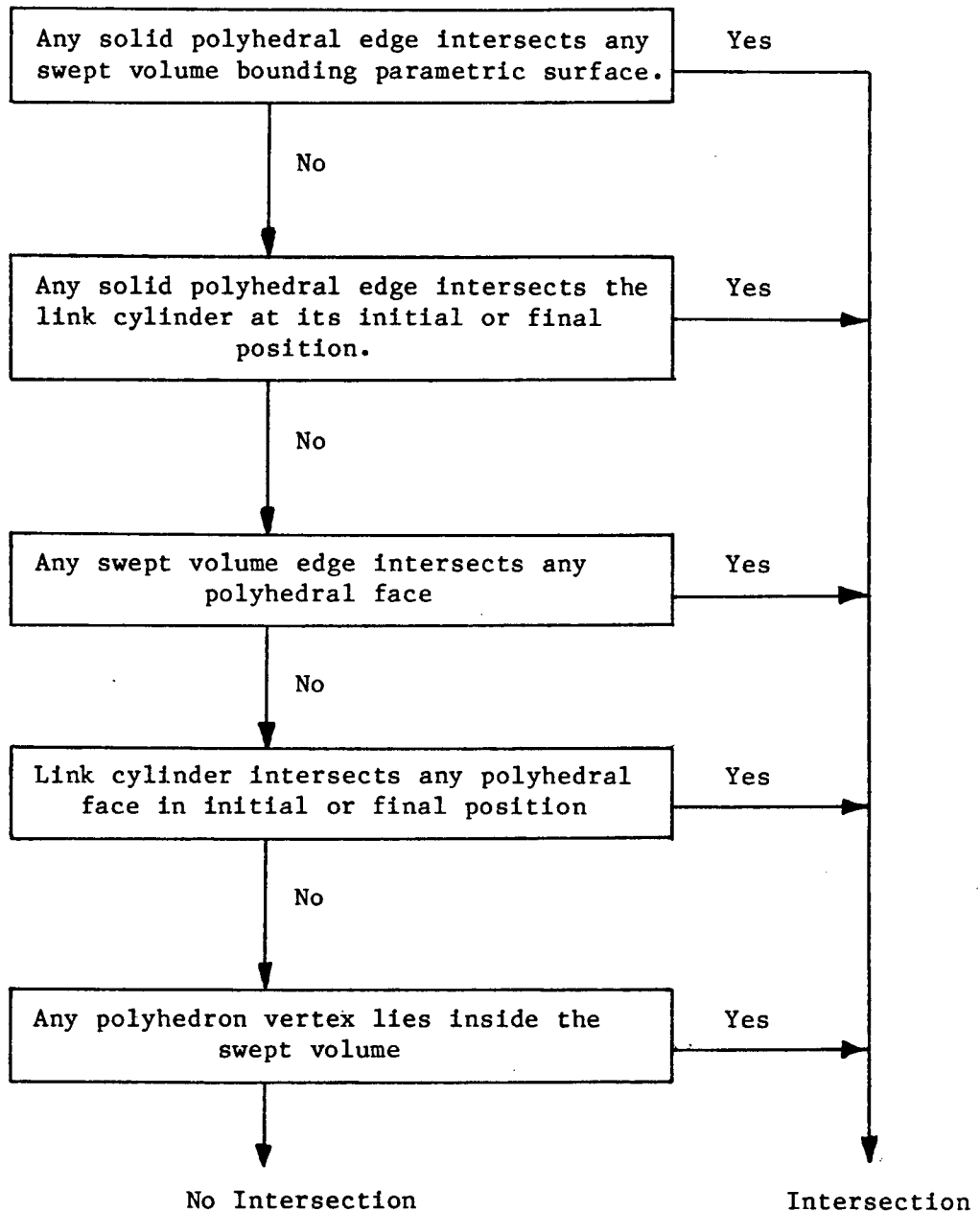


Figure 6.10 Algorithm for Interference Detection.

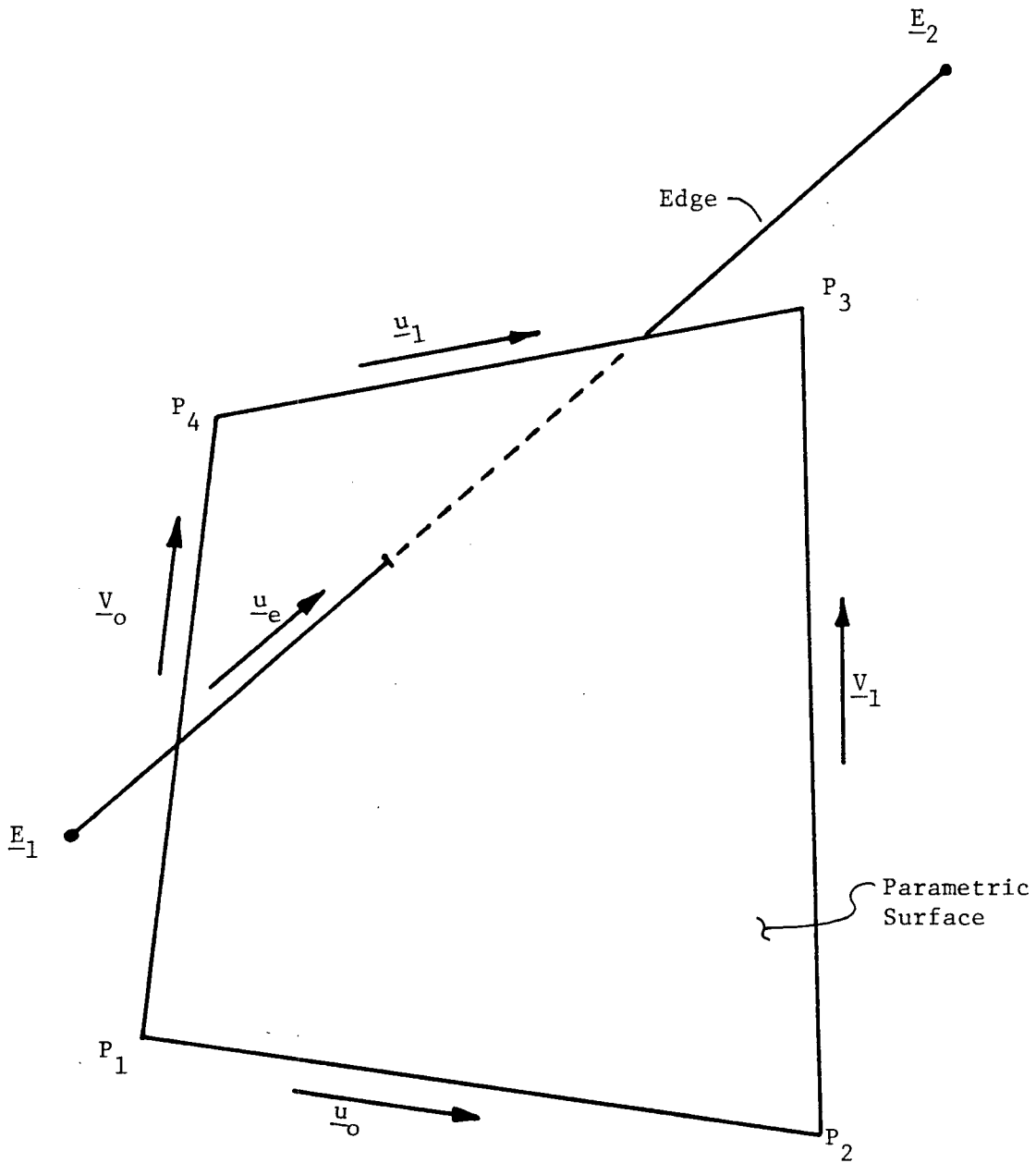


Figure 6.11 Intersection of an Edge and a Parametric Surface.

$$\lambda_{\underline{u}-\underline{0}} + \lambda_{\underline{v}-\underline{0}} - \lambda_{\underline{E}-\underline{E}} + \lambda_{\underline{u}} \lambda_{\underline{v}} (\underline{u}_1 - \underline{u}_0) + \underline{P}_1 - \underline{E}_1 = 0 . \quad (6.137)$$

This system of non-linear equations can be solved iteratively using Newton's method. The edge intersects the surface within its boundaries if $0 < \lambda_{\underline{u}} < 1$, $0 < \lambda_{\underline{v}} < 1$, $0 < \lambda_{\underline{E}} < 1$ at the intersection point.

6.3.3.2 Condition 2 - Intersection of an Edge and a Cylinder (Fig. 6.12)

Suppose the edge is defined as for Condition 1 and is represented by equation (6.136). The cylinder is defined by two points $\underline{C}_1$ and $\underline{C}_2$, and a radius R. $\underline{C}_1$ and $\underline{C}_2$ define the ends of the cylinder and the line joining $\underline{C}_1$ and $\underline{C}_2$ defines the central axis or spine of the cylinder. The ends of the cylinder are flat and normal to the spine.

The edge and the spine lie on two skewed lines in space. We can find a common normal and intersection points $\underline{I}_1$, $\underline{I}_2$ on the edge and spine (see Fig. 6.12).

The common normal and distance between two skewed lines can be found as follows. Suppose the two lines have the following equations:

$$\underline{\text{Line 1}} \text{ (edge)} \quad \underline{r} = \underline{E}_1 + \lambda_{\underline{E}} \underline{u}_{\underline{E}} \quad (6.138)$$

$$\text{where} \quad \underline{u}_{\underline{E}} = \underline{E}_2 - \underline{E}_1 .$$

$$\underline{\text{Line 2}} \text{ (spine)} \quad \underline{r} = \underline{C}_1 + \lambda_{\underline{s}} \underline{u}_{\underline{s}} \quad (6.139)$$

$$\text{where} \quad \underline{u}_{\underline{s}} = \underline{C}_2 - \underline{C}_1 .$$

We can show that the closest distance D is given by

$$D = \left| \frac{(\underline{C}_1 - \underline{E}_1) \cdot (\underline{u}_{\underline{E}} \times \underline{u}_{\underline{s}})}{|\underline{u}_{\underline{E}} \times \underline{u}_{\underline{s}}|} \right| . \quad (6.140)$$

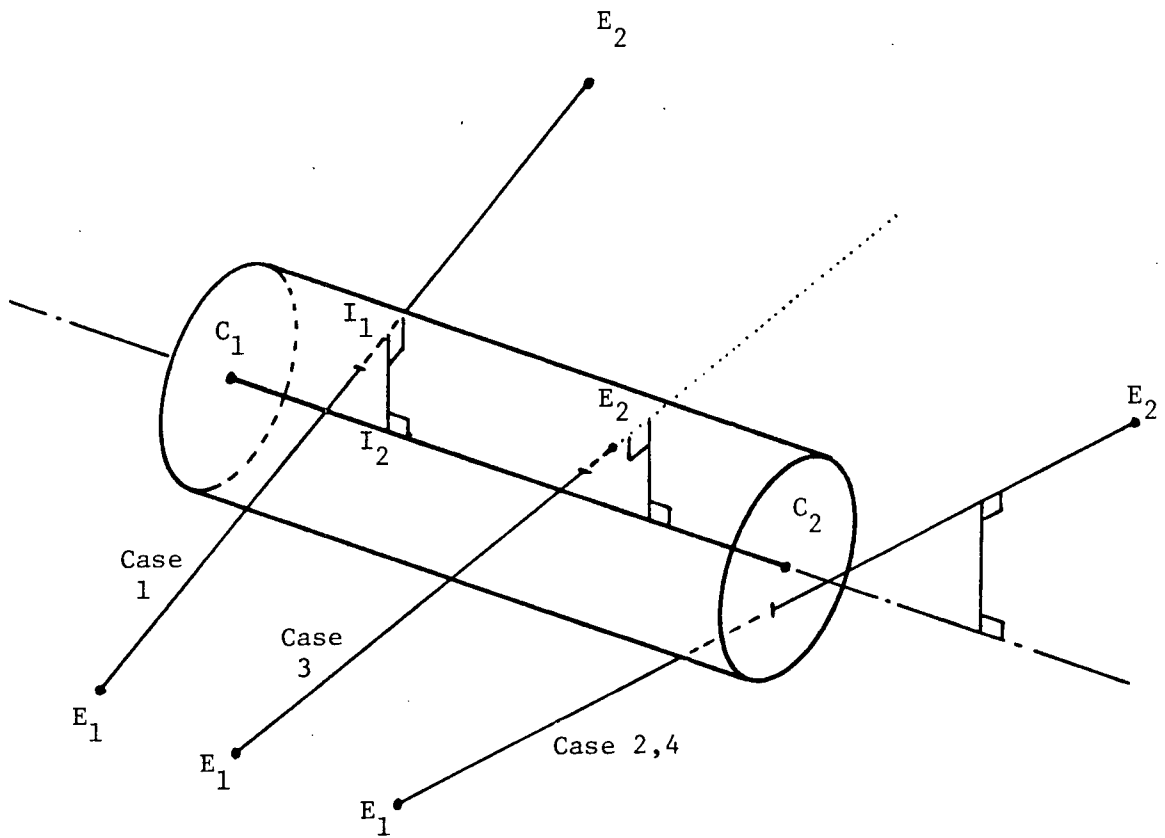


Figure 6.12 Intersection of an Edge and a Cylinder.

The points on each line lying on the common normal at the closest distance are found by substituting the following parameter values into equations (6.138), (6.139):

$$\lambda_E = (\underline{C}_1 - \underline{E}_1) \cdot \underline{u}_E, \quad (6.141)$$

$$\lambda_S = (\underline{E}_1 - \underline{C}_1) \cdot \underline{u}_S. \quad (6.142)$$

Let us call these points $\underline{P}_E$, $\underline{P}_S$.

$\underline{P}_E$ lies on the edge if $0 < \lambda_E < 1$.

$\underline{P}_S$ lies on the spine if $0 < \lambda_S < 1$.

We can now define four different cases of an edge intersecting a cylinder. Each case is handled differently.

Case 1 $\underline{P}_E$ lies on the edge, and

$\underline{P}_S$ lies on the spine.

In this case, intersection occurs if the minimum distance D given by (6.140) is less than the radius R .

Case 2 $\underline{P}_E$ lies on the edge, and

$\underline{P}_S$ does not lie on the spine

In this case, if the edge intersects either end face of the cylinder it intersects the cylinder. This can be tested in several steps:

1. Test if edge intersects face plane of cylinder.
2. If step 1 is true, find intersection point.
3. Find distance from spine to intersection point.
4. If distance is less than the radius, intersection has occurred.

Case 3. $\underline{P_E}$ does not lie on the edge, and

$\underline{P_S}$ lies on the spine.

In this case, we find the minimum distance from each end point of the edge to the spine. If either distance is less than the radius R , intersection occurs.

$$\text{i.e. if } | \underline{E_1} - \underline{P_S} | < R$$

$$\text{or } | \underline{E_2} - \underline{P_S} | < R$$

intersection occurs.

Case 4. $\underline{P_E}$ does not lie on the edge, and

$\underline{P_S}$ does not lie on the spine.

This case is treated the same as case 2.

6.3.3.3 Condition 3 - Intersection of an Edge and a Polyhedral Face (Fig.6.13)

Let us define an edge by equation (6.136). A polyhedral face is defined as the planar region bounded by an n -sided polygon with n coplanar vertices, $\underline{V_1}, \underline{V_2}, \dots \underline{V_n}$.

The face is a region of a known plane with the following equation:

$$\underline{r} \cdot \underline{n} - Q = 0 \quad (6.143)$$

where $\underline{n}$ is the unit surface normal,

Q is the closest distance to the origin.

The edge is defined by equation (6.136),

$$\text{i.e. } \underline{r} = \underline{E_1} + \lambda_E \underline{u_E}$$

$$\text{where } \underline{u_E} = \underline{E_2} - \underline{E_1}.$$

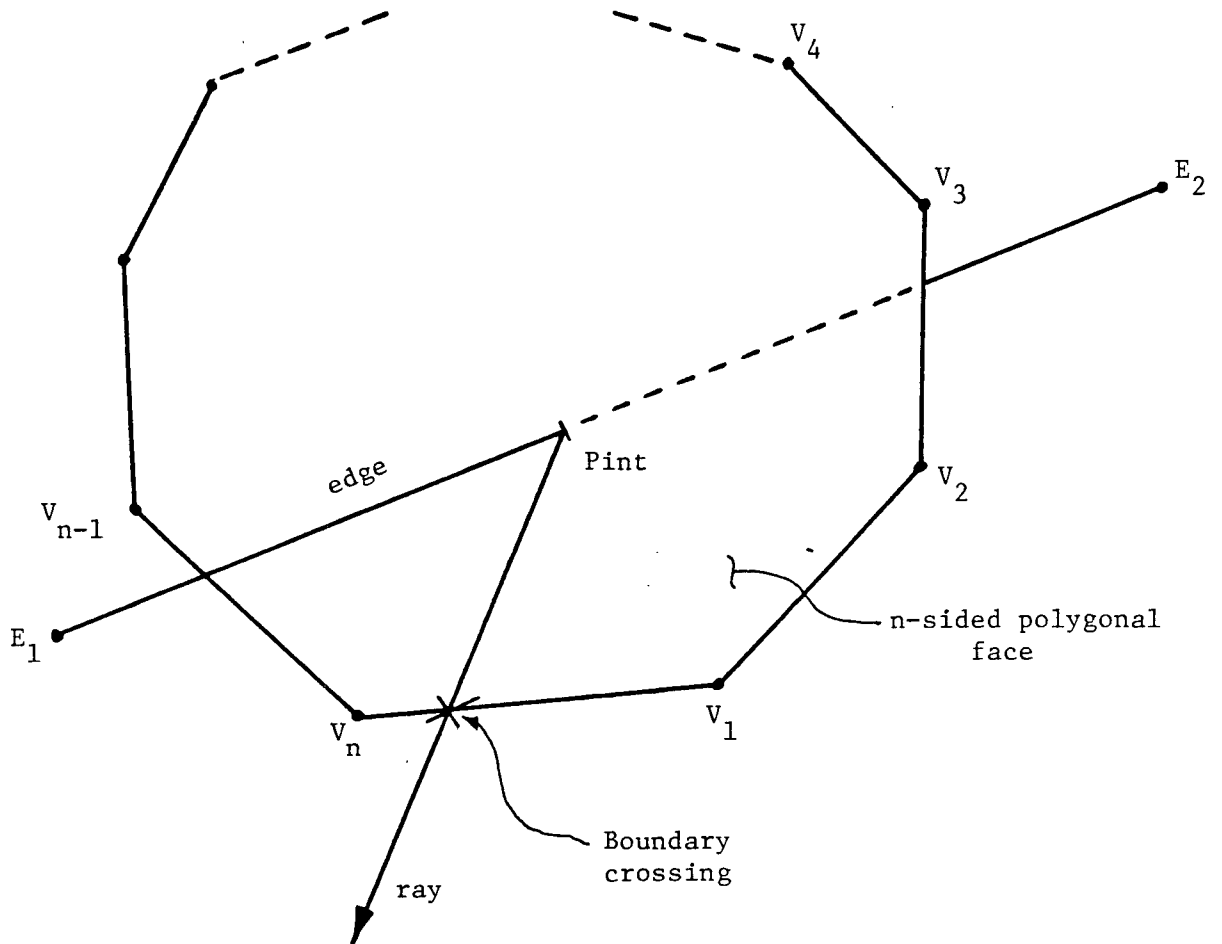


Figure 6.13 Intersection of an Edge and a Polyhedral Face.

Combining (6.136) and (6.143) we get the following equation for the parameter λ_E :

$$\lambda_E = \frac{Q - \underline{E}_1 \cdot \underline{n}}{\underline{u}_E \cdot \underline{n}} .$$

If $\underline{u}_E \cdot \underline{n} = 0$, the line is parallel to the plane and no intersection exists. The edge intersects the face plane if $0 < \lambda_E < 1$.

We can now formulate a series of tests to determine if the edge intersects the face.

1. Test if the edge intersects the face plane.
2. If (1) is true, generate any ray originating from the intersection point $\underline{P}_{int}$ and lying on the face plane.
3. If the ray crosses the face boundaries an odd number of times, intersection has occurred.

6.3.3.4 Condition 4 - Intersection of a Cylinder and a Polyhedral Face

It is possible for a cylinder to intersect a face without any of the face edges intersecting the cylinder. There are two possible intersection cases:

1. The cylinder spine intersects the face plane inside the face boundaries.
2. At least one cylinder end face intersects the face plane inside the face boundary (Fig.6.14).

Tests can be devised for each of these cases.

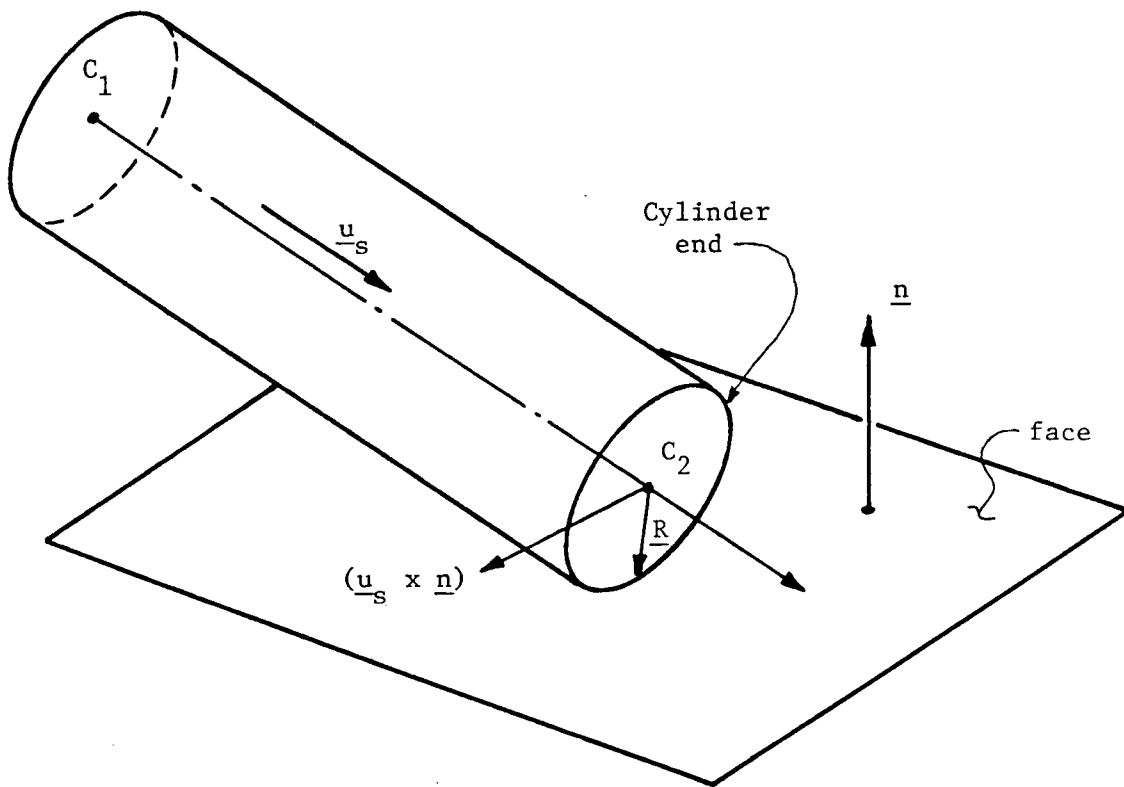


Figure 6.14 Intersection of Cylinder End Face and Polyhedral Face.

Case 1 This test is the same as for intersection between an edge and a face, as previously described.

Case 2 In this case, we will find the shortest distance from the cylinder face center to the intersection line of the cylinder face plane and polyhedral face plane. If the distance is less than the radius R , intersection occurs.

Let us define a vector $\underline{R}'$ in the cylinder end face

$$\underline{R}' = \underline{u}_s \times (\underline{u} \times \underline{n}) \quad (6.145)$$

where $\underline{u}_s = \underline{C}_2 - \underline{C}_1$, spine direction vectors,

$\underline{n}$ = polyhedral face surface normal.

Let us now scale $\underline{R}'$ to length R , the radius

$$\underline{R} = R \frac{\underline{R}'}{|\underline{R}'|} . \quad (6.146)$$

We can now define two lines lying in the cylinder end faces, one in each end;

$$\underline{r}_1 = \underline{C}_1 + \lambda \underline{R} , \quad (6.147)$$

and

$$\underline{r}_2 = \underline{C}_2 + \lambda \underline{R} . \quad (6.148)$$

We now test if either line intersects the polyhedral face. Intersection occurs if $-1 < \lambda < 1$ and if the intersection point $\underline{P}_{int}$ lies inside the polyhedral face boundaries.

6.3.3.5 Condition 5 - Any Polyhedron is Totally Enclosed by a Swept Volume Bounded by Parametric Surfaces

We can now write an equation for a parametric swept volume in the following form:

$$\begin{aligned} \underline{r}(\lambda_u, \lambda_v, \lambda_w) &= \underline{r}_1(\lambda_u, \lambda_v) + \lambda_w \underline{R} \underline{n}, \quad 0 \leq \lambda_u \leq 1, \\ &0 \leq \lambda_v \leq 1, \\ &-1 \leq \lambda_w \leq 1, \end{aligned} \quad (6.149)$$

where $\underline{R}_1 = \underline{P} + \lambda_{u=0} \underline{u} + \lambda_{v=0} \underline{v} + \lambda_u \lambda_v (\underline{u}_1 - \underline{u}_0)$

and $\underline{n} = \frac{\underline{\nabla}_u \times \underline{\nabla}_v}{|\underline{\nabla}_u \times \underline{\nabla}_v|}$,

where $\underline{\nabla}_u, \underline{\nabla}_v$ are as defined in equations (6.132), (6.153). In other words, $\underline{r}_1$ is any point on the parent parametric surface and $\underline{n}$ is the unit surface normal.

Let $\underline{n}' = \underline{\nabla}_u \times \underline{\nabla}_v$. (6.150)

This can be written as

$$\underline{n}' = \underline{L} + \lambda_u \underline{M} + \lambda_v \underline{N} \quad (6.151)$$

We can rewrite (6.149) as

$$\underline{r} = \underline{r}_1 + \lambda_w \underline{n}', \quad -\alpha \leq \lambda_w \leq \alpha, \quad (6.152)$$

where $\alpha = \frac{R}{|\underline{n}'|}$.

Expanding and rearranging,

$$\underline{r} = \lambda_{u=0} \underline{u} + \lambda_{v=0} \underline{v} + \lambda_w \underline{L} + \lambda_u \lambda_v (\underline{u}_1 - \underline{u}_0) + \lambda_w \lambda_u \underline{M} + \lambda_w \lambda_v \underline{N} + \underline{P}, \quad (6.153)$$

$$\begin{aligned}0 &\leq \lambda_u \leq 1, \\0 &\leq \lambda_v \leq 1, \\-\alpha &\leq \lambda_w \leq \alpha.\end{aligned}$$

The result is a system of non-linear equations which can be solved iteratively using Newton's method.

If all previous intersection tests have been false, and if any vertex of a polyhedron lies within the swept volume, then all vertices and thus the entire polyhedron lies within the volume. It is sufficient to test any single arbitrary vertex of the polyhedron.

The method is as follows:

1. Substitute any vertex for $\underline{r}$ and solve for $\lambda_u, \lambda_v, \lambda_w$.
2. Find the surface normal $\underline{n}'$ and its length $|\underline{n}'|$.
3. The vertex is inside the volume if

$$\begin{aligned}0 &< \lambda_u < 1, \text{ and} \\0 &< \lambda_v < 1, \text{ and} \\ \frac{-R}{|\underline{n}'|} &< \lambda_w < \frac{R}{|\underline{n}'|}.\end{aligned}$$

6.4 Interference Detection Demonstration Program TESTIN

The tests and algorithms described above were combined and implemented in a simple test program, TESTIN. TESTIN tests a single polyhedral face against a single swept volume. TESTIN can be expanded easily for the more general case of several many-sided polyhedra intersecting a number of swept volumes.

6.4.1 Functional Description of TESTIN

TESTIN follows the logic shown in Fig.6.11. TESTIN translates and rotates a single cylinder from an initial to a final position and generates a swept volume. The user specifies the cylinder length and radius and the initial and final positions as specified by two homogeneous transformations. The user inputs the number of vertices on the test face, and their coordinates. TESTIN then tests for intersections of the face and the swept volume.

6.4.2 List and Description of all Subroutines used by TESTIN

This section contains a complete list and descriptions of all subroutines used by the interference detection test program TESTIN. All subroutine links are given. Tektronix IGL graphics subroutines are noted with asterisks (*).

Subroutines used in TESTIN:

CORNER	CYLEND	CYLINT
DET2X2	DET3X3	DIFF
DOTPRD	DRCYL	DRPINT
DRVOL	ECROSS	EDGCRS
EDGINT	ELIST	FACE
FACINT	FACYL	GENVOL
INITSV	INPROJ	INSIDE
INVOL	LINK	MATDET
NEWTON	NWTVOL	ONRAY
OPEN	PLANE	ROTK
SHUT	SOL3X3	SOLVE
SURFIN	SWPVOL	TCROSS
TESTIN	TRANS3	UNIT
VCROSS	VLEN	VRTCRS
XPROD	XPROD2	

Subroutines DOTPRD, OPEN, SHUT, UNIT, XPROD are common to the programs AUTOP and TESTIN.

Main Program TESTIN

TESTIN: main program for testing interference detection algorithm

CALLS-CYLINT,DIFF,DRCYL,FACE,FACINT,FACYL,INVOL,SURFIN,SWPVOL,UNIT,
CMCLOS*,CMOPEN*,EYEBAL*,GRSTOP*,GRSTRT*,VRP3D*,WIND3D*,ZPERSP*

Subroutines

CORNER: calculates the corner points of a parametric swept volume

CALLS- NONE

CALLED BY- SWPVOL

CYLEND: tests if an edge intersects either end face of a cylinder

CALLS- DIFF, DOTPRD, VLEN

CALLED BY- CYLINT

CYLINT: tests if an edge intersects a cylinder

CALLS- CYLEND, DIFF, DOTPRD, SOLVE, UNIT, VLEN, XPROD2

CALLED FROM- TESTIN

DIFF: finds the difference of two vectors

CALLS- NONE

CALLED FROM- CYLEND, CYLINT, FACYL, TESTIN

DET2X2: finds the determinant of a 2X2 matrix

CALLS- NONE

CALLED FROM- EDGINT

DOTPRD: finds the dot product of two vectors

CALLS- NONE

CALLED FROM- CYLINT, FACE, FACYL, PLANE, TCROSS

DRCYL: draws the end cylinders as polyhedral approximations

CALLS- ROTK, TRANS3, UNIT, VLEN, XPROD, CMCLOS*, CMOPEN*, DRAW3D*, MOVE3D*

CALLED FROM- TESTIN

DRPINT: draws an intersection point as a small cross

CALLS- CMCLOS*, CMOPEN*, DRAW3D*, MOVE3D*

CALLED FROM- FACINT, FACYL

DRVOL: draws the swept volume

CALLS- CMCLOS*, CMOPEN*, DRAW3D*, MOVE3D*

CALLED FROM- SWPVOL

ECROSS: counts the number of times a ray crosses face boundaries

CALLS- EDGINT

CALLED FROM- INSIDE

EDGINT: tests for intersection between an edge and a test ray

CALLS- DET2X2

CALLED FROM- ECROSS

ELIST: generates a list of edges which have neither endpoint on the
test ray

CALLS- NONE

CALLED FROM- INSIDE

FACE: allows the user to interactively specify a test face

CALLS- DOTPRD,XPROD,CMCLOS*,CMOPEN*,DRAW3D*,MOVE3D*

CALLED FROM- TESTIN

FACINT: tests for intersection between an edge and a face

CALLS- DRPINT,INSIDE,PLANE

CALLED FROM- TESTIN

FACYL: tests for intersection between a cylinder end face and a
polyhedral face

CALLS- DIFF,DOTPRD,DRPINT,FACINT,XPROD

CALLED FROM- TESTIN

GENVOL: generates the parametric volume parameters

CALLS- NONE

CALLED FROM- SWPVOL

INITSV: initializes the manipulator link parameters

CALLS- OPEN,SHUT

CALLED FROM- SWPVOL

INSIDE: tests if an intersection point lies within the boundaries of
a face

CALLS- ECROSS,ELIST,VCROSS

CALLED FROM- FACINT

INVOL: tests if a point lies inside a parametric swept volume

CALLS- NWTVOL,VLEN

CALLED FROM- TESTIN

LINK: calculates the initial and final positions of the link

CALLS- TRANS3

CALLED FROM- SWPVOL

MATDET: finds the determinant of a 3X3 matrix

CALLS- NONE

CALLED FROM- SOL3X3

NEWTON: finds the intersection of a line and a swept surface by
solving a system of non-linear equations iteratively using
Newton's method

CALLS- SOL3X3,VLEN

CALLED FROM- SURFIN

NWTVOL: solves a system of non-linear equations iteratively using
Newton's method to find if a point lies inside a parametric
volume

CALLS- SOL3X3,VLEN

CALLED FROM- INVOL

ONRAY: tests if a point lies on a ray

CALLS- NONE

CALLED FROM- VCROSS

PLANE: tests if a line segment intersects a plane

CALLS- DOTPRD

CALLED FROM- FACINT

ROTK: finds a rotation matrix to rotate about some given direction
vector

CALLS- NONE

CALLED FROM- DRCYL

SHUT: closes a FORTRAN file

CALLS- NONE

CALLED FROM- INITSV

SOL3X3: finds the solution of a 3X3 system of linear equations

CALLS- MATDET

CALLED FROM- NEWTON,NWTVOL

SOLVE: finds the common normal of two skewed lines

CALLS- NONE

CALLED FROM- CYLINT

SURFIN: finds the intersection of a line and a parametric surface

CALLS- NEWTON,CMCLOS*,CMOPEN*,DRAW3D*,MOVE3D*

CALLED FROM- TESTIN

SWPVOL: generates the volume swept out by a cylindrical link moving
from an initial to a final position

CALLS- CORNER,DRVOL,GENVOL,INITSV,LINK

CALLED FROM- TESTIN

TRANS3: performs a 3-D translation

CALLS- NONE

CALLED FROM- LINK

TCROSS: tests if a ray crosses the face boundary at a vertex point
which lies on the ray

CALLS- DOTPRD,XPROD

CALLED FROM- VCROSS

UNIT: converts a vector to unit length

CALLS- NONE

CALLED FROM- CYLINT,DRCYL,TESTIN

VCROSS: counts the number of times a test ray crosses the face
boundary at vertices

CALLS- ONRAY,TCROSS

CALLED FROM- INSIDE

VLEN: finds the length of a vector

CALLS- NONE

CALLED FROM- CYLEND,CYLINT,DRCYL,NEWTON,NWTVOL

XPROD: finds the normalized cross product of two vectors

CALLS- NONE

CALLED FROM- DRCYL,FACE,FACYL,TCROSS

XPROD2: finds the cross product of two vectors

CALLS- NONE

CALLED FROM- CYLINT

6.4.3 Demonstration of TESTIN

The interference detection program TESTIN was tested by defining a swept volume generated from an initial and a final position of a cylindrical link, and then simulating situations known to result in interference. The cylindrical link was defined to have a length of 30

units and a radius of 5 units. The initial cylinder spine position lay along the X-axis, starting at the origin. The link final position was defined by a translation of 30 units along the Y-axis, and a rotation of 45° about the Y-axis. The resulting wire-frame depiction of the resulting swept volume is shown in Figs. 6.15, 6.16, 6.17.

Fig.6.15 shows the intersection of the swept volume with a triangular face defined by the vertices (10, 15, 15), (10, 30, 15), (10, 10, -20). In this case at least one edge of the triangle intersects at least one parametric surface of the swept volume. TESTIN terminates as soon as an intersection is found, so other intersections may exist as well.

Fig. 6.16 shows the swept volume and a rectangular region defined by vertices (40, 10, 20), (40, 10, -20), (-20, 10, -20), (-20, 10, 20). In this case, an edge of a parametric surface intersects the rectangular region.

Fig. 6.17 shows a rectangular region defined by vertices (40, -2, 20), (40, -2, -20), (-20, -2, -20), (-20, -2, 20). In this case, the cylinder intersects the rectangular region.

Fig. 6.18 shows the swept volume resulting from a pure translation with no rotation. An edge of the rectangular region intersects a parametric surface which in this case is planar.

All five intersection cases were tested and verified with numerous examples. The graphical depiction was not very clear for most of them, however, so they have not been reproduced here. During the running of TESTIN, interference messages appear in the text field of the 4027 screen. Unfortunately, these messages are not reproduced by a graphics screen dump for a hard copy record.

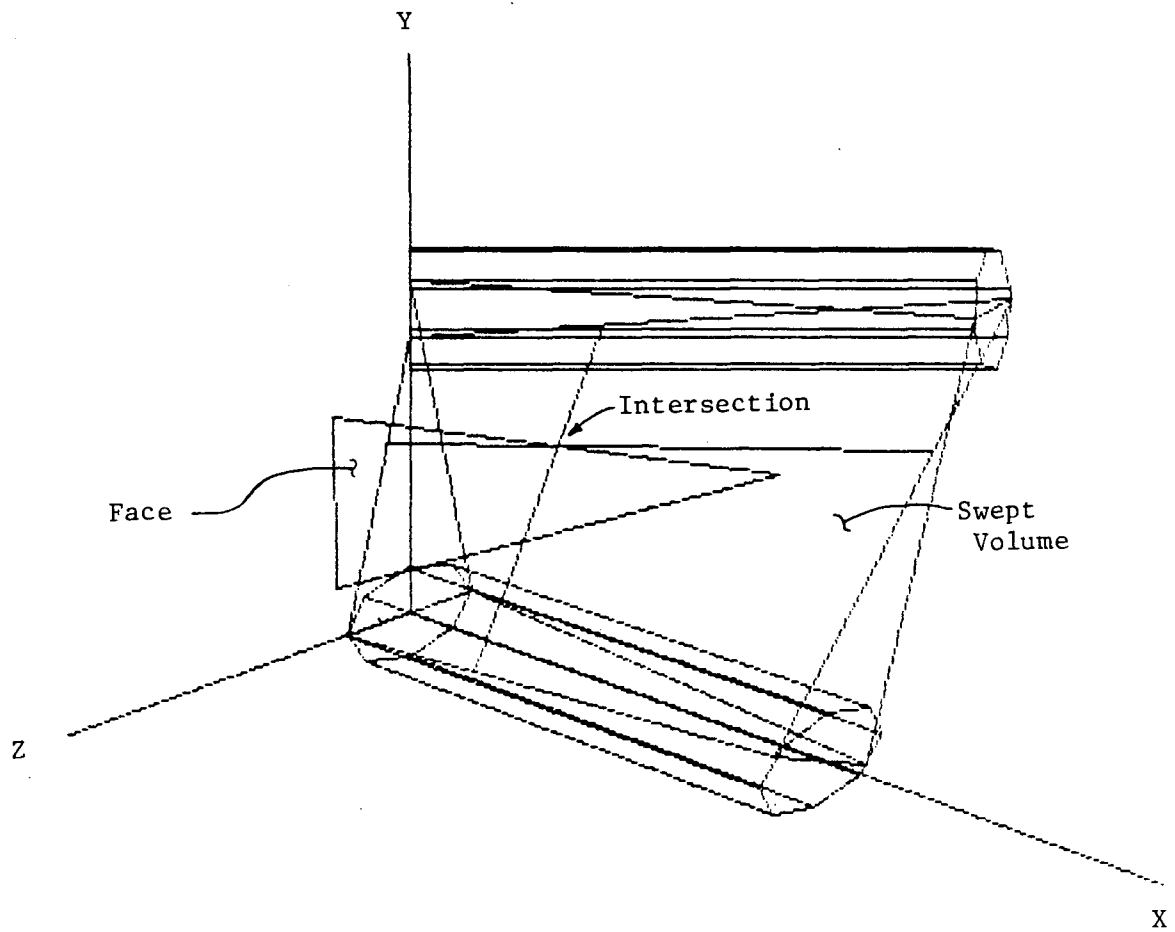


Figure 6.15 Intersection Condition 1 between Face and Swept Volume.

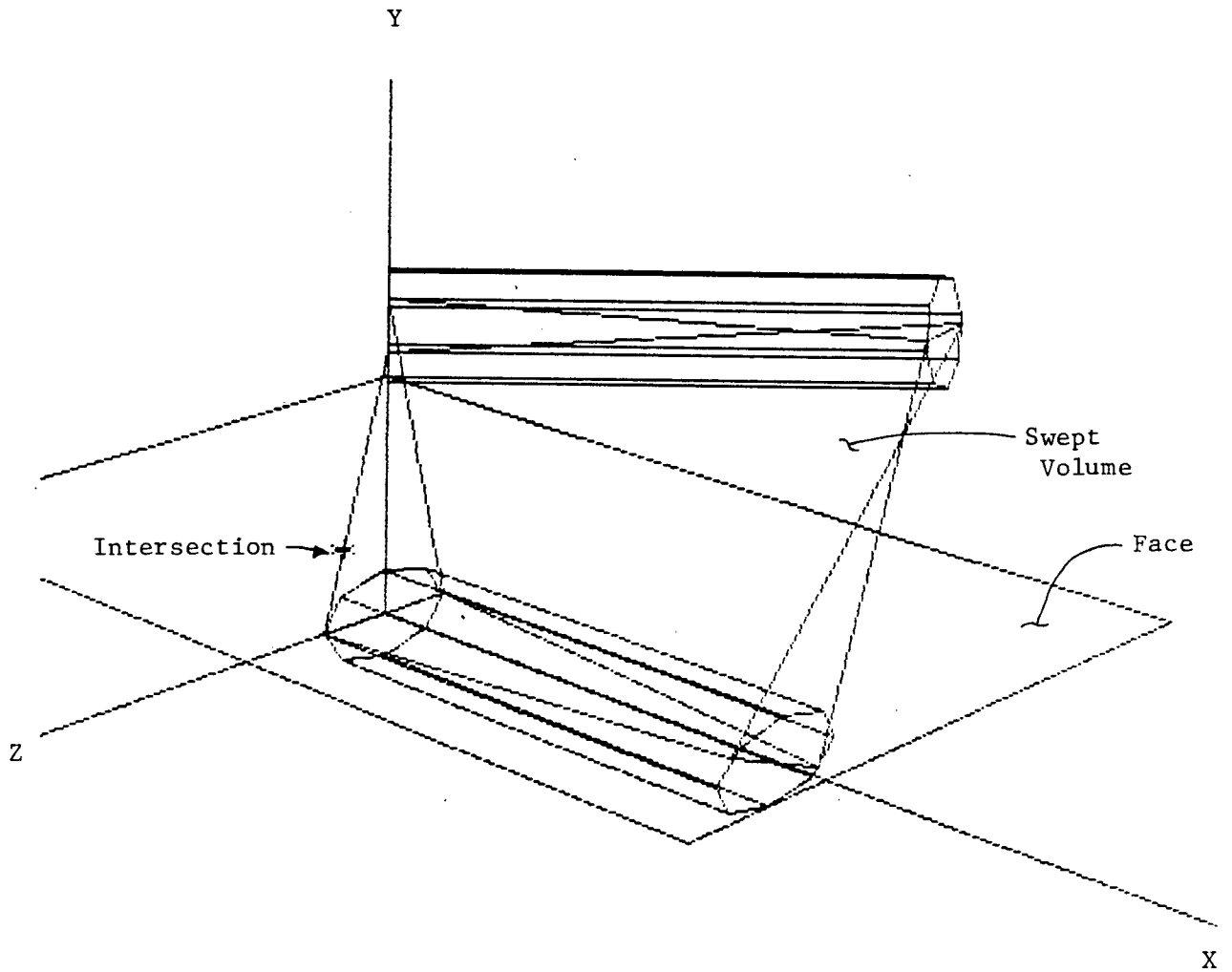


Figure 6.16 Intersection Condition 3 between Face and Swept Volume.

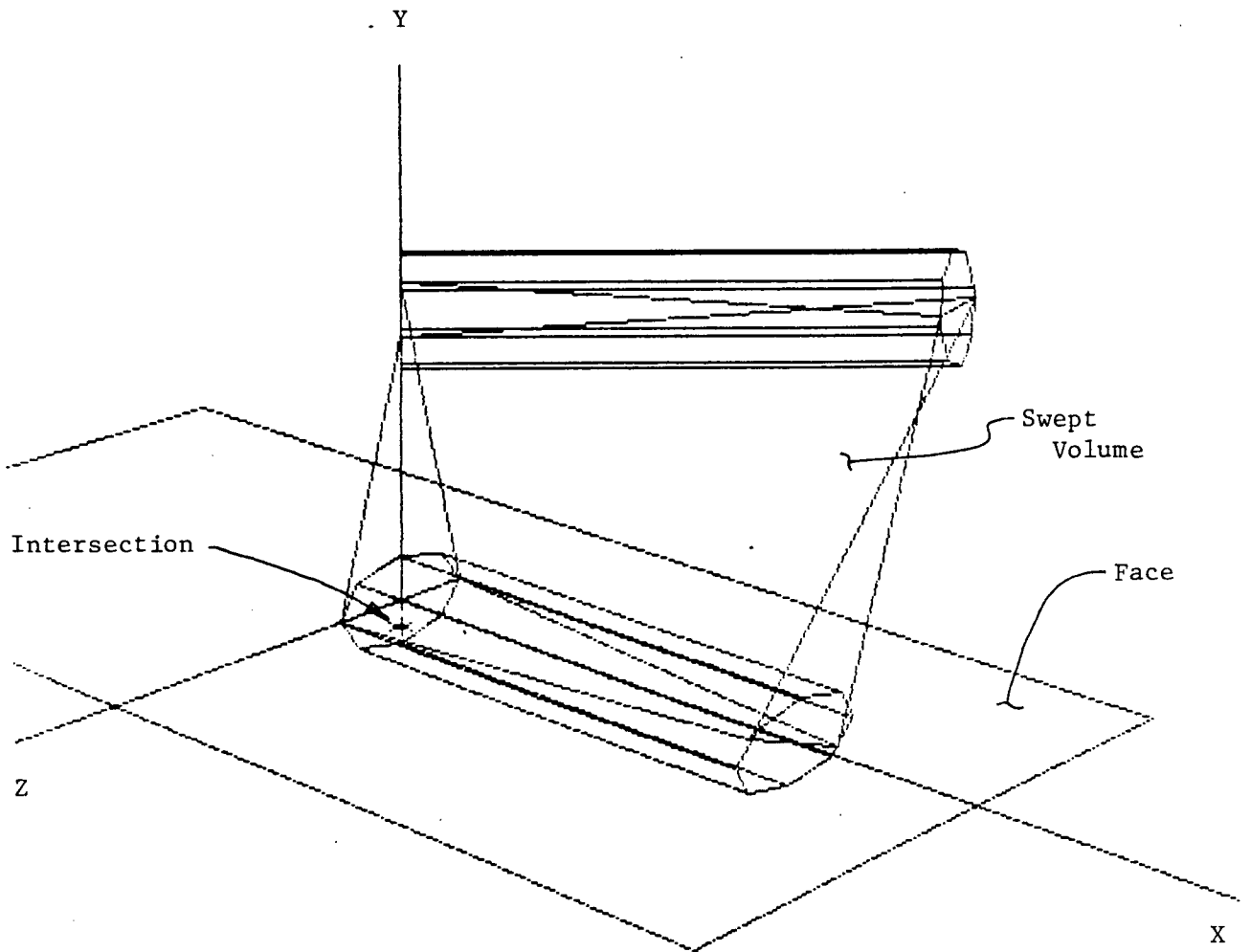


Figure 6.17 Intersection Condition 2 between Face and Swept Volume.

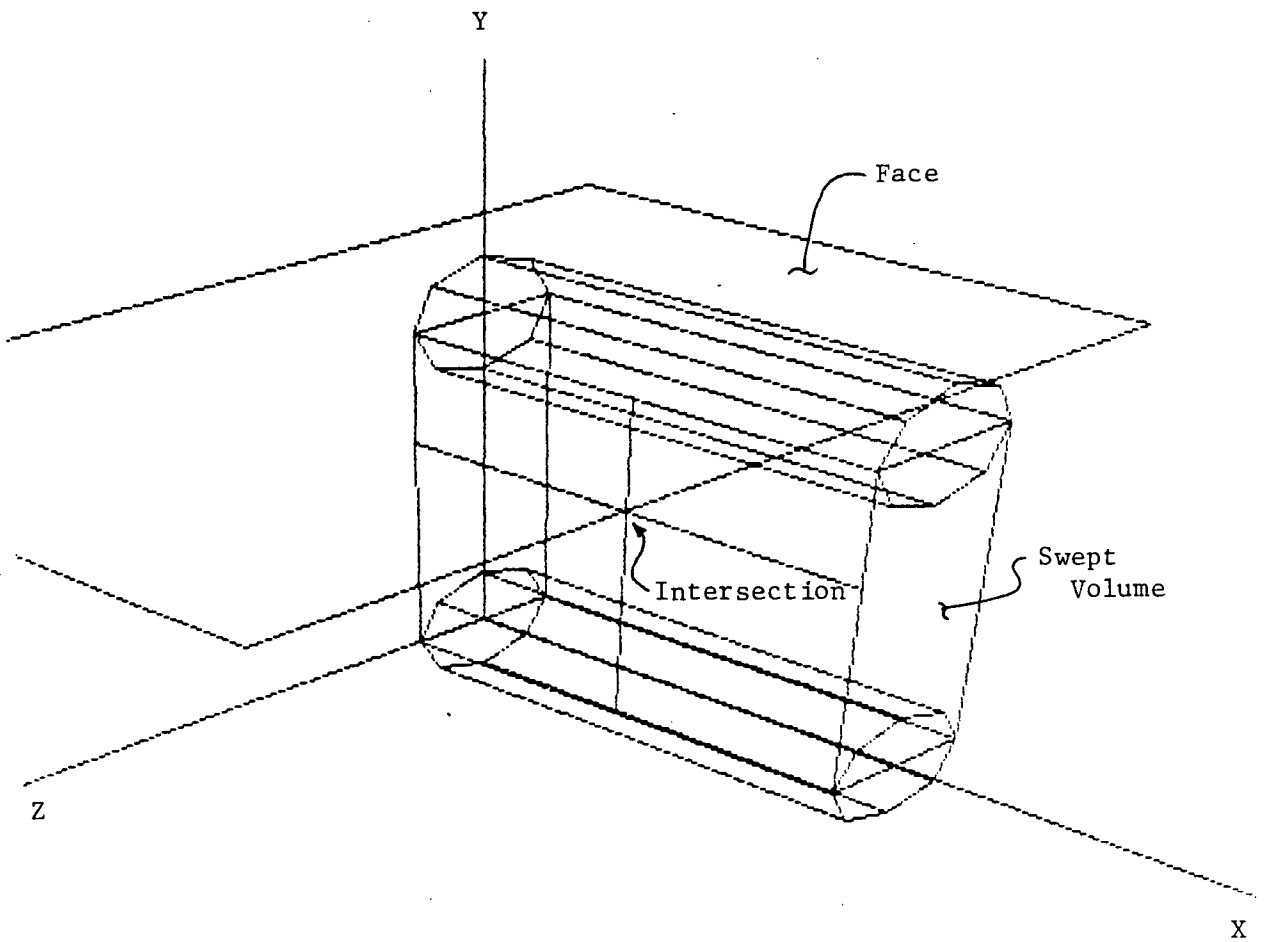


Figure 6.18 Intersection Condition 1 between Face and Swept Volume Generated by Pure Translation.

6.5 Possible Improvements to the Algorithm

We can easily find the smallest bounding boxes containing each of the solid polyhedra and swept volumes. If we choose boxes with faces which are normal to the world coordinate axes, it is very simple to test if any bounding boxes intersect. Further tests need only be conducted on the polyhedra and swept volumes whose bounding boxes intersect.

7. SOFTWARE DESIGN AND DEVELOPMENT

Many of the concepts and much of the theory presented thus far have been successfully incorporated into a substantial FORTRAN software package called AUTOP, for automatically generating a robot program for welding. The software package was developed on a VAX 11-750 computer running under the VMS operating system. The Tektronix Interactive Graphics Library (IGL) software package provided the graphics routines. A DEC VT-101 terminal fitted with a Retrographics DQ-650 enhancement was used, allowing the VT-101 to emulate a Tektronix 4027 graphics terminal with a simple command while still allowing full use of the DEC visual editor for program development. The software package, AUTOP, currently requires about 100 K of memory, and this will increase as AUTOP is expanded.

7.1 Design Approach

Since the software package was to be quite extensive it was essential to carefully formulate some design guidelines and objectives initially, and to develop the software in a planned, consistent and structured manner, following good programming practice. The software was developed with the following goals in mind.

a) Portability

The software was designed to be as system independent as possible. System dependent sections of code have been clearly identified, where possible. No system libraries are called, and no external subroutines are

called, other than the Tektronics IGL graphics subroutines. Standard FORTRAN is followed as closely as possible.

b) Modularity

The software was designed with a modular, hierarchical structure. Each functional module has explicitly defined inputs and outputs. Data is passed via argument lists, common data blocks and files, depending on the requirements.

c) Adaptability and Generality

The software was designed to be easily configured to a wide range of possible workstation implementations. Wherever possible, the software modules are as general as possible, with specific implementation parameters being read from workstation description files. For example, the geometry and dimensions of the robot and positioning table can be easily changed without changes to the software.

7.2 Features and Capabilities of AUTOP

The specific features and capabilities of AUTOP at its current state of development are summarized below:

- a) AUTOP input and output data are stored in files.
- b) Initial system configuration parameters are stored in and read from files. This allows easy modification of parameters.
- c) Interactive graphics is used for communications with the user.
- d) AUTOP generates a welding procedure interactively with the user.

- e) Automatic iterative searching for kinematically feasible torch paths is incorporated.
- f) An Expert Welder module automatically selects welding parameter values. This module was developed separately and is mentioned here for completeness. For details, refer to Buchal et al [1984].

7.3 General Functional Description of AUTOP

It is not practical to describe completely all of the functional details of AUTOP. The software listings are extensively documented so such a description would be largely redundant. A simplified functional description, however, is useful in understanding the basic operation of AUTOP. The following description outlines the major functions in approximate order. No attempt will be made to rigorously follow the actual flow of control of AUTOP. The simplified software structure is shown in Fig.7.1.

AUTOP is designed to interface with a separate CAD system which will generate the CAD data used by AUTOP. No CAD system is currently used, so test CAD files are generated both manually and by specific user-written programs. The data formats which we have specified are closely followed.

The CAD files contain workpiece weld parameters and digitized seam paths, as well as the workpiece geometry. AUTOP converts the CAD data into processed seam data files containing calculated seam tangents and normal vectors at each point.

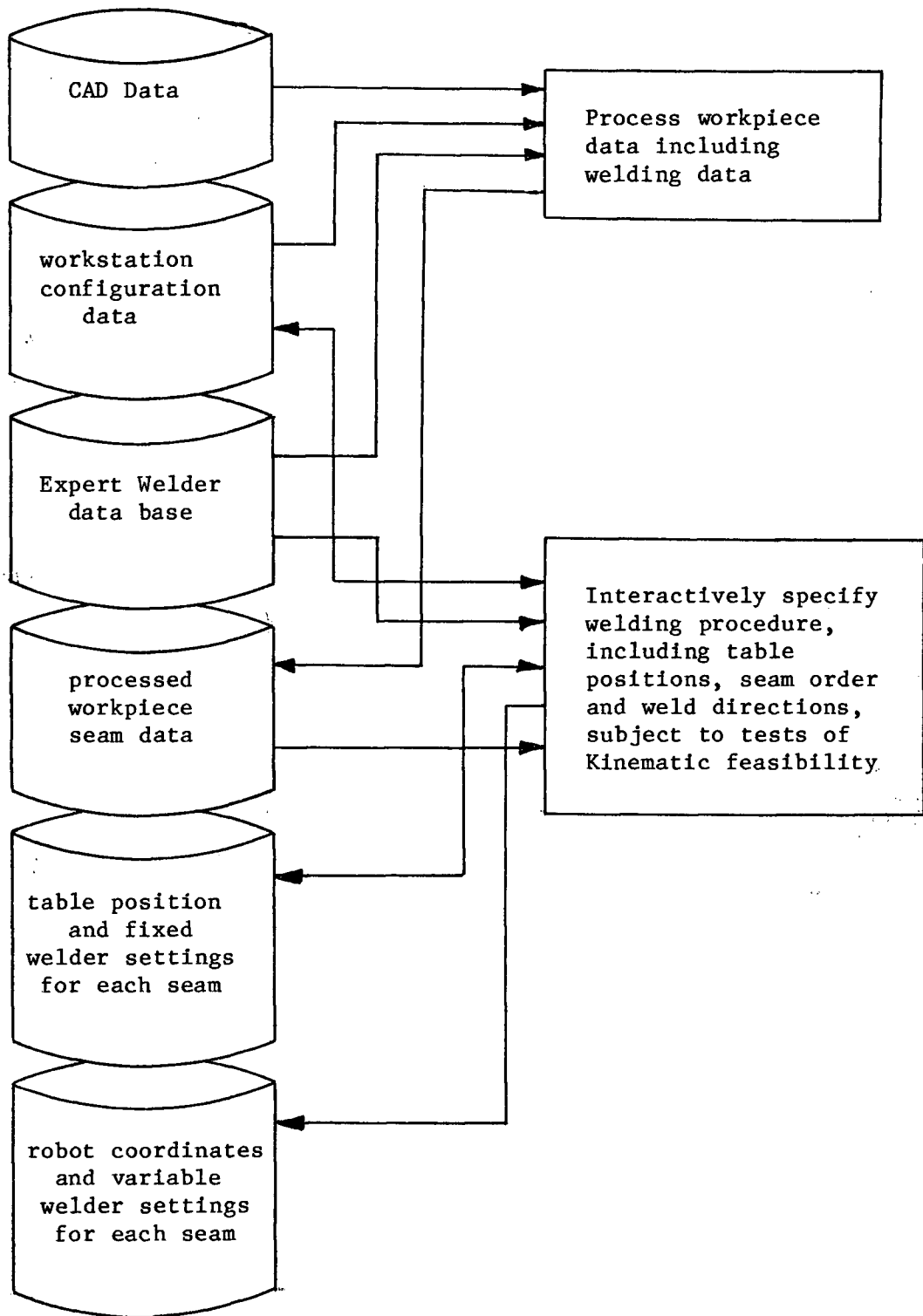


Figure 7.1 Software Structure and Data Flow of AUTOP

Workstation and graphics environment data are read from files and the parameters are initialized. The actual file names for all data files are easily changed.

AUTOP reads the current welder setup, scans all the seam data files and determines if the current setup is appropriate. If not, the user is instructed to reset the welder to the settings determined by the Expert Welder module.

AUTOP draws the workpiece and seams on the screen. The user can change the table position and the eye position. The user interactively 'picks' the seams to be welded. AUTOP then automatically searches for a kinematically feasible robot path to follow the seam. If a path is found, the table position and feasible robot configuration are recorded. A robot coordinate list is generated for the seam and stored in a data file. Also, the Expert Welder module is called to determine appropriate welding parameters such as voltage, current, speed, etc. for the seam. In the present implementation, the current and speed can vary along the seam. The voltage, gas flow rate, wire feed rate and torch orientation are fixed for each seam.

Each seam which has been successfully picked and processed is filled in graphically on the screen. The operator continues moving the workpiece and picking seams until all the seams have been processed. The basic operation of AUTOP is summarized in Fig. 7.2.

7.3.1 List and Description of all Subroutines called from AUTOP

This section lists and briefly describes the function of all

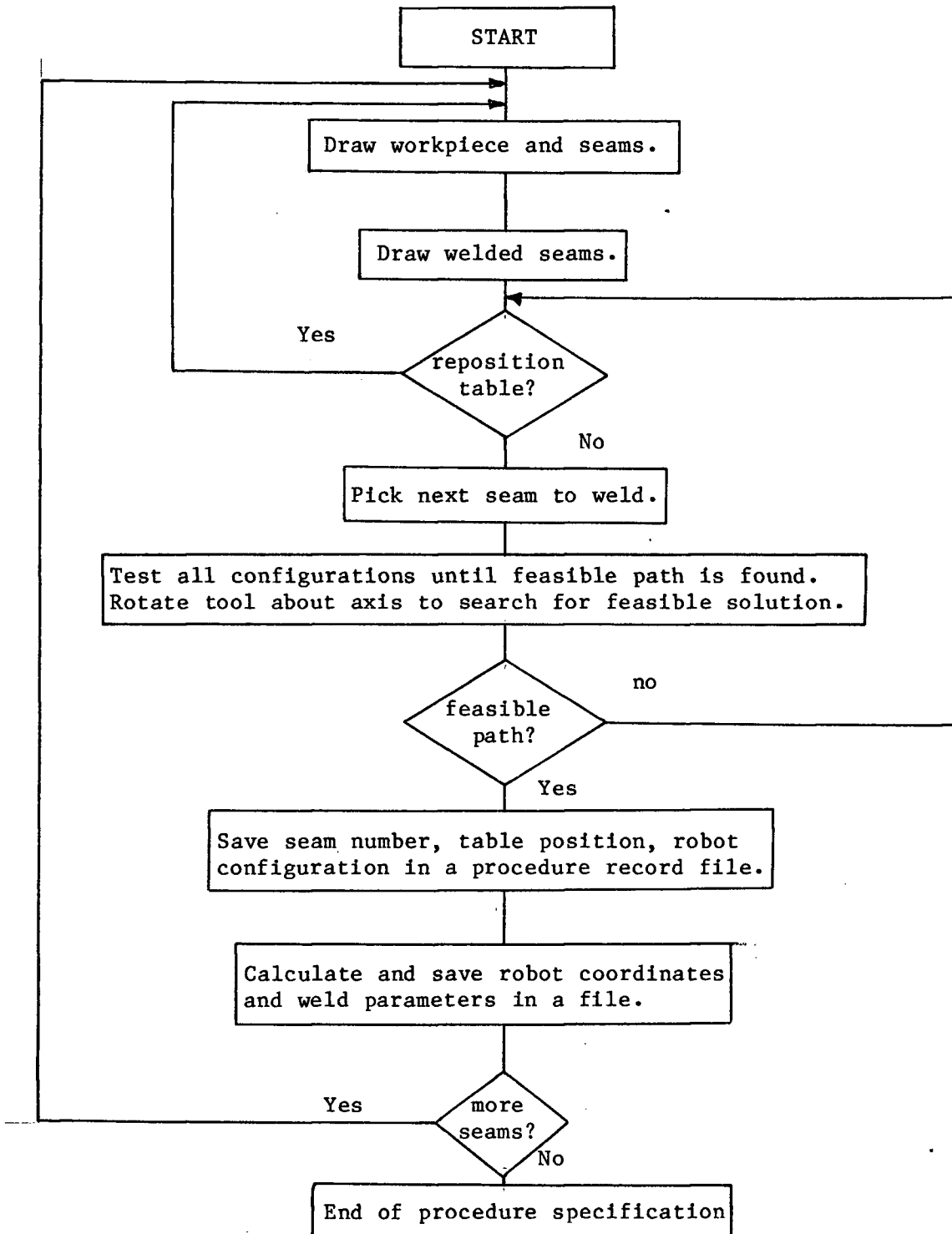


Figure 7.2 Simplified Flowchart of AUTOP.

subroutines called in the software package AUTOP. The subroutines are separated into general functional categories and all links to other subroutines are listed. All Tektronix 'IGL' graphics subroutines are noted with asterisks(*)).

Subroutines used in 'AUTOP':

ARCLEN	CADFIL	CALCS	CHECK
CONFIG	DOTPRD	DRAXES	DRPANL
DRSEAM	DRWELD	EYELOC	EXPERT
HMULT	HMULTS	INIT	JOINTS
LIMITS	LINDIS	MATCNV	MATINV
NOAP1	NOAP5	NORM	OPEN
PROCES	RDFILE	ROTAT	SCENE
SETUP	SHUT	TABPOS	TABSET
TANGEN	TOLROT	TRANS	UNIT
WELSET	WIRE1	WIRE2	WIRE3
WORPOS	XPROD		

Main Program AUTOP

AUTOP: Main Program

CALLS- CADFIL,CHECK,CONFIG,DRWELD,EXPERT,EYELOC,INIT,OPEN,
RDFILE,SCENE,SHUT,WELSET,CMCLOS*,CMOPEN*GETPIK*,GRSTOP*,
GRSTRT*, VUP3D*,VWPT3D*,WIND3D*

Mathematical Operations

DOTPRD: finds the dot product of two vectors

CALLS- NONE

CALLED FROM- DRPANL

HMULT: multiplies two homogeneous transformations in double
precision

CALLS- NONE

CALLED FROM- TABPOS, WORPOS

HMULTS: like HMULT, in single precision

CALLS- NONE

CALLED FROM- WORPOS

MATCHNV: converts double precision homogeneous transformation into
single precision and vice versa

CALLS- NONE

CALLED FROM- WORPOS

MATINV: inverts a homogeneous transformation matrix

CALLS- NONE

CALLED FROM- INIT

ROTAT: performs rotational part of a homogeneous transformation on a
point

CALLS- NONE

CALLED FROM- DRPANL, DRSEAM

TRANS: performs a homogeneous transformation on a point

CALLS- NONE

CALLED FROM- DRPANL, DRSEAM, DRWELD

UNIT: normalizes a vector to unit length

CALLS- NONE

CALLED FROM- DRSEAM, DRWELD

XPROD: performs a vector cross product

CALLS- NONE

CALLED FROM- DRSEAM, DRWELD, LINDIS, NOAP5, PROCES, WORPOS

Graphical Output

DRAXES: draws the world axes

CALLS- MOVE3D*, DRAW3D*

CALLED FROM- SCENE

DRPANL: draws the workpiece

CALLS- DOTPRD, LINDIS, ROTAT, TRANS, APPEAR*, DRAW3D*, MOVE3D*,
PANL3D*, POLY3D*, REMOVE*

CALLED FROM- SCENE

DRSEAM: draws the seams

CALLS- NORM, OPEN, RDFILE, ROTAT, SHUT, TRANS, UNIT, XPROD, APPEAR*,
CLOSEG*, DASHPT*, DRAW3D*, MOVE3D*, OPNSEG*, REMOVE*, SETDET*

CALLED FROM- SCENE

DRWELD: draws the welds

CALLS- NORM, OPEN, RDFILE, SHUT, TRANS, UNIT, XPROD, APPEAR*, CLOSEG*,
DRAW3D*, MOVE3D*, OPNSEG*, REMOVE*, SETDET*, SETHIL*

CALLED FROM- SCENE

EYELOC: chooses eye position through user input

CALLS- NONE

CALLED FROM- MAIN

SCENE: draws complete graphical scene

CALLS- DRAXES,DRPANL,DRSEAM,DRWELD,TABPOS,TABSET,CMCLOS*,

CMOPEN*,DELSEG*,EYEBAL*, NEWPAG*,VRP3D*

CALLED FROM- MAIN

Kinematics

ARCLEN: finds the arc length parameter 's' for a point on a seam
trajectory

CALLS- NONE

CALLED FROM- PROCES

CONFIG: changes robot configuration until a feasible trajectory is
found

CALLS- WORPOS

CALLED FROM- MAIN

JOINTS: calculates the robot joint angles for a given configuration
and tool position

CALLS- NONE

CALLED FROM- WORPOS

LIMITS: checks the robot joint angles against their physical limits

CALLS- NONE

CALLED FROM- WORPOS

NOAP1: finds the PUMA 560 location parameters given the desired tool
location and orientation

CALLS- NONE

CALLED FROM- WORPOS

NOAP5: finds the 5 degree of freedom PUMA location parameters given
the desired tool location and the orientation of its central
axis

CALLS- XPROD

CALLED FROM- WORPOS

NORM: finds the seam normal vector

CALLS- NONE

CALLED FROM- DRSEAM, DRWELD, PROCES

TABPOS: calculates the table frame relative to the world frame given
the table joint positions

CALLS- OPEN, SHUT, HMULT

CALLED FROM- SCENE

TABSET: allows the user to interactively specify new table joint
positions

CALLS- OPEN, SHUT

CALLED FROM- SCENE

TANGEN: calculates the seam tangent vector at each point along a seam

CALLS- NONE

CALLED FROM- PROCES

TOLROT: performs a rotation about the tool axis

CALLS- NONE

CALLED FROM- WORPOS

WORPOS: generates welding parameters, robot location data for a weld

CALLS- JOINTS, HMULT, HMULTS, LIMITS, MATCNV, NOAP1, NOAP5,

OPEN, RDFILE, SHUT, TOLROT, XPROD

CALLED FROM- CONFIG

Expert Welder

CALC: calculates the welding current correction factors as a function
of the gravity vector

CALLS- TRANS

CALLED FROM- EXPERT

EXPERT: finds correct welding parameter values given the initial
welder setup and workpiece parameters

CALLS- CALCS,OPEN,SHUT,WIRE1,WIRE2,WIRE3

CALLED FROM- AUTOP

WELSET: checks if initial welder setup is appropriate for a given job
and recommends changes if necessary

CALLS- OPEN,RDFILE,SHUT

CALLED FROM- AUTOP

WIRE calculates welding parameters for .030 in. wire

CALLS- NONE

CALLED FROM- EXPERT

WIRE2 calculates welding parameters for .035 in. wire.

CALLS- NONE

CALLED FROM- EXPERT

WIRE3 calculates welding parameters for .045 in. wire

CALLS- NONE

CALLED FROM- EXPERT

Data Management

CADFIL: converts object geometry files into internal arrays for faster processing

CALLS- OPEN,RDFILE,SHUT

CALLED FROM- MAIN

INIT: initializes global constants

CALLS- MATINV,OPEN,RDFILE,SHUT

CALLED FROM- MAIN

OPEN: opens a FORTRAN file

CALLS- NONE

CALLED FROM- MAIN,CADFIL,DRSEAM,DRWELD,INIT,PROCESS,TABPOS,
TABSET,WORPOS

RDFILE: reads a given record from a file

CALLS- NONE

CALLED FROM- MAIN,CADFIL,DRSEAM,DRWELD,INIT,WORPOS

SHUT: closes a FORTRAN file

CALLS- NONE

CALLED FROM- MAIN,CADFIL,DRSEAM,DRWELD,INIT,PROCES,
TABPOS,TABSET,WORPOS

Other

CHECK: checks if a picked seam has been picked previously

CALLS- NONE

CALLED FROM- MAIN

LINDIS: finds which of two lines is further away as viewed from the
current eye location

CALLS- XPROD

CALLED FROM- DRPANL

PROCES: processes the initial CAD seam data files into a more useful
form.

CALLS- ARCLEN,NORM,OPEN,SHUT,TANGEN,XPROD

CALLED FROM- SETUP

SETUP: manipulates files for processing by PROCES

CALLS- OPEN,PROCES,RDFILE,SHUT

CALLED FROM- AUTOP

Functions Called

LENGTH finds the length of a character string

CALLED FROM- RDFILE

7.4 Data Structures

AUTOP makes extensive use of external files for data storage and interchange. System initialization parameters are stored in files, where they can be easily changed by AUTOP or by the user through the editor. CAD seam data is also stored in files, in a linked list structure with each seam having a separate file. Output parameters and robot location data are stored in files as well.

The use of internal arrays for temporary data storage is minimized because the data files are large, and of variable length. FORTRAN requires all arrays to have a fixed size, defined at compile time. Arrays

must be dimensioned for the largest conceivable requirement, which wastes memory. Instead, file records are read directly and processed immediately whenever possible. The penalty of slower data access is offset by increased flexibility, simplified data structures and reduced memory requirements.

The largest files, the CAD and processed seam data files, are stored in unformatted form for a substantial reduction in required disk space. The small system parameter and setup files are stored in list-directed format for ease of editing and debugging. The robot location data files are also currently in list-directed format for debugging and ease of verification. When the automatic program generation and downloading module is completed, these files should be changed to unformatted form.

7.4.1 Description of all I/O Files and their Formats

The following is a complete list and description of all the data files accessed by the program AUTOP. The DEC file naming convention used by the VAX computer requires all file names to have the extension .DAT (i.e. NAME.DAT).

FILIST: List of file names used in a particular implementation.

CADLST: List of names of CAD data files corresponding to each seam (i.e. CADDAT).

CADDAT: File containing CAD data for a particular seam.

WPCLST: List of names of processed CAD data files corresponding to each seam (i.e. WPCDAT).

WPCDAT: File containing processed CAD data for a seam.

ROBLST: List of names of robot seam location data files (i.e. ROBDAT).

ROBDAT: File containing robot locations and welding data for a seam path.

OBJLST: List of names of files containing geometric description of objects in workstation.

OBJDAT: File containing polyhedral geometric specification of an object.

WKSTAT: File containing complete specification of robot and welder.

TABLE: File containing specification of worktable.

GRAPH: File containing parameters for graphics system.

LOCSET: File containing workstation setup for each seam to be welded.

STOVAR: Storage file for current table position.

WPARAM: Temporary weld parameter storage file.

The contents and formats of each of these data files follow.

FILE NAME LISTS

FILIST.DAT: assigns names to all the files accessed by AUTOP

TABLE 7.1 - File Name List FILIST

RECORD NO.	CONTENTS
1	CADLST.DAT
2	WPCLST.DAT
3	WKSTAT.DAT
4	TABLE.DAT
5	STOVAR.DAT
6	GRAPH.DAT
7	OBJLST.DAT
8	ROBLST.DAT
9	LOCSET.DAT
10	WPARAM.DAT

CADLST: Contains a list of the CAD files describing each weld seam.

TABLE 7.2 - CAD File List CADLST

RECORD NO.	CONTENTS	DESCRIPTION
1	*TEXT	TEXT HEADING
2	5	NUMBER OF SEAMS
3	CAD1.DAT	SEAM 1 CAD FILE
4	CAD2.DAT	SEAM 2 CAD FILE
5	CAD3.DAT	SEAM 3 CAD FILE
6	CAD4.DAT	SEAM 4 CAD FILE
7	CAD5.DAT	SEAM 5 CAD FILE

WPCLST: Contains a list of processed seam files.

TABLE 7.3 - Processed Seam File List WPCLST

RECORD NO.	CONTENTS	DESCRIPTION
1	*TEXT	TEXT HEADING
2	5	NUMBER OF SEAMS
3	WPC1.DAT	PROCESSED SEAM 1 DATA FILE
4	WPC2.DAT	PROCESSED SEAM 2 DATA FILE
5	WPC3.DAT	PROCESSED SEAM 3 DATA FILE
6	WPC4.DAT	PROCESSED SEAM 4 DATA FILE
7	WPC5.DAT	PROCESSED SEAM 5 DATA FILE

ROBLST: Contains a list of robot location files for each seam

TABLE 7.4 - Robot Location Data File List ROBLST

RECORD NO.	CONTENTS	DESCRIPTION
1	*ROBOT XYZOAT COORDS.	TEXT HEADING
2	PUMA1.DAT	SEAM 1 ROBOT DATA FILE
3	PUMA2.DAT	SEAM 2 ROBOT DATA FILE
4	PUMA3.DAT	SEAM 3 ROBOT DATA FILE
5	PUMA4.DAT	SEAM 4 ROBOT DATA FILE
6	PUMA5.DAT	SEAM 5 ROBOT DATA FILE

OBJLST: Contains a list of geometric CAD files for objects in the world model.

TABLE 7.5 - Object Geometry Data File List OBJLST

RECORD NO.	CONTENTS	DESCRIPTION
1	CYLIND.DAT	GEOMETRY OF WORKPIECE
2	WKTAB.DAT	GEOMETRY OF WORK TABLE
3	TOOL.DAT	GEOMETRY OF ROBOT TOOL
4	FORARM.DAT	GEOMETRY OF ROBOT FOREARM
5	UPARM.DAT	GEOMETRY OF ROBOT UPPER ARM

SEAM DATA FILES

TABLE 7.6 - Seam CAD File

RECORD NO.	CONTENTS	DESCRIPTION
1	JTYPE,XSECT,T1,T2,NPOINT,ICURVE	WELDING PARAMETERS
2	INDEX,R(3),N1(3),N2(3)	TRAJECTORY DATA
	:	:
NPOINT+1	INDEX,R(3),N1(3),N2(3)	TRAJECTORY DATA

JTYPE: joint type code

XSECT: weld cross sectional area

T1,T2: parent metal thicknesses

NPOINT: number of points in file

ICURVE: trajectory type specifier

0: straight line

1: space curve

INDEX: location index

R(3): position vector of current location

N1(3),N2(3): adjacent surface normal vectors

TABLE 7.7 - Processed Seam Data File

RECORD NO.	CONTENTS	DESCRIPTION
1	JTYPE,XSECT,T1,T2,NPOINT,ICURVE	SEAM PARAMETERS
2	INDEX,S,R(3),T(3),A(3),B(3)	SEAM DATA
:	:	:
NPOINT+1	INDEX,S,R(3),T(3),A(3),B(3)	SEAM DATA

JTYPE,XSECT,T1,T2,NPOINT,ICURVE: as defined for CAD file

INDEX: location index

S: arc length to current position along trajectory

R(3): position vector of current position

T(3): unit vector tangent to seam trajectory at current position

A(3): unit vector normal to weld surface

B(3): $B=A \times T$ (to give orthogonal set of basis vectors)

TABLE 7.8 - Robot Location Data File

RECORD NO.	CONTENTS	DESCRIPTION
1	ALPHA,BETA,STKOUT,VOLT,FEED,GFR,NPOINT	SEAM GLOBAL PARAMETERS
2	INDEX,X,Y,Z,O,A,T,CURRENT,SPEED	SEAM LOCAL PARAMETERS
:	:	:
NPOINT+1	INDEX,X,Y,Z,O,A,T,CURRENT,SPEED	

ALPHA: torch lead angle

BETA: torch lateral angle

STKOUT: torch stickout distance

VOLT: welding voltage

FEED: wire feed rate

GFR: shielding gas flow rate

NPOINT: number of locations in file

INDEX: location index

X,Y,Z: robot tool location (translations)

O,A,T: robot tool orientation (rotations)

CURRENT: welding current

SPEED: welding speed

OBJECT GEOMETRY DATA FILES

OBJDAT: File containing polyhedral model of an object in the work station

TABLE 7.9 - Object Geometry Data Files

RECORD NO.	CONTENTS	DESCRIPTION
1	ID1,NORM1,DIST1,NVERT1	FACE1 HEADER
2....NVERT1+1	VERTNO,VERT(3)	VERTEX INDEX,COORDINATES
I:=NVERT1+2	ID2,NORM2,DIST2,NVERT2	FACE2 HEADER
I+1...I+NVERT2+1	VERTNO,VERT(3)	VERTEX INDEX, COORDINATE

etc. for all faces in the object

End of file is indicated by a face id = 0

ID: face ID number

NORM: face outward surface normal

DIST: normal distance from face plane to coordinate frame
origin

NVERT: number of vertices in the face

VERTNO: vertex ID number

VERT(3): vertex coordinates

SYSTEM CONFIGURATION FILES

WKSTAT: contains robot and welder specifications

TABLE 7.10 - System Configuration File WKSTAT

NO. OF RECORDS	CONTENTS	DESCRIPTION
1	*TEXT HEADER	
1	NRDOF	NUMBER OF ROBOT DEGREES OF FREEDOM
1	*TEXT HEADER	
1	*TEXT HEADER	
NRDOF	LINKNO,ALPHA,LEN, DIST,MIN,MAX	LINK PARAMETERS
1	*TEXT	
1	ICONF(6)	CURRENT ROBOT CONFIGURATION
1	TEXT	
4	H00(4,4)	ROBOT LOCATION REL. TO WORLD
1	*TEXT	
4	H6G(4,4)	TOOL FRAME REL. TO LAST LINK
1	*TEXT	
1	IMAX,DELTA	TOOL ROTATION PARAMETERS
1	*TEXT	
1	*TEXT	
1	WSIZE	CURRENTLY INSTALLED WELDER WIRE SIZE

LINKNO: link number
 ALPHA: link twist angle
 LEN: link length
 DIST: joint displacement along its axis
 MIN,MAX: joint limits
 ICONF: current robot default configuration
 H00: homogeneous transformation relating zeroth robot frame to world frame
 H6G: homog. trans. relating tool frame to last link frame
 IMAX: maximum number of rotation steps about tool axis (when searching for a feasible path)
 DELTA: size of incremental angular rotations about tool axis
 WSIZE: current welder wire size

NOTE: The Expert Welder currently assumes all other welding parameter settings are programmable.

TABLE 7.11 - Example of File WKSTAT

CONTENTS						RECORD NO.
*NUMBER OF DEGREES OF FREEDOM, 'NRDOF'						1
6						2
*ROBOT LINK PARAMETERS						3
LINK	ALPHA	LENGTH	DIST.	MIN.	MAX.	4
1	-90.	0.	0.	-160.	160.	5
2	0.	432.	149.5	137.	43.	6
3	90.	-20.5	0.	-52.	-128.	7
4	-90.	0.	433.	-110.	170.	8
5	90.	0.	0.	-100.	100.	9
6	0.	0.	56.5	-266.	266.	10
*INITIAL CONFIGURATION ARRAY ICONF						11
0	0	1	1	0	0	12
*WORLD COORDINATE TO ROBOT COORDINATE TRANSFORMATION MATRIX						13
'HOO'						
1.	0.	0.	0.			14
0.	1.	0.	0.			15
0.	0.	1.	0.			16
0.	0.	0.	1.			17
*GRIPPER FLANGE TO TOOL COORDINATE TRANSFORMATION MATRIX'H6G'						18
0.	0.	1.	120.			19
0.	1.	0.	0.			20
-1	0.	0.	92.			21
0.	0.	0.	1.			22
*TOOL ROTATION PARAMETERS: MAX.No.OF STEPS, ANGLE STEP						
SIZE (DEGREES)						23
18	10					24
*WELDER SETUP						25
*WIRE SIZE						26
0.030						27

POSITIONING TABLE SPECIFICATION FILE

TABLE: contains geometric and kinematic specifications of positioning table

TABLE 7.12 - Positioning Table Specification File TABLE

NO. OF RECORDS	CONTENTS	DESCRIPTION
1	NDOF	NUMBER OF TABLE DEGREES OF FREEDOM
1	*TEXT	TEXT HEADER
NDOF	ID, ITYPE, STEP, RMIN, RMAX, THETA, ALPHA DISP, LEN	TABLE LINK PARAMETERS
1	*TEXT	
4	HTW(4,4)	WORKPIECE RELATIVE TO TABLE FRAME
1	*TEXT	
4	HPRE(4,4)	FIRST LINK RELATIVE TO WORLD
1	*TEXT	
4	HPOST(4,4)	TABLE FRAME RELATIVE TO LAST LINK

NDOF: number of table degrees of freedom

ID: joint type 1=revolute (rotational)
2=prismatic (translational)

ITYPE: joint motion type 1=discrete positions
2=continuous positions

STEP: joint motion step size for itype=1

RMIN, RMAX: joint limits

THETA: initial joint angle

ALPHA: link twist angle

DISP: initial joint displacement along its axis

LEN: link length

HTW(4,4): homog. transformation matrix relating workpiece to table

HPRE(4,4): homog. transformation matrix relating first link to world

HPOST(4,4): homog. transformation relating table frame to last link

TABLE 7.13 - Example of File TABLE

CONTENTS									RECORD NO.
3									1
ID	ITYPE	STEP	RMIN	RMAX	THETA	ALPHA	DISP	LEN	2
2	2	0.	-1000.	1000.	-90.	-90.	0.	0.	3
2	2	0.	-1000.	1000.	-90.	90.	0.	0.	4
1	2	0.	-1000.	1000.	90.	90.	0.	0.	5
*HTW									6
1.		0.	0.	0.					7
0.		1.	0.	0.					8
0.		0.	1.	0.					9
0.		0.	0.	1.					10
*HPRE									11
1.		0.	0.	0.					12
0.		1.	0.	0.					13
0.		0.	1.	0.					14
0.		0.	0.	1.					15
*HPOST									16
1.		0.	0.	0.					17
0.		1.	0.	0.					18
0.		0.	1.	0.					19
0.		0.	0.	1.					20

GRAPHIC SYSTEM INITIALIZATION FILE

GRAPH: contains graphics parameters which can be easily changed

TABLE 7.14 - Graphic System Initialization File GRAPH

NO. OF RECORDS	CONTENTS	DESCRIPTION
1	*TEXT	
1	VRP(3)	VIEW REFERENCE POINT COORDINATES
1	*TEXT	
1	XMIN,XMAX,YMIN, YMAX,ZMIN,ZMAX	VIEW WINDOW LIMITS

TABLE 7.15 - Example of GRAPH

CONTENTS	RECORD NO.
*VIEW REFERENCE POINT	1
0 0 0	2
*WINDOW DIMENSIONS, XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX	3
-600. 600. -600. 500. -600. 600.	4

FILE OF WORKSTATION SETUP FOR EACH SEAM

LOCSET: contains information about the table position and robot configuration to weld a given seam

TABLE 7.16 - Workstation Setup File for Current Seam LOCSET

NO. OF RECORDS:	CONTENTS	DESCRIPTION
1	SEAMID	SEAM IDENTIFICATION NUMBER
1	VAR(6)	TABLE JOINT POSITIONS
1	ICONF(6)	ROBOT CONFIGURATION SPECIFICATION
1	SEAMID	SEAM IDENTIFICATION NUMBER
1	VAR(6)	TABLE JOINT POSITIONS
1	ICONF(6)	ROBOT CONFIGURATION SPECIFICATION
	ETC.	

TABLE 7.17 - Example of LOCSET for Seams 5, 1, 3

CONTENTS	RECORD
5	1
300.0000 600.0000 -90.00000 0.0000000E+00 0.0000000E+00	2
0.0000000E+00	
0 0 1 1 0 0	3
1	4
300.0000 600.0000 -90.00000 0.0000000E+00 0.0000000E+00	5
0.0000000E+00	
0 0 1 1 0 0	6
3	7
300.0000 600.0000 -90.00000 0.0000000E+00 0.0000000E+00	8
0.0000000E+00	
0 0 1 1 0 0	9

CURRENT TABLE POSITION DATA

STOVAR: contains the current table joint positions

TABLE 7.18 - Current Table Position Data STOVAR

NO. OF RECORDS	CONTENTS	DESCRIPTION
1	*TEXT	
1	VAR(1)	JOINT (1) POSITION
	:	:
1	VAR(NDOF)	JOINT (N) POSITION

TABLE 7.19 - Example of STOVAR for 3 Degree of Freedom Table

CONTENTS	RECORD NO.
*JOINT VARIABLES	1
300.0000	2
600.0000	3
-90.00000	4

TEMPORARY WELD PARAMETER FILE

WPARAM: contains the welding current and speed parameters for all the locations of the current seam

TABLE 7.20 - Temporary Weld Parameter File WPARAM

NO. OF RECORDS	CONTENTS	DESCRIPTION
1	CURRNT,SPEED	PARAMETERS FOR LOCATION 1
1	CURRNT,SPEED	PARAMETERS FOR LOCATION 2
	:	:
1	CURRNT,SPEED	PARAMETERS FOR LOCATION 'NPOINT'

TABLE 7.21 - Example of WPARAM for a Seam with 10 Locations

CONTENTS		RECORD NO.
30	10	1
31	10	2
31.5	10.5	3
33	12	4
34.5	12	5
34	11.5	6
33	11	7
33	11	8
32	10	9
31	10	10

TABLE 7.22 - Fortran File Formats

FILE NAME	FORMAT
FILIST	LIST-DIRECTED
CADLST	LIST-DIRECTED
CADDAT	UNFORMATTED
WPCLST	LIST-DIRECTED
WPCDAT	UNFORMATTED
ROBLST	LIST-DIRECTED
ROBDAT	FORMATTED
OBJLST	LIST-DIRECTED
OBJDAT	LIST-DIRECTED
WKSTAT	LIST-DIRECTED
TABLE	LIST-DIRECTED
GRAPH	LIST-DIRECTED
LOCSET	LIST-DIRECTED
STOVAR	LIST-DIRECTED
WPARAM	LIST-DIRECTED

TABLE 7.23 - Subroutines Accessing Data Files

FILE	SUBROUTINES
CADDAT	DRSEAM, DRWELD, PROCES, SETUP
CADLST	AUTOP, DRSEAM, DRWELD, SETUP
FILIST	AUTOP
GRAPH	AUTOP
LOCSET	INIT, WORPOS
OBJDAT	CADFIL
OBJLST	CADFIL
ROBDAT	INIT, WORPOS
ROBLST	INIT, WORPOS
STOVAR	TABSET, WORPOS
TABLE	INIT, TABPOS, TABSET
WKSTAT	EXPERT, INIT, WELSET
WPARAM	EXPERT, WORPOS
WPCDAT	EXPERT, PROCES, SETUP, WESET, WORPOS
WPCLST	SETUP, WORPOS

7.5 Graphical Depiction of Solids Using Panels

If we attempt to represent polyhedra graphically using the usual wire-frame depiction, we find the image ambiguous and confusing unless hidden lines are removed. Hidden line removal is non-trivial, and is generally not available in simple graphics packages such as Tektronix IGL. We have developed an alternative method for depicting polyhedral solids in a clear and unambiguous way without explicit hidden line calculations by taking advantage of the characteristics of the Tektronix 4027 graphics terminal.

The Tektronix 4027 is a raster-type terminal which supports PANEL graphics commands. A PANEL is depicted as a filled polygon on the screen. A new PANEL image is superimposed on anything already on the screen. We have developed an algorithm which depicts a polyhedron as the superposition of its face panels. The panels are superimposed from back to front, relative to the viewpoint. Hidden surfaces are thus automatically covered. Each panel is bounded by a dark vector border so that all edges will be visible.

To superimpose the panels in the correct order, we need a test for determining which of two panels is "closer" to the viewpoint. We note that the face polygons of a polyhedral solid cannot intersect each other, but can contact along common edges only. We can determine heuristically which of two face polygons lies "closer" to the viewpoint by using the following principle:

Suppose we have two lines in space, L_1 and L_2 , and a line of sight vector $\underline{V}_s$ defining the view direction (Fig. 7.3). Let $\underline{N}_{12}$ be the common normal directed from L_1 to L_2 . Then we can say L_1 appears to cross over L_2 from the given view direction if

$$\underline{V}_s \cdot \underline{N}_{12} > 0 . \quad (7.1)$$

We can now state the following algorithm for determining the "stacking" of faces:

1. Each line containing an edge of a face is compared to all other lines containing edges of all other defined faces. Each time the given line is in front of another line, that edges weight factor is incremented by 1. This is repeated for all edges on

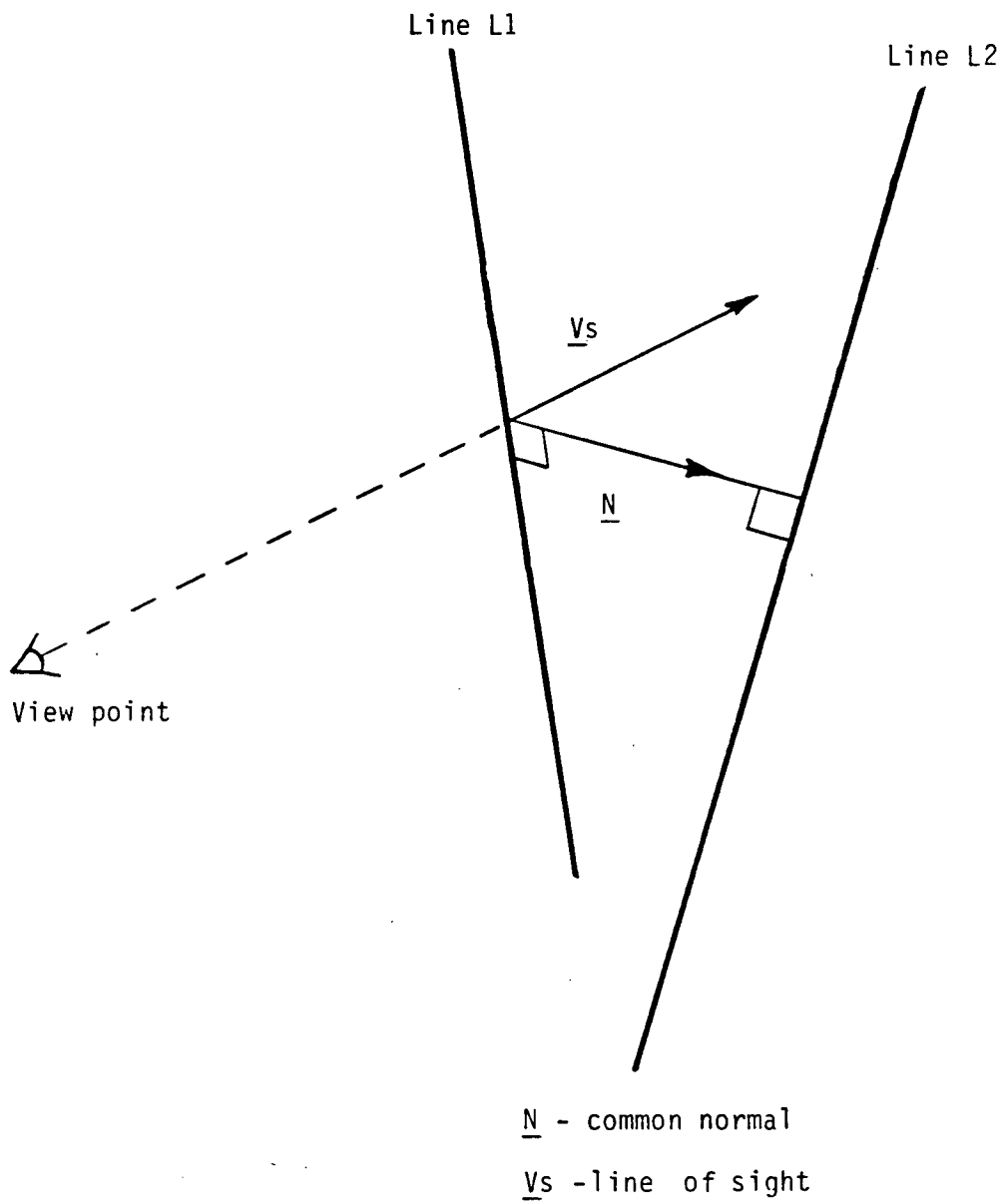


Figure 7.3 Relationship of two lines to view point.

the face, and the edge weights are summed to get a total face weight. An average face weight is obtained by dividing the total face weight by the number of edges.

2. Repeat the above for every face.

3. "Stack" the faces graphically in ascending order of their average weights.

7.6 User's Guide to Running Program AUTOP

AUTOP is a preliminary program only, and does not include all the necessary capabilities of a complete and comprehensive automatic programming system. The following user's guide is not intended for an industrial user, but rather explains the running of AUTOP at its current state of development.

Compiling AUTOP

The main program and all the subroutines can be compiled at once by running the command file COMPIL.COM. (i.e. enter @COMPIL).

Linking AUTOP

All the compiled object modules can be linked into a task image by running the command file AUTOP.COM. (i.e. enter @AUTOP).

Running AUTOP

1. If the initial set-up files FILIST, WKSTAT, TABLE and GRAPH have

not been set up, set them up as described in Section 7.4.1.

2. Generate a pseudo CAD file as described in Section 7.4.1. Currently a direct CAD interface has not been developed. The program CADFIL generates a set of test CAD files. The corresponding geometric model of the workpiece has been generated manually in file CYLIND.DAT.
3. Insure that the terminal emulates a Tektronix 4027.
4. Type RUN AUTOP.
5. The screen will go blank, and AUTOP will prompt the user for the name of the file list FILIST. Currently, the name is FILIST.DAT, in correspondence with the DEC data file naming convention. A different name can be specified if the program is implemented on a system with a different file naming convention. Similarly, FILIST.DAT contains system compatible data file names for all the data files used by AUTOP.
6. AUTOP automatically scans all the seam CAD files and checks if the current welder setup is correct. If not, the operator is instructed to reset the welder. The new setup parameters are recorded in a file.
7. AUTOP next prompts the user for the eye position. The eye position determines the view direction, directed from the eye position to the origin (0.,0.,0.).
8. AUTOP asks the user if a new table position is desired. If the user answers Y (yes), AUTOP prompts for a new position.
9. AUTOP draws the workpiece, seams and a set of reference axes on the screen. The seams are represented by 'herringbone' lines pointing in the default weld directions.

10. AUTOP prompts the user for new eye position and/or table position.
11. If the eye or table position is changed, AUTOP proceeds from step 7.
12. When no more changes are requested, AUTOP draws a set of crosshairs and prompts the user to pick the first seam to be welded. The user moves the crosshairs to the desired seam and picks it by pressing any key.
13. AUTOP checks the selected seam for path feasibility. If the path is feasible, the weld is filled in. Otherwise, the user is prompted to reposition the table and try again.
14. The user continues picking seams and repositioning the table until all seams have been picked. AUTOP stores the table positions and robot configurations, and generates robot locations and welding data for each seam.
15. AUTOP terminates when all the seams have been picked.

7.7 Demonstration of AUTOP

This section illustrates the procedure followed by AUTOP by presenting actual numerical data generated by AUTOP during a test session. For this demonstration, we have extracted and converted the numerical results and contents of data files into a form suitable for inspection and verification.

The workpiece which we have defined for this demonstration consists of a cylinder welded to a block. Five weld seams have been defined; four equal length segments around the circumference of the cylinder and one along the length of the cylinder.

The digitized CAD seam data was generated by a simple FORTRAN program written explicitly for this particular workpiece. The geometric data was entered manually into the object geometry file. The robot was defined to be a PUMA 560 and we arbitrarily defined a positioning table with one revolute and two prismatic degrees of freedom.

AUTOP was run, and the workpiece was interactively repositioned until a kinematically feasible path was found for welding the seam of interest. For this example, we have chosen seam one (see Fig. 7.4). Figs. 7.4, 7.5 and 7.6 show the appearance of the workpiece on the screen, and demonstrate the effect of changing the table position and the user's view point.

The contents of the CAD seam data file for seam one are shown in Table 7.24. These data are processed by AUTOP into the more useful form shown in Table 7.25. For a given seam and table position, AUTOP automatically tests different robot configurations and rotates the torch about its axis within user-defined limits in search of a kinematically feasible path. Table 7.26 contains the calculated joint angles for each robot position along the seam for different configurations and torch rotations. Note that a number of configurations were tested before a feasible one was found.

When a feasible path has been found, AUTOP records the table position and robot configuration for later reference. AUTOP then generates the robot location data in the appropriate form for downloading to the robot. Table 7.27 shows a listing of the resulting X, Y, Z, O, A, T coordinates for the PUMA, as well as the correct welding current and speed at each point. These location values have been verified by manually downloading

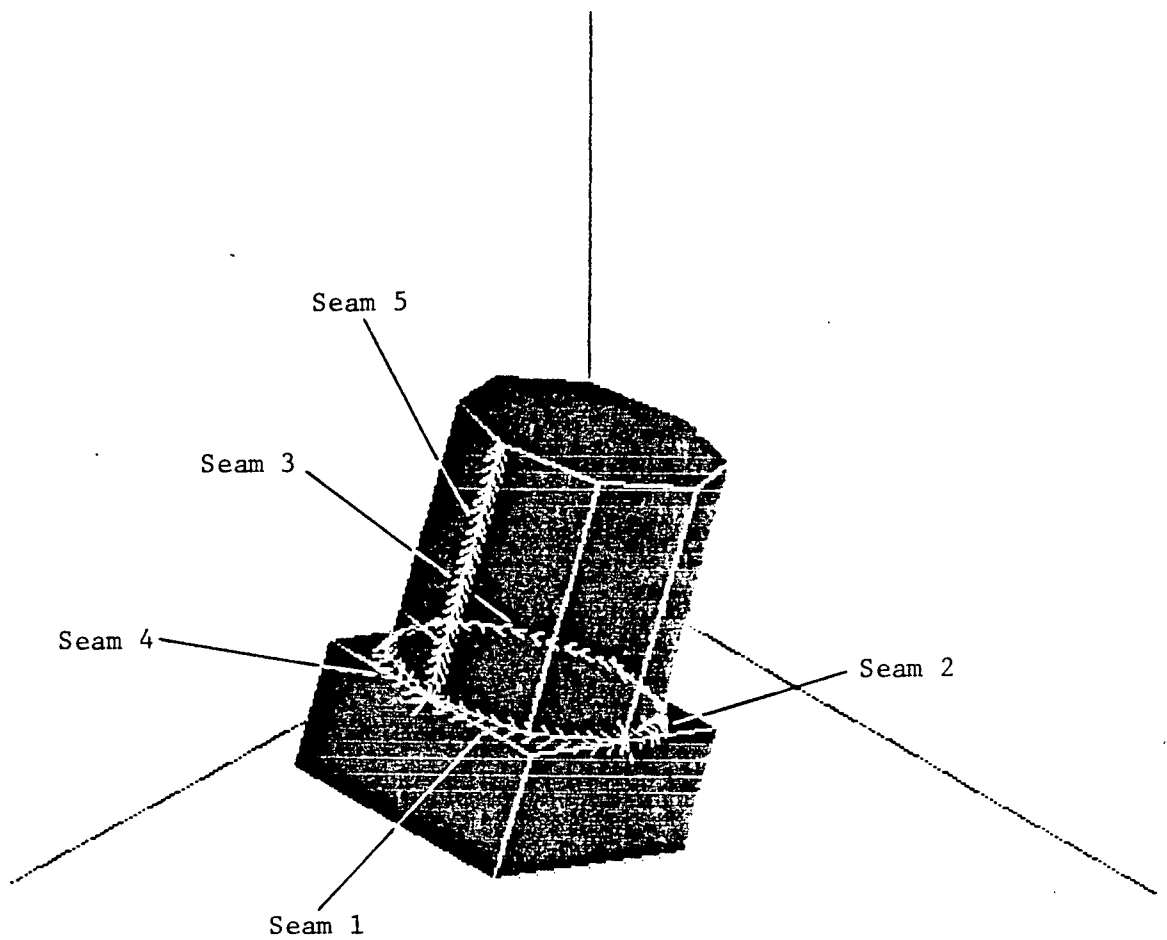


Figure 7.4 Graphical Depiction of Workpiece by AUTOP.

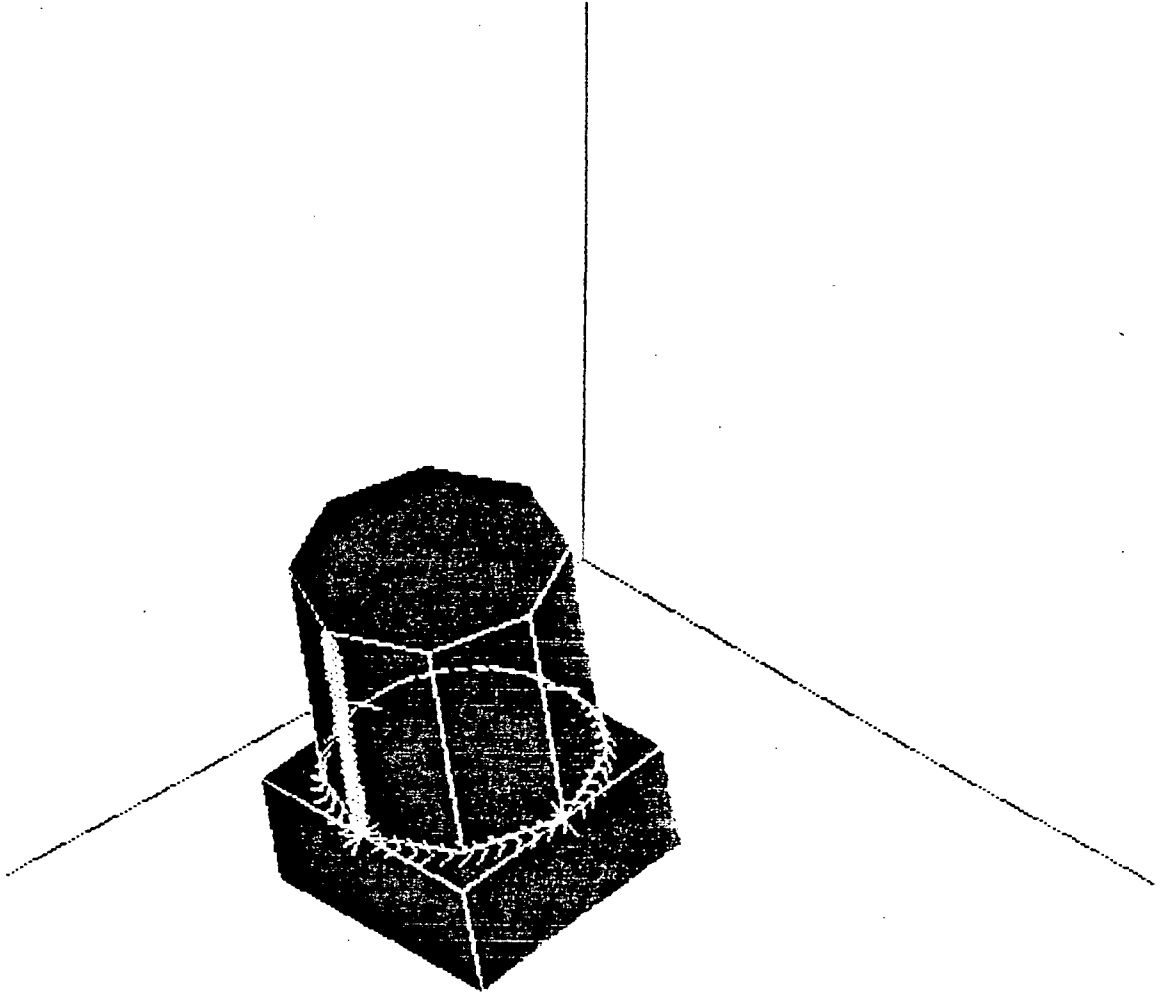


Figure 7.5 New Workpiece Position after Interactively Changing Table Position.

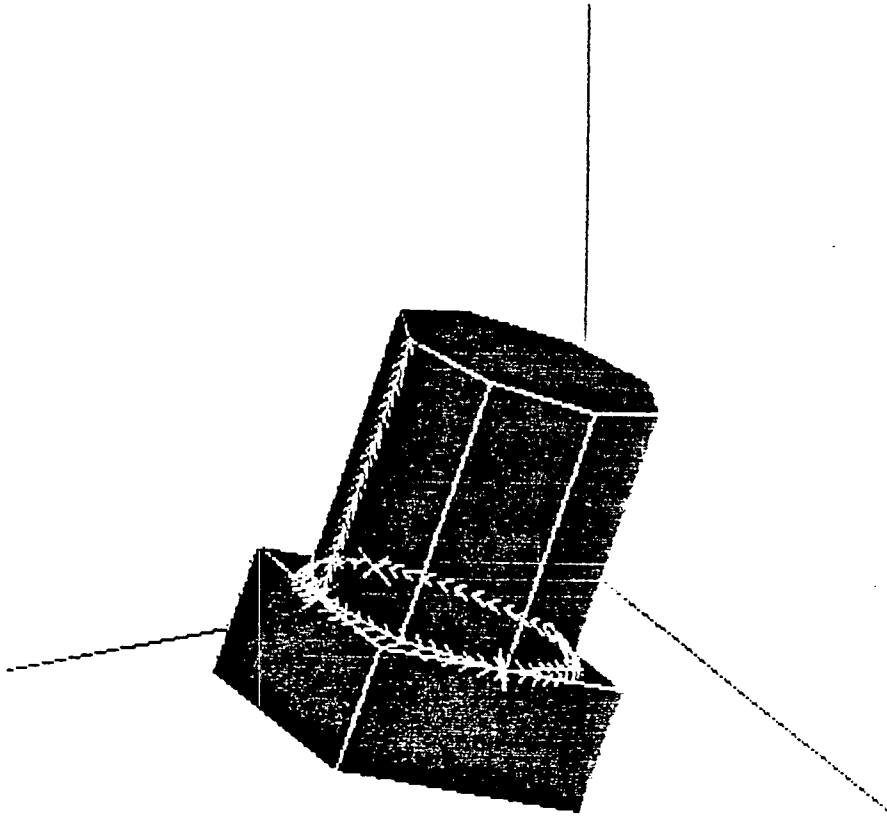


Figure 7.6 Workpiece shown from a Different Viewpoint

Table 7.24 - Contents of Seam CAD File.

ITYPE	XSECT	T1	T2	NPOINT	ICURVE				
1	0.500	0.250	0.500	26	1				
INDEX	R(X)	R(Y)	R(Z)	N1(X)	N1(Y)	N1(Z)	N2(X)	N2(Y)	N2(Z)
1	200.000	100.000	100.000	1.000	0.000	0.000	0.000	0.000	1.000
2	199.803	106.279	100.000	0.998	0.063	0.000	0.000	0.000	1.000
3	199.211	112.533	100.000	0.992	0.125	0.000	0.000	0.000	1.000
4	198.229	118.738	100.000	0.982	0.187	0.000	0.000	0.000	1.000
5	196.858	124.869	100.000	0.969	0.249	0.000	0.000	0.000	1.000
6	195.106	130.902	100.000	0.951	0.309	0.000	0.000	0.000	1.000
7	192.978	136.812	100.000	0.930	0.368	0.000	0.000	0.000	1.000
8	190.483	142.578	100.000	0.905	0.426	0.000	0.000	0.000	1.000
9	187.631	148.175	100.000	0.876	0.482	0.000	0.000	0.000	1.000
10	184.433	153.583	100.000	0.844	0.536	0.000	0.000	0.000	1.000
11	180.902	158.779	100.000	0.809	0.588	0.000	0.000	0.000	1.000
12	177.051	163.742	100.000	0.771	0.637	0.000	0.000	0.000	1.000
13	172.897	168.455	100.000	0.729	0.685	0.000	0.000	0.000	1.000
14	168.455	172.897	100.000	0.685	0.729	0.000	0.000	0.000	1.000
15	163.742	177.051	100.000	0.637	0.771	0.000	0.000	0.000	1.000
16	158.779	180.902	100.000	0.588	0.809	0.000	0.000	0.000	1.000
17	153.583	184.433	100.000	0.536	0.844	0.000	0.000	0.000	1.000
18	148.175	187.631	100.000	0.482	0.876	0.000	0.000	0.000	1.000
19	142.578	190.483	100.000	0.426	0.905	0.000	0.000	0.000	1.000
20	136.813	192.978	100.000	0.368	0.930	0.000	0.000	0.000	1.000
21	130.902	195.106	100.000	0.309	0.951	0.000	0.000	0.000	1.000
22	124.869	196.858	100.000	0.249	0.969	0.000	0.000	0.000	1.000
23	118.738	198.229	100.000	0.187	0.982	0.000	0.000	0.000	1.000
24	112.533	199.211	100.000	0.125	0.992	0.000	0.000	0.000	1.000
25	106.279	199.803	100.000	0.063	0.998	0.000	0.000	0.000	1.000
26	100.000	200.000	100.000	0.000	1.000	0.000	0.000	0.000	1.000

Table 7.25 - Contents of Processed Seam Data File.

JTYPE	XSECT	T1	T2	NPOINT	ICURVE								
1	0.500	0.250	0.500	26	1								
INDEX	S	R(X)	R(Y)	R(Z)	T(X)	T(Y)	T(Z)	A(X)	A(Y)	A(Z)	B(X)	B(Y)	B(Z)
1	0.000	200.000	100.000	100.000	-0.031	1.000	0.000	0.707	0.022	0.707	-0.707	-0.022	0.707
2	6.282	199.803	106.279	100.000	-0.063	0.998	0.000	0.705	0.044	0.708	-0.706	-0.044	0.706
3	12.564	199.211	112.533	100.000	-0.125	0.992	0.000	0.701	0.089	0.708	-0.702	-0.089	0.706
4	18.846	198.229	118.738	100.000	-0.187	0.982	0.000	0.694	0.132	0.708	-0.695	-0.133	0.706
5	25.129	196.858	124.869	100.000	-0.249	0.969	0.000	0.684	0.176	0.708	-0.686	-0.176	0.706
6	31.411	195.106	130.902	100.000	-0.309	0.951	0.000	0.672	0.218	0.708	-0.673	-0.219	0.706
7	37.693	192.978	136.812	100.000	-0.368	0.930	0.000	0.657	0.260	0.708	-0.658	-0.261	0.706
8	43.975	190.483	142.578	100.000	-0.426	0.905	0.000	0.639	0.301	0.708	-0.640	-0.301	0.706
9	50.257	187.631	148.175	100.000	-0.482	0.876	0.000	0.619	0.340	0.708	-0.620	-0.341	0.706
10	56.539	184.433	153.583	100.000	-0.536	0.844	0.000	0.596	0.379	0.708	-0.598	-0.379	0.706
11	62.821	180.902	158.779	100.000	-0.588	0.809	0.000	0.571	0.415	0.708	-0.573	-0.416	0.706
12	69.104	177.051	163.742	100.000	-0.637	0.771	0.000	0.544	0.450	0.708	-0.545	-0.451	0.706
13	75.386	172.897	168.455	100.000	-0.685	0.729	0.000	0.515	0.484	0.708	-0.516	-0.485	0.706
14	81.668	168.455	172.897	100.000	-0.729	0.685	0.000	0.484	0.515	0.708	-0.485	-0.516	0.706
15	87.950	163.742	177.051	100.000	-0.771	0.637	0.000	0.450	0.544	0.708	-0.451	-0.545	0.706
16	94.232	158.779	180.902	100.000	-0.809	0.588	0.000	0.415	0.571	0.708	-0.416	-0.573	0.706
17	100.514	153.583	184.433	100.000	-0.844	0.536	0.000	0.379	0.596	0.708	-0.379	-0.598	0.706
18	106.797	148.175	187.631	100.000	-0.876	0.482	0.000	0.340	0.619	0.708	-0.341	-0.620	0.706
19	113.079	142.578	190.483	100.000	-0.905	0.426	0.000	0.301	0.639	0.708	-0.301	-0.640	0.706
20	119.361	136.813	192.978	100.000	-0.930	0.368	0.000	0.260	0.657	0.708	-0.261	-0.658	0.706
21	125.643	130.902	195.106	100.000	-0.951	0.309	0.000	0.218	0.672	0.708	-0.219	-0.673	0.706
22	131.925	124.869	196.858	100.000	-0.969	0.249	0.000	0.176	0.684	0.708	-0.176	-0.686	0.706
23	138.207	118.738	198.229	100.000	-0.982	0.187	0.000	0.132	0.694	0.708	-0.133	-0.695	0.706
24	144.489	112.533	199.211	100.000	-0.992	0.125	0.000	0.089	0.701	0.708	-0.089	-0.702	0.706
25	150.772	106.279	199.803	100.000	-0.998	0.063	0.000	0.044	0.705	0.708	-0.044	-0.706	0.706
26	157.054	100.000	200.000	100.000	-1.000	0.031	0.000	0.022	0.707	0.707	-0.022	-0.707	0.707

Table 7.26 - Joint Angles Calculated during
Kinematic Feasibility Search.

0 0 1 1 0 0 LEFTY BELOW FLIP CONFIGURATION													
JOINT ANGLES (DEGREES)							JOINT LIMITS EXCEEDED						TOOL ROT.
INDEX	J1	J2	J3	J4	J5	J6	1	2	3	4	5	6	
1	-6.622	16.015	-50.077	-165.980	-148.506	172.931	0	0	0	1	1	0	Y
1	-15.533	13.294	-50.437	-129.587	-147.050	-159.095	0	0	0	1	1	0	Y
1	1.934	19.744	-49.250	163.429	-140.524	151.316	0	0	0	0	1	0	Y
1	-23.432	12.170	-50.308	-104.463	-138.041	-142.106	0	0	0	0	1	0	Y
1	9.213	23.869	-48.006	144.246	-128.539	141.902	0	0	0	0	1	0	Y
1	-29.204	12.847	-49.703	-90.092	-127.680	-135.899	0	0	0	0	1	0	Y
1	14.867	27.948	-46.416	131.030	-116.493	139.435	0	0	0	0	1	0	Y
1	-32.589	14.993	-48.661	-80.849	-118.916	-134.963	0	0	0	0	1	0	Y
1	18.922	31.719	-44.561	120.298	-105.845	140.484	0	0	0	0	1	0	Y
1	-33.991	18.010	-47.247	-73.811	-112.241	-136.320	0	0	0	0	1	0	Y
1	21.564	35.056	-42.524	110.545	-97.011	143.488	0	0	0	0	0	0	
2	23.044	35.421	-42.404	110.839	-96.392	143.565	0	0	0	0	0	0	
3	24.754	35.970	-42.832	112.562	-95.680	142.869	0	0	0	0	0	0	
4	26.424	36.533	-43.311	114.323	-95.033	142.093	0	0	0	0	0	0	
5	28.053	37.107	-43.840	116.123	-94.458	141.240	0	0	0	0	0	0	
6	29.640	37.689	-44.420	117.966	-93.960	140.316	0	0	0	0	0	0	
7	31.182	38.278	-45.049	119.854	-93.547	139.325	0	0	0	0	0	0	
8	32.679	38.871	-45.727	121.788	-93.226	138.271	0	0	0	0	0	0	
9	34.128	39.465	-46.454	123.772	-93.003	137.159	0	0	0	0	0	0	
10	35.526	40.055	-47.227	125.807	-92.886	135.996	0	0	0	0	0	0	
11	36.871	40.639	-48.048	127.894	-92.882	134.786	0	0	0	0	0	0	
12	38.159	41.212	-48.915	130.035	-93.000	133.538	0	0	0	0	0	0	
13	39.385	41.769	-49.828	132.233	-93.248	132.258	0	0	0	0	0	0	
14	40.546	42.304	-50.785	134.489	-93.635	130.955	0	0	0	0	0	0	
15	41.636	42.809	-51.787	136.807	-94.170	129.637	0	0	0	0	0	0	
16	42.647	43.278	-52.833	139.191	-94.866	128.313	0	1	1	0	0	0	Y
16	38.475	37.889	-53.846	148.929	-108.479	126.679	0	0	1	0	1	0	Y
16	45.565	48.083	-51.330	129.697	-82.751	132.566	0	1	0	0	0	0	Y
16	33.086	32.310	-54.290	160.471	-122.654	128.722	0	0	1	0	1	0	Y
16	47.332	52.087	-49.435	119.593	-72.640	138.979	0	1	0	0	0	0	Y
16	26.766	27.090	-54.127	176.471	-135.671	136.549	0	0	1	1	1	0	Y
16	48.106	55.213	-47.252	108.481	-64.807	147.302	0	1	0	0	0	0	Y
16	20.135	22.834	-53.364	-160.138	-144.740	152.536	0	0	1	1	1	0	Y
16	48.055	57.475	-44.881	96.368	-59.387	157.190	0	1	0	0	0	0	Y
16	14.035	20.008	-52.055	-131.926	-147.283	173.638	0	0	1	1	1	0	Y
16	47.335	58.936	-42.410	83.650	-56.382	167.998	0	1	0	0	0	0	Y
16	9.197	18.748	-50.286	-108.091	-144.409	-169.518	0	0	0	0	1	0	Y
16	46.083	59.685	-39.917	70.993	-55.630	178.812	0	1	0	0	0	0	Y
16	5.927	18.834	-48.162	-91.535	-139.649	-159.767	0	0	0	0	1	0	Y
16	44.419	59.811	-37.470	59.058	-56.811	-171.253	0	1	0	0	0	0	Y
16	4.119	19.868	-45.788	-79.772	-135.107	-154.506	0	0	0	0	1	0	Y
16	42.442	59.406	-35.127	48.262	-59.518	-162.748	0	1	0	0	0	0	Y
16	3.473	21.479	-43.267	-70.605	-131.349	-151.436	0	0	0	0	1	0	Y

***** NO FEASIBLE PATH FOUND *****

Table 7.26/cont.

0 0 1 0 0 0 LEFTY BELOW NOFLIP CONFIGURATION												
1	-6.622	16.015	-50.077	14.020	148.506	-7.069	0	0	0	0	1	0
1	-15.533	13.294	-50.437	50.414	147.050	20.906	0	0	0	0	1	0
1	1.934	19.744	-49.250	-16.571	140.524	-28.684	0	0	0	0	1	0
1	-23.432	12.170	-50.308	75.537	138.041	37.894	0	0	0	0	1	0
1	9.213	23.869	-48.006	-35.755	128.539	-38.098	0	0	0	0	1	0
1	-29.204	12.847	-49.703	89.908	127.680	44.102	0	0	0	0	1	0
1	14.867	27.948	-46.416	-48.970	116.493	-40.565	0	0	0	0	1	0
1	-32.589	14.993	-48.661	99.152	118.916	45.038	0	0	0	0	1	0
1	18.922	31.719	-44.561	-59.703	105.845	-39.516	0	0	0	0	1	0
1	-33.991	18.010	-47.247	106.190	112.241	43.680	0	0	0	0	1	0
1	21.564	35.056	-42.524	-69.455	97.011	-36.512	0	0	0	0	0	0
2	23.044	35.421	-42.404	-69.161	96.392	-36.435	0	0	0	0	0	0
3	24.754	35.970	-42.832	-67.438	95.680	-37.131	0	0	0	0	0	0
4	26.424	36.533	-43.311	-65.678	95.033	-37.908	0	0	0	0	0	0
5	28.053	37.107	-43.840	-63.877	94.458	-38.760	0	0	0	0	0	0
6	29.640	37.689	-44.420	-62.034	93.960	-39.684	0	0	0	0	0	0
7	31.182	38.278	-45.049	-60.146	93.547	-40.675	0	0	0	0	0	0
8	32.679	38.871	-45.727	-58.212	93.226	-41.729	0	0	0	0	0	0
9	34.128	39.465	-46.454	-56.228	93.003	-42.841	0	0	0	0	0	0
10	35.526	40.055	-47.227	-54.193	92.886	-44.005	0	0	0	0	0	0
11	36.871	40.639	-48.048	-52.106	92.882	-45.214	0	0	0	0	0	0
12	38.159	41.212	-48.915	-49.965	93.000	-46.462	0	0	0	0	0	0
13	39.385	41.769	-49.828	-47.767	93.248	-47.742	0	0	0	0	0	0
14	40.546	42.304	-50.785	-45.511	93.635	-49.045	0	0	0	0	0	0
15	41.636	42.809	-51.787	-43.193	94.170	-50.364	0	0	0	0	0	0
16	42.647	43.278	-52.833	-40.809	94.866	-51.687	0	1	1	0	0	0
16	38.475	37.889	-53.846	-31.072	108.479	-53.321	0	0	1	0	1	0
16	45.565	48.083	-51.330	-50.303	82.751	-47.434	0	1	0	0	0	0
16	33.086	32.310	-54.290	-19.529	122.654	-51.278	0	0	1	0	1	0
16	47.332	52.087	-49.435	-60.407	72.640	-41.021	0	1	0	0	0	0
16	26.766	27.090	-54.127	-3.529	135.671	-43.451	0	0	1	0	1	0
16	48.106	55.213	-47.252	-71.519	64.807	-32.698	0	1	0	0	0	0
16	20.135	22.834	-53.364	19.862	144.740	-27.464	0	0	1	0	1	0
16	48.055	57.475	-44.881	-83.632	59.387	-22.810	0	1	0	0	0	0
16	14.035	20.008	-52.055	48.074	147.283	-6.363	0	0	1	0	1	0
16	47.335	58.936	-42.410	-96.350	56.382	-12.002	0	1	0	0	0	0
16	9.197	18.748	-50.286	71.909	144.409	10.482	0	0	0	0	1	0
16	46.083	59.685	-39.917	-109.007	55.630	-1.188	0	1	0	0	0	0
16	5.927	18.834	-48.162	88.465	139.649	20.233	0	0	0	0	1	0
16	44.419	59.811	-37.470	-120.943	56.811	8.747	0	1	0	1	0	0
16	4.119	19.868	-45.788	100.228	135.107	25.495	0	0	0	0	1	0
16	42.442	59.406	-35.127	-131.738	59.518	17.252	0	1	0	1	0	0
16	3.473	21.479	-43.267	109.395	131.349	28.564	0	0	0	0	1	0

***** NO FEASIBLE PATH FOUND *****

Table 7.26/cont.

0 0 0 1 0 0 LEFTY ABOVE FLIP CONFIGURATION												
1	-6.622	-127.357	-124.502	-133.965	-10.127	115.326	0	0	1	1	0	0
1	-15.533	-130.443	-124.142	-119.274	-28.721	98.042	0	0	1	1	0	0
1	1.934	-122.787	-125.329	91.741	-10.452	-107.475	0	0	1	0	0	0
1	-23.432	-131.437	-124.270	-116.376	-46.273	92.552	0	0	1	1	0	0
1	9.213	-117.398	-126.573	79.061	-27.743	-91.496	0	0	1	0	0	0
1	-29.204	-130.144	-124.876	-115.454	-61.225	88.686	0	0	1	1	0	0
1	14.867	-111.706	-128.163	77.822	-43.685	-86.588	0	0	0	0	0	0
2	16.438	-111.159	-128.304	77.000	-43.941	-86.257	0	0	0	0	0	0
3	17.992	-111.084	-127.995	74.785	-43.004	-85.888	0	0	1	0	0	0
3	12.386	-116.682	-126.523	75.333	-26.972	-90.091	0	0	1	0	0	0
3	22.065	-105.462	-129.764	75.895	-57.359	-83.042	0	0	0	0	0	0
4	23.651	-105.423	-129.335	73.803	-56.564	-82.802	0	0	0	0	0	0
5	25.192	-105.429	-128.860	71.678	-55.741	-82.549	0	0	0	0	0	0
6	26.687	-105.483	-128.339	69.518	-54.889	-82.279	0	0	0	0	0	0
7	28.134	-105.585	-127.773	67.318	-54.005	-81.990	0	0	1	0	0	0
7	23.730	-111.269	-126.335	65.426	-38.939	-84.115	0	0	1	0	0	0
7	31.182	-99.991	-129.530	69.547	-67.500	-79.883	0	0	0	0	0	0
8	32.679	-100.085	-128.852	67.435	-66.781	-79.659	0	0	0	0	0	0
9	34.128	-100.228	-128.125	65.288	-66.036	-79.420	0	0	0	0	0	0
10	35.526	-100.422	-127.352	63.102	-65.261	-79.164	0	0	1	0	0	0
10	32.162	-106.218	-125.806	60.438	-51.142	-80.967	0	0	1	0	0	0
10	37.670	-94.874	-129.211	65.891	-77.900	-77.146	0	0	0	0	0	0
11	39.124	-95.021	-128.340	63.779	-77.307	-76.913	0	0	0	0	0	0
12	40.528	-95.217	-127.419	61.630	-76.694	-76.667	0	0	1	0	0	0
12	38.159	-100.978	-125.664	58.598	-63.613	-78.594	0	0	1	0	0	0
12	41.814	-89.870	-129.451	64.783	-88.397	-74.504	0	0	0	0	0	0
13	43.273	-89.993	-128.451	62.708	-87.967	-74.248	0	0	0	0	0	0
14	44.681	-90.159	-127.399	60.596	-87.526	-73.982	0	0	1	0	0	0
14	43.172	-95.769	-125.430	57.206	-75.393	-76.134	0	0	1	0	0	0
14	45.239	-85.107	-129.604	64.130	-98.385	-71.561	0	0	0	0	0	0
15	46.698	-85.181	-128.491	62.102	-98.121	-71.260	0	0	0	0	0	0
16	48.106	-85.290	-127.327	60.037	-97.859	-70.950	0	0	1	0	0	0
16	47.332	-90.632	-125.144	56.244	-86.606	-73.418	0	0	1	0	0	0
16	48.055	-80.624	-129.698	64.028	-107.936	-68.144	0	0	0	0	1	0
16	45.565	-96.564	-123.249	52.576	-73.971	-75.537	0	0	1	0	0	0
16	47.335	-76.660	-132.169	68.310	-117.038	-64.914	0	0	0	0	1	0
16	42.647	-102.900	-121.746	48.887	-59.805	-77.134	0	0	1	0	0	0
16	46.083	-73.391	-134.662	73.050	-125.329	-61.085	0	0	0	0	1	0
16	38.475	-109.321	-120.733	44.679	-44.120	-77.653	0	0	1	0	0	0
16	44.419	-70.793	-137.109	78.531	-132.913	-56.367	0	0	0	0	1	0
16	33.086	-115.354	-120.289	37.913	-27.261	-75.019	0	0	1	0	0	0
16	42.442	-68.833	-139.452	85.196	-139.810	-50.318	0	0	0	0	1	0
16	26.766	-120.408	-120.452	13.349	-10.736	-54.043	0	0	1	0	0	0
16	40.235	-67.479	-141.641	93.712	-145.928	-42.294	0	0	0	0	1	0
16	20.135	-123.886	-121.215	-94.731	-11.351	50.982	0	0	1	0	0	0
16	37.865	-66.698	-143.633	104.943	-151.019	-31.470	0	0	0	0	1	0
16	14.035	-125.376	-122.524	-116.452	-26.690	69.617	0	0	1	1	0	0
16	35.387	-66.459	-145.388	119.595	-154.632	-17.215	0	0	0	0	1	0
16	9.197	-124.836	-124.293	-122.530	-41.009	72.580	0	0	1	1	0	0

***** NO FEASIBLE PATH FOUND *****

Table 7.26/cont.

0 0 0 0 0 0						LEFTY ABOVE NOFLIP CONFIGURATION									
1	-6.622	-127.357	-124.502	46.035	10.127	-64.674	0	0	1	0	0	0	0	0	0
1	-15.533	-130.443	-124.142	60.726	28.721	-81.958	0	0	1	0	0	0	0	0	0
1	1.934	-122.787	-125.329	-88.259	10.452	72.525	0	0	1	0	0	0	0	0	0
1	-23.432	-131.437	-124.270	63.625	46.273	-87.448	0	0	1	0	0	0	0	0	0
1	9.213	-117.398	-126.573	-100.939	27.743	88.505	0	0	1	0	0	0	0	0	0
1	-29.204	-130.144	-124.876	64.546	61.225	-91.314	0	0	1	0	0	0	0	0	0
1	14.867	-111.706	-128.163	-102.178	43.685	93.412	0	0	0	0	0	0	0	0	0
2	16.438	-111.159	-128.304	-103.000	43.941	93.743	0	0	0	0	0	0	0	0	0
3	17.992	-111.084	-127.995	-105.215	43.004	94.112	0	0	1	0	0	0	0	0	0
3	12.386	-116.682	-126.523	-104.667	26.972	89.909	0	0	1	0	0	0	0	0	0
3	22.065	-105.462	-129.764	-104.105	57.359	96.958	0	0	0	0	0	0	0	0	0
4	23.651	-105.423	-129.335	-106.197	56.564	97.198	0	0	0	0	0	0	0	0	0
5	25.192	-105.429	-128.860	-108.322	55.741	97.451	0	0	0	0	0	0	0	0	0
6	26.687	-105.483	-128.339	-110.482	54.889	97.721	0	0	0	1	0	0	0	0	0
6	22.371	-111.146	-126.813	-112.146	40.007	95.384	0	0	1	1	0	0	0	0	0
6	29.640	-99.942	-130.159	-108.373	68.194	99.906	0	0	0	0	0	0	0	0	0
7	31.182	-99.991	-129.530	-110.453	67.500	100.117	0	0	0	1	0	0	0	0	0
7	28.134	-105.585	-127.773	-112.682	54.005	98.010	0	0	1	1	0	0	0	0	0
7	33.029	-94.695	-131.521	-107.963	79.566	102.223	0	0	0	0	0	0	0	0	0
8	34.620	-94.713	-130.802	-109.983	79.028	102.422	0	0	0	0	0	0	0	0	0
9	36.167	-94.772	-130.032	-112.031	78.473	102.633	0	0	0	1	0	0	0	0	0
9	34.128	-100.228	-128.125	-114.712	66.036	100.581	0	0	0	1	0	0	0	0	0
9	37.163	-89.737	-132.139	-109.176	89.630	104.791	0	0	0	0	0	0	0	0	0
10	38.757	-89.744	-131.296	-111.162	89.228	105.016	0	0	0	1	0	0	0	0	0
10	37.670	-94.874	-129.211	-114.110	77.900	102.854	0	0	0	1	0	0	0	0	0
10	38.958	-85.129	-133.520	-108.048	99.452	107.341	0	0	0	0	0	0	0	0	0
11	40.590	-85.078	-132.622	-109.966	99.184	107.600	0	0	0	0	0	0	0	0	0
12	42.183	-85.057	-131.670	-111.908	98.917	107.869	0	0	0	1	0	0	0	0	0
12	41.814	-89.870	-129.451	-115.217	88.397	105.496	0	0	0	1	0	0	0	0	0
12	41.795	-80.835	-133.991	-108.401	108.459	110.503	0	0	0	0	1	0	0	0	0
12	40.528	-95.217	-127.419	-118.370	76.694	103.333	0	0	1	1	0	0	0	0	0
12	40.795	-77.224	-136.339	-104.608	117.202	113.516	0	0	0	0	1	0	0	0	0
12	38.159	-100.978	-125.664	-121.402	63.613	101.406	0	0	1	1	0	0	0	0	0
12	39.303	-74.225	-138.648	-100.364	125.282	117.097	0	0	0	0	1	0	0	0	0
12	34.556	-106.944	-124.279	-124.450	49.025	99.895	0	0	1	1	0	0	0	0	0
12	37.426	-71.826	-140.860	-95.387	132.776	121.538	0	0	0	0	1	0	0	0	0
12	29.654	-112.785	-123.350	-128.116	32.968	99.463	0	0	1	1	0	0	0	0	0
12	35.251	-70.010	-142.925	-89.232	139.690	127.284	0	0	0	0	1	0	0	0	0
12	23.564	-118.043	-122.941	-136.077	15.842	103.805	0	0	1	1	0	0	0	0	0
12	32.853	-68.758	-144.801	-81.217	145.921	135.001	0	0	0	0	1	0	0	0	0
12	16.704	-122.169	-123.085	111.544	4.344	-146.938	0	0	1	0	0	0	0	0	0
12	30.293	-68.050	-146.450	-70.406	151.208	145.588	0	0	0	0	1	0	0	0	0
12	9.829	-124.643	-123.774	65.765	20.264	-104.146	0	0	1	0	0	0	0	0	0
12	27.623	-67.865	-147.840	-55.911	155.077	159.867	0	0	0	0	1	0	0	0	0
12	3.826	-125.168	-124.964	60.579	36.131	-101.906	0	0	1	0	0	0	0	0	0

***** NO FEASIBLE PATH FOUND *****

Table 7.26/cont.

	1	1	1	1	0	0	RIGHTY ABOVE FLIP CONFIGURATION									
1	-98.962	-52.644	-50.077	-67.121	-75.397	-98.233	0	0	0	0	0	0	0	0	0	0
2	-98.225	-53.398	-49.854	-67.355	-74.593	-98.659	0	0	0	0	0	0	0	0	0	0
3	-97.183	-53.823	-49.805	-67.582	-72.849	-100.365	0	0	0	0	0	0	0	0	0	0
4	-96.089	-54.198	-49.788	-67.736	-71.153	-102.082	0	0	0	0	0	0	0	0	0	0
5	-94.946	-54.521	-49.803	-67.815	-69.505	-103.814	0	0	0	0	0	0	0	0	0	0
6	-93.755	-54.795	-49.850	-67.817	-67.903	-105.566	0	0	0	0	0	0	0	0	0	0
7	-92.518	-55.018	-49.930	-67.740	-66.349	-107.342	0	0	0	0	0	0	0	0	0	0
8	-91.237	-55.190	-50.042	-67.581	-64.841	-109.146	0	0	0	0	0	0	0	0	0	0
9	-89.912	-55.311	-50.185	-67.336	-63.382	-110.982	0	0	0	0	0	0	0	0	0	0
10	-88.543	-55.380	-50.361	-67.003	-61.973	-112.855	0	0	0	0	0	0	0	0	0	0
11	-87.132	-55.396	-50.567	-66.577	-60.615	-114.769	0	0	0	0	0	0	0	0	0	0
12	-85.678	-55.357	-50.805	-66.056	-59.311	-116.728	0	0	0	0	0	0	0	0	0	0
13	-84.181	-55.263	-51.073	-65.435	-58.065	-118.737	0	0	0	0	0	0	0	0	0	0
14	-82.639	-55.111	-51.371	-64.711	-56.881	-120.798	0	0	0	0	0	0	0	0	0	0
15	-81.053	-54.899	-51.699	-63.881	-55.763	-122.916	0	0	0	0	0	0	0	0	0	0
16	-79.419	-54.624	-52.055	-62.940	-54.717	-125.094	0	0	1	0	0	0	0	0	0	0
16	-86.488	-55.164	-50.286	-51.503	-41.809	-137.531	0	0	0	0	0	0	0	0	0	0
17	-85.015	-55.113	-50.532	-49.679	-41.024	-140.528	0	0	0	0	0	0	0	0	0	0
18	-83.498	-55.005	-50.808	-47.739	-40.346	-143.598	0	0	0	0	0	0	0	0	0	0
19	-81.937	-54.839	-51.115	-45.688	-39.780	-146.735	0	0	0	0	0	0	0	0	0	0
20	-80.330	-54.613	-51.450	-43.536	-39.332	-149.925	0	0	0	0	0	0	0	0	0	0
21	-78.676	-54.323	-51.814	-41.298	-39.009	-153.154	0	0	0	0	0	0	0	0	0	0
22	-76.972	-53.966	-52.206	-38.987	-38.815	-156.406	0	0	1	0	0	0	0	0	0	0
22	-85.331	-57.729	-49.444	-14.283	-32.257	177.540	0	0	0	0	0	0	0	0	0	0
23	-83.691	-57.592	-49.774	-11.733	-32.880	174.018	0	0	0	0	0	0	0	0	0	0
24	-82.007	-57.397	-50.137	-9.302	-33.613	170.664	0	0	0	0	0	0	0	0	0	0
25	-80.278	-57.139	-50.531	-6.999	-34.451	167.488	0	0	0	0	0	0	0	0	0	0
26	-78.432	-56.701	-51.010	-6.902	-35.065	167.575	0	0	0	0	0	0	0	0	0	0

***** FEASIBLE PATH !!! *****

Table 7.27 - Generated Robot Location and Weld Parameter Data

ALPHA	BETA	STICKOUT	VOLTAGE	FEED	GFR	NPOINT		
15.00	0.00	0.403	16.00	217.09	21.11	26		
INDEX	X	Y	Z	O	A	T	CURRENT	SPEED
1	221.4	131.8	205.0	-88.60	155.01	164.32	137.00	1.228
2	220.1	138.6	204.6	-87.19	155.03	163.64	136.95	1.228
3	217.7	145.9	203.6	-84.38	154.94	162.29	136.84	1.228
4	214.7	153.0	202.6	-81.57	154.79	160.94	136.72	1.228
5	211.4	159.8	201.4	-78.76	154.58	159.60	136.59	1.228
6	207.7	166.5	200.0	-75.95	154.31	158.26	136.45	1.228
7	203.6	172.9	198.5	-73.14	153.98	156.95	136.30	1.228
8	199.1	179.0	196.9	-70.34	153.59	155.64	136.15	1.228
9	194.2	184.8	195.1	-67.53	153.15	154.36	136.00	1.228
10	189.1	190.2	193.2	-64.72	152.65	153.09	135.84	1.228
11	183.6	195.3	191.2	-61.92	152.10	151.85	135.68	1.228
12	177.8	200.0	189.1	-59.11	151.49	150.64	135.52	1.228
13	171.8	204.4	187.0	-56.30	150.83	149.45	135.37	1.228
14	165.5	208.3	184.7	-53.49	150.13	148.29	135.21	1.228
15	159.0	211.8	182.3	-50.67	149.37	147.17	135.05	1.228
16	145.5	208.9	186.8	-37.13	143.23	152.08	134.90	1.228
17	159.0	212.4	184.4	-34.29	142.23	151.18	134.75	1.228
18	152.3	215.5	181.9	-31.43	141.20	150.33	134.60	1.228
19	145.4	218.1	179.4	-28.56	140.13	149.53	134.46	1.228
20	138.3	220.3	176.9	-25.65	139.04	148.79	134.33	1.228
21	131.2	222.0	174.3	-22.71	137.92	148.12	134.20	1.228
22	138.9	224.9	177.1	-6.69	131.94	156.81	134.07	1.228
23	131.4	226.7	174.4	-3.43	130.71	156.65	133.96	1.228
24	123.9	227.9	171.6	-0.10	129.48	156.59	133.85	1.228
25	116.3	228.6	168.9	3.33	128.25	156.64	133.75	1.228
26	109.6	228.9	166.4	5.07	127.60	156.72	133.70	1.228

to the PUMA and executing the resulting path program, but the Expert Welder output has not yet been tested.

7.8 Future Capabilities

Several important capabilities have not yet been fully developed and implemented in AUTOP. These capabilities are summarized below:

7.8.1 Real-time Seam Tracking

Eventually, we will require AUTOP to run in a real-time environment with sensory feedback from the welding process. AUTOP will be required to automatically adapt the planned trajectory to any real-world deviations.

7.8.2 CAD System Interface

CAD data will ultimately be generated by a commercial CAD system. The CAD system will operate independently from AUTOP, but additional software will be required to translate the CAD output data into the correct format for input to AUTOP.

7.8.3 Generation of High-level Descriptive Program

A sophisticated welding workstation contains several programmable remotely controlled devices, including a robot, positioning table and welding machine. In order to realize the full benefit of automatic programming it is desirable to be able to program and control the workstation in a single, task-oriented high-level language.

Clearly the welding process can be broken down into a number of actions occurring in a given sequence. Actions must be taken by

different physical devices, either independently or in synchronization with other actions.

An example of independent actions: turn on the welder and reposition the table. An example of synchronized actions: turn on the welder and move the torch along the seam. A simplified high level welding task description is illustrated in Fig. 7.7.

Correct synchronization of device actions requires some form of scheduling based on task precedence requirements. We could create a task precedence table such as in Table 7.28 for scheduling control. All tasks at a given level must be completed before any tasks at the next level can start.

This type of action table can be implemented in a high level descriptive programming language such as FORTRAN, running under a real-time operating system. Standard real-time operating system practices can be used to achieve correct action synchronization. For example, we can use device interrupts, polling of device flags or semaphores to signal completion of an action to the program. The device drivers can be implemented as low-level device dependent macros called as subroutines from the high-level program.

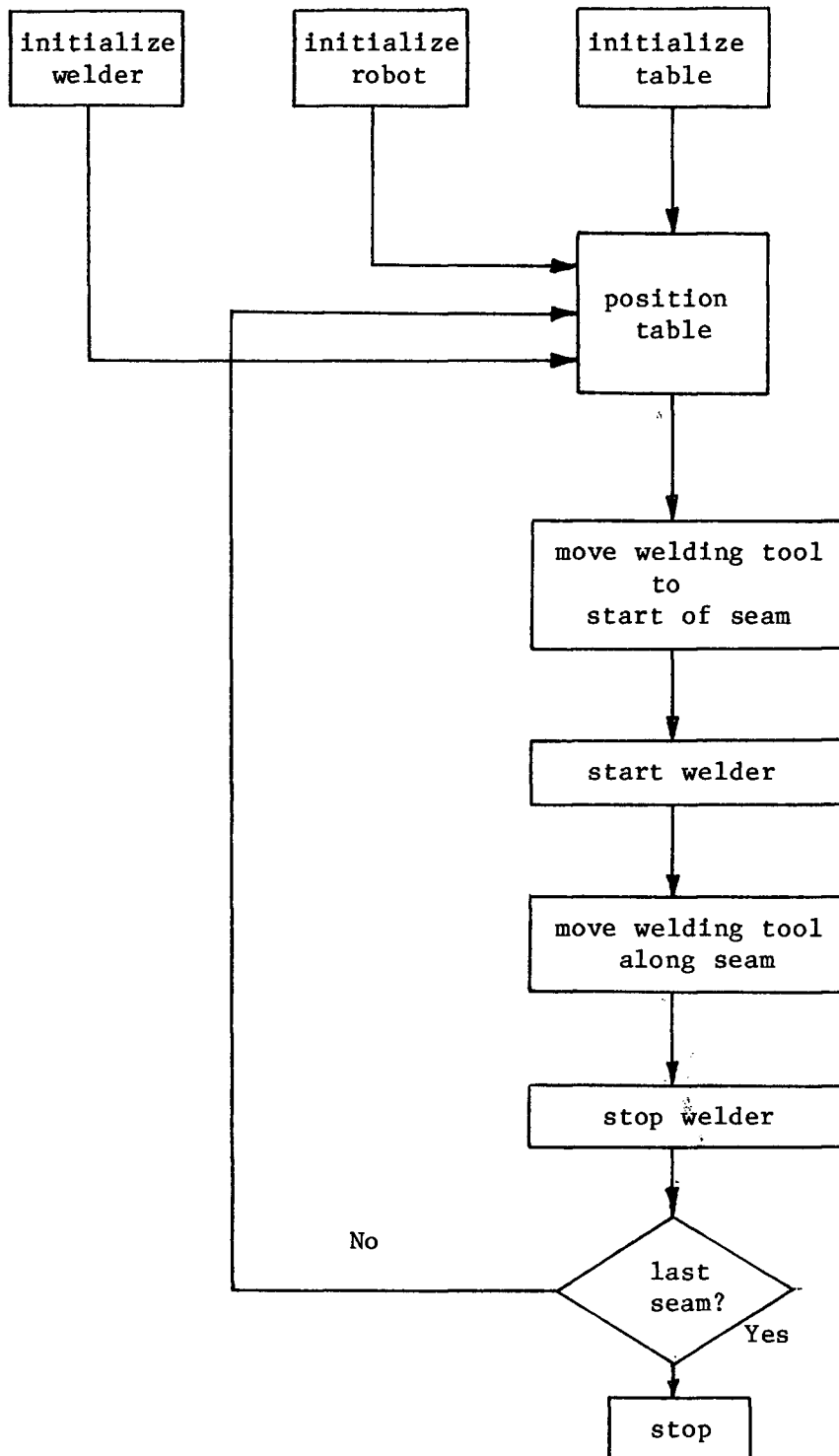


Figure 7.7 Simplified High-level Task Description.

Table 7.28 Task Precedence Table

Welder	Robot	Table
initialize	initialize	initialize
		position
	move to seam	
start welder		
	move along seam	
stop welder		

8. CONCLUSIONS

The work described in this thesis provides a solid basis for a commercially useful automatic programming system. Some necessary additions and desirable improvements are suggested which require more time and effort to develop, debug and implement. The knowledge, experience and techniques gained from this project are highly useful for any robot application, and a fully automatic programming system for a welding robot can be readily adapted to other applications.

8.1 Analysis and Evaluation of Work Completed

Solutions to some of the major problems have been developed and described in this thesis and the solutions implemented, while other problems require further work. The following list summarizes the important capabilities of the software package developed to date:

1. Automatic weld parameter selection.
2. Interactive selection of table positions and seam welding order.
3. Automatic checking for robot trajectory kinematic feasibility.
4. Generation of robot location coordinates for each seam.

We can use the output generated by the software at this stage to simplify the programming of the robot, but some manual programming must still be done. A simple interference detection scheme has been developed, and implemented in computer program TESTIN but has not yet been incorporated into AUTOP.

8.2 Suggestions for Further Work

A number of the problem areas discussed in this thesis have not been considered in any depth. We can identify a number of areas which require further work before a commercially feasible system can be realized:

1. Develop and generate high-level task-oriented descriptive programming methods for an integrated workstation. This is necessary if we wish to control a number of machines (welder, robot, table) as an integrated system from a central computer. To date, there has been little standardization of languages or communications protocols for this purpose.
2. Refine the method for specifying the welding procedure in AUTOP. The welding procedure should include duty cycle optimization, weld seam orientation optimization, and continuous optimization of welding parameters. A continuously repositioning table can be accommodated with modifications to the current software.
3. Develop a practical path planning or collision avoidance scheme. Initially, an interactive system employing graphical simulation and collision detection should be developed. Eventually a more sophisticated path-planning algorithm should be used to minimize operator input.

4. Develop and incorporate real-time seam tracking. As a minimum requirement, the seam tracker should provide positional error information to AUTOP in real-time. AUTOP will then modify the nominal path appropriately. Real-time weld puddle analysis and appropriate weld parameter modification is a desirable capability for ensuring uniformly high weld quality, and is a worthwhile area for future research.
5. Interface the software to a commercial CAD system and/or develop a simple dedicated CAD system for generating object specifications from a drawing or prototype.
6. Interface AUTOP to a real welding workstation. This requires high-level task-oriented programs and real-time host computer/machine communication. The interface must be customized to each installation due to lack of standardization.
7. Refine the Expert Welder module, and develop a reliable, comprehensive welding data base.
8. Continually evaluate, modify and refine AUTOP as problems are encountered and experience is gained.

REFERENCES

1. Ambler, A.P., Popplestone, R.J., Kempf, K.G., "An Experiment in the Offline Programming of Robots", Proc. 12th Int. Symp. on Industrial Robots, Paris 1982.
2. Baer, A., Eastman, C., Henrion, M., "Geometric Modelling: a Survey", Computer-aided Design, Vol.11, No.5, 1979, pp.253-72.
3. Benati, M., Morasso, P., Tagliasco, V., "The Inverse Kinematic Problem for Anthropomorphic Manipulator Arms", Transactions of the ASME, Vol.104, March 1982, pp.110-113.
4. Boyse, J., "Interference Detection Among Solids and Surfaces", Communications of the ACM, Vol.22, No.1, 1979, pp.3-9.
5. Brady, Hollerbach, Johnson, Lozano-Pérez, Mason, eds., Robot Motion: Planning and Control, MIT Press, 1982.
6. Braid, I.C., "The Synthesis of Solids Bounded by Many Faces", Communications of the ACM, Vol.18, No.4, 1975, pp.209-216.
7. Brooks, R., "Planning Collision-Free Motions for Pick-and-Place Operations", International Journal of Robotics Research, Vol2, No.4, 1983, pp.19-44.
8. Buchal, R., Sassani, F., Cherchas, D., Duncan, J.P., "Development of an Automatic Welding System for a Welding Robot: Phase I", University of B.C. Research Contract Report for National Research Council, Industrial Materials Research Institute, 1984.
9. Denavit, J., Hartenberg, R.S., "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices", Journal of Applied Mechanics, Vol.22, No.1, 1955, pp.215-221.
10. Fournier, A., Khalil, W., "Coordination and Reconfiguration of Mechanical Redundant Systems", Proc. International Conf. Cybernetics Society, 1977.
11. Gaglio, S., Morasso, P., Tagliasco, V., Zaccaria, R., "Computation of Inverse Kinematics and Inverse Dynamics in Manipulator Arm Control", Proc. 11th Int. Symp. on Industrial Robots, Tokyo, 1981.
12. Gruver, W., Soroka, B., Craig, J., Turner, T., "Evaluation of Commercially Available Robot Programming Languages", Proc. 13th Int. Symp. on Industrial Robots, Chicago, 1983.
13. Hosaka, M., Kimura, F., Kakishita, N., "A Unified Method for Processing Polyhedra", Proc. Information Processing 74, 1974.

14. Kempf, K., "Robot Command Languages and Artificial Intelligence", Proc. Robots VI, Detroit, 1982.
15. Konstantinov, M., Markov, M., "Discrete Positions Method in Kinematics and Control of Spatial Linkages", Mechanism and Machine Theory, Vol.15, 1980, pp.47-60.
16. Konstantinov, M., Genova, P., Zahariev, E., "Direct Kinematic Control of Industrial Manipulators and Robots", Proc. 11th Int. Symp. on Industrial Robots, Tokyo, 1981.
17. Konstantinov, M., Markov, M., Nechev, D., "Kinematic Control of Redundant Manipulators", Proc. 11th Int. Symp. on Industrial Robots, Tokyo, 1981.
18. Konstantinov, M., Patarinski, S., "A Contribution to the Inverse Kinematic Problem for Industrial Robots", Proc. 12th Int. Symp. on Industrial Robots, Paris, 1982.
19. Lee, C., Ziegler, M., "A Geometric Approach to Solving the Inverse Kinematics of PUMA Robots", Proc. 13th Int. Symp. on Industrial Robots, Chicago, 1983.
20. Liegeois, A., "Automatic Supervisory Control of the Configuration and Behaviour of Multi-body Mechanisms", IEEE Transactions on Systems, Man and Cybernetics, Vol.SMC-7, No.12, 1977, pp.868-871.
21. Lozano-Pérez, T., Wesley, M., "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles", Communications of the ACM, Vol.22, No.10, 1979, pp.560-570.
22. Lozano-Pérez, T., "Spatial Planning: A Configuration Space Approach", IEEE Transactions on Computers, Vol.C-32, No.2, 1983, pp.108-120.
23. Maruyama, K., "A Procedure to Determine Intersections Between Polyhedral Objects", Int. Journal of Computer and Information Sciences, Vol.1, No.3, 1972, pp.255-265.
24. Milenkovic, V., Huang, B., "Kinematics of Major Robot Linkage", Proc. 13th Int. Symp. on Industrial Robots, Chicago, 1983.
25. Paul, R., Shimano, B., Mayer, G., "Kinematic Control Equations for Simple Manipulators", IEEE Transactions on Systems, Man, and Cybernetics, Vol.SMC-11, No.6, June 1981, pp.449-455.
26. Paul, R. Robot Manipulators: Mathematics, Programming and Control, MIT Press, 1981.
27. Requicha, A., "Representations of Solids: Theory, Methods and Systems", Computing Surveys, Vol.12, No.4, 1980, pp.437-465.

28. Sata, T., Fumihiko, K., Akio, A., "Robot Simulation System as a Task Programming Tool", Proc. 11th Int. Symp. on Industrial Robots, Tokyo, 1981.
29. Schwartz, J., "Finding the Minimum Distance Between Two Convex Polygons", Information Processing Letters, Vol.13, No.4,5, 1981, pp.168-170.
30. Sjolund, P., Donath, M., "Robot Task Planning: Programming Using Interactive Computer Graphics", Proc. 13th Int. Symp. on Industrial Robots, Chicago, 1983.
31. Tarvin, R., "Considerations for Off-line Programming a Heavy Duty Industrial Robot", Proc. 10th Int. Symp. on Industrial Robots, Milan, 1980.
31. Udupa, S., "Collision Detection and Avoidance in Computer Controlled Manipulators", Proc. 5th Int. Joint Conf. on Artificial Intelligence, Cambridge, 1977.
32. Wesley, M, Lozano-Pérez, T., Lieberman, L., Lavin, M., Grossman, D., "A Geometric Modelling System for Automated Mechanical Assembly", IBM Journal of Research and Development, Vol.24, No.1, 1980, pp.64-74.
33. Whitney, D., "Resolved Motion Rate Control of Manipulators and Human Prostheses", IEEE Transactions on Man-Machine Systems, Vol.MMS-10, No.2, 1969, pp.47-53.
34. Whitney, D., "The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators", ASME Journal of Dynamic Systems, Measurement and Control, Dec. 1972, pp.303-309.

APPENDIX I

SOFTWARE DOCUMENTATION

APPENDIX I.1

IMPLEMENTATION SPECIFIC SOFTWARE

Subroutines OPEN, SHUT

The method of accessing external files from a FORTRAN program varies from one FORTRAN implementation to another. DEC FORTRAN uses a FORTRAN OPEN subroutine call to open a file, using a DEC specific argument list. In AUTOP, all files are opened and closed by calls to the AUTOP subroutines OPEN and SHUT. The software can then be easily transported to another system with different file management procedures by rewriting the subroutines OPEN and SHUT.

APPENDIX I.2

AUTOP PROGRAM LISTING

Listing of AUTOP.FOR at 14:39:07 on AUG 9, 1984 for CCId=DAC0 Page 1

```

1      C      PROGRAM AUTOP
2      C
3      C      THIS MAIN PROGRAM IS FOR THE AUTOMATIC PROGRAMMING OF A
4      C      WELDING ROBOT.
5      C
6      C      INPUTS: NONE
7      C
8      C      OUTPUTS: NONE
9      C
10     C      FILES READ:      FILIST,CADLST,GRAPH
11     C
12     C      FILES WRITTEN:  NONE
13     C
14     C      SUBROUTINES CALLED:      CADFIL
15     C                                CHECK
16     C                                CONFIG
17     C                                DRWELD
18     C                                EXPERT
19     C                                EYELOC
20     C                                INIT
21     C                                OPEN
22     C                                RDFILE
23     C                                SCENE
24     C                                SHUT
25     C                                WELSET
26     C
27     C      'IGL' GRAPHICS SUBROUTINES:      CMCLOS
28     C                                          CMOPEN
29     C                                          GETPIK
30     C                                          GRSTOP
31     C                                          GRSTRT
32     C                                          VUP3D
33     C                                          VWPT3D
34     C                                          WIND3D
35     C
36     C
37     C*****
38     C
39     C      BYTE FILIST(20),CADLST(20),WPCLST(20),WKSTAT(20)
40     C      BYTE TABLE(20),STOVAR(20),GRAPH(20),OBJLST(20),ROBLST(20)
41     C      BYTE LOCSET(20),WPARAM(20),WPCDAT(20)
42     C      BYTE BEL(3),YES,NO,ANS,ANS2
43     C
44     C      DOUBLE PRECISION HOT(4,4),HOW(4,4),HTW(4,4)
45     C      REAL VERT(1000,3),FACES(100,4),H(4,4),EYEPOS(3),VRP(3),VAR(6)
46     C      REAL SHOT(4,4),SHOW(4,4),HOO(4,4),HGG(4,4),INHOO(4,4),INHGA(4,4)
47     C      REAL SHTW(4,4),DELTA
48     C      REAL WINSIZ(6),VUPO(3),VUP1(3)
49     C      REAL ALPHA(6),LEN(6),DIST(6),RMIN(6),RMAX(6)
50     C      INTEGER ICONF(6),IMAX
51     C
52     C      INTEGER IOBJ(10,2),IFACE(100,2),IFLAG(1000),IDIR(1)
53     C      INTEGER IPICK(100)
54     C
55     C      COMMON /DATA/VERT,FACES,IFACE,IOBJ,IFLAG
56     C      COMMON /VIEW/EYEPOS,VRP
57     C      COMMON /WINSIZ/WINSIZ
58     C      COMMON /FILE/FILIST

```

Listing of AUTOP.FOR at 14:39:07 on AUG 9, 1984 for CCId=DACO Page 2

```
59      COMMON /CADLST/CADLST
60      COMMON /WPCLST/WPCLST
61      COMMON /WKSTAT/WKSTAT
62      COMMON /TABLE/TABLE
63      COMMON /STOVAR/STOVAR
64      COMMON /GRAPH/GRAPH
65      COMMON /OBJLST/OBJLST
66      COMMON /ROBLST/ROBLST
67      COMMON /LOCSET/LOCSET
68      COMMON /MATRIX/HOT, HOW, SHOT, SHOW
69      COMMON /ROBOT/NRDOF, ALPHA, LEN, DIST, RMIN, RMAX, HOO, H6G, INHOO, INH6A
70      COMMON /CONFIG/ICONF
71      COMMON /TABDAT/HTW, SHTW
72      COMMON /TOLROT/IMAX, DELTA
73      COMMON /EXPERT/WPARAM, VOLT, WSIZE, FEED, GFR, IALPHA, IBETA, ISTOUT
74      COMMON /NSEAMS/NSEAMS
75      C
76      DATA BEL/3*7/, YES/89/, NO/78/
77      DATA VUPO/3*0/, VUP1/0., 0., 1./
78      C
79      C***** ENTER THE NAME OF THE FILE LIST 'FILIST'.
80      C
81      WRITE(5,10)
82      10      FORMAT(' ENTER THE NAME OF THE FILE LIST... ', $)
83      CALL RDFILE(5,1,FILIST,20,IEND)
84      C
85      C***** READ NAMES OF SYSTEM FILES FROM FILE 'FILIST'.
86      C
87      CALL OPEN(1,FILIST,4,1)
88      CALL RDFILE(1,1,CADLST,20,IEND)
89      CALL RDFILE(1,2,WPCLST,20,IEND)
90      CALL RDFILE(1,3,WKSTAT,20,IEND)
91      CALL RDFILE(1,4,TABLE,20,IEND)
92      CALL RDFILE(1,5,STOVAR,20,IEND)
93      CALL RDFILE(1,6,GRAPH,20,IEND)
94      CALL RDFILE(1,7,OBJLST,20,IEND)
95      CALL RDFILE(1,8,ROBLST,20,IEND)
96      CALL RDFILE(1,9,LOCSET,20,IEND)
97      CALL RDFILE(1,10,WPARAM,20,IEND)
98      CALL SHUT(1)
99      C
100     C***** GENERATE THE WORKPIECE SEAM FILES FROM THE SEAM CAD FILES
101     C
102     CALL SETUP
103     C
104     C
105     C***** OPEN CAD FILE LIST, 'CADLST'.
106     C
107     CALL OPEN(1,CADLST,4,1)
108     C
109     C***** SKIP THE FIRST RECORD AND READ THE NUMBER OF SEAMS, 'NSEAMS'.
110     C
111     READ(1,*)
112     READ(1,*) NSEAMS
113     CALL SHUT(1)
114     C
115     C***** OPEN GRAPHICS PARAMETERS FILE, 'GRAPH'.
116     C
```

Listing of AUTOP.FOR at 14:39:07 on AUG 9, 1984 for CCId=DAC0 Page 3

```
117          CALL OPEN(1,GRAPH,4,1)
118          READ(1,*)
119          C
120          C***** READ CURRENT VIEW REFERENCE POINT, 'VRP'.
121          C
122          READ(1,*) (VRP(I),I=1,3)
123          READ(1,*)
124          C
125          C***** READ CURRENT WINDOW PARAMETERS, 'WINSIZ'.
126          C
127          READ(1,*) (WINSIZ(I),I=1,6)
128          CALL SHUT(1)
129          C
130          C
131          C***** INITIALIZE THE ROBOT PARAMETERS
132          C
133          CALL INIT(NSEAMS)
134          C
135          C
136          C***** INITIALIZE WELDER SETUP
137          C
138          CALL WELSET(WPCLST,WKSTAT)
139          C
140          C
141          C***** INITIALIZE IGL GRAPHICS SYSTEM
142          C
143          CALL GRSTRT(4027,1)
144          CALL VUP3D(VUPO,VUP1)
145          C
146          C***** DEFINE 3D WINDOW AND VIEWPORT.
147          C
148          CALL WIND3D(WINSIZ(1),WINSIZ(2),WINSIZ(3),WINSIZ(4),WINSIZ(5),
149          * WINSIZ(6))
150          CALL VWPT3D(0.,100.,0.,100.,0.,100.)
151          C
152          C***** INITIALIZE THE WELDED SEAM COUNT
153          C
154          ICOUNT=1
155          C
156          C***** DEFINE EYEBALL LOCATION.
157          C
158          CALL CMCLOS
159          CALL EYELOC(VRP,EYEPOS)
160          CALL CMOPEN
161          C
162          C
163          C***** CONVERT EXTERNAL CAD OBJECT DESCRIPTION FILES INTO INTERNAL
164          C***** ARRAYS FOR FAST PROCESSING.
165          C
166          CALL CMCLOS
167          CALL CADFIL
168          CALL CMOPEN
169          C
170          C
171          C***** DRAW COMPLETE SCENE, INCLUDING SEAMS
172          C
173          CALL SCENE
174          C
```

Listing of AUTOP.FOR at 14:39:07 on AUG 9, 1984 for CCid=DACD Page 4

```
175      C
176      C** UNTIL (ALL SEAMS PICKED)
177      C
178      600          CONTINUE
179      C
180      C** UNTIL (NO MORE CHANGES)
181      C
182      70          CONTINUE
183      C
184          CALL CMCLOS
185          WRITE(5,161)
186      161      FORMAT($,' DO YOU WISH TO CHANGE THE EYEBALL POSITION? (Y/N) ')
187          READ(5,170) ANS
188      C
189          IF(ANS.NE.YES) GO TO 50
190          CALL EYELOC(VRP,EYEPOS)
191      50          CONTINUE
192      C
193          WRITE(5,162)
194      162      FORMAT($,' DO YOU WISH TO REPOSITION TABLE? (Y/N) ')
195          READ(5,170) ANS2
196      170      FORMAT(A1)
197          CALL CMOPEN
198      C
199      C** IF (EITHER ANSWER IS YES) TRANSFORM AND REDRAW THE SCENE.
200      C
201          IF(ANS.NE.YES.AND.ANS2.NE.YES) GO TO 60
202          CALL SCENE
203      C
204      C** IF (ICOUNT.GT.1) DRAW ALL PREVIOUSLY PICKED WELDS.
205      C
206          IF(ICOUNT.EQ.1) GO TO 80
207          DO 8 K=1,ICOUNT-1
208      C
209      C*****      FILL IN WELD. 'SHOW' IS A HOMOGENEOUS TRANSFORMATION
210      C*****      MATRIX RELATING THE WORKPIECE TO WORLD COORDINATES.
211      C*****      'IPICK' IS AN ARRAY OF PREVIOUSLY PICKED WELDS.
212      C
213          CALL DRWELD(SHOW,IPICK(K))
214      C
215      8          CONTINUE
216      C
217      C** END IF
218      C
219      80          CONTINUE
220      C
221          GO TO 70
222      C
223      C** END IF
224      C
225      C** END UNTIL
226      C
227      60          CONTINUE
228      C
229      C***** PICK THE NEXT SEAM TO BE WELDED
230      C
231      C
232      C** UNTIL (PICK SUCCESSFUL)
```

Listing of AUTOP.FOR at 14:39:07 on AUG 9, 1984 for CCid=DACD Page 5

```
233      C
234      611      CONTINUE
235      C
236              CALL CMCLOS
237              WRITE(5,260)
238      260      FORMAT(' PICK SEAM TO BE WELDED NEXT- HIT ANY KEY TO PICK')
239              CALL CMOPEN
240      C
241      C*****      PICK SEAM WITH CROSSHAIRS. GETPIK RETURNS THE SEAM ID,
242      C*****      'IDSEAM'.
243      C
244              CALL GETPIK(IDSEAM,IDPIK)
245      C
246      C**      IF (PICK UNSUCCESSFUL)
247      C
248              IF(IDSEAM.NE.O) GO TO 190
249      C
250              CALL CMCLOS
251              WRITE(5,191) BEL
252      191      FORMAT($,' PICK UNSUCCESSFUL-TRY AGAIN (HIT RETURN)',3A1)
253              CALL CMOPEN
254              GO TO 610
255      C
256      C**      END IF
257      C
258      190      CONTINUE
259      C
260      C
261      C*****      CHECK IF THE SEAM HAS ALREADY BEEN PICKED.
262      C
263              CALL CMCLOS
264      C
265      C*****      'IPICK' IS AN ARRAY CONTAINING ALL PICKED SEAMS.
266      C*****      'ICOUNT' IS THE NUMBER OF SEAMS ALREADY PICKED.
267      C*****      'IDSEAM' IS THE CURRENT SEAM ID
268      C*****      'IPREV' IS A STATUS FLAG. IF 'IDSEAM' ALREADY PICKED, IPREV=1
269      C
270              CALL CHECK(IPICK,ICOUNT,IDSEAM,IPREV)
271      C
272              CALL CMOPEN
273      C
274      C**      IF (SEAM PREVIOUSLY PICKED)
275      C
276              IF(IPREV.EQ.O) GO TO 350
277      C
278              CALL CMCLOS
279              WRITE(5,130) BEL
280      130      FORMAT($,' SEAM ALREADY PICKED- TRY AGAIN ',3A1)
281              CALL CMOPEN
282      C
283              GO TO 610
284      C
285      C**      ELSE (PICK SUCCESSFUL)
286      C
287      350      CONTINUE
288      C
289      C*****      ADD 'IDSEAM' TO ARRAY OF PICKED SEAMS, 'IPICK'
290      C
```

Listing of AUTOP.FOR at 14:39:07 on AUG 9, 1984 for CCid=DACO Page 6

```
291             IPICK(ICOUNT)=IDSEAM
292     C
293             GO TO 360
294     C
295     C** END IF
296     C
297     610         CONTINUE
298             GO TO 611
299     C
300     C** END UNTIL
301     C
302     360         CONTINUE
303     C
304             CALL CMCLOS
305             WRITE(5,160) IDSEAM
306     160         FORMAT($,' IDSEAM=',I2)
307     C
308     C***** FIND THE CORRESPONDING SEAM FILE 'WPCDAT'.
309     C
310             CALL OPEN(2,WPCLS,4,1)
311             CALL RDFILE(2,IDSEAM+2,WPCDAT,20,IEND)
312             CALL SHUT(2)
313     C
314     C***** ENTER DIRECTION IN WHICH SEAM IS TO BE WELDED.
315     C
316     290         CONTINUE
317             WRITE(5,140)
318     140         FORMAT($,' ENTER WELD DIRECTION- (0=NORMAL, 1=REVERSE) ')
319             READ(5,*) IDIR
320     C
321     C***** CALL THE EXPERT WELDER MODULE TO FIND THE TORCH ANGLES AND
322     C***** STICKOUT, AND THE WELDING PARAMETERS.
323     C
324             CALL EXPERT(WPCDAT)
325     C
326     C***** SEARCH FOR A FEASIBLE ROBOT SOLUTION. ISTAT: 0=OK,
327     C***** 1=ROBOT ENVELOPE EXCEEDED, 2=ROBOT JOINT LIMITS EXCEEDED
328     C
329             CALL CONFIG(WPCDAT,IDSEAM,IDIR,ISTAT)
330     C
331             CALL CMOPEN
332     C
333     C** IF (ROBOT JOINT LIMITS EXCEEDED) REPOSITION WORKPIECE AND RETRY.
334     C
335             IF(ISTAT.EQ.0) GO TO 285
336             IF(ISTAT.EQ.1) GO TO 286
337             WRITE(5,850)
338     850         FORMAT(' JOINT LIMITS EXCEEDED-TRY AGAIN')
339     C
340     C***** REMOVE SEAM FROM PICKED LIST
341     C
342             IPICK(ICOUNT)=0
343     C
344             GO TO 600
345     C
346     C** OR IF (ROBOT ENVELOPE EXCEEDED) REPOSITION WORKPIECE AND RETRY.
347     C
348     286         CONTINUE
```

Listing of AUTOP.FOR at 14:39:07 on AUG 9, 1984 for CCid=DACO Page 7

```
349          WRITE(5,851)
350      851          FORMAT(' ROBOT ENVELOPE EXCEEDED-REPOSITION')
351      C
352      C*****      REMOVE SEAM FROM PICKED LIST
353      C
354          IPICK(ICOUNT)=0
355      C
356          GO TO 600
357      285      CONTINUE
358      C
359      C**      END IF
360      C
361      C**      OR IF (IDIR NOT 0 OR 1) RE-ENTER IDIR
362      C
363          IF(ISTAT.NE.1) GO TO 295
364          GO TO 290
365      295      CONTINUE
366      C
367      C**      ELSE (PATH OK)
368      C**      END IF
369      C
370      C*****      FILL IN THE WELDED SEAM 'IDSEAM'. 'SHOW' IS A HOMOGENEOUS
371      C*****      TRANSFORMATION MATRIX RELATING THE WORKPIECE FRAME TO THE WORLD
372      C*****      COORDINATE FRAME.
373      C
374          CALL DRWELD(SHOW,IDSEAM)
375      C
376      C*****      INCREMENT THE WELDED SEAM COUNTER
377      C
378          ICOUNT=ICOUNT+1
379          IF(ICOUNT.LE.NSEAMS) GO TO 600
380      C
381      C**      END UNTIL
382      C
383      C
384          CALL GRSTOP
385      C
386          STOP
387          END
```

Listing of ARCLLEN.FOR at 14:39:10 on AUG 9, 1984 for CCId=DAC0 Page 1

```
1
2      SUBROUTINE ARCLLEN(R,DELS)
3      C
4      C      THIS SUBROUTINE CALCULATES ARC LENGTH S.
5      C
6      C      INPUTS:
7      C          R(3,3)=POSITION VECTORS AT POINTS I-1, I, I+1.
8      C      OUTPUT:
9      C          DELS=ARC LENGTH FROM POINT I TO POINT I+1.
10     C
11     C*****
12     C
13     C
14     REAL R(3,3),X,Y,Z,LASTX,LASTY,LASTZ,DELS
15     DOUBLE PRECISION DELX, DELY, DELZ
16     C
17     X=R(3,1)
18     Y=R(3,2)
19     Z=R(3,3)
20     LASTX=R(2,1)
21     LASTY=R(2,2)
22     LASTZ=R(2,3)
23     DELX=X-LASTX
24     DELY=Y-LASTY
25     DELZ=Z-LASTZ
26     DELS=DSQRT(DELX**2+DELY**2+DELZ**2)
27     RETURN
28     END
```

Listing of CADFIL.FOR at 14:39:12 on AUG 9, 1984 for CCid=DACO Page 1

```
1
2      SUBROUTINE CADFIL
3      C
4      C      THIS SUBROUTINE CONVERTS CAD DATA FILE INTO ARRAYS IN MAIN
5      C      MEMORY FOR FAST PROCESSING.
6      C
7      C      INPUTS: NONE
8      C
9      C      OUTPUTS: NONE
10     C
11     C      COMMON BLOCKS CHANGED: /DATA/
12     C
13     C      FILES READ:      OBJLST, FILES LISTED IN OBJLST
14     C
15     C      OBJLST- CONTAINS LIST OF SEPARATE OBJECT CAD FILES
16     C      IE. FILE1- WORKPIECE CAD FILE
17     C      FILE2- TABLE CAD FILE (MOVING PART)
18     C      FILE3- TABLE CAD FILE (FIXED PART)
19     C      FILE4- ROBOT FOREARM
20     C      FILE5- ROBOT HAND AND TOOL
21     C
22     C      DATA ARRAYS (COMMON):
23     C      VERT(1000,3)- VERTEX LIST Vx,Vy,Vz
24     C      FACES(100,4)- FACE PARAMETERS U=(Ux,Uy,Uz),P
25     C      WHERE U=SURFACE NORMAL
26     C      P=DISTANCE TO ORIGIN
27     C      IOBJ(10,2)- LOCATION OF FIRST FACE,NUMBER OF FACES
28     C      IN FACE INDEX 'IFACE'
29     C      IFACE(100,2)- LOCATION OF FIRST VERTEX,NUMBER OF
30     C      VERTICES IN VERTEX INDEX 'IVERT'
31     C
32     C      FOR EACH FACE, THE ROUTINE REPEATS THE FIRST VERTEX AT THE END OF
33     C      THE VERTEX LIST SINCE EVERY FACE IS A CLOSED POLYGON. THIS
34     C      SIMPLIFIES LATER PROCESSING.
35     C      IE. FOR A FACE WITH 3 VERTICES, V1,V2,V3, THE VERTEX LIST WOULD
36     C      CONTAIN V1, V2, V3, V1.
37     C
38     C      SUBROUTINES CALLED:      OPEN
39     C      RDFILE
40     C      SHUT
41     C
42     C*****
43     C
44     C      BYTE FILE(20),OBJLST(20)
45     C      REAL VERT(1000,3),FACES(100,4)
46     C      INTEGER IOBJ(10,2),IFACE(100,2)
47     C
48     C      COMMON /OBJLST/OBJLST
49     C      COMMON /DATA/VERT,FACES,IFACE,IOBJ
50     C
51     C***** INITIALIZE COUNTERS
52     C
53     C      NVERT=1
54     C      NFACE=1 -
55     C      NOBJ=1
56     C
57     C** REPEAT UNTIL THERE ARE NO MORE OBJECT FILES
58     C
```

Listing of CADFIL.FOR at 14:39:12 on AUG 9, 1984 for CCid=DACO Page 2

```

59      105      CONTINUE
60      C
61      C
62      C***** OPEN OBJECT LIST
63      C
64      CALL OPEN(1,OBJLST,4,1)
65      C
66      C***** FIND OBJECT FILE
67      C
68      CALL RDFILE(1,NOBJ,FILE,20,IEND)
69      CALL SHUT(1)
70      C
71      C***** IF THE END OF THE OBJECT LIST IS REACHED, GO TO END
72      C
73      IF (IEND.EQ.1.) GO TO 500
74      C
75      C***** OPEN THE CURRENT OBJECT FILE
76      C
77      CALL OPEN(1,FILE,4,1)
78      C
79      C***** STORE THE LOCATION OF THE FIRST FACE OF OBJECT 'NOBJ'
80      C
81      IOBJ(NOBJ,1)=NFACE
82      C
83      C***** INITIALIZE FACE COUNTER
84      C
85      NUFACE=0
86      C
87      C** REPEAT UNTIL THERE ARE NO MORE FACES
88      C
89      100      CONTINUE
90      C
91      C
92      C***** READ THE FACE ID,FACE PARAMETERS AND NUMBER OF VERTICES
93      C
94      READ(1,*) ID,(FACES(NFACE,J),J=1,4),NUVERT
95      C
96      C***** IF ID=0, THERE ARE NO MORE FACES
97      C
98      IF (ID.EQ.0) GO TO 150
99      C
100     C***** STORE THE LOCATION OF THE FIRST VERTEX, AND THE NUMBER OF
101     C***** VERTICES
102     C
103     IFACE(NFACE,1)=NVERT
104     IFACE(NFACE,2)=NUVERT
105     C
106     C***** READ AND STORE THE COORDINATES OF THE VERTICES.
107     C
108     DO 110 J=1,NUVERT
109         READ(1,*) ITEMP,(VERT(NVERT,K),K=1,3)
110         NVERT=NVERT+1
111     110     CONTINUE
112     C
113     C***** REPEAT THE FIRST VERTEX AT THE END OF THE LIST
114     C
115     DO 115 J=1,3
116         VERT(NVERT,J)=VERT(NVERT-NUVERT,J)
```

Listing of CADFIL.FOR at 14:39:12 on AUG 9, 1984 for CCId=DACO Page 3

```
117      115      CONTINUE
118              NVERT=NVERT+1
119      C
120      C***** INCREMENT THE FACE POINTER AND THE FACE COUNTER
121      C
122              NFACE=NFACE+1
123              NUFACE=NUFACE+1
124      C
125              GO TO 100
126      C
127      150      CONTINUE
128      C
129              CALL SHUT(1)
130      C
131      C***** STORE THE NUMBER OF FACES OF OBJECT 'NOBJ'
132      C
133              IOBJ(NOBJ,2)=NUFACE
134      C
135      C***** INCREMENT THE OBJECT COUNTER, AND REPEAT UNTIL THERE ARE NO MORE
136      C***** OBJECTS
137      C
138              NOBJ=NOBJ+1
139      C
140              GO TO 105
141      C
142      500      CONTINUE
143      C
144              RETURN
145              END
```

Listing of CALCS.FOR at 14:39:13 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE CALCS(CPHI,CTHE)
2      C
3      C      THIS SUBROUTINE CALCULATES THE ORIENTATION FACTORS 'CPHI',
4      C      'CTHE' AS FUNCTIONS OF THE TORCH ORIENTATION RELATIVE
5      C      TO THE GRAVITY VECTOR.
6      C
7      DOUBLE PRECISION HOT(4,4),HOW(4,4)
8      COMMON/COEF/TW(3),AW(3),T(3),A(3)
9      COMMON/MATRIX/HOT,HOW,SHOT(4,4),SHOW(4,4)
10     C
11     CONV=180./3.14159
12     C
13     C***** 'CONV' CONVERTS ANGLES IN RADIAN TO DEGREES
14     C
15     C***** ROTATE THE SEAM TANGENT AND NORMAL VECTORS INTO THE WORLD
16     C***** COORDINATE FRAME. TW(3), AW(3) ARE THE COMPONENTS OF TW,AW
17     C***** ALONG THE GRAVITY VECTOR.
18     C
19     CALL ROTATE(SHOW,T,TW)
20     CALL ROTATE(SHOW,A,AW)
21     C
22     C***** 'PHI' IS THE ANGLE BETWEEN +Z (UPWARD) AXIS AND THE TANGENT
23     C***** TO THE WELD SEAM
24     C
25     PHI=ACOS(TW(3)/SQRT(TW(1)**2+TW(2)**2+TW(3)**2))
26     C
27     C***** 'THETA' IS THE ANGLE BETWEEN +Z (UPWARD) AXIS AND THE NORMAL
28     C***** TO THE WELD SEAM
29     C
30     THETA=ACOS(AW(3)/SQRT(AW(1)**2+AW(2)**2+AW(3)**2))
31     PHI=PHI*CONV
32     C
33     C***** DETERMINE ORIENTATION FACTORS 'CPHI' AND 'CTHE' USING -
34     C***** THE ORIENTATION ANGLES
35     C
36     THETA=THETA*CONV
37     CPHI=5.889*(10.0**-4.)*PHI+0.95
38     IF(PHI.GT.90.0) CPHI=-5.89*(10.0**-4.)*PHI+1.05
39     CTHE=-5.556*(10.0**-4.)*THETA+1.
40     IF(THETA.GT.90.0) CTHE=-1.111*(10.0**-3.)*THETA+1
41     C
42     RETURN
43     END
```

Listing of CHECK.FOR at 14:39:13 on AUG 9, 1984 for CC1d=DAC0 Page 1

```

1
2      SUBROUTINE CHECK(IPICK,ICOUNT,IDSEAM,IPREV)
3      C
4      C      THIS SUBROUTINE CHECKS IF A SEGMENT HAS ALREADY BEEN
5      C      PICKED.
6      C
7      C      INPUTS: IPICK(100)- ARRAY OF PICKED SEGMENT ID INTEGERS
8      C
9      C      ICOUNT- TOTAL NUMBER OF SEGMENTS
10     C
11     C      IDSEAM- ID OF SEGMENT TO BE CHECKED
12     C
13     C      OUTPUTS: IPREV- 0 IF SEAM NOT PREVIOUSLY PICKED
14     C      1 OTHERWISE
15     C
16     C*****
17     C
18     C      INTEGER IPICK(100)
19     C
20     C      IPREV=1
21     C
22     C      DO 100 I=1,ICOUNT
23     C      IF(IPICK(I).EQ.IDSEAM) GO TO 200
24     C      100 CONTINUE
25     C
26     C      IPREV=0
27     C
28     C      200 CONTINUE
29     C
30     C      RETURN
31     C      END
```

Listing of CONFIG.FOR at 14:39:33 on AUG 9, 1984 for CCId=DACO Page 1

```

1      SUBROUTINE CONFIG(WPCDAT,IDSEAM,IDIR,ISTAT)
2      C
3      C      THIS SUBROUTINE TESTS ALL CONFIGURATIONS OF
4      C      THE ROBOT UNTIL A FEASIBLE SOLUTION IS FOUND.
5      C
6      C      INPUTS: WPCDAT- FILE NAME OF CURRENT SEAM TO BE SOLVED
7      C                  IDSEAM- ID OF CURRENT SEAM TO BE SOLVED
8      C                  IDIR-  DESIRED DIRECTION OF TRAVEL.
9      C
10     C      OUTPUTS: ISTAT- FEASIBILITY STATUS
11     C                  0=FEASIBLE TRAJECTORY FOUND
12     C                  1=ROBOT ENVELOPE EXCEEDED
13     C                  2=ROBOT JOINT LIMIT EXCEEDED
14     C
15     C      COMMON BLOCKS CHANGED:  /CONFIG/
16     C
17     C      VALUES OF ICONF(6):      (FOR PUMA 560 TYPE RQBOT)
18     C
19     C      0 0 0 0 0 0      L A NF      MODIFY CONFIGURATION IN THIS
20     C      0 0 0 1 0 0      L A F        DIRECTION ONLY
21     C      0 0 1 0 0 0      L B NF      :
22     C      0 0 1 1 0 0      L B F        :
23     C      1 1 1 0 0 0      R A NF      :
24     C      1 1 1 1 0 0      R A F        :
25     C      1 1 0 0 0 0      R B NF      V
26     C      1 1 0 1 0 0      R B F
27     C
28     C      WHERE:  R- RIGHT HANDED ARM CONFIGURATION
29     C              L- LEFT HANDED ARM CONFIG.
30     C
31     C              A- ELBOW ABOVE CONFIGURATION
32     C              B- ELBOW BELOW CONFIG.
33     C
34     C              NF- WRIST NOT FLIPPED
35     C              F- WRIST FLIPPED
36     C
37     C      SUBROUTINES CALLED:      WORPOS
38     C
39     C
40     C*****
41     C
42     C      BYTE WPCDAT(20)
43     C      REAL ALPHA(6),LEN(6),DIST(6),RMIN(6),RMAX(6),HOO(4,4),
44     C      * H6G(4,4),INHOO(4,4),INH6A(4,4)
45     C      INTEGER ICONF(6)
46     C
47     C      COMMON /ROBOT/NRDOF,ALPHA,LEN,DIST,RMIN,RMAX,HOO,H6G,INHOO,INH6A
48     C      COMMON /CONFIG/ICONF
49     C
50     C      I=2
51     C      J=2
52     C      K=2
53     C
54     C      100      CONTINUE
55     C
56     C***** TEST CURRENT CONFIGURATION FOR FEASIBILITY
57     C
58     C      CALL WORPOS(WPCDAT,IDSEAM,IDIR,ISTAT)

```

Listing of CONFIG.FOR at 14:39:33 on AUG 9, 1984 for CCid=DACO Page 2

```
59      C
60      C***** IF A FEASIBLE SOLUTION WAS FOUND, GO TO END
61      C
62          IF(ISTAT.EQ.O) GO TO 200
63      C
64      C***** ELSE TRY A DIFFERENT CONFIGURATION UNTIL ALL POSSIBLE
65      C***** CONFIGURATIONS HAVE BEEN TRIED
66      C
67      C***** REVERSE WRIST FLIP CONFIGURATION
68      C
69          IF(ICONF(4).EQ.O) GO TO 10
70          ICONF(4)=0
71          GO TO 20
72      10      CONTINUE
73          ICONF(4)=1
74      20      CONTINUE
75      C
76          I=I-1
77          IF(I.GT.O) GO TO 100
78      C
79          I=2
80      C
81      C***** REVERSE ELBOW ABOVE/BELOW CONFIGURATION
82      C
83          IF(ICONF(3).EQ.O) GO TO 11
84          ICONF(3)=0
85          GO TO 21
86      11      CONTINUE
87          ICONF(3)=1
88      21      CONTINUE
89      C
90          J=J-1
91          IF(J.GT.O) GO TO 100
92      C
93          J=2
94      C
95      C***** REVERSE LEFT/RIGHT ARM CONFIGURATION
96      C
97      300      CONTINUE
98          IF(ICONF(1).EQ.O) GO TO 12
99          ICONF(1)=0
100         ICONF(2)=0
101         GO TO 22
102     12      CONTINUE
103         ICONF(1)=1
104         ICONF(2)=1
105     22      CONTINUE
106     C
107         K=K-1
108         IF(K.GT.O) GO TO 100
109         IF(K.EQ.O) GO TO 300
110     C
111     200      CONTINUE
112     C
113     RETURN
114     END
```

Listing of DOTPRD.FOR at 14:39:35 on AUG 9, 1984 for CCid=DACO Page 1

```
1          SUBROUTINE DOTPRD(V1,V2,PROD)
2      C
3      C      THIS SUBROUTINE FINDS THE DOT PRODUCT OF TWO VECTORS,
4      C      V1(3),V2(3).
5      C
6      C      INPUTS: V1(3),V2(3)- TWO VECTORS
7      C
8      C      OUTPUTS: PROD- THE DOT PRODUCT PROD=V1.V2
9      C
10     C*****
11     REAL V1(3),V2(3)
12     C
13     PROD=V1(1)*V2(1)+V1(2)*V2(2)+V1(3)*V2(3)
14     C
15     RETURN
16     END
```

Listing of DRAXES.FOR at 14:39:35 on AUG 9, 1984 for CCid=DACD Page 1

```
1          SUBROUTINE DRAXES
2      C
3      C      THIS SUBROUTINE DRAWS THE WORLD COORDINATE AXES.
4      C
5      C*****
6      C
7          CALL MOVE3D(0.,0.,0.)
8          CALL DRAW3D(1000.,0.,0.)
9          CALL MOVE3D(0.,0.,0.)
10         CALL DRAW3D(0.,1000.,0.)
11         CALL MOVE3D(0.,0.,0.)
12         CALL DRAW3D(0.,0.,1000.)
13     C
14     RETURN
15     END
```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE DRPANL(H,NOBJ)
2      C
3      C      THIS SUBROUTINE DRAWS OBJECT 'NOBJ', TRANSFORMED BY
4      C      MATRIX 'H'. THE OBJECT IS REPRESENTED BY A SOLID
5      C      POLYHEDRON, DRAWN AS A SET OF SUPERIMPOSED FACE PANELS.
6      C
7      C      INPUTS: H(4,4)- HOMOGENEOUS TRANSFORMATION MATRIX
8      C                  DEFINING LOCATION OF OBJECT IN VIEW VOLUME.
9      C
10     C      NOBJ- IDENTITY OF OBJECT TO BE DRAWN
11     C
12     C      OUTPUTS: NONE
13     C
14     C      COMMON BLOCKS ACCESSED: /DATA/,/VIEW/
15     C
16     C      COMMON BLOCKS CHANGED: NONE
17     C
18     C      FILES READ OR WRITTEN: NONE
19     C
20     C      SUBROUTINES CALLED: DOTPRD
21     C                          LINDIS
22     C                          ROTAT
23     C                          TRANS
24     C
25     C      'IGL' SUBROUTINES: APPEAR
26     C                          DRAW3D
27     C                          MOVE3D
28     C                          PANL3D
29     C                          POLY3D
30     C                          REMOVE
31     C
32     C*****
33     C
34     REAL VERT(1000,3),FACES(100,4),V(3),V1(3),H(4,4),V1ST(3)
35     REAL VX(4),VY(4),VZ(4),P1(3),P2(3),Q1(3),Q2(3),UFACE1(3)
36     REAL A1(3),A2(3),B1(3),B2(3)
37     REAL EYEPOS(3),VRP(3),USIGHT(3),UFACE(3),WEIGHT(20),PQ(3)
38     REAL VX3(3),VY3(3),VZ3(3),VX4(4),VY4(4),VZ4(4)
39     REAL VX5(5),VY5(5),VZ5(5),VX6(6),VY6(6),VZ6(6)
40     REAL VX7(7),VY7(7),VZ7(7),VX8(8),VY8(8),VZ8(8)
41     REAL VX9(9),VY9(9),VZ9(9),VX10(10),VY10(10),VZ10(10)
42     INTEGER IOBJ(10,2),IFACE(100,2)
43     INTEGER IFLIST(20),IORDER(20)
44     C
45     COMMON /DATA/VERT,FACES,IFACE,IOBJ
46     COMMON /VIEW/EYEPOS,VRP
47     C
48     C
49     C***** FIND STARTING LOCATION OF FIRST FACE, 'IFASTA', AND THE NUMBER
50     C***** OF FACES, 'NFACES', FROM THE OBJECT PARAMETER ARRAY 'IOBJ'.
51     C
52     IFASTA=IOBJ(NOBJ,1)
53     NFACES=IOBJ(NOBJ,2)
54     C
55     C***** CALCULATE THE LINE OF SIGHT VECTOR 'USIGHT', FROM THE GIVEN
56     C***** EYE POSITION.
57     C
58     DO 11 I=1,3

```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCId=DACO Page 2

```
59          USIGHT(I)=-EYEPOS(I)
60      11      CONTINUE
61      C
62      C***** THIS SECTION CALCULATES WHICH FACES WILL BE VISIBLE FROM THE
63      C***** GIVEN LINE OF SIGHT ('USIGHT'). THE DOT PRODUCTS OF 'USIGHT'
64      C***** AND THE FACE SURFACE NORMALS ARE TAKEN. A NEGATIVE RESULT
65      C***** INDICATES VISIBILITY.
66      C
67      C***** INITIALIZE THE VISIBLE FACE COUNTER
68      C
69      NSHOW=0
70      C
71      C***** DO FOR EACH FACE
72      C
73      DO 200 I=1,NFACES
74      C
75      C***** GET SURFACE NORMALS 'UFACE' FROM THE FACE PARAMETER ARRAY
76      C***** 'FACES'.
77      C
78          DO 201 J=1,3
79          UFACE(J)=FACES(IFASTA+I-1,J)
80      201      CONTINUE
81      C
82      C***** ROTATE 'UFACE' BY TRANSFORMATION 'H' TO GET 'UFACE1' IN WORLD
83      C***** COORDINATES
84      C
85          CALL ROTAT(H,UFACE,UFACE1)
86      C
87      C***** TEST FOR 'VISIBILITY'
88      C
89          CALL DOTPRD(UFACE1,USIGHT,UDOT)
90      C
91          IF(UDOT.GE.-.0001) GO TO 500
92      C
93      C***** IF THE FACE IS 'VISIBLE', INCREMENT THE VISIBLE FACE COUNTER
94      C
95          NSHOW=NSHOW+1
96      C
97      C***** STORE VISIBLE FACE POINTERS IN ARRAY 'IFLIST'
98      C
99          IFLIST(NSHOW)=IFASTA+I-1
100     C
101     500      CONTINUE
102     200      CONTINUE
103     C
104     C
105     C***** THIS SECTION CALCULATES THE ORDER OF FACE GENERATION SO THAT
106     C***** CLOSER FACES ARE SUPERIMPOSED ON, OR COVER, MORE DISTANT FACES.
107     C***** WEIGHTS ARE ASSIGNED TO EACH FACE, THE HIGHEST BEING THE
108     C***** CLOSEST. THE ALGORITHM IS AS FOLLOWS: GIVEN A PAIR OF EDGES,
109     C***** A WEIGHT OF ONE IS ASSIGNED IF THE SECOND EDGE IS BEHIND THE
110     C***** FIRST EDGE AS VIEWED ALONG THE LINE OF SIGHT. THIS IS TESTED
111     C***** FOR EACH EDGE OF A FACE AGAINST ALL EDGES IN THE OBJECT, AND
112     C***** THE SUMMED WEIGHT DIVIDED BY THE NUMBER OF EDGES ON THE FACE
113     C***** GIVES THE FINAL WEIGHT.
114     C
115     C***** DO FOR EACH VISIBLE FACE
116     C
```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCid=DAC0 Page 3

```

117          DO 202 I=1,NSHOW
118      C
119      C***** INITIALIZE FACE WEIGHT
120      C
121          WEIGHT(I)=0.
122      C
123      C***** FIND LOCATION OF FIRST VERTEX 'NSTART' AND NUMBER OF VERTICES
124      C***** 'NVERTS' FROM FACE PARAMETER ARRAY 'IFACE'.
125      C
126          NSTART=IFACE(IFLIST(I),1)
127          NVERTS=IFACE(IFLIST(I),2)
128      C
129      C***** DO FOR EACH EDGE BOUNDING FACE
130      C
131          DO 203 J=1,NVERTS
132      C
133      C***** FIND ENDPOINTS A1,A2 OF FACE EDGE.
134      C
135          DO 204 K=1,3
136              A1(K)=VERT(NSTART+J-1,K)
137              A2(K)=VERT(NSTART+J,K)
138              IF(J.EQ.NVERTS) A2(K)=VERT(NSTART,K)
139      204      CONTINUE
140      C
141      C***** TRANSFORM INTO WORLD COORDS
142      C
143          CALL TRANS(H,A1,P1)
144          CALL TRANS(H,A2,P2)
145      C
146      C***** DO FOR ALL VISIBLE FACES
147      C
148          DO 205 L=1,NSHOW
149      C
150      C***** FIND SECOND FACE
151      C
152          ISTART=IFACE(IFLIST(L),1)
153          IVERTS=IFACE(IFLIST(L),2)
154      C
155          DO 206 M=1,IVERTS
156      C
157      C***** FIND ENDPOINTS B1,B2 OF SECOND FACE EDGES
158      C
159          DO 301 N=1,3
160              B1(N)=VERT(ISTART+M-1,N)
161              B2(N)=VERT(ISTART+M,N)
162              IF(M.EQ.IVERTS) B2(N)=VERT(ISTART,N)
163      301      CONTINUE
164      C
165      C***** TRANSFORM SECOND EDGE INTO WORLD COORDS.
166      C
167          CALL TRANS(H,B1,Q1)
168          CALL TRANS(H,B2,Q2)
169      C
170      C***** TEST IF SECOND EDGE IS BEHIND FIRST EDGE.
171      C***** IF TRUE, ISIGN=1.
172      C
173          CALL LINDIS(P1,P2,Q1,Q2,USIGHT,ISIGN)
174      C

```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCid=DAC0 Page 4

```
175             IF(ISIGN.EQ.0) GO TO 600
176         C
177         C***** IF SECOND EDGE IS BEHIND FIRST EDGE, INCREMENT FACE WEIGHT
178         C
179             WEIGHT(I)=WEIGHT(I)+1.
180         C
181         600             CONTINUE
182         C
183         206             CONTINUE
184         C
185         205             CONTINUE
186         C
187         203             CONTINUE
188         C
189         C
190         C***** CALCULATE AVERAGED WEIGHT
191         C
192             WEIGHT(I)=WEIGHT(I)/FLOAT(NVERTS)
193         C
194         202             CONTINUE
195         C
196         C***** THIS SECTION GENERATES A FACE LIST IN ORDER OF INCREASING FACE
197         C***** WEIGHT.
198         C
199             DO 210 I=1,NSHOW
200                 TEST=10000.
201                 DO 211 J=1,NSHOW
202                     IF(WEIGHT(J).GE.TEST) GO TO 700
203                     TEST=WEIGHT(J)
204                     JSAVE=J
205             700             CONTINUE
206             211             CONTINUE
207                 WEIGHT(JSAVE)=10000.
208                 IORDER(I)=JSAVE
209             210             CONTINUE
210         C
211         C
212         C***** THIS SECTION PERFORMS THE GRAPHIC OUTPUT
213         C
214         C***** DO FOR EACH VISIBLE FACE
215         C
216             DO 220 I=1,NSHOW
217         C
218         C***** FIND THE FURTHEST FACE STARTING POINT 'NSTART', NO. OF VERTICES
219         C***** 'NVERTS'.
220         C
221             IND=IORDER(I)
222             NSTART=IFACE(IFLIST(IND),1)
223             NVERTS=IFACE(IFLIST(IND),2)
224         C
225         C***** DRAW THE FACE ON THE SCREEN
226         C***** GRAPHICS ROUTINE DEPENDS ON NUMBER OF VERTICES ON FACE.
227         C***** 3,4,5,6,7,8,9,10 CORNERED POLYGONS ARE ALLOWED.
228         C
229             GO TO (803,804,805,806,807,808,809,810), NVERTS-2
230         C
231         803             CONTINUE
232             DO 333 J=1,NVERTS
```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCid=DAC0 Page 5

```
233      C
234      C***** FIND FACE VERTICES FROM VERTEX LIST 'VERT'
235      C
236          DO 303 K=1,3
237              V(K)=VERT(NSTART+J-1,K)
238      303      CONTINUE
239      C
240      C***** TRANSFORM VERTICES INTO WORLD COORDINATES.
241      C
242          CALL TRANS(H,V,V1)
243          VX3(J)=V1(1)
244          IF(VX3(J).LT..01) VX3(J)=0.
245          VY3(J)=V1(2)
246          IF(VY3(J).LT..01) VY3(J)=0.
247          VZ3(J)=V1(3)
248          IF(VZ3(J).LT..01) VZ3(J)=0.
249      333      CONTINUE
250      C
251      C***** DRAW FACE AS A FILLED PANEL SURROUNDED BY A DARK BORDER.
252      C
253          CALL PANL3D(3,VX3,VY3,VZ3)
254          CALL MOVE3D(VX3(1),VY3(1),VZ3(1))
255          CALL REMOVE
256          CALL POLY3D(3,VX3,VY3,VZ3)
257          CALL DRAW3D(VX3(1),VY3(1),VZ3(1))
258          CALL APPEAR
259          GO TO 900
260      C
261      804      CONTINUE
262          DO 344 J=1,NVERTS
263              DO 304 K=1,3
264                  V(K)=VERT(NSTART+J-1,K)
265      304      CONTINUE
266          CALL TRANS(H,V,V1)
267          VX4(J)=V1(1)
268          IF(ABS(VX4(J)).LT..01) VX4(J)=0.
269          VY4(J)=V1(2)
270          IF(ABS(VY4(J)).LT..01) VY4(J)=0.
271          VZ4(J)=V1(3)
272          IF(ABS(VZ4(J)).LT..01) VZ4(J)=0.
273      C
274      344      CONTINUE
275          CALL PANL3D(4,VX4,VY4,VZ4)
276          CALL MOVE3D(VX4(1),VY4(1),VZ4(1))
277          CALL REMOVE
278          CALL POLY3D(4,VX4,VY4,VZ4)
279          CALL DRAW3D(VX4(1),VY4(1),VZ4(1))
280          CALL APPEAR
281          GO TO 900
282      C
283      805      CONTINUE
284          DO 355 J=1,NVERTS
285              DO 305 K=1,3
286                  V(K)=VERT(NSTART+J-1,K)
287      305      CONTINUE
288          CALL TRANS(H,V,V1)
289          VX5(J)=V1(1)
290          IF(VX5(J).LT..01) VX5(J)=0.
```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCid=DACO Page 6

```

291          VY5(J)=V1(2)
292          IF(VY5(J).LT..01) VY5(J)=0.
293          VZ5(J)=V1(3)
294          IF(VZ5(J).LT..01) VZ5(J)=0.
295      355      CONTINUE
296          CALL PANL3D(5,VX5,VY5,VZ5)
297          CALL MOVE3D(VX5(1),VY5(1),VZ5(1))
298          CALL REMOVE
299          CALL POLY3D(5,VX5,VY5,VZ5)
300          CALL DRAW3D(VX5(1),VY5(1),VZ5(1))
301          CALL APPEAR
302          GO TO 900
303      C
304      806      CONTINUE
305          DO 366 J=1,NVERTS
306          DO 306 K=1,3
307              V(K)=VERT(NSTART+J-1,K)
308      306      CONTINUE
309          CALL TRANS(H,V,V1)
310          VX6(J)=V1(1)
311          IF(VX6(J).LT..01) VX6(J)=0.
312          VY6(J)=V1(2)
313          IF(VY6(J).LT..01) VY6(J)=0.
314          VZ6(J)=V1(3)
315          IF(VZ6(J).LT..01) VZ6(J)=0.
316      366      CONTINUE
317          CALL PANL3D(6,VX6,VY6,VZ6)
318          CALL MOVE3D(VX6(1),VY6(1),VZ6(1))
319          CALL REMOVE
320          CALL POLY3D(6,VX6,VY6,VZ6)
321          CALL DRAW3D(VX6(1),VY6(1),VZ6(1))
322          CALL APPEAR
323          GO TO 900
324      807      CONTINUE
325          DO 377 J=1,NVERTS
326          DO 307 K=1,3
327              V(K)=VERT(NSTART+J-1,K)
328      307      CONTINUE
329          CALL TRANS(H,V,V1)
330          VX7(J)=V1(1)
331          IF(ABS(VX7(J)).LT..01) VX7(J)=0.
332          VY7(J)=V1(2)
333          IF(ABS(VY7(J)).LT..01) VY7(J)=0.
334          VZ7(J)=V1(3)
335          IF(ABS(VZ7(J)).LT..01) VZ7(J)=0.
336      377      CONTINUE
337          CALL PANL3D(7,VX7,VY7,VZ7)
338          CALL MOVE3D(VX7(1),VY7(1),VZ7(1))
339          CALL REMOVE
340          CALL POLY3D(7,VX7,VY7,VZ7)
341          CALL DRAW3D(VX7(1),VY7(1),VZ7(1))
342          CALL APPEAR
343          GO TO 900
344      808      CONTINUE
345          DO 388 J=1,NVERTS
346          DO 308 K=1,3
347              V(K)=VERT(NSTART+J-1,K)
348      308      CONTINUE
```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCid=DAC0 Page 7

```
349      CALL TRANS(H,V,V1)
350      VX8(J)=V1(1)
351      IF(ABS(VX8(J)).LT..01) VX8(J)=0.
352      VY8(J)=V1(2)
353      IF(ABS(VY8(J)).LT..01) VY8(J)=0.
354      VZ8(J)=V1(3)
355      IF(ABS(VZ8(J)).LT..01) VZ8(J)=0.
356      388      CONTINUE
357      CALL PANL3D(8,VX8,VY8,VZ8)
358      CALL MOVE3D(VX8(1),VY8(1),VZ8(1))
359      CALL REMOVE
360      CALL POLY3D(8,VX8,VY8,VZ8)
361      CALL DRAW3D(VX8(1),VY8(1),VZ8(1))
362      CALL APPEAR
363      GO TO 900
364      809      CONTINUE
365      DO 399 J=1,NVERTS
366      DO 309 K=1,3
367      V(K)=VERT(NSTART+J-1,K)
368      309      CONTINUE
369      CALL TRANS(H,V,V1)
370      VX9(J)=V1(1)
371      IF(ABS(VX9(J)).LT..01) VX9(J)=0.
372      VY9(J)=V1(2)
373      IF(ABS(VY9(J)).LT..01) VY9(J)=0.
374      VZ9(J)=V1(3)
375      IF(ABS(VZ9(J)).LT..01) VZ9(J)=0.
376      399      CONTINUE
377      CALL PANL3D(9,VX9,VY9,VZ9)
378      CALL MOVE3D(VX9(1),VY9(1),VZ9(1))
379      CALL REMOVE
380      CALL POLY3D(9,VX9,VY9,VZ9)
381      CALL DRAW3D(VX9(1),VY9(1),VZ9(1))
382      CALL APPEAR
383      GO TO 900
384      810      CONTINUE
385      DO 311 J=1,NVERTS
386      DO 310 K=1,3
387      V(K)=VERT(NSTART+J-1,K)
388      310      CONTINUE
389      CALL TRANS(H,V,V1)
390      VX10(J)=V1(1)
391      IF(ABS(VX10(J)).LT..01) VX10(J)=0.
392      VY10(J)=V1(2)
393      IF(ABS(VY10(J)).LT..01) VY10(J)=0.
394      VZ10(J)=V1(3)
395      IF(ABS(VZ10(J)).LT..01) VZ10(J)=0.
396      311      CONTINUE
397      CALL PANL3D(10,VX10,VY10,VZ10)
398      CALL MOVE3D(VX10(1),VY10(1),VZ10(1))
399      CALL REMOVE
400      CALL POLY3D(10,VX10,VY10,VZ10)
401      CALL DRAW3D(VX10(1),VY10(1),VZ10(1))
402      CALL APPEAR
403      GO TO 900
404      C
405      900      CONTINUE
406      220      CONTINUE
```

Listing of DRPANL.FOR at 14:39:35 on AUG 9, 1984 for CCId=DACO Page 8

```
407      C
408      RETURN
409      END
```

Listing of DRSEAM.FOR at 14:39:57 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE DRSEAM(HOW)
2      C
3      C      THIS SUBROUTINE DRAWS THE SEAMS
4      C
5      C      INPUTS: HOW(4,4)- MATRIX SPECIFYING WORKPIECE POSITION RELATIVE
6      C                  TO WORLD COORDINATES.
7      C
8      C      OUTPUTS: NONE
9      C
10     C      INPUT FILES: WPCLST
11     C
12     C      OUTPUT FILES: NONE
13     C
14     C      COMMON BLOCKS ACCESSED: /WPCLST/
15     C
16     C      COMMON BLOCKS ALTERED: NONE
17     C
18     C      SUBROUTINES CALLED: NORM
19     C
20     C
21     C
22     C
23     C
24     C
25     C
26     C
27     C      IGL SUBROUTINES:
28     C
29     C
30     C
31     C
32     C
33     C
34     C
35     C
36     C*****
37     C
38     C
39     C      BYTE WPCDAT(20)
40     C      BYTE WPCLST(20)
41     C      REAL WINSIZ(6)
42     C      REAL R(3),HOW(4,4),RW(3),A(3),AW(3),AUW(3)
43     C      REAL T(3),B(3),BW(3),BUW(3),RLW(3),TANW(3),UTANW(3)
44     C
45     C      COMMON /WPCLST/WPCLST
46     C      COMMON /WINSIZ/WINSIZ
47     C
48     C***** INITIALIZE THE TICK SPACING PARAMETER 'TICSPA', BASED ON THE
49     C***** VIEW WINDOW SIZE
50     C
51     C      TICSPA=(WINSIZ(2)-WINSIZ(1))/100.
52     C
53     C***** OPEN THE LIST OF SEAM DATA FILES, 'WPCLST'.
54     C
55     C      CALL OPEN(1,WPCLST,4,1)
56     C
57     C***** READ THE NUMBER OF SEAMS, 'NSEAMS'.
58     C
```

Listing of DRSEAM.FOR at 14:39:57 on AUG 9, 1984 for CCId=DACO Page 2

```
59      READ(1,*)
60      READ(1,*) NSEAMS
61      C
62      C***** SET THE GRAPHIC ATTRIBUTES: DETECTABILITY=TRUE, REMOVE=TRUE
63      C
64      CALL SETDET(-2,.TRUE.)
65      CALL REMOVE
66      C
67      C***** REPEAT FOR ALL SEAMS
68      C
69      DO 100 I=1,NSEAMS
70      C
71      C***** CREATE A SEGMENT I
72      C
73      CALL OPNSEG(I)
74      C
75      C***** FIND AND OPEN THE SEAM DATA FILE FOR SEAM I.
76      C
77      CALL RDFILE(1,I+2,WPCDAT,20,IEND)
78      CALL OPEN(2,WPCDAT,1,1)
79      C
80      C***** SKIP THE FIRST RECORD
81      C
82      READ(2)
83      C
84      C***** READ SEAM DATA FILE I
85      C
86      READ(2,END=550) J,S,(R(K),K=1,3),(T(L),L=1,3),(A(M),M=1,3),
87      * (B(N),N=1,3)
88      C
89      C***** TRANSFORM INTO WORLD COORDINATES
90      C
91      CALL TRANS(HOW,R,RW)
92      C
93      C***** MOVE TO FIRST POINT
94      C
95      CALL MOVE3D(RW(1),RW(2),RW(3))
96      C
97      C***** INITIALIZE SIGN
98      C
99      SIGN=1.
100     C
101     C***** DRAW A CROSS AT SEAM START
102     C
103     CALL ROTAT(HOW,A,AW)
104     CALL UNIT(AW,AUW)
105     CALL ROTAT(HOW,B,BW)
106     CALL UNIT(BW,BUW)
107     CALL ROTAT(HOW,T,TANW)
108     CALL UNIT(TANW,UTANW)
109     DO 22 K=1,3
110         RLW(K)=RW(K)+SIGN*2.*TICSPA*BUW(K)
111     22 CONTINUE
112     CALL MOVE3D(RLW(1),RLW(2),RLW(3))
113     DO 33 K=1,3
114         RLW(K)=RW(K)-SIGN*2.*TICSPA*BUW(K)
115     33 CONTINUE
116     CALL DRAW3D(RLW(1),RLW(2),RLW(3))
```

Listing of DRSEAM.FOR at 14:39:57 on AUG 9, 1984 for CCid=DACO Page 3

```

117          DO 44 K=1,3
118             RLW(K)=RW(K)+SIGN*2.*TICSPA*AUX(K)
119          44      CONTINUE
120             CALL MOVE3D(RLW(1),RLW(2),RLW(3))
121             DO 55 K=1,3
122                RLW(K)=RW(K)-SIGN*2.*TICSPA*AUX(K)
123          55      CONTINUE
124             CALL DRAW3D(RLW(1),RLW(2),RLW(3))
125          C
126          C*****      INITIALIZE COUNTERS
127          C
128             SLAST=0.
129             ICOUNT=1
130          C
131          C*****      REPEAT FOR ALL POINTS ALONG SEAM
132          C
133          150      CONTINUE
134          C
135          C*****      READ SEAM DATA FOR CURRENT POINT
136          C
137             READ(2,END=550) INDEX,S,(R(K),K=1,3),(T(L),L=1,3),
138             *      (A(M),M=1,3),(B(N),N=1,3)
139          C
140             IF((S-SLAST).LT.TICSPA) GO TO 500
141          C
142             SLAST=S
143          C
144          C*****      TRANSFORM 'R','B','T' INTO WORLD COORDS 'RW','BW','UTANW'
145          C*****      AND NORMALIZE
146          C
147             CALL TRANS(HOW,R,RW)
148             CALL ROTAT(HOW,B,BW)
149             CALL UNIT(BW,BUW)
150             CALL ROTAT(HOW,T,TANW)
151             CALL UNIT(TANW,UTANW)
152          C
153          C*****      CHANGE SIGN
154          C
155             SIGN=-SIGN
156          C
157          C*****      DRAW HERRINGBONE TICK PATTERN, POINTING IN DIRECTION OF
158          C*****      SEAM
159          C
160             DO 2 K=1,3
161                RLW(K)=RW(K)+SIGN*TICSPA*BUW(K)
162          2      CONTINUE
163             CALL MOVE3D(RLW(1),RLW(2),RLW(3))
164             DO 8 K=1,3
165                RLW(K)=RW(K)+TICSPA*UTANW(K)
166          8      CONTINUE
167             CALL DRAW3D(RLW(1),RLW(2),RLW(3))
168             DO 5 K=1,3
169                RLW(K)=RW(K)-SIGN*TICSPA*BUW(K)
170          5      CONTINUE
171             CALL DRAW3D(RLW(1),RLW(2),RLW(3))
172          C
173          500      CONTINUE
174          C
```

Listing of DRSEAM.FOR at 14:39:57 on AUG 9, 1984 for CCid=DAC0 Page 4

175		GO TO 150
176	C	
177	550	CONTINUE
178	C	
179		CALL CLOSEG
180		CALL SHUT(2)
181	100	CONTINUE
182	C	
183		CALL SHUT(1)
184	C	
185		CALL APPEAR
186	C	
187		RETURN
188		END

Listing of DRWELD.FOR at 14:39:59 on AUG 9, 1984 for CCId=DACO Page 1

```

1      SUBROUTINE DRWELD(HOW,IDSEAM)
2      C
3      C      THIS SUBROUTINE FILLS IN THE WELDED SEAMS GRAPHICALLY
4      C
5      C      INPUTS: HOW(4,4)- MATRIX SPECIFYING POSITION OF WORKPIECE
6      C                  RELATIVE TO THE WORLD COORDINATE FRAME.
7      C
8      C                  IDSEAM- IDENTIFIES THE SEAM NUMBER WHICH HAS BEEN
9      C                  WELDED (TO BE GRAPHICALLY FILLED IN).
10     C
11     C      OUTPUTS: NONE
12     C
13     C      INPUT FILES: WPCLST- CONTAINS A LIST OF SEAM DATA FILES
14     C
15     C                  WPCDAT- DATA FOR A GIVEN SEAM
16     C
17     C      OUTPUT FILES: NONE
18     C
19     C      COMMON BLOCKS ACCESSED: /IOFILE/
20     C
21     C      COMMON BLOCKS ALTERED: NONE
22     C
23     C      SUBROUTINES CALLED:      NORM
24     C                               OPEN
25     C                               RDFILE
26     C                               SHUT
27     C                               TRANS
28     C                               UNIT
29     C                               XPROD
30     C
31     C      IGL SUBROUTINES:      APPEAR
32     C                               CLOSEG
33     C                               DRAW3D
34     C                               MOVE3D
35     C                               OPNSEG
36     C                               REMOVE
37     C                               SETDET
38     C                               SETHIL
39     C
40     C*****
41     C
42     C      BYTE WPCDAT(20)
43     C      BYTE WPCLST(20)
44     C      REAL WINSIZ(6)
45     C      REAL R(3),HOW(4,4),RW(3),N1(3,3),N2(3,3),A(3),AW(3),RL(3)
46     C      REAL TAN(3),B(3),BW(3),BUW(3),T(3)
47     C
48     C      COMMON /WPCLST/WPCLST
49     C      COMMON /WINSIZ/WINSIZ
50     C
51     C***** CALCULATE TICK SPACING PARAMETER 'TICSPA',BASED ON GRAPHIC
52     C***** VIEW WINDOW SIZE
53     C
54     C      TICSPA=(WINSIZ(2)-WINSIZ(1))/100.
55     C
56     C***** OPEN SEAM DATA FILE LIST, 'WPCLST'
57     C
58     C      CALL OPEN(1,WPCLST,4,1)

```

Listing of DRWELD.FOR at 14:39:59 on AUG 9, 1984 for CCid=DACO Page 2

```
59      C
60          READ(1,*)
61          READ(1,*) NSEAMS
62      C
63      C***** SET SEGMENT DETECTABILITY OFF, HILIGHTING ON, 'REMOVE' ATTRIBUTE
64      C***** TRUE
65      C
66          CALL SETDET(-2,.FALSE.)
67          CALL SETHIL(-2,.TRUE.)
68          CALL REMOVE
69      C
70      C***** DETERMINE SEGMENT ID AND OPEN SEGMENT
71      C
72          NSEG=NSEAMS+IDSEAM
73      C
74          CALL OPNSEG(NSEG)
75      C
76      C***** FIND AND OPEN SEAM DATA FILE
77      C
78          CALL RDFILE(1,IDSEAM+2,WPCDAT,20,IEND)
79          CALL OPEN(2,WPCDAT,1,1)
80      C
81      C***** SKIP FIRST RECORD
82      C
83          READ(2)
84      C
85      C***** READ FIRST POINT DATA TO FIND SEAM STARTING POINT.
86      C
87          READ(2,END=550) J,S,(R(K),K=1,3),(T(L),L=1,3),(A(M),M=1,3),
88          * (B(N),N=1,3)
89      C
90      C***** TRANSFORM INTO WORLD COORDINATES
91      C
92          CALL TRANS(HOW,R,RW)
93      C
94      C***** MOVE TO STARTING POINT
95      C
96          CALL MOVE3D(RW(1),RW(2),RW(3))
97      C
98          SIGN=-1.
99      C
100     C***** REPEAT FOR ALL POINTS ALONG THE SEAM
101     C
102     150    CONTINUE
103     C
104     C***** READ CURRENT POINT
105     C
106         READ(2,END=550) J,S,(R(K),K=1,3),(T(L),L=1,3),(A(M),M=1,3),
107         * (B(N),N=1,3)
108     C
109     C***** TRANSFORM INTO WORLD COORDINATES
110     C
111         CALL TRANS(HOW,R,RW)
112     C
113     C***** ROTATE TRANSVERSE VECTOR 'B' INTO WORLD COORDINATES, 'BW'
114     C
115         CALL ROTAT(HOW,B,BW)
116     C
```

Listing of DRWELD.FOR at 14:39:59 on AUG 9, 1984 for CCid=DACO Page 3

```
117 C***** NORMALIZE 'BW' TO GET 'BUW'
118 C
119 CALL UNIT(BW,BUW)
120 C
121 C***** DRAW A ZIG-ZAG PATTERN OVER THE SEAM TO REPRESENT A FILLED
122 C***** WELD
123 C
124 SIGN=-SIGN
125 C
126 DO 2 K=1,3
127 RW(K)=RW(K)+SIGN*TICSPA*BUW(K)
128 2 CONTINUE
129 C
130 CALL MOVE3D(RW(1),RW(2),RW(3))
131 C
132 DO 6 K=1,3
133 RW(K)=RW(K)-SIGN*2.*TICSPA*BUW(K)
134 6 CONTINUE
135 C
136 CALL DRAW3D(RW(1),RW(2),RW(3))
137 C
138 C***** THEN GO TO NEXT WELDED SEAM
139 C
140 GO TO 150
141 C
142 550 CONTINUE
143
144 CALL CLOSEG
145 CALL SHUT(2)
146 C
147 CALL SHUT(1)
148 C
149 CALL APPEAR
150 C
151 RETURN
152 END
```

Listing of EYELOC.FOR at 14:40:00 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE EYELOC(VRP,EYEPOS)
2      C
3      C      THIS SUBROUTINE ALLOWS THE USER TO CHOOSE THE 'IGL' EYE
4      C      POSITION. THE VIEW UP VECTOR IS ASSUMED TO BE THE Z-DIRECTION,
5      C      SO AN ERROR OCCURS IF THE VIEW DIRECTION COINCIDES WITH THE
6      C      Z AXIS.
7      C
8      C      INPUTS:          VRP(3)-      THE 'IGL' VIEW REFERENCE POINT.
9      C
10     C      OUTPUTS:        EYEPOS(3)- THE THREE COMPONENTS OF THE EYE
11     C                      POSITION IN WORLD COORDINATES.
12     C
13     C*****
14     C
15     C      BYTE BEL(3),YES,NO
16     C      REAL EYEPOS(3),VRP(3)
17     C
18     C      DATA BEL/3*7/,YES/89/,NO/78/
19     C
20     C**  UNTIL (EYE POSITION IS VALID)
21     C
22     400    CONTINUE
23     C
24     C*****  PROMPT THE USER FOR THE DESIRED EYE POSITION.
25     C
26     C      WRITE(5,100)
27     100    FORMAT(5,' ENTER EYE POSITION (X,Y,Z)  ')
28     C      READ(5,*) (EYEPOS(K),K=1,3)
29     C
30     C*****  CHECK X,Y COMPONENTS OF VIEW DIRECTION.
31     C
32     C      IF(EYEPOS(1).NE.O..OR.EYEPOS(2).NE.O.) GO TO 200
33     C
34     C      WRITE(5,300) BEL
35     300    FORMAT(' EYE POSITION ON Z-AXIS, TRY AGAIN  ',3A1)
36     C      GO TO 400
37     C
38     C**  END UNTIL
39     C
40     200    CONTINUE
41     C
42     C      RETURN
43     C      END
```

Listing of EXPERT.FOR at 14:40:00 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE EXPERT(WPCDAT)
2      C
3      C      THIS SUBROUTINE IS THE EXPERT WELDER MODULE. FIXED AND
4      C      CONTINUOUSLY VARYING PARAMETERS ARE CHOSEN AS A FUNCTION OF
5      C      GIVEN WORKSTATION AND WORKPIECE PARAMETERS.
6      C
7      C      INPUT ARGUMENTS: WPCDAT- NAME OF SEAM DATA FILE
8      C
9      C      OUTPUT ARGUMENTS: NONE
10     C
11     C      FILES READ:      WPCDAT- FILE CONTAINING SEAM DATA
12     C                      WKSTAT- FILE CONTAINING ROBOT AND WELDER
13     C                      PARAMETERS.
14     C
15     C      FILES WRITTEN:  WPARAM- TEMPORARY FILE CONTAINING CURRENT,SPEED
16     C                      DATA FOR EACH POINT ON THE CURRENT SEAM.
17     C
18     C      COMMON BLOCKS ACCESSED: /MATRIX/HOT,HOW,SHOT,SHOW
19     C
20     C      COMMON BLOCKS CHANGED: /EXPERT/WPARAM,VOLT,WSIZE,FEED,GFR,
21     C                          WALPHA,BETA,STKOUT
22     C                          /WKSTAT/WKSTAT
23     C
24     C      FORMAT OF FILE 'WPCDAT'.
25     C
26     C      RECORD 1:
27     C
28     C          JTYPE,XSECT,T1,T2,NPOINT,ICURVE
29     C
30     C
31     C      RECORDS 2 TO NPOINT+1:
32     C
33     C          INDEX,S,R(3),T(3),A(3),B(3)
34     C
35     C
36     C      FORMAT OF FILE 'WKSTAT'.
37     C
38     C      RECORDS 1 TO 26:
39     C
40     C          CONTAIN HEADINGS, FIXED ROBOT PARAMETERS.
41     C
42     C      RECORD 27:
43     C
44     C          WSIZE
45     C
46     C
47     C      VARIABLE DEFINITIONS:
48     C
49     C          JTYPE- CODE IDENTIFYING JOINT TYPE
50     C
51     C          XSECT- WELD CROSS SECTIONAL AREA
52     C
53     C          T1,T2- THICKNESSES OF PARENT MATERIAL
54     C
55     C          NPOINT- NUMBER OF POINTS DEFINED ALONG SEAM
56     C
57     C          ICURVE- SEAM TYPE- 0=STRAIGHT LINE
58     C                      1=SPACE CURVE
```

Listing of EXPERT.FOR at 14:40:00 on AUG 9, 1984 for CCId=DACD Page 2

```

59      C
60      C          IFXVAR- WELDING METHOD- O=FIXED PARAMETERS (USED ONLY
61      C                      WHEN ICURVE=1)
62      C                      1=VARYING PARAMETERS (USED ONLY
63      C                      WHEN ICURVE=1)
64      C
65      C          VOLT- WELDER VOLTAGE SETTING
66      C
67      C          WSIZE- WIRE SIZE
68      C
69      C          FEED- WIRE FEED RATE
70      C
71      C          GFR- SHIELDING GAS FLOW RATE
72      C
73      C          CURRNT- WELDER CURRENT
74      C
75      C          SPEED- WELDING SPEED
76      C
77      C          T(3),A(3)- VECTORS DEFINING SEAM TANGENT AND NORMAL
78      C                      DIRECTIONS RELATIVE TO THE WORKPIECE
79      C
80      C          TW(3),AW(3)- 'T','A' RELATIVE TO WORLD FRAME
81      C
82      C          SHOW(4,4)- HOMOGENEOUS TRANSFORMATION MATRIX DEFINING
83      C                      CURRENT WORKPIECE LOCATION AND ORIENTATION
84      C                      RELATIVE TO WORLD COORDINATES.
85      C
86      C          SUBROUTINES CALLED:      CALCS
87      C                                      OPEN
88      C                                      SHUT
89      C                                      WIRE1
90      C                                      WIRE2
91      C                                      WIRE3
92      C
93      C*****
94      C
95      C          BYTE WKSTAT(20),WPARAM(20)
96      C          BYTE WPCDAT(20)
97      C          BYTE YES,ANS
98      C          REAL SHOT(4,4),SHOW(4,4)
99      C          REAL TEMP(3)
100     C          DOUBLE PRECISION HOT(4,4),HOW(4,4)
101     C
102     C          COMMON /EXPERT/WPARAM,VOLT,WSIZE,FEED,GFR,WALPHA,BETA,STKOUT
103     C          COMMON /WKSTAT/WKSTAT
104     C          COMMON /MATRIX/HOT,HOW,SHOT,SHOW
105     C          COMMON /W123/CURRNT,SPEED,XSECT,TX
106     C          COMMON/COEF/TW(3),AW(3),T(3),A(3)
107     C
108     C          DATA YES/89/
109     C
110     C*****  INITIALIZE PARAMETER 'XMIN'.
111     C
112     C          XMIN=1.
113     C
114     C*****  OPEN FILES.
115     C
116     C          CALL OPEN(12,WPARAM,4,1)

```

Listing of EXPERT.FOR at 14:40:00 on AUG 9, 1984 for CCId=DACO Page 3

```
117          CALL OPEN(13,WPCDAT,1,1)
118          CALL OPEN(14,WKSTAT,4,1)
119      C
120      C***** SKIP FIRST 26 RECORDS OF FILE 'WKSTAT'.
121      C
122          DO 150 I=1,26
123              READ(14,*)
124          150 CONTINUE
125      C
126      C***** READ FIXED WELDING PARAMETERS FROM 'WKSTAT'.
127      C
128          READ(14,*) WSIZE
129      C
130      C
131          WRITE(5,300)
132      300      FORMAT(' ENTER METHOD OF WELDING: 0=FIXED PARAMETERS,1=VARIABLE
133      * PARAMETERS ', $)
134      C
135          READ(5,*) IFXVAR
136      C
137      C***** READ FIXED SEAM WELDING PARAMETERS FROM 'WPCDAT'.
138      C
139          READ(13) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
140
141      C***** DETERMINE THE MINIMUM (SMALLER) THICKNESS
142      C
143          TX=AMIN1(T1,T2)
144      C
145      C***** CHECK WIRE SIZE AND BRANCH OFF TO CORRESPONDING ROUTINE
146      C
147      220      IF(WSIZE.EQ.0.030) GO TO 240
148              IF(WSIZE.EQ.0.035) GO TO 250
149              IF(WSIZE.EQ.0.045) GO TO 260
150              WRITE(5,*) 'WIRE SIZE ERROR'
151              ERROR=2
152              GO TO 1001
153      C
154      C***** SUB. WIRE1 DETERMINES WELDING PARAMETERS FOR WIRE SIZE OF 0.030
155      C***** INCHES.
156      C
157      240          CALL WIRE1
158      C
159      C***** GO TO 500 FOR POSSIBLE ORIENTATION ADJUSTMENT
160      C
161          GO TO 500
162      C
163      C***** SUB. WIRE2 DETERMINES WELDING PARAMETERS FOR WIRE SIZE OF 0.035
164      C***** INCHES.
165      C
166      250          CALL WIRE2
167      C
168      C***** GO TO 500 FOR POSSIBLE ORIENTATION ADJUSTMENT
169      C
170          GO TO 500
171      C
172      C***** SUB. WIRE3 DETERMINES WELDING PARAMETERS FOR WIRE SIZE OF 0.045
173      C***** INCHES.
174      C
```

Listing of EXPERT.FOR at 14:40:00 on AUG 9, 1984 for CCId=DAC0 Page 4

```

175      260      CALL WIRE3
176      C
177      C***** GO TO 500 FOR ORIENTATION ADJUSTMENT
178      C
179      500      CURNNT=CURRNT
180      IF(ICURVE.EQ.1) GO TO 700
181      C
182      C***** CURRENT ADJUSTMENT FOR SIMPLE (STRAIGHT) SEAMS
183      C
184      READ(13) INDEX,S,(TEMP(L),L=1,3),(T(J),J=1,3),
185      *(A(K),K=1,3),(TEMP(L),L=1,3)
186      C
187      C***** CALL 'CALCS 'TO DETERMINE AND RETURN ADJUSTMENT (ORIENTATION)
188      C***** FACTORS FOR SIMPLE SEAMS THIS IS DONE FOR ONE OF THE POINTS
189      C***** ON THE SEAM SINCE ALL THE POINTS ARE IDENTICAL.
190      C
191      CALL CALCS(CPHI,CTHE)
192      CURRNT=CURNNT*CPHI*CTHE
193      GO TO 900
194      C
195      700      CONTINUE
196      C
197      C***** CURRENT ADJUSTMENT FOR COMPLEX (CURVED) SEAMS
198      C
199      READ(13,END=750) INDEX,S,(TEMP(L),L=1,3),(T(J),J=1,3),
200      *(A(K),K=1,3),(TEMP(L),L=1,3)
201      C
202      C***** CALL 'CALCS 'TO DETERMINE AND RETURN ADJUSTMENT (ORIENTATION)
203      C***** FACTORS POINT BY POINT
204      C
205      CALL CALCS(CPHI,CTHE)
206      CURRNT=CURNNT*CPHI*CTHE
207      IF(IFXVAR.EQ.0) GO TO 850
208      C
209      C***** RECORD CALCULATED CURRENT AND SPEED POINT BY POINT FOR COMPLEX
210      C***** SEAMS.
211      C
212      WRITE(12,*) CURRNT,SPEED
213      C
214      C***** FOR FIXED WELDING PARAMETERS MINIMUM ADJUSTMENT FACTOR IS USED
215      C***** (THE MAXIMUM ADJUSTMENT FACTOR IS 1.00)
216      C
217      850      XMIN=AMIN1(XMIN,CPHI*CTHE)
218      GO TO 700
219      C
220      750      CONTINUE
221      C
222      IF(IFXVAR.EQ.1) GO TO 1001
223      C
224      C***** FIXED WELDING PARAMETERS, OVERRIDE OPTION MAY BE USED
225      C
226      CURRNT=CURNNT*XMIN
227      C
228      C
229      C
230      C ***** OBSERVE AND/OR OVERRIDE *****
231      C
232      C

```

Listing of EXPERT.FOR at 14:40:00 on AUG 9, 1984 for CCid=DAC0 Page 5

```
233      C
234      900      WRITE(5,301)
235      301      FORMAT(' DO YOU WISH TO OBSERVE/OVERRIDE(Y/N)? ', $)
236      READ(5,325) ANS
237      325      FORMAT(A1)
238      IF (ANS.NE.YES) GO TO 1000
239      C
240      201      WRITE(5,302) CURRNT
241      302      FORMAT(' PRESENT CURRENT=',F6.1,'AMPS..DO YOU WISH TO
242      * OVERRIDE(Y/N)? ', $)
243      READ(5,325) ANS
244      IF(ANS.NE.YES) GO TO 202
245      WRITE(5,352)
246      352      FORMAT(' ENTER NEW VALUE ', $)
247      READ(5,*) CURRNT
248      C
249      202      WRITE(5,303) VOLT
250      303      FORMAT(' PRESENT VOLTAGE=',F6.2,'(VOLTS)..DO YOU WISH TO
251      * OVERRIDE(Y/N)? ', $)
252      READ(5,325) ANS
253      IF(ANS.NE.YES) GO TO 203
254      WRITE(5,352)
255      READ (5,*) VOLT
256      C
257      203      WRITE(5,304) FEED
258      304      FORMAT(' PRESENT WIRE FEED=',F6.1,'(IPM)..DO YOU WISH TO
259      * OVERRIDE(Y/N)? ', $)
260      READ(5,325) ANS
261      IF(ANS.NE.YES) GO TO 204
262      WRITE(5,352)
263      READ (5,*) FEED
264      C
265      204      WRITE(5,305) GFR
266      305      FORMAT(' PRESENT GAS FLOW RATE=',F6.2,'(CFH)..DO YOU WISH TO
267      * OVERRIDE(Y/N)? ', $)
268      READ(5,325) ANS
269      IF(ANS.NE.YES) GO TO 205
270      WRITE(5,352)
271      READ (5,*) GFR
272      C
273      205      WRITE(5,306) SPEED
274      306      FORMAT(' PRESENT SPEED=',F6.2,'(IPM)..DO YOU WISH TO
275      * OVERRIDE(Y/N)? ', $)
276      READ(5,325) ANS
277      IF(ANS.NE.YES) GO TO 206
278      WRITE(5,352)
279      READ (5,*) , SPEED
280      C
281      206      WRITE(5,307) STKOUT
282      307      FORMAT(' PRESENT STICK-OUT=',F6.3,'(INCHES)..DO YOU WISH TO
283      * OVERRIDE(Y/N)? ', $)
284      READ(5,325) ANS
285      IF(ANS.NE.YES) GO TO 207
286      WRITE(5,352)
287      READ (5,*) STKOUT
288      C
289      207      WRITE(5,308) WALPHA
290      308      FORMAT(' PRESENT LONGITUDINAL ANGLE=',F6.2,'(DEG)..DO YOU WISH
```

Listing of EXPERT.FOR at 14:40:00 on AUG 9, 1984 for CCId=DACO Page 6

```
291      * TO OVERRRIDE(Y/N)? ',$)
292      READ(5,325) ANS
293      IF(ANS.NE.YES) GO TO 208
294      WRITE(5,352)
295      READ (5,*)   WALPHA
296      C
297      208  WRITE(5,309) BETA
298      309  FORMAT(' PRESENT TRANSVERSE ANGLE=',F6.2,'(DEG)..DO YOU WISH TO
299      * OVERRIDE(Y/N)? ',$)
300      READ(5,325) ANS
301      IF(ANS.NE.YES) GO TO 209
302      WRITE(5,352)
303      READ (5,*)   BETA
304      C
305      209  CONTINUE
306      GO TO 900
307      C
308      C
309      C
310      C*****
311      C
312      1000  CONTINUE
313      IF(IFXVAR.EQ.1.AND.ICURVE.EQ.1) GO TO 1001
314      C
315      C***** RECORD CALCULATED CURRENT AND SPEED FOR SIMPLE SEAMS
316      C
317      DO 120 I=1,NPOINT
318      WRITE(12,*) CURRNT,SPEED
319      120  CONTINUE
320      C
321      1001  CONTINUE
322      C
323      C***** CLOSE FILES.
324      C
325      CALL SHUT(12)
326      CALL SHUT(13)
327      CALL SHUT(14)
328      C
329      RETURN
330      END
```

Listing of HMULT.FOR at 14:40:20 on AUG 9, 1984 for CCId=DACD Page 1

```
1          SUBROUTINE HMULT(A,B,C)
2      C
3      C      THIS SUBROUTINE MULTIPLIES TWO 4X4 HOMOGENEOUS TRANSFORMATIONS.
4      C      C=AB, WHERE A, B, C ARE HOMOGENEOUS TRANSFORMATIONS.
5      C
6      C      THE MULTIPLICATION IS DOUBLE PRECISION.
7      C
8      C*****
9      C
10     DOUBLE PRECISION A(4,4),B(4,4),C(4,4)
11     C
12     DO 100 I=1,3
13         DO 200 J=1,4
14             C(I,J)=A(I,1)*B(1,J)+A(I,2)*B(2,J)+A(I,3)*B(3,J)
15             IF(J.EQ.4) C(I,J)=C(I,J)+A(I,J)
16             C(4,1)=0.
17             C(4,2)=0.
18             C(4,3)=0.
19             C(4,4)=1.
20     200 CONTINUE
21     100 CONTINUE
22     C
23     RETURN
24     END
```

Listing of HMULTS.FOR at 14:40:21 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE HMULTS(A,B,C)
2      C
3      C      THIS SUBROUTINE MULTIPLIES TWO 4X4 HOMOGENEOUS TRANSFORMATIONS.
4      C      C=AB, WHERE A, B, C ARE HOMOGENEOUS TRANSFORMATIONS.
5      C
6      C      THE MULTIPLICATION IS **SINGLE** PRECISION.
7      C
8      C*****
9      C
10     REAL A(4,4),B(4,4),C(4,4)
11     C
12     DO 100 I=1,3
13     DO 200 J=1,4
14     C(I,J)=A(I,1)*B(1,J)+A(I,2)*B(2,J)+A(I,3)*B(3,J)
15     IF(J.EQ.4) C(I,J)=C(I,J)+A(I,J)
16     C(4,1)=0.
17     C(4,2)=0.
18     C(4,3)=0.
19     C(4,4)=1.
20     200 CONTINUE
21     100 CONTINUE
22     C
23     RETURN
24     END
```

Listing of INIT.FOR at 14:40:21 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE INIT(NSEAMS)
2      C
3      C      THIS SUBROUTINE INITIALIZES THE VALUES OF ALL CONSTANT
4      C      PARAMETERS WHICH ARE STORED IN FILES AND SAVES THEM IN
5      C      COMMON BLOCKS.
6      C
7      C      INPUTS: NSEAMS- THE NUMBER OF SEAMS TO BE WELDED
8      C
9      C      OUTPUTS: NONE
10     C
11     C      COMMON BLOCKS CHANGED:  /ROBOT/ - ROBOT GEOMETRIC PARAMETERS
12     C                               /TABDAT/ - FIXED TABLE TRANSFORMATIONS
13     C                               /TOLROT/ - NUMBER AND SIZE OF TOOL
14     C                                   ROTATIONS ABOUT TOOL AXIS
15     C                                   FOR 6 DEGREE OF FREEDOM ROBOT.
16     C
17     C      FILES READ: WKSTAT- ROBOT DESCRIPTION
18     C                     TABLE- TABLE DESCRIPTION
19     C
20     C      FILES CREATED: ROBLST- LIST OF ROBOT COORDINATE OUTPUT FILES
21     C                     ROBDAT- ROBOT COORDINATE FILES
22     C                     LOCSET- FILE RECORDING WELDING SEQUENCE, ROBOT
23     C                     CONFIGURATIONS AND TABLE POSITIONS.
24     C
25     C      SUBROUTINES CALLED:      MATINV
26     C                               OPEN
27     C                               RDFILE
28     C                               SHUT
29     C
30     C*****
31     C
32     C      BYTE TABLE(20),ROBLST(20),LOCSET(20),WKSTAT(20)
33     C      DOUBLE PRECISION HTW(4,4)
34     C      REAL ALPHA(6),LEN(6),DIST(6),RMIN(6),RMAX(6),HOO(4,4),H6G(4,4),
35     C      * INHOO(4,4),INH6G(4,4),SHTW(4,4)
36     C      REAL DELTA
37     C      INTEGER ICONF(6),IMAX
38     C
39     C      COMMON /ROBOT/NRDOF,ALPHA,LEN,DIST,RMIN,RMAX,HOO,H6G,INHOO,INH6G
40     C      COMMON /WKSTAT/WKSTAT
41     C      COMMON /TABLE/TABLE
42     C      COMMON /ROBLST/ROBLST
43     C      COMMON /LOCSET/LOCSET
44     C      COMMON /TABDAT/HTW,SHTW
45     C      COMMON /TOLROT/IMAX,DELTA
46     C      COMMON /CONFIG/ICONF
47     C
48     C
49     C***** READ FIXED ROBOT PARAMETERS *****
50     C
51     C      CALL OPEN(1,WKSTAT,4,1)
52     C
53     C      READ(1,*)
54     C
55     C***** READ THE NUMBER OF DEGREES OF FREEDOM
56     C
57     C      READ(1,*) NRDOF
58     C      READ(1,*)

```

Listing of INIT.FOR at 14:40:21 on AUG 9, 1984 for CCId=DACO Page 2

```
59      READ(1,*)
60      C
61      C***** READ THE ROBOT GEOMETRY PARAMETERS AND JOINT LIMITS
62      C
63          DO 10 I=1,6
64              READ(1,*) LINK,ALPHA(I),LEN(I),DIST(I),RMIN(I),RMAX(I)
65          10  CONTINUE
66      C
67      C***** READ INITIAL ROBOT CONFIGURATION
68      C
69          READ(1,*)
70          READ(1,*) (ICONF(I),I=1,6)
71      C
72      C***** READ THE WORLD TO ROBOT BASE HOMOGENEOUS TRANSFORMATION 'HOO'
73      C
74          READ(1,*)
75          DO 20 I=1,4
76              READ(1,*) (HOO(I,J),J=1,4)
77          20  CONTINUE
78      C
79      C***** READ THE TOOL TO LAST LINK HOMOGENEOUS TRANSFORMATION 'H6G'
80      C
81          READ(1,*)
82          DO 30 I=1,4
83              READ(1,*) (H6G(I,J),J=1,4)
84          30  CONTINUE
85      C
86      C***** READ THE NUMBER AND SIZE OF THE DESIRED ROTATIONS ABOUT THE TOOL
87      C***** AXIS (6 DEG. OF FREEDOM ROBOT ONLY)
88      C
89          READ(1,*)
90          READ(1,*) IMAX,DELTA
91      C
92          CALL SHUT(1)
93      C
94          DELTA=DELTA*3.14159/180.
95      C
96      C***** CALCULATE INVERSE MATRICES FOR USE LATER.
97      C
98          CALL MATINV(HOO,INHOO)
99          CALL MATINV(H6G,INH6G)
100     C
101     C***** READ TABLE PARAMETERS. *****
102     C
103         CALL OPEN(1,TABLE,4,1)
104     C
105     C***** READ THE NUMBER OF DEGREES OF FREEDOM
106     C
107         READ(1,*) NUMDOF
108         DO 40 I=1,NUMDOF+2
109             READ(1,*)
110         40  CONTINUE
111     C
112     C***** READ THE WORKPIECE TO TABLE TRANSFORMATION MATRIX 'HTW'
113     C
114         DO 50 I=1,4
115             READ(1,*) (HTW(I,J),J=1,4)
116         50  CONTINUE
```

Listing of INIT.FOR at 14:40:21 on AUG 9, 1984 for CCid=DACD Page 3

```
117      C
118      C***** CONVERT 'HTW' TO SINGLE PRECISION 'SHTW'
119      C
120          DO 60 I=1,4
121              DO 70 J=1,4
122                  SHTW(I,J)=HTW(I,J)
123      70      CONTINUE
124      60      CONTINUE
125      C
126          CALL SHUT(1)
127      C
128      C***** CREATE EMPTY ROBOT COORDINATE FILES TO BE FILLED LATER. *****
129      C
130          CALL OPEN(1,ROBLST,4,1)
131          DO 80 ISEAM=1,NSEAMS
132              CALL RDFILE(1,ISEAM+1,ROBDAT,20,IEND)
133              CALL OPEN(2,ROBDAT,6,1)
134              CALL SHUT(2)
135      80      CONTINUE
136          CALL SHUT(1)
137      C
138      C***** CREATE EMPTY SEAM, TABLE POSITION AND ROBOT CONFIGURATION
139      C***** FILE TO BE FILLED LATER.
140      C
141          CALL OPEN(1,LOCSET,6,1)
142          CALL SHUT(1)
143      C
144          RETURN
145      END
```

Listing of JOINTS.FOR at 14:40:22 on AUG 9, 1984 for CCId=DACO Page 1

```

1      SUBROUTINE JOINTS(HO6,THETA,IFLAG)
2      C
3      C      THIS SUBROUTINE CALCULATES THE JOINT ANGLES OF A 6 DEGREE OF
4      C      FREEDOM MANIPULATOR GIVEN THE DESIRED LOCATION AND ORIENTATION
5      C      OF THE GRIPPER. THE MANIPULATOR TO BE SOLVED MUST BE
6      C      GEOMETRICALLY IDENTICAL TO A UNIMATION PUMA 560, WITH
7      C      THE JOINT DIMENSIONS BEING SPECIFIABLE.
8      C
9      C      INPUTS: HO6(4,4)- HOMOGENEOUS TRANSFORMATION MATRIX FROM THE
10     C      SIXTH JOINT FRAME 'F6' TO THE ROBOT BASE
11     C      WORLD FRAME 'FO'. 'HO6' IS OF THE FORM
12     C      [n,o,a,p], WHERE n,o,a ARE UNIT VECTORS
13     C      DESCRIBING THE ORIENTATION OF FRAME 'F6',
14     C      AND p IS THE POSITION VECTOR FOR THE ORIGIN
15     C      OF 'F6' RELATIVE TO 'FO'.
16     C
17     C      OUTPUTS: THETA(6): A REAL ARRAY CONTAINING THE JOINT ANGLES IN
18     C      DEGREES.
19     C
20     C      IFLAG: THIS FLAG IS SET TO 1 IF THE ROBOT WORK
21     C      ENVELOPE IS EXCEEDED, OTHERWISE IT IS 0.
22     C
23     C      COMMON: /CONFIG/ICONF(6)-INTEGER ARRAY DESCRIBING DESIRED
24     C      CONFIGURATION OF EACH JOINT, WHEN MORE THAN
25     C      ONE CONFIGURATION IS POSSIBLE.
26     C
27     C      POSSIBLE CONFIGURATIONS:
28     C
29     C      0 0 0 0 0 0 - L A NF
30     C      0 0 0 1 0 0 - L A F
31     C      0 0 1 0 0 0 - L B NF
32     C      0 0 1 1 0 0 - L B F
33     C      1 1 1 0 0 0 - R A NF
34     C      1 1 1 1 0 0 - R A F
35     C      1 1 0 0 0 0 - R B NF
36     C      1 1 0 1 0 0 - R B F
37     C
38     C      WHERE- R=RIGHT SHOULDER CONFIG.
39     C      L=LEFT SHOULDER CONFIG.
40     C      A=ELBOW ABOVE HAND
41     C      B=ELBOW BELOW HAND
42     C      NF=WRIST NOT FLIPPED
43     C      F=WRIST FLIPPED
44     C
45     C      NOTE: ICONF(5-6) NOT USED
46     C
47     C      COMMON: /ROBOT/NRDOF,ALPHA(6),LEN(6),DIST(6),RMIN(6),RMAX(6),
48     C      H00(4,4),H6G(4,4),INH00(4,4),INH6A(4,4)
49     C
50     C
51     C
52     C      FILES: NONE
53     C
54     C      SUBROUTINES CALLED: NONE
55     C
56     C      *****
57     C
58     C      REAL H0G(4,4),THETA(6),ALPHA(6),LEN(6),DIST(6),RMIN(6),RMAX(6)

```

Listing of JOINTS.FOR at 14:40:22 on AUG 9, 1984 for CCId=DACD Page 2

```

59      REAL X,Y,TEMP1,TEMP2,TEMP3,TEMP4,TEMP5,TEMP6,H6G(4,4),
60      *   INHOO(4,4),HOO(4,4),HOG(4,4),Q(6),INH6A(4,4)
61      INTEGER ICONF(6),IFLAG
62      C
63      COMMON /ROBOT/NRDOF,ALPHA,LEN,DIST,RMIN,RMAX,HOO,H6G,INHOO,INH6A
64      COMMON /CONFIG/ICONF
65      C
66      C
67      C***** INITIALIZE 'IFLAG'
68      C
69      IFLAG=0
70      C
71      C***** CALCULATE COMPONENTS OF VECTOR 'Q(3)'
72      C
73      DO 20 I=1,3
74          Q(I)=HOG(I,4)-DIST(6)*HOG(I,3)
75      20  CONTINUE
76      Q(4)=1.
77      C
78      C***** FIND THETA(1) *****
79      C
80      TEST=Q(1)**2+Q(2)**2-DIST(2)**2
81      C
82      C***** IF(ENVELOPE EXCEEDED) SET FLAG AND RETURN
83      C
84      IF(TEST.LT.O.) GO TO 850
85      C
86      TEMP=SQRT(Q(1)**2+Q(2)**2-DIST(2)**2)
87      T1=Q(2)*TEMP
88      T2=Q(1)*TEMP
89      C
90      C***** CHOOSE SOLUTION DEPENDING ON ROBOT CONFIGURATION.
91      C
92      IF (ICONF(1).GT.1) GO TO 900
93      IF (ICONF(1).NE.1) GO TO 100
94          T1=-T1
95          T2=-T2
96      100  CONTINUE
97      C
98      Y=T1-DIST(2)*Q(1)
99      X=T2+DIST(2)*Q(2)
100      C
101      THETA(1)=ATAN2(Y,X)
102      C
103      C***** FIND THETA(3) *****
104      C
105      Y=(Q(1)**2+Q(2)**2+Q(3)**2-DIST(4)**2-LEN(2)**2-DIST(2)**2-
106      *   LEN(3)**2)/(2.*LEN(2))
107      RSQ=LEN(3)**2+DIST(4)**2
108      TEST2=RSQ-Y**2
109      C
110      C***** IF(ENVELOPE EXCEEDED) SET FLAG AND RETURN
111      C
112      IF(TEST2.LT.O.) GO TO 850
113      C
114      X=SQRT(TEST2)
115      THETA1=ATAN2(LEN(3),-DIST(4))
116      C

```

Listing of JOINTS.FOR at 14:40:22 on AUG 9, 1984 for CCId=DACO Page 3

```
117 C***** CHOOSE SOLUTION DEPENDING ON CONFIGURATION.
118 C
119 IF (ICONF(3).GT.1) GO TO 900
120 IF (ICONF(3).EQ.1) X=-X
121 C
122 THETA(3)=THETA1-ATAN2(Y,X)
123 C
124 C***** FIND THETA(2) *****
125 C
126 C3=COS(THETA(3))
127 S3=SIN(THETA(3))
128 TEMP1=DIST(4)*C3-LEN(3)*S3
129 TEMP2=DIST(4)*S3+LEN(3)*C3+LEN(2)
130 THETA1=ATAN2(TEMP1,TEMP2)
131 RSQ=TEMP1**2+TEMP2**2
132 TEST3=RSQ-Q(3)**2
133 C
134 C***** IF(ENVELOPE EXCEEDED) SET FLAG AND RETURN
135 C
136 IF(TEST3.LT.O.) GO TO 850
137 C
138 Y=Q(3)
139 X=SQRT(TEST3)
140 C
141 C***** CHOOSE SOLUTION DEPENDING ON CONFIGURATION.
142 C
143 IF(ICONF(2).NE.O) X=-X
144 C
145 THETA(2)=THETA1-ATAN2(Y,X)
146 C
147 C***** FIND THETA(4) *****
148 C
149 Y=COS(THETA(1))*HO6(2,3)-SIN(THETA(1))*HO6(1,3)
150 X=COS(THETA(1))*COS(THETA(2)+THETA(3))*HO6(1,3)+
151 * SIN(THETA(1))*COS(THETA(2)+THETA(3))*HO6(2,3)-
152 * SIN(THETA(2)+THETA(3))*HO6(3,3)
153 C
154 C***** CHOOSE SOLUTION DEPENDING ON CONFIGURATION.
155 C
156 IF(ICONF(4).GT.1) GO TO 900
157 IF(ICONF(4).NE.1) GO TO 101
158 X=-X
159 Y=-Y
160 101 CONTINUE
161 C
162 THETA(4)=ATAN2(Y,X)
163 C
164 C***** FIND THETA(5) *****
165 C
166 TEMP1=(COS(THETA(1))*COS(THETA(2)+THETA(3))*COS(THETA(4))-
167 * SIN(THETA(1))*SIN(THETA(4)))*HO6(1,3)
168 TEMP2=(SIN(THETA(1))*COS(THETA(2)+THETA(3))*COS(THETA(4))+
169 * COS(THETA(1))*SIN(THETA(4)))*HO6(2,3)
170 TEMP3=COS(THETA(4))*SIN(THETA(2)+THETA(3))*HO6(3,3)
171 TEMP4=COS(THETA(1))*SIN(THETA(2)+THETA(3))*HO6(1,3)
172 TEMP5=SIN(THETA(1))*SIN(THETA(2)+THETA(3))*HO6(2,3)
173 TEMP6=COS(THETA(2)+THETA(3))*HO6(3,3)
174 C
```

Listing of JOINTS.FOR at 14:40:22 on AUG 9, 1984 for CCid=DAC0 Page 4

```
175      Y=TEMP1+TEMP2-TEMP3
176      X=TEMP4+TEMP5+TEMP6
177      C
178      THETA(5)=ATAN2(Y,X)
179      C
180      C***** FIND THETA(6) *****
181      C
182      TEMP1=(-SIN(THETA(1))*COS(THETA(4))-COS(THETA(1))*COS(THETA(2)+
183      *      THETA(3))*SIN(THETA(4)))*H06(1,1)
184      TEMP2=(COS(THETA(1))*COS(THETA(4))-SIN(THETA(1))*COS(THETA(2)+
185      *      THETA(3))*SIN(THETA(4)))*H06(2,1)
186      TEMP3=SIN(THETA(4))*SIN(THETA(2)+THETA(3))*H06(3,1)
187      TEMP4=(-SIN(THETA(1))*COS(THETA(4))-COS(THETA(1))*COS(THETA(2)+
188      *      THETA(3))*SIN(THETA(4)))*H06(1,2)
189      TEMP5=(COS(THETA(1))*COS(THETA(4))-SIN(THETA(1))*COS(THETA(2)+
190      *      THETA(3))*SIN(THETA(4)))*H06(2,2)
191      TEMP6=SIN(THETA(4))*SIN(THETA(2)+THETA(3))*H06(3,2)
192      C
193      Y=TEMP1+TEMP2+TEMP3
194      X=TEMP4+TEMP5+TEMP6
195      C
196      THETA(6)=ATAN2(Y,X)
197      C
198      C***** CONVERT ANGLES INTO DEGREES, INSURE THAT -180.LE.THETA.LE.180.
199      C
200      CONV=180./3.14159
201      DO 30 I=1,6
202      THETA(I)=THETA(I)*CONV
203      IF(THETA(I).GT.180.) THETA(I)=THETA(I)-360.
204      IF(THETA(I).LT.-180) THETA(I)=THETA(I)+360.
205      30 CONTINUE
206      C
207      GO TO 800
208      C
209      C***** POSITION OUTSIDE ENVELOPE, SET FLAG
210      C
211      850 CONTINUE
212      IFLAG=1
213      GO TO 800
214      C
215      C***** ERROR MESSAGES *****
216      C
217      900 WRITE(5,901)
218      901 FORMAT(' VALUE OF PARAMETER ICONF NOT 0 OR 1')
219      C
220      800 CONTINUE
221      C
222      RETURN
223      END
```

Listing of LIMITS.FOR at 14:40:47 on AUG 9, 1984 for CCid=DACO Page. 1

```

1      SUBROUTINE LIMITS(THETA,IFLAG,IJOINT)
2      C
3      C      THIS SUBROUTINE CHECKS THE CALCULATED JOINT ANGLES 'THETA'
4      C      AGAINST THEIR PHYSICAL LIMITS.  IF ANY JOINT ANGLE IS
5      C      EXCEEDED, 'IFLAG' IS SET.
6      C
7      C
8      C      INPUTS: THETA(6)-JOINT ANGLES IN DEGREES
9      C
10     C      COMMON /ROBOT/ALPHA,LEN,DIST,RMIN,RMAX,HOO,H6G,INHOO,
11     C      INH6A
12     C
13     C
14     C      OUTPUTS: IFLAG- 0-ANGLES WITHIN LIMITS
15     C      1-JOINT LIMIT EXCEEDED
16     C
17     C      IJOINT(6)-ARRAY WHICH IDENTIFIES JOINTS
18     C      WHOSE LIMITS HAVE BEEN EXCEEDED.
19     C      IE. 0 0 0 1 0 0 - JOINT 4 EXCEEDED
20     C
21     C
22     C
23     C*****
24     REAL THETA(6),RMIN(6),RMAX(6),ALPHA(6),DIST(6),LEN(6)
25     REAL HOO(4,4),H6G(4,4),INHOO(4,4),INH6A(4,4)
26     INTEGER IJOINT(6)
27     C
28     COMMON /ROBOT/NRDOF,ALPHA,LEN,DIST,RMIN,RMAX,HOO,H6G,INHOO,INH6A
29     C
30     C
31     C*****  INITIALIZE 'IJOINT','IFLAG'
32     C
33     DO 11 I=1,6
34         IJOINT(I)=0
35     11 CONTINUE
36     C
37     IFLAG=0
38     C
39     C*****  TEST EACH JOINT ANGLE 'THETA' AGAINST ITS SPECIFIED LIMITS
40     C*****  'RMIN','RMAX'.
41     C
42     DO 20 I=1,6
43     C
44         IF(RMIN(I).LE.RMAX(I)) GO TO 200
45     C
46         IF(THETA(I).GE.RMIN(I).OR.THETA(I).LE.RMAX(I)) GO TO 100
47         IFLAG=1
48         IJOINT(I)=1
49         GO TO 100
50     200 CONTINUE
51     C
52         IF(THETA(I).GE.RMIN(I).AND.THETA(I).LE.RMAX(I)) GO TO 100
53         IFLAG=1
54         IJOINT(I)=1
55     100 CONTINUE
56     C
57     20 CONTINUE
58     C
```

Listing of LIMITS.FOR at 14:40:47 on AUG 9, 1984 for CCId=DACO Page 2

59	RETURN
60	END

Listing of LINDIS.FOR at 14:40:47 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE LINDIS(P1,P2,Q1,Q2,USIGHT,ISIGN)
2      C
3      C      THIS SUBROUTINE FINDS THE SHORTEST VECTOR FROM
4      C      LINE P1P2 TO LINE Q1Q2, AND TESTS IF P1P2 IS IN FRONT OF
5      C      OR BEHIND Q1Q2 FOR A GIVEN LINE OF SIGHT VECTOR 'USIGHT'.
6      C
7      C      INPUTS: P1(3),P2(3)- ENDPOINTS OF FIRST LINE
8      C      Q1(3),Q2(3)- ENDPOINTS OF SECOND LINE
9      C      USIGHT(3)- LINE OF SIGHT VECTOR
10     C
11     C      OUTPUTS: ISIGN- 0= Q1Q2 IN FRONT OF P1P2
12     C      1= Q1Q2 BEHIND P1P2
13     C
14     C*****
15     C
16     REAL P1(3),P2(3),Q1(3),Q2(3),USIGHT(3),UP(3),UQ(3),DELT(3)
17     REAL UXP1(3),UXP(3)
18     C
19     C***** INITIALIZE ISIGN
20     C
21     ISIGN=0
22     C
23     C***** FIND THE EDGE VECTORS 'UP','UQ' FROM THE ENDPOINTS
24     C***** FIND A VECTOR 'DELT' FROM EDGE 'UP' TO EDGE 'UQ'
25     C
26     DO 10 I=1,3
27         UP(I)=P2(I)-P1(I)
28         UQ(I)=Q2(I)-Q1(I)
29         DELT(I)=Q1(I)-P1(I)
30     10 CONTINUE
31     C
32     C***** FIND THE CROSS PRODUCT UXP = UP X UQ
33     C***** THIS GIVES US THE COMMON NORMAL OF THE TWO EDGES
34     C
35     CALL XPROD(UP,UQ,UXP)
36     C
37     C***** TEST IF 'UXP' IS ZERO (IE. 'UP','UQ' ARE PARALLEL)
38     C
39     DOT1=UXP(1)**2+UXP(2)**2+UXP(3)**2
40     IF(DOT1.LT..001) GO TO 500
41     GO TO 600
42     C
43     C***** IF 'UXP' IS ZERO, FIND THE CROSS PRODUCT UXP1 = UP X DELT
44     C
45     500 CONTINUE
46     CALL XPROD(UP,DELT,UXP1)
47     C
48     C***** IF 'UXP' IS AGAIN ZERO, 'UP'='UQ'. THIS IS AN ERROR CONDITION.
49     C
50     DIST=UXP1(1)**2+UXP1(2)**2+UXP1(3)**2
51     IF(DIST.LT..001) GO TO 700
52     C
53     C***** THE COMMON NORMAL 'UXP' IS NOW FOUND FROM UXP = UP X UXP1
54     C
55     CALL XPROD(UP,UXP1,UXP)
56     C
57     600 CONTINUE
58     C
```

Listing of LINDIS.FOR at 14:40:47 on AUG 9, 1984 for CCid=DACO Page 2

```
59 C***** FIND THE DOT PRODUCT OF THE COMMON NORMAL 'UXP' AND THE LINE OF
60 C***** SIGHT VECTOR 'USIGHT'.
61 C
62 CALL DOTPRD(UXP,USIGHT,DOT)
63 C
64 C***** FIND THE DIRECTION OF 'UXP' RELATIVE TO EDGES 'UP','UQ' BY
65 C***** TAKING THE DOT PRODUCT OF ANY VECTOR 'DELT' DIRECTED FROM
66 C***** 'UP' TO 'UQ' AND 'UXP'. THE RESULT WILL BE POSITIVE
67 C***** IF 'UXP' IS DIRECTED FROM 'UP' TO 'UQ'.
68 C
69 CALL DOTPRD(DELT,UXP,DOT2)
70 C
71 C***** 'UQ' IS BEHIND 'UP' IF DOT*DOT2 IS POSITIVE
72 C
73 IF(DOT*DOT2.GT.O.OOO1) ISIGN=1
74 C
75 700 CONTINUE
76 C
77 RETURN
78 END
```

Listing of MATCHV.FOR at 14:40:48 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE MATCHV(DH,SH,ICONV)
2      C
3      C      THIS SUBROUTINE CONVERTS A DOUBLE PRECISION 4X4 MATRIX
4      C      TO SINGLE PRECISION IF ICONV=1.  THE INVERSE OCCURS IF
5      C      ICONV=2.
6      C
7      REAL SH(4,4)
8      DOUBLE PRECISION DH(4,4)
9      C
10     IF(ICONV.EQ.2) GO TO 100
11     C
12     DO 1 I=1,4
13         DO 2 J=1,4
14             SH(I,J)=DH(I,J)
15         2    CONTINUE
16     1    CONTINUE
17     GO TO 200
18     C
19     100  CONTINUE
20     C
21     DO 3 I=1,4
22         DO 4 J=1,4
23             DH(I,J)=SH(I,J)
24         4    CONTINUE
25     3    CONTINUE
26     C
27     200  CONTINUE
28     C
29     RETURN
30     END
```

Listing of MATINV.FOR at 14:40:48 on AUG 9, 1984 for CCid=DACD Page 1

```

1      SUBROUTINE MATINV(H,HINV)
2      C
3      C      THIS SUBROUTINE FINDS THE INVERSE 'HINV' OF A HOMOGENEOUS
4      C      TRANSFORMATION MATRIX 'H'.
5      C
6      C      INPUTS: H(4,4)- HOMOGENEOUS TRANSFORMATION MATRIX
7      C
8      C      OUTPUTS: HINV(4,4)- INVERSE OF 'H'
9      C
10     C*****
11     C
12     REAL H(4,4),HINV(4,4)
13     C
14     DO 1 I=1,3
15         DO 2 J=1,3
16             HINV(I,J)=H(J,I)
17     2    CONTINUE
18     1    CONTINUE
19     C
20     HINV(1,4)=- (H(1,4)*H(1,1)+H(2,4)*H(2,1)+H(3,4)*H(3,1))
21     HINV(2,4)=- (H(1,4)*H(1,2)+H(2,4)*H(2,2)+H(3,4)*H(3,2))
22     HINV(3,4)=- (H(1,4)*H(1,3)+H(2,4)*H(2,3)+H(3,4)*H(3,3))
23     C
24     HINV(4,1)=0.
25     HINV(4,2)=0.
26     HINV(4,3)=0.
27     HINV(4,4)=1.
28     C
29     RETURN
30     END
```

Listing of NOAP1.FOR at 14:41:10 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE NOAP1(HO6,XYZDAT)
2      C
3      C      THIS SUBROUTINE FINDS THE PUMA LOCATION PARAMETERS X,Y,Z,O,A,T
4      C      GIVEN THE HOMOGENEOUS TRANSFORMATION 'HO6' FOR THE PUMA 560
5      C      ROBOT.
6      C
7      C
8      C      INPUTS: HO6(4,4) -HOMOGENEOUS COORD TRANSFORMATION FROM 6TH
9      C                      LINK FRAME F6 TO ROBOT BASE FRAME FO.
10     C
11     C      OUTPUTS: XYZDAT(6) -ARRAY CONTAINING ROBOT LOCATION PARAMETERS
12     C                      X,Y,Z,O,A,T WHERE X,Y,Z ARE THE COORDINATES
13     C                      OF THE TOOL CENTER REL. TO FO AND O,A,T ARE
14     C                      THE WRIST ROTATION ANGLES ABOUT THE
15     C                      AXES OF FO.
16     C
17     C*****
18     C
19     REAL HO6(4,4),PO,PA,PT
20     REAL XYZDAT(6)
21     C
22     C*****  CALCULATE PO,PA,PT
23     C
24     CONV=180./3.14159
25     C
26     C*****  TEST IF THE APPROACH VECTOR 'a' IS VERTICAL (ALONG ROBOT BASE
27     C*****  Z-AXIS)
28     C
29     IF(ABS(HO6(1,3)).LT..001.AND.ABS(HO6(2,3)).LT..001) GO TO 400
30     C
31     C*****  IF FALSE
32     C
33     PO=ATAN2(-HO6(1,3),HO6(2,3))
34     TEMP=SIN(PO)*HO6(1,3)-COS(PO)*HO6(2,3)
35     PA=ATAN2(-HO6(3,3),SIN(PO)*HO6(1,3)-COS(PO)*HO6(2,3))
36     PT=ATAN2(COS(PO)*HO6(1,1)+SIN(PO)*HO6(2,1),COS(PO)*
37     * HO6(1,2)+SIN(PO)*HO6(2,2))
38     GO TO 410
39     C
40     C*****  IF TRUE
41     C
42     400    CONTINUE
43     C
44     IF(HO6(3,3).GT.O.) PA=90.
45     IF(HO6(3,3).LT.O.) PA=-90.
46     PT=O.
47     PO=ATAN2(HO6(2,2),HO6(1,2))
48     410    CONTINUE
49     C
50     C*****  COORDINATES X,Y,Z ARE THE SAME AS THE LAST COLUMN OF 'HO6'
51     C
52     DO 303 J=1,3
53         XYZDAT(J)=HO6(J,4)
54     303    CONTINUE
55     C
56     XYZDAT(4)=PO*CONV
57     XYZDAT(5)=PA*CONV
58     XYZDAT(6)=PT*CONV
```

Listing of NOAP1.FOR at 14:41:10 on AUG 9, 1984 for CCid=DAC0 Page 2

```
59      C
60      RETURN
61      END
```

Listing of NOAP5.FOR at 14:41:11 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE NOAP5(ISOLN)
2      C
3      C      THIS SUBROUTINE FINDS THE 5 DEGREE OF FREEDOM PUMA LOCATION
4      C      PARAMETERS X,Y,Z,O,A,T GIVEN THE TOOL CENTER LOCATION VECTOR
5      C      'P', AND THE TOOL AXIS ORIENTATION VECTOR 'A'.
6      C
7      C      INPUTS: ISOLN- FLAG SPECIFYING WHICH OF THE TWO 5 DEGREE OF
8      C      FREEDOM SOLUTIONS TO FIND.
9      C      O- SOLUTION 1
10     C      1- SOLUTION 2
11     C
12     C      COMMON BLOCKS READ:      /ROBOT/- ROBOT PARAMETERS
13     C      /CONFIG/- CURRENT ROBOT CONFIGURATION
14     C      /ROBPOS/- TOOL FRAME, HOG=[n,o,a,p]
15     C
16     C      COMMON BLOCKS ALTERED: /ROBPOS/- TOOL FRAME, HOG=[n,o,a,p]
17     C
18     C*****
19     C
20     REAL N(3),O(3),A(3),P(3)
21     REAL HOG(4,4)
22     REAL ALPHA(6),LEN(6),DIST(6),RMIN(6),RMAX(6),HOO(4,4),H6G(4,4),
23     *   INHOO(4,4),INH6A(4,4)
24     INTEGER ICONF(6)
25     C
26     COMMON /ROBOT/NRDOF,ALPHA,LEN,DIST,RMIN,RMAX,HOO,H6G,INHOO,INH6A
27     COMMON /CONFIG/ICONF
28     COMMON /ROBPOS/HOG
29     C
30     C***** READ THE VALUES OF TOOL FRAME VECTORS 'A','P'.
31     C
32     DO 30 I=1,3
33         A(I)=HOG(I,3)
34         P(I)=HOG(I,4)
35     30 CONTINUE
36     C
37     C***** FIND THE TOOL OFFSET IN THE X DIRECTION RELATIVE TO THE TOOL
38     C***** MOUNTING FLANGE.
39     C
40     DT=H6G(1,4)
41     C
42     C***** CALCULATE THETA1
43     C
44     T1=P(1)-DT*A(1)
45     T2=P(2)-DT*A(2)
46     RSQ=T1**2+T2**2
47     ATAN1=ATAN2(T2,T1)
48     T3=SQRT(RSQ-DIST(2)**2)
49     IF(ICONF(1).EQ.1) T3=-T3
50     ATAN=ATAN2(DIST(2),T3)
51     THET1=ATAN1-ATAN
52     C
53     C***** CALCULATE N(1). THERE ARE TWO POSSIBLE SOLUTIONS, SELECTED BY THE
54     C***** VALUE OF 'ISOLN'.
55     C
56     C1=A(2)**2+A(3)**2
57     C2=C1*A(3)**2
58     C3=C1**2*TAN(THET1)**2

```

Listing of NOAP5.FOR at 14:41:11 on AUG 9, 1984 for CCId=DACO Page 2

```
59          C4=2.*C1*TAN(THET1)*A(1)*A(2)
60          C5=C1*(A(1)**2+A(3)**2)
61          N(1)=SQRT(C2/(C3+C4+C5))
62          IF(ISOLN.EQ.1) N(1)=-N(1)
63          C
64          C***** CALCULATE N(2)
65          C
66          D1=N(1)*A(1)*A(2)
67          D2=SQRT(4.*D1**2-4.*(A(2)**2+A(3)**2)*N(1)**2*(A(1)**2*
68          * A(3)**2)-A(3)**2)
69          D3=2.*(A(2)**2+A(3)**2)
70          N(2)=N(1)*TAN(THET1)
71          C
72          C***** CALCULATE N(3)
73          C
74          N(3)=SQRT(1.-N(1)**2-N(2)**2)
75          DOT=ABS(N(1)*A(1)+N(2)*A(2)+N(3)*A(3))
76          IF(DOT.GT..0001) N(3)=-N(3)
77          C
78          C
79          C***** CALCULATE BASIS VECTOR 'O'
80          C
81          CALL XPROD(A,N,O)
82          C
83          C
84          C***** WRITE BASIS VECTORS 'N','O' INTO 'HOG'.
85          C
86          DO 200 I=1,3
87             HOG(I,1)=N(I)
88             HOG(I,2)=O(I)
89          200 CONTINUE
90          C
91          RETURN
92          END
```

Listing of NORM.FOR at 14:41:11 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE NORM(N1,N2,A)
2      C
3      C      THIS SUBROUTINE CALCULATES THE NORMAL VECTOR, 'A', WHICH
4      C      BISECTS THE TWO PARENT SURFACE NORMALS 'N1','N2'.
5      C
6      C      INPUTS:
7      C      N1(3,3),N2(3,3)=NORMAL VECTORS OF TWO SURFACES AT
8      C      INTERSECTION POINTS I-1, I, I+1.
9      C
10     C      OUTPUT:
11     C      A(3)=UNIT BISECTOR OF N1 AND N2 AT POINT I+1.
12     C
13     C*****
14     C
15     C
16     REAL N1(3,3),N2(3,3),A(3),LX,LY,LZ,L
17     C
18     D1=N1(3,1)+N2(3,1)
19     D2=N1(3,2)+N2(3,2)
20     D3=N1(3,3)+N2(3,3)
21     C
22     LX=D1**2
23     LY=D2**2
24     LZ=D3**2
25     C
26     L=SQRT(LX+LY+LZ)
27     C
28     A(1)=D1/L
29     A(2)=D2/L
30     A(3)=D3/L
31     C
32     RETURN
33     END
```

Listing of OPEN.FOR at 14:41:11 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE OPEN(IUNIT,NAME,ITYPE,IREC)
2      C
3      C      THIS SUBROUTINE OPENS A FILE CALLED (NAME), AND ASSIGNS
4      C      IT TO LOGIC UNIT (IUNIT). THE TYPE OF FILE IS DETERMINED
5      C      BY (ITYPE). THIS SUBROUTINE IS COMPATIBLE WITH VAX-11 FORTRAN.
6      C      (IREC) IS THE RECORD LENGTH FOR DIRECT ACCESS FILES.
7      C
8      C      INPUTS:
9      C          IUNIT=A VALID LOGIC UNIT NUMBER
10     C          NAME=A COMPLETE FILE NAME
11     C          ITYPE=FILE TYPE
12     C              1=UNFORMATTED,EXISTING
13     C              2=UNFORMATTED,NEW
14     C              3=UNFORMATTED,DIRECT ACCESS,EXISTING
15     C              4=FORMATTED,SEQUENTIAL,EXISTING
16     C              5=UNFORMATTED,DIRECT ACCESS,NEW
17     C              6=FORMATTED,SEQUENTIAL,NEW
18     C          IREC=RECORD LENGTH (FOR FILE TYPE 5 ONLY)
19     C
20     C      OUTPUTS: NONE.
21     C
22     C
23     C*****
24     C
25     C      BYTE NAME(20)
26     C
27     C** CASE OF (ITYPE)
28     C
29     C      IF(ITYPE.LT.1.OR.ITYPE.GT.6) GO TO 600
30     C      GO TO (100,200,300,400,500,510) ITYPE
31     C
32     C** CASE (1)
33     C
34     C***** OPEN EXISTING, UNFORMATTED, SEQUENTIAL FILE.
35     C
36     100    CONTINUE
37     C      OPEN (UNIT=IUNIT,FILE=NAME,FORM='UNFORMATTED',STATUS='OLD',
38     C      *    ERR=10)
39     C      GO TO 900
40     C
41     C** CASE (2)
42     C
43     C***** OPEN NEW, UNFORMATTED, SEQUENTIAL FILE.
44     C
45     200    CONTINUE
46     C      OPEN (UNIT=IUNIT,FILE=NAME,FORM='UNFORMATTED',STATUS='NEW',
47     C      *    ERR=10)
48     C      GO TO 900
49     C
50     C** CASE(3)
51     C
52     C***** OPEN EXISTING, DIRECT ACCESS FILE.
53     C
54     300    CONTINUE
55     C      OPEN (UNIT=IUNIT,FILE=NAME,ACCESS='DIRECT',STATUS='OLD',
56     C      *    ERR=10)
57     C      GO TO 900
58     C
```

Listing of OPEN.FOR at 14:41:11 on AUG 9, 1984 for CCid=DAC0 Page 2

```
59      C** CASE(4)
60      C
61      C***** OPEN EXISTING, FORMATTED, SEQUENTIAL FILE.
62      C
63      400      CONTINUE
64              OPEN (UNIT=IUNIT,FILE=NAME,STATUS='OLD',ERR=10)
65              GO TO 900
66      C
67      C** CASE(5)
68      C
69      C***** OPEN NEW, DIRECT ACCESS FILE.
70      C
71      500      CONTINUE
72              OPEN (UNIT=IUNIT,FILE=NAME,FORM='UNFORMATTED',
73                  * ACCESS='DIRECT',RECL=IREC,STATUS='NEW',ERR=10)
74              GO TO 900
75      C
76      C***** OPEN NEW,FORMATTED FILE.
77      C
78      510      CONTINUE
79              OPEN (UNIT=IUNIT,FILE=NAME,STATUS='NEW',ERR=10)
80              GO TO 900
81      C
82      C** ELSE
83      C
84      600      CONTINUE
85              WRITE(5,20)
86      20      FORMAT(' VALUE OF PARAMETER ITYPE NOT VALID')
87              GO TO 900
88      C
89      10      CONTINUE
90              WRITE(5,30)
91      30      FORMAT(' ERROR ENCOUNTERED IN OPEN STATEMENT IN SUBROUTINE
92              * OPEN')
93      C
94      C** END CASE
95      C
96      900      CONTINUE
97              RETURN
98              END
```

Listing of PROCES.FOR at 14:41:34 on AUG 9, 1984 for CCId=DACD Page 1

```

1      SUBROUTINE PROCES
2      C
3      C      THIS PROGRAM READS THE WELDING PARAMETERS, R, N1, N2 FROM CAD
4      C      FILE, CALCULATES ARCLength S, THE POSITION VECTOR R, AND THE
5      C      TOOL FRAME AXES T, A, B.
6      C      THE RESULTS ARE STORED IN AN OUTPUT FILE.
7      C
8      C      INPUTS: FROM CAD FILE
9      C
10     C      JTYPE- JOINT TYPE CODE
11     C      XSECT- WELD CROSS SECTIONAL AREA
12     C      T1,T2- PARENT METAL THICKNESSES
13     C      NPOINT- NUMBER OF LOCATIONS IN FILE
14     C      ICURVE- SEAM TYPE- 0=STRAIGHT LINE
15     C      1=SPACE CURVE
16     C      INDEX=POINT INDEX.
17     C      R(3,3)=POSITION VECTORS FOR POINTS I-1, I, I+1.
18     C      N1(3,3),N2(3,3)=UNIT NORMAL VECTORS OF TWO SURFACES AT
19     C      POINTS I-1, I, I+1.
20     C
21     C      OUTPUTS: TO OUTPUT FILE.
22     C
23     C      JTYPE- JOINT TYPE CODE
24     C      XSECT- WELD CROSS SECTIONAL AREA
25     C      T1,T2- PARENT METAL THICKNESSES
26     C      NPOINT- NUMBER OF LOCATIONS IN FILE
27     C      ICURVE- SEAM TYPE- 0=STRAIGHT LINE
28     C      1=SPACE CURVE
29     C      INDEX- POINT INDEX.
30     C      S(3)=ARCLength AT POINTS I-1, I, I+1.
31     C      R(3,3)=POSITION VECTORS FOR POINTS I-1, I, I+1.
32     C      T(3)=TANGENT VECTOR OF CURVE AT POINT I.
33     C      A(3)=UNIT BISECTOR OF N1, N2 AT POINT I.
34     C      B(3)=CROSS PRODUCT T X A.
35     C
36     C      FILES READ: CAD- THE CAD FILE
37     C
38     C      FILES WRITTEN: WPCDAT- THE OUTPUT FILE
39     C
40     C      SUBROUTINES CALLED:      ARCLen
41     C                               NORM
42     C                               OPEN
43     C                               SHUT
44     C                               TANGEN
45     C                               XPROD
46     C
47     C*****
48     C
49     C      IMPLICIT DOUBLE PRECISION (D)
50     C      BYTE CAD(20), WPCDAT(20)
51     C      REAL R(3,3),N1(3,3),N2(3,3),T(3),A(3),B(3),S(3)
52     C      COMMON /CAD/CAD
53     C      COMMON /WPCDAT/WPCDAT
54     C      COMMON /CADLST/CADLST
55     C      COMMON /WPCLST/WPCLST
56     C      DATA R/9*0./
57     C      DATA ICAD/1/,IOUT/2/
58     C

```

Listing of PROCES.FOR at 14:41:34 on AUG 9, 1984 for CCId=DAC0 Page 2

```
59      C***** OPEN CAD INPUT FILE AS L.U. ICAD.
60      C***** OPEN OUTPUT FILE AS L.U. IOUT.
61      C
62      CALL OPEN(ICAD,CAD,1,1)
63      CALL OPEN(IOUT,WPCDAT,2,1)
64      C
65      C***** INITIALIZE ARCLENGTH 'S', FLAG 'IFLAG'
66      C
67      S(3)=0.
68      IFLAG=0
69      C
70      C***** READ JOINT TYPE, X-SECTIONAL AREA, PARENT THICKNESSES FROM
71      C***** 1ST RECORD OF 'CAD' AND WRITE THEM INTO 1ST RECORD OF 'WPCDAT'.
72      C
73      READ(ICAD) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
74      WRITE(IOUT) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
75      C
76      C
77      C** WHILE (INPUT .LE. EOF)
78      C
79      150      CONTINUE
80      C
81      170      CONTINUE
82      C
83      C***** READ INDEX, R, N1, N2, FROM CAD FILE.
84      C
85      READ(ICAD,END=555) INDEX,(R(3,I),I=1,3),(N1(3,K),K=1,3),
86      *      (N2(3,L),L=1,3)
87      C
88      C***** READ FIRST TWO POINTS BEFORE BEGINNING CALCULATIONS.
89      C***** SET FLAGS IF SECOND POINT OR LAST POINT IS READ.
90      C***** THESE POINTS REQUIRE A MODIFIED PROCEDURE.
91      C
92      C***** FIRST POINT
93      C
94      IF(INDEX.EQ.1) GO TO 700
95      IFLAG=2
96      C
97      C***** SECOND POINT
98      C
99      IF(INDEX.EQ.2) IFLAG=1
100     GO TO 556
101     C
102     C***** LAST POINT
103     C
104     555     CONTINUE
105     INDEX=INDEX+1
106     IFLAG=3
107     556     CONTINUE
108     C
109     C
110     C***** CALCULATE ARCLENGTH S AT POSITION R. 'DELS' IS THE CHANGE IN
111     C***** ARCLENGTH FROM R(I-1) TO R(I)
112     C
113     CALL ARCLEN(R,DELS)
114     S(3)=S(2)+DELS
115     C
116     C***** CALCULATE THE TANGENT VECTOR 'T' AT POSITION 'R'
```

Listing of PROCES.FOR at 14:41:34 on AUG 9, 1984 for CCid=DACO Page 3

```
117 C
118 CALL TANGEN(R,S,T,IFLAG)
119 C
120 C***** CALCULATE THE NORMAL VECTOR 'A' FROM THE SURFACE NORMALS
121 C***** 'N1','N2'
122 C
123 CALL NORM(N1,N2,A)
124 C
125 C
126 C***** CALCULATE THE LATERAL VECTOR 'B', B=A X T (CROSS PRODUCT)
127 C
128 CALL XPROD(A,T,B)
129 C
130 C***** A=T X A (CROSS PRODUCT). THIS STEP ENSURES AN ORTHOGONAL
131 C***** FRAME.
132 C
133 CALL XPROD(T,B,A)
134 C
135 C***** WRITE RESULTS INTO OUTPUT FILE.
136 C
137 JJ=INDEX-1
138 WRITE(IOUT) JJ,S(2),(R(2,I),I=1,3),(T(K),K=1,3),(A(L),L=1,3),
139 * (B(M),M=1,3)
140 C
141 C***** UPDATE CURRENT POSITION, I, BY REDEFINING POINT I AS
142 C***** POINT I-1.
143 C
144 DO 300 K=1,3
145 R(1,K)=R(2,K)
146 300 CONTINUE
147 S(1)=S(2)
148 C
149 700 CONTINUE
150 C
151 C***** SIMILARLY, REDEFINE POINT I+1 AS POINT I.
152 C
153 DO 301 K=1,3
154 R(2,K)=R(3,K)
155 N1(2,K)=N1(3,K)
156 N2(2,K)=N2(3,K)
157 301 CONTINUE
158 S(2)=S(3)
159 C
160 100 IF (IFLAG.NE.3) GOTO 150
161 C
162 C** END WHILE
163 C
164 C
165 C***** CLOSE INPUT AND OUTPUT FILES.
166 C
167 CALL SHUT(ICAD)
168 CALL SHUT(IOUT)
169 C
170 RETURN
171 END
```

Listing of RDFILE.FOR at 14:41:35 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE RDFILE(IUNIT,IRECNO,NAME,LEN,IEND)
2      C
3      C      SUBROUTINE RDFILE READS RECORD NUMBER 'IRECNO' OF A FILE OPEN
4      C      ON LOGICAL UNIT 'IUNIT'. THE RECORD IS RETURNED AS A
5      C      CHARACTER STRING 'NAME' OF LENGTH 'LEN'.
6      C
7      C      INPUTS: IUNIT- LOGICAL UNIT NUMBER OF OPEN FILE
8      C      IRECNO- RECORD NUMBER TO BE READ
9      C
10     C      OUTPUTS: NAME- BYTE ARRAY OF LENGTH 'LEN' CONTAINING CONTENTS
11     C      OF RECORD 'IRECNO'. IE. BYTE NAME(LEN)
12     C      LEN- LENGTH OF BYTE ARRAY.
13     C      IEND- FLAG INDICATING END OF FILE ENCOUNTERED DURING
14     C      READ.
15     C      IEND=0   READ SUCCESSFUL
16     C      IEND=1   END OF FILE ENCOUNTERED
17     C
18     C
19     C      FUNCTIONS CALLED:      LENGTH- FINDS THE LENGTH OF A CHARACTER
20     C      STRING
21     C
22     C
23     C*****
24     C
25     C      BYTE NAME(LEN)
26     C
27     C      IEND=0
28     C
29     C** IF (IRECNO .GT. 1)
30     C
31     C      IF (IRECNO .EQ. 1) GO TO 100
32     C
33     C***** IF (IRECNO.GT.1) REWIND FILE, STEP FORWARD TO RECORD 'IRECNO'
34     C***** AND READ CONTENTS.
35     C
36     C      REWIND IUNIT
37     C      DO 300 I=1,IRECNO-1
38     C          READ(IUNIT,150,END=400)
39     C      300 CONTINUE
40     C      READ(IUNIT,150,END=400) (NAME(J),J=1,LEN)
41     C      GO TO 200
42     C
43     C** ELSE
44     C
45     C      100 CONTINUE
46     C
47     C***** IF FIRST RECORD IS TO BE READ, REWIND FILE AND READ CONTENTS
48     C***** OF FIRST RECORD.
49     C
50     C      REWIND IUNIT
51     C      READ(IUNIT,150,END=400) (NAME(J),J=1,LEN)
52     C
53     C** END IF
54     C
55     C      200 CONTINUE
56     C
57     C      150 FORMAT(80A1)
58     C
```

Listing of RDFILE.FOR at 14:41:35 on AUG 9, 1984 for CCid=DACO Page 2

```
59      C***** FIND LENGTH OF CHARACTER STRING AND PAD TO RIGHT WITH ASCII
60      C***** ZEROS.
61      C
62          L=LENGTH(NAME,LEN)
63          NAME(L+1)=0
64      C
65          GO TO 500
66      C
67      C***** END OF FILE ENCOUNTERED. SET FLAG 'IEND'.
68      C
69      400      CONTINUE
70              IEND=1
71      C
72      500      CONTINUE
73              RETURN
74              END
```

Listing of ROTAT.FOR at 14:41:35 on AUG 9, 1984 for CCId=DAC0 Page 1

```
1          SUBROUTINE ROTAT(H,A,B)
2      C
3      C      THIS SUBROUTINE PERFORMS THE ROTATIONAL PART OF A
4      C      HOMOGENEOUS TRANSFORMATION ,B=H*A, WHERE H(4,4) IS A
5      C      HOMOGENEOUS TRANSFORMATION MATRIX AND A(3),B(3) ARE VECTORS.
6      C
7      C      INPUTS: H(4,4)- HOMOGENEOUS TRANSFORMATION MATRIX
8      C      A(3) - VECTOR
9      C
10     C      OUTPUTS: B(3) - VECTOR
11     C
12     C*****
13     C
14     C      REAL H(4,4),A(3),B(3)
15     C
16     C      B(1)=H(1,1)*A(1)+H(1,2)*A(2)+H(1,3)*A(3)
17     C      B(2)=H(2,1)*A(1)+H(2,2)*A(2)+H(2,3)*A(3)
18     C      B(3)=H(3,1)*A(1)+H(3,2)*A(2)+H(3,3)*A(3)
19     C
20     C      RETURN
21     C      END
```

Listing of SCENE.FOR at 14:41:35 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE SCENE
2      C
3      C      THIS SUBROUTINE DRAWS THE WORKSTATION ,OBJECT AND SEAMS.
4      C
5      C      INPUTS: NONE
6      C
7      C      OUTPUTS: NONE
8      C
9      C      COMMON BLOCKS ACCESSED: /VIEW/,/MATRIX/
10     C
11     C      COMMON BLOCKS CHANGED: /MATRIX/
12     C
13     C      FILES READ OR WRITTEN: NONE
14     C
15     C      SUBROUTINES CALLED:      DRAXES
16     C                               DRPANL
17     C                               DRSEAM
18     C                               TABPOS
19     C                               TABSET
20     C
21     C      'IGL' GRAPHICS SUBROUTINES:  CMCLOS
22     C                                     CMOPEN
23     C                                     DELSEG
24     C                                     EYEBAL
25     C                                     NEWPAG
26     C                                     VRP3D
27     C
28     C
29     C*****
30     C
31     C      BYTE BEL(3),YES,NO
32     C
33     C      REAL VRP(3),EYEPOS(3),VAR(6),SHOT(4,4),SHOW(4,4),H(4,4)
34     C      REAL VERT(1000),FACES(100)
35     C      DOUBLE PRECISION HOT(4,4),HOW(4,4)
36     C
37     C      INTEGER IOBJ(10,2),IFACE(100,2),IDIR(1),IPICK(100)
38     C
39     C      DATA BEL/3*7/,YES/89/,NO/78/
40     C
41     C      COMMON /DATA/VERT,FACES,IFACE,IOBJ
42     C      COMMON /VIEW/EYEPOS,VRP
43     C      COMMON /MATRIX/HOT,HOW,SHOT,SHOW
44     C
45     C***** CLOSE IGL COMMUNICATIONS
46     C
47     C      CALL CMCLOS
48     C
49     C***** SET WORKTABLE PARAMETERS
50     C
51     C      CALL TABSET(VAR)
52     C
53     C***** CALCULATE RESULTING HOMOGENEOUS TRANSFORMATIONS 'HOT','HOW'.
54     C
55     C      CALL TABPOS(VAR,HOT,HOW)
56     C
57     C***** CONVERT DOUBLE PRECISION 'HOT','HOW', INTO SINGLE PRECISION
58     C***** 'SHOT', 'SHOW'.
```

Listing of SCENE.FOR at 14:41:35 on AUG 9, 1984 for CCid=DAC0 Page 2

```
59      C
60          CALL MATCHNV(HOT,SHOT,1)
61          CALL MATCHNV(HOW,SHOW,1)
62      C
63      C***** OPEN IGL COMMUNICATIONS
64      C
65          CALL CMOPEN
66      C
67      C***** DELETE ALL SEGMENTS AND CLEAR SCREEN
68      C
69          CALL DELSEG(-1)
70          CALL NEWPAG
71      C
72      C***** SET VIEW REFERENCE POINT 'VRP', EYE POSITION 'EYEPOS'
73      C
74          CALL VRP3D(VRP(1),VRP(2),VRP(3))
75          CALL EYEBAL(EYEPOS(1),EYEPOS(2),EYEPOS(3))
76      C
77      C***** DRAW WORLD COORDINATE FRAME AXES
78      C
79          CALL DRAXES
80      C
81      C***** DRAW OBJECT AS A POLYHEDRON
82      C
83          CALL DRPANL(SHOT,1)
84      C
85      C***** DRAW THE SEAMS
86      C
87          CALL DRSEAM(SHOW)
88      C
89          RETURN
90      END
```

Listing of SETUP.FOR at 14:42:07 on AUG 9, 1984 for CCid=DACO Page 1

```

1          SUBROUTINE SETUP
2      C
3      C      THIS SUBROUTINE PROCESSES THE CAD SEAM DATA
4      C
5      C
6      C
7      C      FILES READ:          FILIST
8      C                          CADLST
9      C                          CAD
10     C                          WPCLST
11     C
12     C      FILES WRITTEN:  WPCDAT
13     C
14     C      SUBROUTINES:      OPEN
15     C                          PROCES
16     C                          RDFILE
17     C                          SHUT
18     C
19     C
20     C*****
21     C
22     C
23     C      BYTE CAD(20), WPCDAT(20),CADLST(20)
24     C      BYTE WPCLST(20),FILIST(20)
25     C
26     C      COMMON /FILIST/FILIST
27     C      COMMON /CADLST/CADLST
28     C      COMMON /WPCLST/WPCLST
29     C      COMMON /CAD/CAD
30     C      COMMON /WPCDAT/WPCDAT
31     C
32     C***** OPEN CAD SEAM DATA FILE LIST, 'CADLST'
33     C
34     C      CALL OPEN(1,CADLST,4,1)
35     C
36     C***** OPEN PROCESSED SEAM DATA FILE LIST, 'WPCLST'
37     C
38     C      CALL OPEN(2,WPCLST,4,1)
39     C
40     C***** SKIP FIRST RECORD, THEN READ THE NUMBER OF SEAMS 'NSEAMS' FROM
41     C***** 'CADLST'
42     C
43     C      READ(1,*)
44     C      READ(1,*) NSEAMS
45     C
46     C***** REPEAT FOR ALL SEAMS...
47     C
48     C      DO 110 I=1,NSEAMS
49     C
50     C*****      FIND 'CAD','WPCDAT' FILES FOR SEAM I
51     C
52     C      CALL RDFILE(1,I+2,CAD,20,IEND)
53     C      CALL RDFILE(2,I+2,WPCDAT,20,IEND)
54     C
55     C*****      CLOSE LOGIC UNITS 1,2
56     C
57     C      CALL SHUT(1)
58     C      CALL SHUT(2)
```

Listing of SETUP.FOR at 14:42:07 on AUG 9, 1984 for CCId=DAC0 Page 2

```
59      C
60      C*****      PROCESS 'CAD' SEAM DATA AND GENERATE 'WPCDAT' SEAM DATA
61      C
62      CALL PROCES
63      C
64      C*****      REOPEN LOGIC UNITS 1,2
65      C
66      CALL OPEN(1,CADLST,4,1)
67      CALL OPEN(2,WPCLST,4,1)
68      C
69      110      CONTINUE
70      C
71      C
72      CALL SHUT(1)
73      CALL SHUT(2)
74      C
75      RETURN
76      END
```

Listing of SHUT.FOR at 14:42:07 on AUG 9, 1984 for CCId=DACD Page 1

```
1          SUBROUTINE SHUT(IUNIT)
2      C
3      C          THIS SUBROUTINE CLOSSES LOGIC UNIT IUNIT.
4      C
5      C*****
6      C
7          CLOSE(UNIT=IUNIT)
8      C
9          RETURN
10         END
```

Listing of TABPOS.FOR at 14:42:08 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE TABPOS(VAR,HOT,HOW)
2      C
3      C      THIS SUBROUTINE CALCULATES THE TABLE COORDINATE TO WORLD
4      C      COORDINATE TRANSFORMATION MATRIX, 'HOT'. THE FIXED TABLE
5      C      PARAMETERS ARE READ FROM FILE 'TABLE'. TABLE POSITION
6      C      VARIABLES, 'VAR(I)', ARE PASSED FROM THE MAIN PROGRAM. THESE
7      C      VARIABLES ARE ANGLES AND/OR DISPLACEMENTS, DEPENDING ON THE
8      C      TABLE DESIGN. THE PROGRAM CALCULATES THE TABLE TRANSFORMATION,
9      C      RELATIVE TO THE FIRST JOINT FRAME. THIS TRANSFORMATION IS THEN
10     C      PREMULIPLIED BY A TRANSFORMATION 'HPRE' FROM THE WORLD FRAME
11     C      TO THE FIRST JOINT FRAME AND THE RESULT IS POSTMULTIPLIED BY
12     C      A TRANSFORMATION 'HPOST' FROM THE LAST JOINT FRAME TO THE
13     C      TABLE REFERENCE FRAME.
14     C
15     C      INPUTS:
16     C          VAR(6)- ARRAY OF TABLE JOINT COORDINATES
17     C                  IE. ANGLES (DEG) AND DISPLACEMENTS
18     C      OUTPUTS:
19     C          HOT(4,4)- HOMOGENEOUS TRANSFORMATION MATRIX FROM
20     C                     TABLE FRAME 'FT' TO WORLD FRAME 'FO'
21     C          HOW(4,4)- HOMOGENEOUS TRANSFORMATION MATRIX FROM
22     C                     WORKPIECE FRAME 'FW' TO WORLD FRAME 'FO'
23     C
24     C      COMMON BLOCKS ACCESSED: /IOFILE/
25     C
26     C      FILES READ:
27     C
28     C          STOVAR- CONTAINS CURRENT 'VAR' AND CURRENT 'HOT'
29     C          TABLE- FILE CONTAINING WORK TABLE SPECIFICATIONS
30     C
31     C      *      *      *      *      *      *      *      *
32     C      FORMAT OF TABLE: (LIST-DIRECTED FORMAT)
33     C
34     C      RECORD  JOINT  CONTENTS      DESCRIPTION
35     C      NO.    NO.
36     C      1              NUMDOF        NUMBER OF DEGREE; OF FREEDOM
37     C
38     C      2              TEXT HEADING
39     C
40     C      3              1      ID,ITYPE,STEP.  ID-JOINT TYPE, 1=ROTARY
41     C                                     2=PRISMATIC
42     C                                     ITYPE- 1=DISCRETE 2=CONTINUOUS
43     C                                     ALPHA,DIST,LEN  STEP- STEP SIZE FOR DISCRETE
44     C                                                         POSITIONS
45     C                                     RMIN,RMAX- MINIMUM,MAXIMUM
46     C                                                         LIMITS OF JOINT MOTION
47     C                                     ALPHA- TWIST ANGLE
48     C                                     DISP- JOINT DISPLACEMENT
49     C                                     LEN- JOINT LENGTH
50     C
51     C      4              2      AS ABOVE, FOR
52     C                          JOINT 2
53     C
54     C      5              3      JOINT 3      ''
55     C
56     C      6              4      JOINT 4      ''
57     C
58     C      7              5      JOINT 5      ''
```

Listing of TABPOS.FOR at 14:42:08 on AUG 9, 1984 for CCId=DACO Page 2

```

59      C
60      C      8      6      JOINT 6
61      C
62      C      9      TEXT HEADING
63      C
64      C      10     N/A     HTW(1,J),J=1,4  FIRST ROW OF WORKPIECE FRAME
65      C                                     TO TABLE FRAME TRANSFORMATION
66      C                                     MATRIX 'HTW'
67      C
68      C      11     N/A     HTW(2,J),J=1,4  SECOND ROW OF 'HTW'
69      C
70      C      12     N/A     HTW(3,J),J=1,4  THIRD ROW OF 'HTW'
71      C
72      C      13     N/A     HTW(4,J),J=1,4  FOURTH ROW OF 'HTW'
73      C
74      C      14     TEXT HEADING
75      C
76      C      15     N/A     HPRE(1,J),J=1,4  FIRST ROW OF FIRST JOINT TO
77      C                                     WORLD FRAME TRANSFORMATION
78      C                                     MATRIX 'HPRE'
79      C
80      C      16     N/A     HPRE(2,J),J=1,4  SECOND ROW OF 'HPRE'
81      C
82      C      17     N/A     HPRE(3,J),J=1,4  THIRD ROW OF 'HPRE'
83      C
84      C      18     N/A     HPRE(4,J),J=1,4  FOURTH ROW OF 'HPRE'
85      C
86      C      19     N/A     TEXT HEADING
87      C
88      C      20     N/A     HPOST(1,J),J=1,4 FIRST ROW OF TABLE FRAME TO
89      C                                     LAST JOINT TRANSFORMATION
90      C                                     MATRIX 'HPOST'
91      C
92      C      21     N/A     HPOST(2,J),J=1,4 SECOND ROW OF 'HPOST'
93      C
94      C      22     N/A     HPOST(3,J),J=1,4 THIRD ROW OF 'HPOST'
95      C
96      C      23     N/A     HPOST(4,J),J=1,4 FOURTH ROW OF 'HPOST'
97      C
98      C
99      C      FORMAT OF 'STOVAR'
100     C
101     C      RECORD  CONTENTS      DESCRIPTION
102     C      NO.
103     C      1      TEXT HEADING
104     C      2      VAR(1)          CURRENT POSITION OF JOINT 1
105     C      3      VAR(2)          CURRENT POSITION OF JOINT 2
106     C
107     C      7      VAR(6)          CURRENT POSITION OF JOINT 6
108     C
109     C
110     C      C*****
111     C
112     C      IMPLICIT DOUBLE PRECISION (D)
113     C      BYTE TABLE(20)
114     C      DOUBLE PRECISION HA(4,4),HB(4,4),HOT(4,4),HPRE(4,4),
115     C      * HPOST(4,4),HTEMP(4,4),DTHETA(6),DALPHA(6),HTW(4,4),HOW(4,4)
116     C      REAL STEP(6),RMIN(6),RMAX(6),THETA(6),ALPHA(6),DISP(6),LEN(6)

```

Listing of TABPOS.FOR at 14:42:08 on AUG 9, 1984 for CCId=DACO Page 3

```
117      REAL VAR(6)
118      INTEGER ID(6),ITYPE(6)
119      C
120      COMMON /TABLE/TABLE
121      C
122      C
123      C***** OPEN THE TABLE SPECIFICATION FILE 'TABLE'
124      C
125      CALL OPEN(1,TABLE,4,1)
126      C
127      C***** READ THE NUMBER OF DEGREES OF FREEDOM, 'NUMDOF'.
128      C
129      READ(1,*) NUMDOF
130      C
131      C***** SKIP A RECORD, THEN READ THE TABLE LINK PARAMETERS
132      C
133      READ(1,*)
134      DO 123 I=1,NUMDOF
135          READ(1,*) ID(I),ITYPE(I),STEP(I),RMIN(I),RMAX(I),THETA(I),
136          *      ALPHA(I),DISP(I),LEN(I)
137      123 CONTINUE
138      C
139      C***** SKIP A RECORD, THEN READ THE TRANSFORMATION MATRIX 'HTW'
140      C
141      READ(1,*)
142      DO 131 I=1,4
143          READ(1,*) (HTW(I,J),J=1,4)
144      131 CONTINUE
145      C
146      C***** SKIP A RECORD, THEN READ THE TRANSFORMATION MATRIX 'HPRE'
147      C
148      READ(1,*)
149      DO 124 I=1,4
150          READ(1,*) (HPRE(I,J),J=1,4)
151      124 CONTINUE
152      C
153      C***** SKIP A RECORD, THEN READ THE TRANSFORMATION MATRIX 'HPOST'
154      C
155      READ(1,*)
156      DO 125 I=1,4
157          READ(1,*) (HPOST(I,J),J=1,4)
158      125 CONTINUE
159      C
160      CALL SHUT(1)
161      C
162      C***** 'DCONV' IS A CONVERSION FACTOR FROM DEGREES TO RADIANS
163      C
164      DCONV=4.DO*DATAN(1.DO)/180.
165      C
166      C** WHILE (I .LE. NUMDOF)
167      C
168          DO 400 I=1,NUMDOF
169              IF(ID(I).EQ.1) VAR(I)=VAR(I)*DCONV
170              DTHETA(I)=THETA(I)*DCONV
171              DALPHA(I)=ALPHA(I)*DCONV
172      C
173      C** IF THE JOINT IS REVOLUTE (ROTATIONAL)
174      C
```

Listing of TABPOS.FOR at 14:42:08 on AUG 9, 1984 for CCid=DACO Page 4

```
175          IF(ID(I).NE.1) GO TO 100
176      C
177      C*****      CALCULATE THE DENAVIT-HARTENBERG TRANSFORMATION MATRIX
178      C*****      RELATING LINK I TO LINK I-1
179      C
180          HA(1,1)=COS(VAR(I))
181          HA(1,2)=-SIN(VAR(I))*COS(DALPHA(I))
182          HA(1,3)=SIN(VAR(I))*SIN(DALPHA(I))
183          HA(1,4)=LEN(I)*COS(VAR(I))
184          HA(2,1)=SIN(VAR(I))
185          HA(2,2)=COS(VAR(I))*COS(DALPHA(I))
186          HA(2,3)=-COS(VAR(I))*SIN(DALPHA(I))
187          HA(2,4)=LEN(I)*SIN(VAR(I))
188          HA(3,1)=0.
189          HA(3,2)=SIN(DALPHA(I))
190          HA(3,3)=COS(DALPHA(I))
191          HA(3,4)=DISP(I)
192          HA(4,1)=0.
193          HA(4,2)=0.
194          HA(4,3)=0.
195          HA(4,4)=1.
196      C
197          GO TO 200
198      C
199      C**      ELSE THE JOINT IS PRISMATIC (TRANSLATIONAL)
200      C
201      100      CONTINUE
202      C
203      C*****      CALCULATE THE DENAVIT-HARTENBERG TRANSFORMATION MATRIX
204      C*****      RELATING LINK I TO LINK I-1
205      C
206          HA(1,1)=COS(DTHETA(I))
207          HA(1,2)=-SIN(DTHETA(I))*COS(DALPHA(I))
208          HA(1,3)=SIN(DTHETA(I))*SIN(DALPHA(I))
209          HA(1,4)=LEN(I)*COS(DTHETA(I))
210          HA(2,1)=SIN(DTHETA(I))
211          HA(2,2)=COS(DTHETA(I))*COS(DALPHA(I))
212          HA(2,3)=-COS(DTHETA(I))*SIN(DALPHA(I))
213          HA(2,4)=LEN(I)*SIN(DTHETA(I))
214          HA(3,1)=0.
215          HA(3,2)=SIN(DALPHA(I))
216          HA(3,3)=COS(DALPHA(I))
217          HA(3,4)=DISP(I)+VAR(I)
218          HA(4,1)=0.
219          HA(4,2)=0.
220          HA(4,3)=0.
221          HA(4,4)=1.
222      C
223      200      CONTINUE
224      C
225      C**      ENDIF
226      C
227      C
228      C**      IF (I.EQ.1)
229      C
230          IF(I.GT.1) GO TO 300
231      C
232          DO 101 J=1,4
```

Listing of TABPOS.FOR at 14:42:08 on AUG 9, 1984 for CCId=DACO Page 5

```
233          DO 102 K=1,4
234             HB(J,K)=HA(J,K)
235     102      CONTINUE
236     101      CONTINUE
237     C
238          GO TO 500
239     C
240 C**  ELSE
241     C
242     300      CONTINUE
243     C
244 C*****  MULTIPLY INTERMEDIATE TRANSFORMATION MATRICES 'HA','HB'
245     C
246          CALL HMULT(HB,HA,HOT)
247     C
248 C*****  THEN LET 'HB' := 'HOT'
249     C
250          DO 666 J=1,4
251             DO 777 K=1,4
252                HB(J,K)=HOT(J,K)
253     777      CONTINUE
254     666      CONTINUE
255     C
256     500      CONTINUE
257     C
258 C**  ENDIF
259     C
260     400      CONTINUE
261     C
262 C**  END WHILE
263     C
264     C
265 C*****  PREMULIPLY BY 'HPRE' AND POSTMULTIPLY BY 'HPOST' TO GET THE
266 C*****  REQUIRED TRANSFORMATION MATRIX 'HOW'
267     C
268          CALL HMULT(HPRE,HOT,HTEMP)
269          CALL HMULT(HTEMP,HPOST,HOT)
270          CALL HMULT(HOT,HTW,HOW)
271     C
272          RETURN
273          END
```

Listing of TABSET.FOR at 14:42:09 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE TABSET(VAR)
2      C
3      C      THIS SUBROUTINE ALLOWS THE USER TO INTERACTIVELY MODIFY THE
4      C      WORK TABLE DESCRIPTION FILE 'TABLE'.  THE USER IS PROMPTED FOR
5      C      CHANGES IN THE WORK TABLE POSITION.  THE FIXED WORK TABLE
6      C      PARAMETERS ARE INITIALLY ENTERED INTO 'TABLE' USING THE FILE
7      C      EDITOR, AND ARE NOT CHANGED CONSEQUENTLY.
8      C
9      C      VARIABLE TABLE PARAMETERS, JOINT VARIABLES 'VAR' ARE STORED IN
10     C      FILE 'STOVAR'.
11     C
12     C      INPUTS:          NONE
13     C
14     C      OUTPUTS:          VAR(6)- TABLE JOINT VARIABLES
15     C
16     C      COMMON BLOCKS READ:      /TABLE/
17     C                          /STOVAR/
18     C
19     C      FILES READ:      TABLE,STOVAR
20     C
21     C      FILES WRITTEN:  STOVAR
22     C
23     C      SUBROUTINES CALLED:      OPEN,SHUT
24     C
25     C*****
26     C
27     C      BYTE YCAP,YLOW,NCAP,NLOW,ANS,LABEL(10)
28     C      BYTE TABLE(20),STOVAR(20)
29     C      BYTE BELL
30     C
31     C      REAL VAR(6),STEP(6),RMIN(6),RMAX(6),ALPHA(6),DISP(6),RLEN(6)
32     C      DOUBLE PRECISION HOT(4,4),HPRE(4,4),HPOST(4,4)
33     C      INTEGER ITYPE(6),ID(6)
34     C
35     C      COMMON /TABLE/TABLE
36     C      COMMON /STOVAR/STOVAR
37     C
38     C      DATA BELL/7/
39     C      DATA YCAP/89/,YLOW/121/,NCAP/78/,NLOW/110/
40     C
41     C***** READ AND SAVE CONTENTS OF FILE 'TABLE'
42     C***** NOTE THAT RECORDS MUST BE READ IN CORRECT ORDER
43     C
44     C      CALL OPEN(1,TABLE,4,1)
45     C
46     C***** READ RECORD 1, NUMBER OF DEGREES OF FREEDOM, 'NUMDOF'
47     C
48     C      READ(1,*) NUMDOF
49     C
50     C***** SKIP RECORD 2
51     C
52     C      READ(1,*)
53     C
54     C***** READ RECORDS 3 TO 8, FIXED TABLE PARAMETERS
55     C
56     C      DO 170 J=1,NUMDOF
57     C          READ(1,*) ID(J),ITYPE(J),STEP(J),RMIN(J),RMAX(J),ALPHA(J),
58     C          *      DISP(J),RLEN(J)

```

Listing of TABSET.FOR at 14:42:09 on AUG 9, 1984 for CCid=DACO Page 2

```
59      170      CONTINUE
60      C
61      C***** SKIP RECORD 9
62      C
63      READ(1,*)
64      C
65      C***** SKIP RECORDS 10 TO 13, TRANSFORMATION MATRIX 'HTW' (NOT USED)
66      C
67      DO 175 J=1,4
68      READ(1,*)
69      175      CONTINUE
70      C
71      C***** SKIP RECORD 14
72      C
73      READ(1,*)
74      C
75      C***** READ RECORDS 15 TO 18, TRANSFORMATION MATRIX 'HPRE'
76      C
77      DO 180 J=1,4
78      READ(1,*) (HPRE(J,K),K=1,4)
79      180      CONTINUE
80      C
81      C***** SKIP RECORD 19
82      C
83      READ(1,*)
84      C
85      C***** READ RECORDS 20 TO 23, TRANSFORMATION MATRIX 'HPOST'
86      C
87      DO 185 J=1,4
88      READ(1,*) (HPOST(J,K),K=1,4)
89      185      CONTINUE
90      C
91      CALL SHUT(1)
92      C
93      C***** READ CURRENT VALUES OF JOINT VARIABLES 'VAR' FROM CURRENT
94      C***** FILE 'STOVAR'.
95      C
96      CALL OPEN(1,STOVAR,4,1)
97      C
98      C***** SKIP RECORD 1
99      C
100     READ(1,*)
101     C
102     C***** READ RECORDS 2,NUMDOF+1
103     C
104     DO 190 J=1,NUMDOF
105     READ(1,*) VAR(J)
106     190      CONTINUE
107     C
108     CALL SHUT(1)
109     C
110     C** UNTIL (ANSWER .NE. YES)
111     C
112     250      CONTINUE
113     C
114     C***** WRITE THE CURRENT TABLE JOINT POSITIONS ONTO THE SCREEN, AND
115     C***** PROMPT FOR CHANGES
116     C
```

Listing of TABSET.FOR at 14:42:09 on AUG 9, 1984 for CCId=DACD Page 3

```
117      WRITE(5,230) (VAR(J),J=1,3)
118      230      FORMAT(' JOINT VAR(1)=' ,F8.2, ' JOINT VAR(2)=' ,F8.2,
119      *      ' JOINT VAR(3)=' ,F8.2)
120      C
121      WRITE(5,100)
122      100      FORMAT($, ' DO YOU WISH TO CHANGE THE TABLE POSITION? (Y/N) ')
123      READ(5,301) ANS
124      301      FORMAT(A1)
125      C
126      C** IF (ANS .EQ. YES) CONTINUE
127      C
128      IF(ANS.EQ.YCAP.OR.ANS.EQ.YLOW) GO TO 500
129      C
130      C** ELSE GO TO END
131      C
132      GO TO 650
133      C
134      500      CONTINUE
135      C
136      C** END IF
137      C
138      WRITE(5,101)
139      101      FORMAT($, ' ENTER JOINT NUMBER YOU WISH TO CHANGE (INTEGER) ')
140      READ(5,*) JOINT
141      C
142      C** IF (JOINT NOT VALID) PRINT MESSAGE, TRY AGAIN
143      C
144      IF (JOINT.LT.1.OR.JOINT.GT.NUMDOF) GO TO 701
145      C
146      C** END IF
147      C
148      WRITE(5,102)
149      102      FORMAT($, ' ENTER THE NEW POSITION (DEG. OR MM., REAL) ')
150      READ(5,*) VAR(JOINT)
151      C
152      C** IF (VARIABLE OUT OF RANGE) PRINT MESSAGE, TRY AGAIN
153      C
154      IF(VAR(JOINT).LT.RMIN(JOINT).OR.VAR(JOINT).GT.RMAX(JOINT))
155      * GO TO 800
156      C
157      C** END IF
158      C
159      C** IF (JOINT TYPE IS DISCRETE) TEST FOR VALID VALUE OF 'VAR'
160      C
161      IF(ITYPE(JOINT).NE.1) GO TO 850
162      C
163      TEMP=VAR(JOINT)/STEP(JOINT)
164      ITEMP=INT(TEMP)
165      REM=ABS(TEMP-FLOAT(ITEMP))
166      C
167      IF(REM.GT..001) GO TO 900
168      C
169      850      CONTINUE
170      C
171      C** END IF
172      C
173      GO TO 250
174      C
```

Listing of TABSET.FOR at 14:42:09 on AUG 9, 1984 for CCId=DACO Page 4

```
175      C**  END UNTIL
176      C
177      650      CONTINUE
178      C
179      C*****  STORE NEW VALUES OF JOINT VARIABLES 'VAR' IN FILE 'STOVAR'.
180      C
181              CALL OPEN(1,STOVAR,4,1)
182              READ(1,*)
183              DO 123 J=1,NUMDOF
184                  WRITE(1,*) VAR(J)
185      123      CONTINUE
186              CALL SHUT(1)
187      C
188      C*****  GO TO END
189      C
190              GO TO 600
191      C
192      C
193      C*****  ERROR MESSAGES
194      C
195      C
196      800      WRITE(5,106) BELL
197      106      FORMAT($,' POSITION ENTERED OUTSIDE OF RANGE-TRY AGAIN ',A1)
198              READ(5,*)
199              GO TO 250
200      C
201      900      WRITE(5,107) BELL
202      107      FORMAT($,' POSITION ENTERED NOT ONE OF THE POSSIBLE VALUES-TRY
203              * AGAIN',A1)
204              READ(5,*)
205              GO TO 250
206      C
207      701      WRITE(5,108) BELL
208      108      FORMAT($,' JOINT NUMBER NOT VALID-TRY AGAIN ',A1)
209              READ(5,*)
210              GO TO 250
211      C
212      600      CONTINUE
213              RETURN
214              END
```

Listing of TANGEN.FOR at 14:42:31 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE TANGEN(R,S,T,IFLAG)
2      C
3      C      THIS SUBROUTINE CALCULATES TANGENT OF CURVE, T.
4      C
5      C      INPUTS:
6      C          R(3,3)=POSITION VECTORS FOR POINTS I-1, I, I+1.
7      C          S(3)=ARC LENGTHS AT POINTS I-1, I, I+1.
8      C          IFLAG=CONDITION FLAG WHERE
9      C              I=1...SECOND POINT ON CURVE.
10     C              I=2...INTERMEDIATE POINTS.
11     C              I=3...LAST POINT ON CURVE.
12     C
13     C      OUTPUTS:
14     C          T(3)=TANGENT VECTOR AT POINT I.
15     C
16     C*****
17     C
18     C
19     C      REAL R(3,3), S(3), T(3)
20     C
21     C** CASE OF (IFLAG)
22     C
23     C      IF (IFLAG.LT.1.OR.IFLAG.GT.3) GO TO 400
24     C      GO TO (100,200,300), IFLAG
25     C
26     C** CASE (1)
27     C
28     100     CONTINUE
29     C      DO 101 J=1,3
30     C          T(J)=(R(3,J)-R(2,J))/S(3)
31     101     CONTINUE
32     C      GO TO 500
33     C
34     C** CASE (2)
35     C
36     200     CONTINUE
37     C      DO 102 J=1,3
38     C          D1=R(3,1)-R(1,1)
39     C          D2=R(3,2)-R(1,2)
40     C          D3=R(3,3)-R(1,3)
41     C          DS=SQRT(D1**2+D2**2+D3**2)
42     C          T(J)=(R(3,J)-R(1,J))/DS
43     102     CONTINUE
44     C      GO TO 500
45     C
46     C** CASE (3)
47     C
48     300     CONTINUE
49     C      DO 103 J=1,3
50     C          T(J)=(R(2,J)-R(1,J))/(S(2)-S(1))
51     103     CONTINUE
52     C      GO TO 500
53     C
54     C** CASE. ELSE
55     C
56     400     CONTINUE
57     C      WRITE(5,150)
58     150     FORMAT(' CASE INDEX SPECIFICATION ERROR.')
```

Listing of TANGEN.FOR at 14:42:31 on AUG 9, 1984 for CCid=DACD Page 2

59	C	
60	500	CONTINUE
61		RETURN
62		END

Listing of TOLROT.FOR at 14:42:32 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE TOLROT(ROTOLD,ICOUNT,ROTNEW,INHAG)
2      C
3      C      THIS SUBROUTINE CALCULATES AN INVERSE ROTATION MATRIX
4      C      FOR A ROTATION 'ROTNEW' ABOUT THE TOOL AXIS.
5      C      THE TOOL IS ROTATED ALTERNATELY BY AN ANGLE OF ICOUNT*DELTA
6      C      IN THE POSITIVE AND NEGATIVE DIRECTIONS.
7      C
8      C      INPUTS: ROTOLD- PREVIOUS ROTATION ANGLE
9      C      ICOUNT- NUMBER OF ROTATIONS
10     C
11     C      OUTPUTS: ROTNEW- NEW ROTATION ANGLE
12     C      INHAG(4,4)- INVERSE ROTATION MATRIX
13     C
14     C*****
15     C
16     REAL INHAG(4,4),DELTA
17     INTEGER IMAX
18     C
19     COMMON /TOLROT/IMAX,DELTA
20     C
21     C***** FIND THE NEW ROTATION ANGLE
22     C
23     ROTNEW=ROTOLD+FLOAT((-1)**ICOUNT*ICOUNT/2)*DELTA
24     C
25     C***** CALCULATE THE ROTATION MATRIX RELATING THE ROTATED TOOL TO THE
26     C***** UNROTATED TOOL.
27     C
28     INHAG(1,1)=COS(ROTNEW)
29     INHAG(1,2)=SIN(ROTNEW)
30     INHAG(2,1)=-INHAG(1,2)
31     INHAG(2,2)=INHAG(1,1)
32     INHAG(3,3)=1.
33     INHAG(4,4)=1.
34     C
35     RETURN
36     END
```

Listing of TRANS.FOR at 14:42:32 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE TRANS(H,P1,P2)
2      C
3      C      THIS SUBROUTINE PERFORMS THE FOLLOWING HOMOGENEOUS
4      C      TRANSFORMATION.
5      C
6      C      P2(4)=H(4,4)*P1(4)
7      C
8      C      WHERE P1,P2 ARE 4X1 VECTORS AND H IS A 4X4 HOMOGENEOUS
9      C      COORDINATE TRANSFORMATION MATRIX.
10     C
11     C*****
12     C
13     REAL H(4,4),P1(4),P2(4)
14     C
15     DO 100 I=1,3
16         P2(I)=H(I,1)*P1(1)+H(I,2)*P1(2)+H(I,3)*P1(3)+H(I,4)
17     100 CONTINUE
18     C
19     P2(4)=1.
20     C
21     RETURN
22     END
```

Listing of UNIT.FOR at 14:42:33 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE UNIT(VECT,UVECT)
2      C
3      C      THIS SUBROUTINE NORMALIZES A VECTOR.
4      C
5      C      INPUTS: VECT(3)- A VECTOR
6      C
7      C      OUTPUTS: UVECT(3)- VECT NORMALIZED.
8      C
9      C*****
10     C
11     REAL VECT(3),UVECT(3),LENGTH
12     C
13     LENGTH=SQRT(VECT(1)**2+VECT(2)**2+VECT(3)**2)
14     C
15     UVECT(1)=VECT(1)/LENGTH
16     UVECT(2)=VECT(2)/LENGTH
17     UVECT(3)=VECT(3)/LENGTH
18     C
19     RETURN
20     END
```

Listing of WELSET.FOR at 14:42:59 on AUG 9, 1984 for CCId=DACD Page 1

```
1      SUBROUTINE WELSET(WPCLST,WKSTAT)
2      C
3      C      THIS SUBROUTINE SCANS THE WELD PARAMETERS JTYPE,XSECT,T1,T2,
4      C      FOR ALL SEAMS TO BE WELDED. THE CURRENT WELDING MACHINE
5      C      SETTINGS ARE READ FROM THE WELDER SETUP FILE. IF NECESSARY,
6      C      THE SETUP FILE IS MODIFIED AND THE OPERATOR IS INSTRUCTED TO
7      C      RESET WELDER.
8      C
9      C      INPUTS: WPCLST- NAME OF WORKPIECE SEAM DATA FILE
10     C              WKSTAT- NAME OF WORKSTATION DATA FILE
11     C
12     C      FILES READ: WPCLST
13     C                   WKSTAT
14     C
15     C      FILES WRITTEN: WKSTAT
16     C
17     C      SUBROUTINES:   OPEN
18     C                      RDFILE
19     C                      SHUT
20     C
21     C*****
22     C
23     C      BYTE WPCLST(20),WPCDAT(20),WKSTAT(20)
24     C      REAL WIRES(3,2),WRSIZE(3)
25     C
26     C      DATA WRSIZE/.030,.035,.045/
27     C      DATA WIRES/.025,.125,.037,.25,.125,.5/
28     C
29     C      COMMON /NSEAMS/NSEAMS
30     C
31     C      CALL OPEN(1,WPCLST,4,1)
32     C
33     C      TMIN=1000.
34     C      TMAX=0.
35     C
36     C      DO 100 I=1,NSEAMS
37     C          CALL RDFILE(1,I+2,WPCDAT,20,IEND)
38     C          CALL OPEN(2,WPCDAT,1,1)
39     C          READ(2) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
40     C          CALL SHUT(2)
41     C
42     C*****      PROCESS DATA HERE, SAVING MINIMUM,MAXIMUM VALUES ETC.
43     C*****      FOR SELECTION OF OPTIMUM WELDER SETTINGS.
44     C
45     C          TX=AMIN1(T1,T2)
46     C          TMIN=AMIN1(TMIN,TX)
47     C          TMAX=AMAX1(TMAX,TX)
48     C
49     C      100 CONTINUE
50     C
51     C
52     C      CALL SHUT(1)
53     C
54     C*****      OPEN CURRENT WELDER SETUP FILE 'WKSTAT'.
55     C
56     C      CALL OPEN(1,WKSTAT,4,1)
57     C
58     C*****      SKIP FIRST 26 RECORDS
```

Listing of WELSET.FOR at 14:42:59 on AUG 9, 1984 for CCId=DACO Page 2

```
59      C
60          DO 10 I=1,26
61              READ(1,*)
62      10      CONTINUE
63      C
64      C***** READ CURRENT WELDER SETUP FROM 'WKSTAT'.
65      C
66          READ(1,*) WSIZE
67      C
68      C***** INTERACTIVELY UPDATE WELDER SETUP IF REQUIRED. WHEN NEW
69      C***** PARAMETERS HAVE BEEN CHOSEN, UPDATE 'WKSTAT'
70      C
71          DO 20 J=1,3
72              IF(TMIN.GE.WIRES(J,1).AND.TMAX.LE.WIRES(J,2)) GO TO 500
73      20      CONTINUE
74      C
75          WRITE(5,30)
76      30      FORMAT(' NO APPROPRIATE WIRE SIZE CAN BE FOUND. HIT RETURN TO
77      * CONTINUE ')
78          READ(5,*)
79          GO TO 900
80      C
81      500      CONTINUE
82      C
83          IF(WSIZE.EQ.WRSIZE(J)) GO TO 600
84      C
85          BACKSPACE(1)
86          WRITE(1,*) WRSIZE(J)
87          WRITE(5,31) WSIZE,WRSIZE(J)
88      31      FORMAT(' CURRENT WIRE SIZE= ',F5.3,' IN., NEW WIRE SIZE= ',
89      * F5.3,' IN')
90          WRITE(5,33)
91      33      FORMAT(' INSTALL NEW WIRE, THEN HIT RETURN TO CONTINUE')
92          READ(5,*)
93          GO TO 900
94      C
95      600      CONTINUE
96      C
97          WRITE(5,32)
98      32      FORMAT(' CURRENT WIRE SIZE IS OK. HIT RETURN TO CONTINUE ')
99          READ(5,*)
100      C
101      900      CONTINUE
102      C
103          CALL SHUT(1)
104      C
105          RETURN
106          END
```

Listing of WIRE1.FOR at 14:42:59 on AUG 9, 1984 for CCId=DAC0 Page 1

```
1      SUBROUTINE WIRE1
2      C
3      C***** THIS SUBROUTINE DETERMINES CURRENT, VOLTAGE, WIRE FEED, WELDING
4      C***** WELDING SPEED, GAS FLOW RATE, WIRE STICK-OUT, LONGITUDINAL ANGLE
5      C***** AND TRANSVERSE ANGLE AS A FUNCTION OF THICKNESS AND CROSS
6      C***** SECTIONAL AREA.
7      C
8      C      DEFINITION OF VARIABLES:
9      C          CURRNT=CURRENT (AMPS)
10     C          VOLT=VOLTAGE (VOLTS)
11     C          FEED=WIRE FEED RATE (IPM)
12     C          GFR=GAS FLOW RATE (CFM)
13     C          SPEED=WELDING SPEED (IPM)
14     C          STKOUT=WIRE STICK-OUT (INCHES)
15     C          WALPHA=LONGITUDINAL TORCH ANGLE
16     C          BETA=TRANSVERSE TORCH ANGLE
17     C          TX=MINIMUM PARENT METAL THICKNESS
18     C          XSECT=WELD CROSS SECTIONAL AREA (SQ.IN.)
19     C
20     C      BYTE WPARAM(20)
21     C      COMMON/EXPERT/WPARAM,VOLT,WSIZE,FEED,GFR,WALPHA,BETA,STKOUT
22     C      COMMON/W123/CURRNT,SPEED,XSECT,TX
23     C
24     C      CURRNT=199.07+43.04*ALOG(TX)
25     C      VOLT=16.
26     C      FEED=290.38+52.87*ALOG(TX)
27     C      GFR=23.38+1.64*ALOG(TX)
28     C      SPEED=3.1415*(.030**2)*FEED/(XSECT)
29     C      STKOUT=.75+.25*ALOG(TX)
30     C      WALPHA=15.
31     C      BETA=0.0
32     C
33     C      RETURN
34     C      END
```

Listing of WIRE2.FOR at 14:42:59 on AUG 9, 1984 for CCId=DAC0 Page 1

```
1          SUBROUTINE WIRE2
2      C
3      C***** WIRE SIZE IS 0.035 INCHES
4      C***** THIS SUBROUTINE DETERMINES CURRENT, VOLTAGE, WIRE FEED,
5      C***** WELDING SPEED, GAS FLOW RATE, WIRE STICK-OUT, LONGITUDINAL
6      C***** ANGLE AND TRANSVERSE ANGLE AS A FUNCTION OF THICKNESS
7      C***** AND CROSS SECTIONAL AREA.
8      C
9      BYTE WPARAM(20)
10     COMMON/EXPERT/WPARAM,VOLT,WSIZE,FEED,GFR,WALPHA,BETA,STKOUT
11     COMMON/W123/CURRNT,SPEED,XSECT,TX
12     C
13     CURRNT=222.64+44.41*ALOG(TX)
14     IF(CURRNT.GT.150.) CURRNT=150.
15     VOLT=23.91+2.12*ALOG(TX)
16     FEED=435.77+101.51*ALOG(TX)
17     IF(FEED.GT.265.) FEED=265.
18     GFR=40.55+7.17*ALOG(TX)
19     SPEED=3.1415*(.035**2)*FEED/(XSECT*4.)
20     IF(GFR.GT.22.5) GFR=22.5
21     STKOUT=.75+.25*ALOG(TX)
22     WALPHA=15.
23     BETA=0.0
24     C
25     RETURN
26     END
```

Listing of WIRE3.FOR at 14:42:59 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE WIRE3
2      C
3      C***** WIRE SIZE IS 0.045 INCHES
4      C***** THIS SUBROUTINE DETERMINES CURRENT, VOLTAGE, WIRE FEED,
5      C***** WELDING SPEED, GAS FLOW RATE, WIRE STICK-OUT, LONGITUDINAL
6      C***** ANGLE AND TRANSVERSE ANGLE AS A FUNCTION OF THICKNESS
7      C***** AND CROSS SECTIONAL AREA.
8      C
9      BYTE WPARAM(20)
10     COMMON/EXPERT/WPARAM,VOLT,WSIZE,FEED,GFR,WALPHA,BETA,STKOUT
11     COMMON/W123/CURRNT,SPEED,XSECT,TX
12     C
13     CURRNT=207.88+8.72*ALOG(TX)
14     VOLT=22.
15     FEED=296.88+21.77*ALOG(TX)
16     GFR=22.5
17     SPEED=3.1415*(.045**2)*FEED/(XSECT*4.)
18     STKOUT=.75+.25*ALOG(TX)
19     WALPHA=15.
20     BETA=0.0
21     C
22     RETURN
23     END
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE WORPOS(WPCDAT,IDSEAM,IDIR,ISTAT)
2      C
3      C      THIS SUBROUTINE DOES THE COORDINATE TRANSFORMATIONS TO FIND
4      C      THE TOOL LOCATION DATA FOR THE TOOL COORDINATE FRAME FG RELATIVE
5      C      TO THE WORLD COORDINATE FRAME FO. THE SUBROUTINE FINDS THE
6      C      HOMOGENEOUS TRANSFORMATION HOG, AND STORES THE TOOL POSITION
7      C      AND ORIENTATION IN AN OUTPUT FILE.
8      C
9      C      INPUTS: IDSEAM- SEAM IDENTIFICATION NUMBER
10     C
11     C      IDIR- SEAM TRAVERSE DIRECTION
12     C      O=NORMAL DIRECTION
13     C      1=REVERSE DIRECTION
14     C
15     C      OUTPUTS: ISTAT- FEASIBILITY STATUS
16     C      O=TRAJECTORY IS FEASIBLE
17     C      1=ROBOT WORK ENVELOPE EXCEEDED
18     C      2=ROBOT JOINT LIMIT EXCEEDED
19     C      3=IDIR NOT 0 OR 1
20     C
21     C      INPUT FILES: TABLE - A FILE CONTAINING THE WORKPIECE AND
22     C      TABLE TRANSFORMATION MATRICES, HTW, HOT.
23     C
24     C      WPCNST - A FILE SPECIFYING NAME OF WPCDAT.
25     C
26     C      WPCDAT - A FILE CONTAINING THE
27     C      POSITION AND ORIENTATION DATA FOR THE
28     C      WELD FRAME FC, RELATIVE TO THE
29     C      WORKPIECE FRAME FW.
30     C
31     C      ROBLST - A FILE SPECIFYING NAME OF LOCDAT.
32     C
33     C      OUTPUT FILES: ROBDAT - AN EXTERNALLY SPECIFIED FILE CONTAINING
34     C      THE LOCATION AND ORIENTATION DATA OF
35     C      THE TOOL RELATIVE TO THE ROBOT
36     C      COORDINATE FRAME FO, THE WELDING
37     C      CURRENT AND SPEED SETTINGS, AND
38     C      DIRECTION VECTORS FOR NORMAL AND
39     C      LATERAL PATH CORRECTIONS.
40     C
41     C      LOCSET - CONTAINS, IN WELDING SEQUENCE, THE
42     C      FOLLOWING:
43     C
44     C      SEAM NUMBER
45     C      TABLE JOINT VARIABLES
46     C      ROBOT CONFIGURATION
47     C
48     C      SUBROUTINES CALLED - HMULT
49     C      HMULTS
50     C      JOINTS
51     C      LIMITS
52     C      MATCNV
53     C      NOAP1
54     C      NOAP5
55     C      OPEN
56     C      RDFILE
57     C      SHUT
58     C      TOLROT
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCId=DACD Page 2

```

59      C                                XPROD
60      C
61      C*****
62      C
63      C
64      IMPLICIT DOUBLE PRECISION (D)
65      DOUBLE PRECISION HOT(4,4),HTW(4,4),HWC(4,4),HCG(4,4)
66      DOUBLE PRECISION HOW(4,4),HOC(4,4),HOG(4,4)
67      REAL R(3),T(3),A(3),B(3),N(3),O(3),X(3),Y(3),Z(3).
68      * THETA(6),SHOW(4,4),SHOT(4,4),INHOO(4,4)
69      REAL XYZOAT(6),SHOG(4,4),INHAG(4,4),H6G(4,4),HOG(4,4),HOG(4,4)
70      REAL INH6A(4,4),HOA(4,4),DELTA,TABVAR(6),HOG5(4,4)
71      REAL ALPHA(6),LEN(6),DIST(6),RMIN(6),RMAX(6),VAR(6),HOO(4,4)
72      REAL SHOC(4,4),HOC(4,4),ENORM(3),ELAT(3)
73      INTEGER ITOOL(3),IOFF(2),IJOINT(6),ICONF(6),IMAX
74      BYTE WPCLST(20),STOVAR(20),ROBLST(20),LOCLST(20)
75      BYTE WPCDAT(20),LOC DAT(20),ROBDAT(20),LOCSET(20)
76      BYTE BELL(3),WPARAM(20)
77      C
78      COMMON /FILE/FILIST
79      COMMON /LOCSET/LOCSET
80      COMMON /ROBLST/ROBLST
81      COMMON /STOVAR/STOVAR
82      COMMON /WPCLST/WPCLST
83      COMMON /ROBOT/NRODF,ALPHA,LEN,DIST,RMIN,RMAX,HOO,H6G,INHOO,INH6A
84      COMMON /CONFIG/ICONF
85      COMMON /MATRIX/HOT,HOW,SHOT,SHOW
86      COMMON /TABDAT/HTW,SHTW
87      COMMON /TOLROT/IMAX,DELTA
88      COMMON /ROBPOS/HOG
89      COMMON /EXPERT/WPARAM,VOLT,WSIZE,FEED,GFR,WALPHA,BETA,STKOUT
90      C
91      DATA BELL/3*7/
92      C
93      C***** INITIALIZE PARAMETERS
94      C
95      ISOLN=0
96      ISTAT=0
97      ICOUNT=1
98      IFEAS=1
99      ROTOLD=0.
100     C
101     C
102     C***** FIND AND OPEN FILES 'WPCDAT','ROBDAT','LOCSET','WPARAM'
103     C
104     CALL OPEN(2,WPCDAT,1,1)
105     C
106     CALL OPEN(3,ROBLST,4,1)
107     CALL RDFILE(3,IDSEAM+1,ROBDAT,20,IEND)
108     CALL SHUT(3)
109     CALL OPEN(3,ROBDAT,4,1)
110     C
111     CALL OPEN(4,LOCSET,4,1)
112     C
113     CALL OPEN(7,WPARAM,4,1)
114     C
115     C
116     C***** READ FIRST RECORD OF 'WPCDAT', THEN BACKSPACE TO BEGINNING.

```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCid=DACO Page 3

```
117      C
118          READ(2) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
119          BACKSPACE 2
120      C
121      C
122      C***** STORE WELDING PARAMETERS 'WALPHA','BETA','STKOUT','VOLT',
123      C***** 'FEED','GFR' FOUND BY THE EXPERT WELDER MODULE, AND NUMBER OF
124      C***** LOCATIONS 'NPOINT' AT HEAD OF ROBOT LOCATION FILE 'ROBDAT' FOR
125      C***** CURRENT SEAM
126      C
127          WRITE(3,*) WALPHA,BETA,STKOUT,VOLT,FEED,GFR,NPOINT
128      C
129      C
130      C
131      C***** REPEAT UNTIL ALL CONFIGURATIONS HAVE BEEN TESTED OR UNTIL A
132      C***** FEASIBLE TRAJECTORY HAS BEEN FOUND
133      C
134      C
135      160      CONTINUE
136      C
137      C*****      READ FIRST RECORD OF 'WPCDAT'
138      C
139          READ(2) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
140      C
141      C*****      SET INDEX
142      C
143          INDEX=NPOINT
144      C
145      C
146      C***      IF (DIRECTION.EQ.REVERSE.AND.PATH IS FEASIBLE) STEP TO END OF
147      C***      FILES 'WPCDAT','WPARAM'
148      C
149          IF (IDIR.EQ.1.AND.IFEAS.EQ.0) GO TO 890
150          GO TO 891
151      890      CONTINUE
152      C
153      C*****      REPEAT UNTIL END OF FILE REACHED
154      C
155      560      CONTINUE
156          READ(2,END=556)
157          READ(7,*)
158          GO TO 560
159      556      CONTINUE
160      C
161      C***      END IF
162      C
163      C
164      891      CONTINUE
165      C
166      C
167      C***      REPEAT FOR ALL LOCATIONS IN FILE 'WPCDAT'
168      C
169      150      CONTINUE
170      C
171      C***      IF (DIRECTION .EQ. NORMAL) (IE. IDIR=0)
172      C
173          IF (IDIR.NE.0.AND.IFEAS.EQ.0) GO TO 855
174      C
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCId=DACO Page 4

```
175 C***** READ DATA FROM INPUT FILE 'WPCDAT'.
176 C
177 READ(2,END=555) JJ,S,(R(I),I=1,3),(T(K),K=1,3),
178 * (A(L),L=1,3),(B(M),M=1,3)
179 C
180 GO TO 880
181 C
182 855 CONTINUE
183 C
184 C
185 C*** OR IF (DIRECTION .EQ. REVERSE .AND. PATH IS FEASIBLE)
186 C
187 IF (IDIR.NE.1) GO TO 860
188 C
189 C***** READ FILE IN REVERSE ORDER FROM END TO BEGINNING
190 C***** 'INDEX' IS NUMBER OF RECORDS REMAINING
191 C
192 IF (INDEX.EQ.0) GO TO 555
193 BACKSPACE 2
194 READ(2) JJ,S,(R(I),I=1,3),(T(K),K=1,3),(A(L),L=1,3),
195 * (B(M),M=1,3)
196 BACKSPACE 2
197 INDEX=INDEX-1
198 JJ=NPOINT-JJ+1
199 GO TO 880
200 C
201 C*** ELSE (ERROR)
202 C
203 860 CONTINUE
204 GO TO 861
205 C
206 C*** END IF
207 C
208 880 CONTINUE
209 C
210 C
211 C***** SET UP TRANSFORMATION MATRIX 'HWC'. THE COLUMNS CONTAIN
212 C***** THE VECTORS T, B, A, R, OBTAINED FROM 'WPCDAT'.
213 C
214 HWC(1,1)=T(1)
215 HWC(2,1)=T(2)
216 HWC(3,1)=T(3)
217 HWC(4,1)=0.
218 HWC(1,2)=B(1)
219 HWC(2,2)=B(2)
220 HWC(3,2)=B(3)
221 HWC(4,2)=0.
222 HWC(1,3)=A(1)
223 HWC(2,3)=A(2)
224 HWC(3,3)=A(3)
225 HWC(4,3)=0.
226 HWC(1,4)=R(1)
227 HWC(2,4)=R(2)
228 HWC(3,4)=R(3)
229 HWC(4,4)=1.
230 C
231 C***** CONVERT ANGLES 'WALPHA', 'BETA' INTO RADIANS AND 'RHO'
232 C***** INTO MILLIMETERS
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCid=DAC0 Page 5

```

233      C
234      DPI=4.DO*DATAN(1.DO)
235      DE2RAD=DPI/180.
236      DALPHA=WALPHA*DE2RAD
237      DBETA=BETA*DE2RAD
238      DRHO=STKQUT
239      C
240      C*****      CALCULATE ELEMENTS OF MATRIX 'HCG'.
241      C
242      X(1)=0.
243      X(2)=DCOS(DALPHA)*DCOS(DBETA)
244      X(3)=DSIN(DBETA)
245      Z(1)=-DSIN(DALPHA)
246      Z(2)=DSIN(DBETA)
247      Z(3)=-DCOS(DALPHA)*DCOS(DBETA)
248      HCG(1,4)=-DRHO*Z(1)
249      HCG(2,4)=-DRHO*Z(2)
250      HCG(3,4)=-DRHO*Z(3)
251      HCG(4,4)=1.
252      C
253      C*****      THE SECOND COLUMN IS THE CROSS PRODUCT OF THE THIRD AND
254      C*****      FIRST COLUMNS
255      C
256      CALL XPROD(Z,X,Y)
257      C
258      DO 65 I=1,3
259      HCG(I,1)=X(I)
260      HCG(I,2)=Y(I)
261      HCG(I,3)=Z(I)
262      65      CONTINUE
263      C
264      HCG(4,1)=0.
265      HCG(4,2)=0.
266      HCG(4,3)=0.
267      C
268      C
269      C*****      PERFORM MATRIX MULTIPLICATIONS TO OBTAIN HOG.
270      C*****      HOG=HOT*HTW*HWC*HCG
271      C
272      CALL HMULT(HOT,HTW,HOW)
273      CALL HMULT(HOW,HWC,HOC)
274      CALL HMULT(HOC,HCG,HOG)
275      C
276      C
277      C*****      CONVERT DOUBLE PRECISION 'HOT','HOW','HOG' INTO SINGLE
278      C*****      PRECISION 'SHOT','SHOW','SHOG'.
279      C
280      CALL MATCNV(HOC,SHOC,1)
281      CALL MATCNV(HOG,SHOG,1)
282      C
283      C
284      C*****      CALCULATE HOG=INHOO*HOG
285      C*****      'INHOO' IS THE INVERSE HOMOGENEOUS TRANSFORMATION RELATING
286      C*****      THE WORLD COORDINATE FRAME TO THE ROBOT BASE FRAME
287      C
288      CALL HMULTS(INHOO,SHOG,HOG)
289      C
290      C

```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCid=DAC0 Page 6

```
291      895          CONTINUE
292      C
293      C
294      C*****      CALCULATE ROBOT JOINT ANGLES, TEST FOR LIMITS
295      C
296      C*****      IF THE ROBOT HAS 5 DEGREES OF FREEDOM, THEN CALCULATE
297      C*****      VECTORS 'N','O' AS FUNCTIONS OF VECTOR 'A' AND POSITION
298      C*****      'P'. THERE ARE TWO SOLUTIONS, SELECTED BY SETTING
299      C*****      'ISOLN'=0 OR 1. THE RESULT IS ONE OF TWO UNIQUE
300      C*****      TRANSFORMATIONS 'HOG'=[n,o,a,p]
301      C
302      IF(NRDOF.EQ.5) CALL NOAP5(ISOLN)
303      C
304      C
305      C***      WHILE(PATH NOT FEASIBLE AND COUNT LT. MAX) ROTATE TOOL
306      C
307      C*****      SEARCH FOR A FEASIBLE SOLUTION BY ROTATING THE TOOL
308      C*****      ABOUT ITS AXIS. THIS APPLIES TO 6 DEGREE OF FREEDOM
309      C*****      ROBOTS ONLY, FOR WHICH THE TOOL FRAME VECTORS 'N','O'
310      C*****      CAN BE ARBITRARILY SPECIFIED. THE INITIAL 'N','O' ARE
311      C*****      AS CALCULATED PREVIOUSLY. THE FIRST CALL TO 'TOLROT'
312      C*****      YIELDS A ROTATION OF ZERO.
313      C
314      CALL TOLROT(ROTOLD,ICOUNT,ROTNEW,INHAG)
315      C
316      C
317      C*****      CALCULATE ROBOT TRANSFORMATION MATRIX 'HOG'.
318      C*****      HOG=HOG*INHAG*INH6A
319      C
320      CALL HMULTS(HOG,INHAG,HOA)
321      CALL HMULTS(HOA,INH6A,HOG)
322      C
323      C
324      C*****      CALCULATE JOINT ANGLES FOR A GIVEN ROBOT TOOL MOUNTING
325      C*****      FLANGE FRAME MATRIX 'HOG'. IFLAG1: 0- OK, 1- ENVELOPE
326      C*****      EXCEEDED
327      C
328      CALL JOINTS(HOG,THETA,IFLAG1)
329      C
330      C
331      IF(IFLAG1.NE.0) GO TO 861
332      C
333      C
334      C*****      TEST THE CALCULATED JOINT POSITIONS AGAINST THEIR PHYSICAL
335      C*****      LIMITS. IFLAG2: 0- OK, 1- JOINT LIMIT EXCEEDED
336      C
337      CALL LIMITS(THETA,IFLAG2,IJOINT)
338      C
339      C
340      IF(IFLAG2.NE.0) GO TO 776
341      C
342      C
343      C*****      TEST JOINT 6 LIMITS (CHECK IF JOINT 6 ROTATES IN ONE
344      C*****      DIRECTION THROUGH A JOINT LIMIT). ALTHOUGH JOINT 6
345      C*****      ALLOWS MORE THAN 360. DEGREES OF ROTATION, WE CANNOT
346      C*****      DRIVE THE JOINT CONTINUOUSLY IN ONE DIRECTION. IF A
347      C*****      LIMIT IS REACHED, THE JOINT WILL ROTATE ONE ROTATION
348      C*****      AWAY FROM THE LIMIT BEFORE PROCEEDING. THIS CAN RESULT
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9. 1984 for CCId=DACO Page 7

```

349      C*****      IN A CATASTROPHIC DISCONTINUITY OF MOTION EVEN THOUGH
350      C*****      EACH LOCATION HAS A VALID SOLUTION.
351      C*****
352      C*****      TGLAST- LAST JOINT 6 ANGLE
353      C*****      T6SUM- JOINT 6 NET ROTATION FROM INITIAL POSITION
354      C*****      T6SUM2- T6SUM + 360.
355      C*****      J6FLG1- 1=T6SUM EXCEEDED JOINT LIMITS
356      C*****      J6FLG2- 1=T6SUM2 EXCEEDED JOINT LIMITS
357      C*****      T6INIT- INITIAL JOINT POSITION
358      C*****      DNEROT- ONE ROTATION (360. DEG.)
359      C*****      FTEST- 1=ONE ROTATION IN EITHER DIRECTION FROM INITIAL
360      C*****      ANGLE EXCEEDS JOINT LIMITS
361      C
362      IF(JJ.GT.1) GO TO 777
363      C
364      C*****      INITIALIZE PARAMETERS AT START OF SEAM.
365      C
366      TGLAST=0.
367      T6SUM=0.
368      T6SUM2=0.
369      J6FLG1=0
370      J6FLG2=0
371      T6INIT=THETA(6)
372      DNEROT=360.
373      C
374      C*****      CHECK IF JOINT 6 CAN BE ROTATED ONE ROTATION
375      C
376      IF(T6INIT+DNEROT.LT.RMAX(6)) GO TO 111
377      C
378      C*****      IF NOT, CHECK IF JOINT CAN BE ROTATED MINUS ONE
379      C*****      ROTATION
380      C
381      IF(T6INIT-DNEROT.GT.RMIN(6)) GO TO 112
382      GO TO 113
383      C
384      112      CONTINUE
385      DNEROT=-360.
386      GO TO 111
387      113      CONTINUE
388      C
389      C*****      IF JOINT 6 CANNOT BE ROTATED ONE ROTATION IN EITHER
390      C*****      DIRECTION FROM THE INITIAL POSITION, SET 'FTEST',
391      C*****      'J6FLG2'
392      C
393      FTEST=1
394      J6FLG2=1
395      C
396      111      CONTINUE
397      777      CONTINUE
398      C
399      C*****      UPDATE CURRENT JOINT POSITION 'T6SUM'
400      C
401      T6SUM=T6SUM+THETA(6)-TGLAST
402      TGLAST=THETA(6)
403      C
404      C*****      IF 'T6SUM' EXCEEDS JOINT LIMIT, SET 'J6FLG1'
405      C
406      IF(T6SUM.LT.RMIN(6).OR.T6SUM.GT.RMAX(6)) J6FLG1=1

```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCId=DAC0 Page 8

```

407      C
408          IF(FTEST.EQ.1) GO TO 118
409          T6SUM2=T6SUM+ONEROT
410      C
411      C*****          IF 'T6SUM2' EXCEEDS JOINT LIMIT, SET 'J6FLG2'
412      C
413          IF(T6SUM2.LT.RMIN(6).OR.T6SUM2.GT.RMAX(6)) J6FLG2=1
414      C
415      118          CONTINUE
416      C
417      C
418      C*****          IF EITHER 'J6FLG1' OR 'J6FLG2' IS SET, JOINT 6 HAS MOVED
419      C*****          THROUGH A JOINT LIMIT
420      C
421          IF(J6FLG1.EQ.0.OR.J6FLG2.EQ.0) GO TO 885
422      C
423      776          CONTINUE
424      C
425      C
426      C*****          IF PATH NOT FEASIBLE, ROTATE TOOL AND TRY AGAIN UNTIL
427      C*****          ALL SPECIFIED ROTATIONS HAVE BEEN TRIED.
428      C*****          FOR A 5 DEGREE OF FREEDOM ROBOT, TEST THE TWO POSSIBLE
429      C*****          JOINT SOLUTIONS.
430      C
431          IF(NRDOF.NE.5) GO TO 830
432      C
433      C*****          'ISOLN'=0 OR 1 SELECTS THE TWO POSSIBLE 5 DEGREE OF
434      C*****          FREEDOM SOLUTIONS
435      C
436          ISOLN=ISOLN+1
437          IF(ISOLN.GT.1) GO TO 850
438          GO TO 895
439      830          CONTINUE
440      C
441      C*****          FOR A 6 DEGREE OF FREEDOM ROBOT, INCREMENT THE TOOL AXIS
442      C*****          ROTATION COUNTER 'ICOUNT' AND RETRY, UNTIL A MAXIMUM
443      C*****          ROTATION IS REACHED.
444      C
445          ICOUNT=ICOUNT+1
446          IF(ICOUNT.LE.IMAX) GO TO 895
447      C
448      C***          END WHILE
449      C
450      C*****          RESET TOOL ROTATION COUNTER 'ICOUNT'
451      C
452          ICOUNT=1
453          GO TO 850
454      C
455      885          CONTINUE
456      C
457      C*****          RESET TOOL ROTATION COUNTER 'ICOUNT' AND REDEFINE THE
458      C*****          INITIAL ROTATION OF THE TOOL AS THE CURRENT ROTATION
459      C
460          ICOUNT=1
461          ROTOLD=ROTNEW
462      C
463      C*****          IF A FEASIBLE PATH HAS NOT BEEN FOUND, SKIP THE NEXT
464      C*****          SECTION.
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCid=DACO Page 9

```
465      C
466          IF(IFEAS.NE.O) GO TO 789
467      C
468      C
469      C*****      CALCULATE HOC=INHOO*HOC
470      C
471          CALL HMULTS(INHOO,SHOC,HOC)
472      C
473      C*****      FIND SEAM LATERAL AND NORMAL DIRECTIONS RELATIVE TO ROBOT
474      C*****      COORDS.
475      C
476          DO 115 I=1,3
477              ENORM(I)=HOC(I,3)
478              ELAT(I)=HOC(I,2)
479      115      CONTINUE
480      C
481      C
482      C*****      CALCULATE PUMA COORDINATES X,Y,Z,D,A,T FROM THE ROBOT TOOL
483      C*****      MOUNTING FLANGE FRAME MATRIX 'HO6'.
484      C
485          CALL NOAP1(HO6,XYZOAT)
486      C
487      C
488      C*****      READ THE WELDING CURRENT AND SPEED FROM THE WELD PARAMETER
489      C*****      FILE 'WPARAM'. IF WELDING DIRECTION IS REVERSE, READ THE
490      C*****      FILE FROM END TO BEGINNING
491      C
492          IF(IDIR.EQ.O) GO TO 225
493              BACKSPACE 7
494              READ(7,*) CURRNT,SPEED
495              BACKSPACE 7
496              GO TO 226
497      225      CONTINUE
498              READ(7,*) CURRNT,SPEED
499      226      CONTINUE
500      C
501      C
502      C*****      WRITE ROBOT COORDINATES, WELDING CURRENT AND SPEED,
503      C*****      AND SEAM LATERAL AND NORMAL DIRECTION VECTORS INTO
504      C*****      FILE 'ROBDAT'.
505      C
506          WRITE(3,99) JJ,(XYZOAT(I),I=1,6),CURRNT,SPEED
507      99      FORMAT(1X,I4,3F8.1,3F8.2,F8.2,F7.3)
508          WRITE(3,98) (ENORM(I),I=1,3),(ELAT(J),J=1,3)
509      98      FORMAT(5X,6F11.3)
510      C
511      C
512      789      CONTINUE
513      C
514      C***      END WHILE
515      C
516          GO TO 150
517      C
518      C*****      ERROR MESSAGES
519      C
520      C*****      ROBOT ENVELOPE EXCEEDED
521      C
522      861      CONTINUE
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCId=DACO Page 10

```
523          ISTAT=1
524          GO TO 555
525      C
526      C***** JOINT LIMIT EXCEEDED
527      C
528      850      CONTINUE
529          ISTAT=2
530          GO TO 555
531      C
532      555      CONTINUE
533      C
534      C
535      C***** IF A FEASIBLE PATH HAS BEEN FOUND, REPEAT THE LAST PATH
536      C***** AND STORE THE RESULTS IN THE OUTPUT FILES. 'IFEAS'=0
537      C***** INDICATES A FEASIBLE PATH HAS BEEN FOUND.
538      C
539          IF(ISTAT.NE.0) GO TO 75
540          IFEAS=IFEAS-1
541          IF(IFEAS.LT.0) GO TO 75
542      C
543      C
544      C***** GET CURRENT TABLE POSITION FROM TEMPORARY FILE 'STOVAR'.
545      C
546          CALL OPEN(1,STOVAR,4,1)
547      C
548      C***** 'IVAR' IS A TEMPORARY COUNTER
549      C
550          IVAR=1
551          READ(1,*)
552      420      CONTINUE
553          READ(1,*,END=410) TABVAR(IVAR)
554          IVAR=IVAR+1
555          GO TO 420
556      410      CONTINUE
557          CALL SHUT(1)
558      C
559      C***** WRITE SEAM NUMBER, TABLE POSITION AND ROBOT CONFIGURATION
560      C***** INTO FILE 'LOCSET'.
561      C
562      C***** GO TO CURRENT END OF FILE
563      C
564      450      CONTINUE
565          READ(4,*,END=400)
566          GO TO 450
567      400      CONTINUE
568      C
569      C***** APPEND PARAMETERS DESCRIBING SEAM 'IDSEAM'
570      C
571          WRITE(4,*) IDSEAM
572          WRITE(4,*) (TABVAR(I),I=1,6)
573          WRITE(4,*) (ICONF(I),I=1,6)
574          WRITE(4,*) VOLT,WSIZE,FEED,GFR
575      C
576      C***** REWIND FILE 'WPCDAT'
577      C
578          REWIND(2)
579      C
580      C***** RESET THE TOOL ROTATION ANGLE TO ZERO
```

Listing of WORPOS.FOR at 14:43:16 on AUG 9, 1984 for CCid=DACO Page 11

```
581      C
582              ROTOLD=0.
583      C
584      C
585      C***** REPEAT LAST PATH, WHICH IS FEASIBLE, AND GENERATE THE ROBOT DATA
586      C***** FILE 'ROBDAT' THIS TIME.
587      C
588              GO TO 160
589      C
590      C
591      75      CONTINUE
592      C
593      C***** CLOSE DATA FILES 'WPCDAT', 'ROBDAT', 'LOCSET'.
594      C
595              CALL SHUT(2)
596              CALL SHUT(3)
597              CALL SHUT(4)
598              CALL SHUT(7)
599      C
600      C
601              RETURN
602              END
```

Listing of XPROD.FOR at 14:43:20 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE XPROD(A,B,C)
2      C
3      THIS SUBROUTINE CALCULATES VECTOR C = A X B (CROSS PRODUCT).
4      C
5      INPUTS:
6      C          A(3)=FIRST VECTOR
7      C          B(3)=SECOND VECTOR
8      C
9      OUTPUT:
10     C          C(3)=NORMALIZED CROSS PRODUCT, A X B
11     C
12     C*****
13     C
14     C
15     REAL A(3), B(3), C(3), ABS
16     C
17     C(1)=A(2)*B(3)-A(3)*B(2)
18     C(2)=A(3)*B(1)-A(1)*B(3)
19     C(3)=A(1)*B(2)-A(2)*B(1)
20     C
21     ABS=SQRT(C(1)**2+C(2)**2+C(3)**2)
22     IF(ABS.EQ.0.) GO TO 100
23     C
24     C(1)=C(1)/ABS
25     C(2)=C(2)/ABS
26     C(3)=C(3)/ABS
27     C
28     100 CONTINUE
29     RETURN
30     END
```

APPENDIX I.3

CAD FILE GENERATING PROGRAM

CADGEN LISTING

Listing of CADGEN.FOR at 14:50:38 on AUG 9, 1984 for CCid=DACO Page 1

```
1
2      PROGRAM CADGEN
3      C
4      C      THIS PROGRAM GENERATES LOCATION AND SURFACE NORMAL DATA
5      C      FOR A VERTICAL CYLINDER INTERSECTING A HORIZONTAL PLATE.
6      C      THE DATA IS STORED IN CAD FILES IN THE CORRECT FORMAT
7      C      FOR TESTING AND DEMONSTRATION OF PROGRAM 'AUTOP'.
8      C
9      C
10     C      SUBROUTINES CALLED:      OPEN
11     C                               SHUT
12     C
13     C*****
14     C
15     C      BYTE BUF(80),BUF2(80),BUF3(80),BUF4(80),BUF5(80)
16     C      REAL R(3),N1(3),N2(3),RQ(3)
17     C      INTEGER ISTAT,IWELD
18     C      DATA ISTAT/O/,IWELD/O/
19     C      DATA BUF/'C','A','D','1',.,.,.,'D','A','T',72*O/
20     C      DATA BUF2/'C','A','D','2',.,.,.,'D','A','T',72*O/
21     C      DATA BUF3/'C','A','D','3',.,.,.,'D','A','T',72*O/
22     C      DATA BUF4/'C','A','D','4',.,.,.,'D','A','T',72*O/
23     C      DATA BUF5/'C','A','D','5',.,.,.,'D','A','T',72*O/
24     C      DATA RQ/100.,100.,100./
25     C
26     C***** CREATE FORMATTED FILES
27     C
28     C      IEND=100
29     C
30     C      RADIUS=100.
31     C      PI=3.14159
32     C      TWOPI=2.*PI
33     C      HALFPI=.5*PI
34     C      STEP=TWOPI/100.
35     C      THETA=0.
36     C
37     C***** DUMMY WELD PARAMETERS
38     C
39     C      JTYPE=1
40     C      XSECT=.5
41     C      T1=.25
42     C      T2=.5
43     C      NPOINT=26
44     C      ICURVE=1
45     C
46     C      DO 500 IND=1,4
47     C      INDEX=1
48     C      IF(IND.EQ.1) CALL OPEN(1,BUF,2,1)
49     C      IF(IND.EQ.2) CALL OPEN(1,BUF2,2,1)
50     C      IF(IND.EQ.3) CALL OPEN(1,BUF3,2,1)
51     C      IF(IND.EQ.4) CALL OPEN(1,BUF4,2,1)
52     C
53     C***** WRITE JTYPE,XSECT,T1,T2 INTO CAD FILE.
54     C
55     C      WRITE(1) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
56     C
57     C      THETA=FLOAT(IND-1)*HALFPI
58     C
```

Listing of CADGEN.FOR at 14:50:38 on AUG 9, 1984 for CCId=DACO Page 2

```
59      C      WHILE (THETA .LT. TWOPI)
60      C
61          DO 100 IPOINT=1,26
62              R(1)=RO(1)+RADIUS*COS(THETA)
63              R(2)=RO(2)+RADIUS*SIN(THETA)
64              R(3)=RO(3)
65              N1(1)=COS(THETA)
66              N1(2)=SIN(THETA)
67              N1(3)=0.
68              N2(1)=0.
69              N2(2)=0.
70              N2(3)=1.
71      C
72      C      WRITE RESULTS TO FILE.
73      C
74          WRITE(1) INDEX,(R(I),I=1,3),(N1(K),K=1,3),(N2(L),L=1,3),
75          * ISTAT,IWELD
76          WRITE(5,900) INDEX,(R(I),I=1,3)
77      900    FORMAT(I5,3F12.4)
78      C
79      C      INCREMENT THETA,INDEX
80      C
81          THETA=THETA+STEP
82          INDEX=INDEX+1
83      100    CONTINUE
84      C
85      C      END WHILE
86      C
87          CALL SHUT(1)
88      500    CONTINUE
89      C
90          R(1)=200.
91          R(2)=100.
92          R(3)=100.
93          N1(1)=1.
94          N1(2)=0.
95          N1(3)=0.
96          N2(1)=1.
97          N2(2)=0.
98          N2(3)=0.
99          CALL OPEN(1,BUF5,2,1)
100     C
101         NPOINT=50
102         ICURVE=0
103     C
104         WRITE(1) JTYPE,XSECT,T1,T2,NPOINT,ICURVE
105     C
106         DO 300 I=1,50
107             WRITE(1) I,(R(K),K=1,3),(N1(L),L=1,3),(N2(M),M=1,3),ISTAT,IWELD
108             R(3)=R(3)+4.
109             WRITE(5,900) I,(R(J),J=1,3)
110     300    CONTINUE
111     C
112         GO TO 999
113     C
114     10      WRITE(5,11)
115     11      FORMAT(' ERROR ENCOUNTERED IN OPEN STATEMENT')
116     C
```

Listing of CADGEN.FOR at 14:50:38 on AUG 9, 1984 for CCId=DAC0 Page 3

117	999	CONTINUE
118	C	
119		CALL SHUT(1)
120	C	
121		STOP
122		END

APPENDIX I.4

TESTIN PROGRAM LISTING

Listing of TESTIN.FOR at 14:48:01 on AUG 9, 1984 for CCId=DACD Page 1

```

1      C      PROGRAM TESTIN
2      C
3      C      THIS PROGRAM TESTS FOR INTERFERENCE BETWEEN A FACE OF A SOLID
4      C      POLYHEDRON AND THE VOLUME SWEEPED OUT BY A CYLINDER AS IT IS
5      C      TRANSLATED AND ROTATED FROM AN INITIAL POSITION TO A FINAL
6      C      POSITION.
7      C
8      C      DEFINITION OF VARIABLES AND CONSTANTS:
9      C
10     C      A1,B1- ENDPOINTS OF INITIAL CYLINDER SPINE POSITION.
11     C      A2,B2- ENDPOINTS OF FINAL CYLINDER SPINE POSITION.
12     C      P1,P2,...P8- CORNER POINTS OF PARAMETRIC SWEEPED VOLUME.
13     C      VERT- POLYHEDRA VERTEX LIST.
14     C      FACES- POLYHEDRA FACE PLANE PARAMETER LIST.
15     C      IFACE- POLYHEDRA FACE LIST.
16     C      RADIUS- RADIUS OF CYLINDER
17     C      E1,E2- ENDPOINTS OF CURRENT EDGE
18     C      K1,K2- SPINE DIRECTION VECTORS.
19     C      K1N,K2N- NORMALIZED K1,K2
20     C
21     C      SUBROUTINES CALLED:
22     C
23     C          CYLINT
24     C          DIFF
25     C          DRCYL
26     C          FACE
27     C          FACINT
28     C          FACYL
29     C          INVOL
30     C          SURFIN
31     C          SWPVOL
32     C          UNIT
33     C
34     C      'IGL' SUBROUTINES CALLED:
35     C
36     C          CMCLOS
37     C          CMOPEN
38     C          EYEBAL
39     C          GRSTOP
40     C          GRSTRT
41     C          VRP3D
42     C          WIND3D
43     C          ZPERSP
44     C
45     C
46     C*****
47     C
48     C      REAL E1(3),E2(3)
49     C      REAL K1(3),K2(3),K1N(3),K2N(3)
50     C
51     C      COMMON /SPINES/A1(3),B1(3),A2(3),B2(3)
52     C      COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
53     C      COMMON /CORNER/P1(3),P2(3),P3(3),P4(3),P5(3),P6(3),P7(3),P8(3)
54     C      COMMON /RADIUS/RADIUS
55     C
56     C***** INITIALIZE 'IGL' GRAPHICS ENVIRONMENT
57     C
58     C      CALL GRSTRT(4027,1)
```

Listing of TESTIN.FOR at 14:48:01 on AUG 9, 1984 for CCid=DACO Page 2

```
59          CALL ZPERSP
60          CALL EYEBAL(100.,100.,100.)
61          CALL VRP3D(0.,0.,0.)
62          CALL WIND3D(-50.,50.,-50.,50.,-50.,50.)
63          CALL CMCLOS
64          C
65          C***** GENERATE SWEPT VOLUME.
66          C
67          CALL SWPVOL
68          C
69          C***** CALCULATE AND DRAW CYLINDERS AT INITIAL AND FINAL POSITIONS.
70          C
71          CALL DIFF(B1,A1,K1)
72          CALL UNIT(K1,K1N)
73          CALL DIFF(B2,A2,K2)
74          CALL UNIT(K2,K2N)
75          CALL DRCYL(A1,B1,K1N)
76          CALL DRCYL(A2,B2,K2N)
77          C
78          C***** INTERACTIVELY SPECIFY A TYPICAL POLYHEDRON FACE FOR TESTING.
79          C
80          CALL FACE
81          C
82          C***** FIND INTERSECTIONS BETWEEN FACE EDGES AND SWEPT VOLUME SURFACES.
83          C
84          C***** REPEAT FOR EACH EDGE OF THE FACE...
85          C
86          DO 10 I=1,IFACE(1,2)
87          C
88          C***** FIND THE ENDPOINTS E1,E2 OF THE CURRENT EDGE.
89          C
90              DO 15 J=1,3
91                  E1(J)=VERT(I,J)
92                  E2(J)=VERT(I+1,J)
93          15      CONTINUE
94          C
95          C***** TEST FOR INTERSECTION BETWEEN THE CURRENT EDGE AND THE FOUR
96          C***** BOUNDING PARAMETRIC FACES OF THE SWEPT VOLUME.
97          C
98              CALL SURFIN(P1,P2,P3,P4,E1,E2,INT)
99              IF (INT.EQ.1) GO TO 900
100         C
101             CALL SURFIN(P5,P6,P7,P8,E1,E2,INT)
102             IF (INT.EQ.1) GO TO 900
103         C
104             CALL SURFIN(P1,P5,P6,P2,E1,E2,INT)
105             IF (INT.EQ.1) GO TO 900
106         C
107             CALL SURFIN(P3,P4,P8,P7,E1,E2,INT)
108             IF (INT.EQ.1) GO TO 900
109         C
110         C***** FIND INTERSECTIONS BETWEEN EDGES AND CYLINDRICAL VOLUMES
111         C
112             CALL CYLINT(A1,B1,E1,E2,INT)
113             IF (INT.EQ.1) GO TO 901
114         C
115             CALL CYLINT(A2,B2,E1,E2,INT)
116             IF (INT.EQ.1) GO TO 901
```

Listing of TESTIN.FOR at 14:48:01 on AUG 9, 1984 for CCId=DACD Page 3

```
117      C
118      10      CONTINUE
119      C
120      C***** END OF DO LOOP *****
121      C
122      C***** TEST IF ANY SWEEPED VOLUME EDGE INTERSECTS FACE.
123      C
124              CALL FACINT(1,P1,P2,INT)
125              IF(INT.EQ.1) GO TO 980
126      C
127              CALL FACINT(1,P2,P3,INT)
128              IF(INT.EQ.1) GO TO 980
129      C
130              CALL FACINT(1,P3,P4,INT)
131              IF(INT.EQ.1) GO TO 980
132      C
133              CALL FACINT(1,P4,P1,INT)
134              IF(INT.EQ.1) GO TO 980
135      C
136              CALL FACINT(1,P1,P5,INT)
137              IF(INT.EQ.1) GO TO 980
138      C
139              CALL FACINT(1,P2,P6,INT)
140              IF(INT.EQ.1) GO TO 980
141      C
142              CALL FACINT(1,P3,P7,INT)
143              IF(INT.EQ.1) GO TO 980
144      C
145              CALL FACINT(1,P4,P8,INT)
146              IF(INT.EQ.1) GO TO 980
147      C
148              CALL FACINT(1,P5,P6,INT)
149              IF(INT.EQ.1) GO TO 980
150      C
151              CALL FACINT(1,P6,P7,INT)
152              IF(INT.EQ.1) GO TO 980
153      C
154              CALL FACINT(1,P7,P8,INT)
155              IF(INT.EQ.1) GO TO 980
156      C
157              CALL FACINT(1,P8,P5,INT)
158              IF(INT.EQ.1) GO TO 980
159      C
160      C***** TEST IF CYLINDER IN INITIAL OR FINAL POSITION INTERSECTS ANY FACE.
161      C
162              CALL FACYL(1,A1,B1,INT)
163              IF(INT.EQ.1) GO TO 990
164      C
165              CALL FACYL(1,A2,B2,INT)
166              IF(INT.EQ.1) GO TO 990
167      C
168      C***** TEST IF A POLYHEDRON LIES INSIDE THE SWEEPED VOLUME.
169      C
170              CALL INVOL(1,INT)
171              IF(INT.EQ.1) GO TO 991
172      C
173              GO TO 998
174      C
```

Listing of TESTIN.FOR at 14:48:01 on AUG 9, 1984 for CCId=DACD Page 4

```
175 C***** INTERSECTON MESSAGES *****
176 C
177 900 WRITE(5,*) ' FACE EDGE INTERSECTED SWEPT SURFACE '
178 GO TO 999
179 901 WRITE(5,*) ' FACE EDGE INTERSECTED CYLINDRICAL ENDS '
180 GO TO 999
181 980 CONTINUE
182 WRITE(5,*) ' SWEPT VOLUME EDGE INTERSECTED FACE '
183 GO TO 999
184 990 WRITE(5,*) ' CYLINDER INTERSECTED FACE '
185 GO TO 999
186 991 WRITE(5,*) ' POLYHEDRON LIES ENTIRELY INSIDE SWEPT VOLUME '
187 GO TO 999
188 C
189 998 CONTINUE
190 WRITE(5,*) ' NO INTERSECTIONS WERE DETECTED '
191 C
192 999 CONTINUE
193 C
194 CALL CMOPEN
195 CALL GRSTOP
196 C
197 STOP
198 END
```

Listing of CORNER.FOR at 14:43:49 on AUG 9, 1984 for CCId=DAC0 Page 1

```

1      SUBROUTINE CORNER
2      C
3      C      THIS SUBROUTINE FINDS THE CORNER POINTS OF THE SWEEP VOLUME.
4      C
5      C      COMMON BLOCKS ACCESSED: /LINK/,/RADIUS/,/VOLUME/,/SNORM/
6      C      COMMON BLOCKS MODIFIED: /CORNER/
7      C
8      C*****
9      C
10     REAL LEN,RADIUS
11     REAL AN(3),BN(3),CN(3)
12     REAL N(3)
13     C
14     COMMON /LINK/H1(4,4),H2(4,4),LEN
15     COMMON /RADIUS/RADIUS
16     COMMON /VOLUME/P(3),UO(3),U1(3),VO(3),V1(3)
17     COMMON /SNORM/DAN(3),DBN(3),DCN(3)
18     COMMON /CORNER/P1(3),P2(3),P3(3),P4(3),P5(3),P6(3),P7(3),P8(3)
19     C
20     C
21     C***** LAMBD AU=LAMBD AV=0
22     C
23     DO 10 I=1,3
24         N(I)=DAN(I)
25     10 CONTINUE
26     C
27     ALPHA=SQRT(N(1)**2+N(2)**2+N(3)**2)
28     C
29     DO 20 I=1,3
30         P1(I)=P(I)-RADIUS*DAN(I)/ALPHA
31         P5(I)=P1(I)+2.*RADIUS*DAN(I)/ALPHA
32     20 CONTINUE
33     C
34     C
35     C***** LAMBD AU=1,LAMBD AV=0
36     C
37     DO 30 I=1,3
38         N(I)=DAN(I)+DBN(I)
39     30 CONTINUE
40     C
41     ALPHA=SQRT(N(1)**2+N(2)**2+N(3)**2)
42     C
43     DO 40 I=1,3
44         P2(I)=P(I)+UO(I)-RADIUS*N(I)/ALPHA
45         P6(I)=P2(I)+2.*RADIUS*N(I)/ALPHA
46     40 CONTINUE
47     C
48     C
49     C***** LAMBD AU=1,LAMBD AV=1
50     C
51     DO 50 I=1,3
52         N(I)=DAN(I)+DBN(I)+DCN(I)
53     50 CONTINUE
54     C
55     ALPHA=SQRT(N(1)**2+N(2)**2+N(3)**2)
56     C
57     DO 60 I=1,3
58         P3(I)=P(I)+UO(I)+V1(I)-RADIUS*N(I)/ALPHA

```

Listing of CORNER.FOR at 14:43:49 on AUG 9, 1984 for CCid=DACO Page 2

```
59          P7(I)=P3(I)+2.*RADIUS*N(I)/ALPHA
60      60      CONTINUE
61      C
62      C
63      C***** LAMBD AU=0, LAMBD AV=1
64      C
65          DO 70 I=1,3
66              N(I)=DAN(I)+DCN(I)
67      70      CONTINUE
68      C
69          ALPHA=SQRT(N(1)**2+N(2)**2+N(3)**2)
70      C
71          DO 80 I=1,3
72              P4(I)=P(I)+VO(I)-RADIUS*N(I)/ALPHA
73              P8(I)=P4(I)+2.*RADIUS*N(I)/ALPHA
74      80      CONTINUE
75      C
76          RETURN
77          END
```

Listing of CYLEND.FOR at 14:43:50 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE CYLEND(P,U,PC,U1,Q1,Q2,INT)
2      C
3      C      THIS SUBROUTINE TESTS IF AN EDGE WITH ENDPOINTS Q1,Q2
4      C      INTERSECTS THE END OF A CYLINDER OF RADIUS 'RADIUS'.
5      C      P,U(3) DEFINE THE PLANE OF THE CIRCLE AND PC(3)
6      C      DEFINES ITS CENTER. U1 IS THE EDGE DIRECTION VECTOR.
7      C
8      C      INPUT ARGUMENTS: P,U(3),PC(3),U1(3),Q1(3),Q2(3)
9      C
10     C      OUTPUT ARGUMENTS: INT- 0=NO INTERSECTION
11     C      1=INTERSECTION
12     C
13     C      SUBROUTINES CALLED:      DIFF
14     C                          DOTPRD
15     C                          VLEN
16     C
17     C*****
18     C
19     REAL U(3),PC(3),U1(3),Q1(3),Q2(3),TEMP(3),PINT(3)
20     C
21     COMMON /RADIUS/RADIUS
22     C
23     C
24     INT=0
25     C
26     C***** TEST IF BOTH ENDPOINTS ARE ON THE SAME SIDE OF THE PLANE.
27     C
28     CALL DOTPRD(Q1,U,Q1U)
29     CALL DOTPRD(Q2,U,Q2U)
30     C
31     Z=(Q1U-P)*(Q2U-P)
32     C
33     IF(Z.GT.0.) GO TO 800
34     C
35     C***** TEST IF THE SPINE IS PARALLEL TO THE PLANE.
36     C
37     IF(Q1U.EQ.Q2U) GO TO 800
38     C
39     C***** IF THE SPINE INTERSECTS THE PLANE, FIND THE INTERSECTION POINT.
40     C
41     RLAMB=(P-Q1U)/(Q2U-Q1U)
42     C
43     DO 20 I=1,3
44     PINT(I)=Q1(I)+RLAMB*U1(I)
45     20 CONTINUE
46     C
47     C***** FIND THE DISTANCE FROM THE INTERSECTION POINT TO THE CENTER OF THE
48     C***** CYLINDER FACE.
49     C
50     CALL DIFF(PINT,PC,TEMP)
51     CALL VLEN(TEMP,DIST)
52     C
53     C***** TEST IF THE DISTANCE IS LESS THAN THE RADIUS.
54     C
55     IF(DIST.GT.RADIUS) GO TO 800
56     C
57     INT=1
58     C
```

Listing of CYLEND.FOR at 14:43:50 on AUG 9, 1984 for CCId=DAC0 Page 2

59	800	CONTINUE
60		RETURN
61		END

Listing of CYLINT.FOR at 14:43:50 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE CYLINT(P1,P2,Q1,Q2,INT)
2      C
3      C      THIS SUBROUTINE TESTS IF A LINE SEGMENT WITH ENDPJINTS
4      C      Q1,Q2 INTERSECTS A CYLINDER WITH SPINE ENDPOINTS
5      C      P1,P2, AND A RADIUS 'RADIUS'.
6      C
7      C      INPUT ARGUMENTS: P1(3),P2(3)-CYLINDER SPINE ENDPOINTS.
8      C                        Q1(3),Q2(3)-LINE SEGMENT ENDPOINTS.
9      C
10     C      , OUTPUT ARGUMENTS: INT- 0=NO INTERSECTION
11     C                        1=INTERSECTION
12     C
13     C
14     C      DEFINITION OF CONSTANTS AND VARIABLES.
15     C
16     C      LAMB1,LAMB2- PARAMETERS SPECIFYING LOCATION OF COMMON NORMAL.
17     C      DIST- DISTANCE BETWEEN EDGE LINE AND SPINE LINE.
18     C      PA,PB- DISTANCES FROM ORIGIN TO CYLINDER END FACE PLANES.
19     C      UCYL- SURFACE NORMAL OF CYLINDER END FACES.
20     C
21     C      SUBROUTINES CALLED:      CYLEND
22     C                               DIFF
23     C                               DOTPRD
24     C                               SOLVE
25     C                               UNIT
26     C                               VLEN
27     C                               XPROD
28     C
29     C*****
30     C
31     REAL LAMB1,LAMB2
32     REAL P1(3),P2(3),Q1(3),Q2(3),U1(3),U2(3),UN1(3),UN2(3),TEMP(3)
33     REAL T1(3),T2(3),UN(3),U(3),T3(3),T4(3),DIST1(3),DIST2(3)
34     REAL UTEMP(3)
35     C
36     COMMON /PARAM/LAMB1,LAMB2,DIST,D1,D2
37     COMMON /RADIUS/RADIUS
38     COMMON /ENDS/PA,PB,UCYL(3)
39     C
40     C***** FIND DISTANCE BETWEEN EDGE LINE AND SPINE LINE
41     C
42     C***** FIRST FIND COMMON NORMAL
43     C
44     CALL DIFF(P1,Q1,T1)
45     CALL DIFF(P2,P1,U1)
46     CALL DIFF(Q2,Q1,U2)
47     CALL UNIT(U1,UN1)
48     CALL UNIT(U2,UN2)
49     CALL XPROD2(UN1,UN2,U)
50     CALL VLEN(U,A2)
51     C
52     C***** TEST IF LINES ARE PARALLEL.
53     C
54     IF(A2.LT..00001) GO TO 120
55     GO TO 130
56     C
57     120 CONTINUE
58     CALL XPROD2(T1,UN1,UDIST)

```

Listing of CYLINT.FOR at 14:43:50 on AUG 9, 1984 for CCid=DAC0 Page 2

```

59          CALL XPROD2(UDIST,UN1,U)
60          CALL VLEN(UDIST,DIST)
61          IPARAL=1
62          C
63          C*****      TEST IF LINES ARE COINCIDENT.
64          C
65          IF(DIST.LE..000001) GO TO 800
66          GO TO 140
67          130      CONTINUE
68          CALL DOTPRD(T1,U,A1)
69          DIST=ABS(A1/A2)
70          140      CONTINUE
71          C
72          C
73          C*****      TEST DISTANCE AGAINST RADIUS
74          C
75          IF(DIST.GT.RADIUS) GO TO 900
76          C
77          C*****      IF THE SPINE AND EDGE ARE PARALLEL, INTERSECTION CAN ONLY OCCUR
78          C*****      IF EITHER END OF THE EDGE IS INSIDE THE CYLINDER.
79          C
80          IF(IPARAL.EQ.1) GO TO 200
81          C
82          C*****      FIND LENGTHS OF SPINE,RLEN1,AND EDGE,RLEN2.
83          C
84          CALL VLEN(U1,RLEN1)
85          CALL VLEN(U2,RLEN2)
86          C
87          C*****      FIND VALUES OF LAMB1,LAMB2
88          C
89          CALL UNIT(U,UN)
90          C
91          CALL SOLVE(P1,Q1,U1,U2,UN,DIST,LAMB1,LAMB2)
92          C
93          C*****      TEST LAMB1,LAMB2
94          C
95          IF(LAMB1.GE.0..AND.LAMB1.LE.1.) IFLG1=1
96          IF(LAMB2.GE.0..AND.LAMB2.LE.1.) IFLG2=1
97          C
98          IF(IFLG1.EQ.1.AND.IFLG2.EQ.1) GO TO 800
99          C
100         C*****      FIND DISTANCES OF EDGE ENDPOINTS Q1,Q2 FROM SPINE
101         C
102         CALL XPROD2(T1,UN1,DIST1)
103         CALL VLEN(DIST1,D1)
104         CALL DIFF(P1,Q2,T2)
105         CALL XPROD2(T2,UN1,DIST2)
106         CALL VLEN(DIST2,D2)
107         C
108         C*****      TEST IF DISTANCES ARE GREATER THAN THE RADIUS.
109         C
110         IF(D1.GT.RADIUS.AND.D2.GT.RADIUS) GO TO 900
111         C
112         C*****      CALCULATE PA,PB,UCYL FOR CYLINDER ENDS
113         C
114         CALL DIFF(P2,P1,UTEMP)
115         CALL UNIT(UTEMP,UCYL)
116         CALL DOTPRD(P1,UCYL,PA)
```

Listing of CYLINT.FOR at 14:43:50 on AUG 9, 1984 for CCId=DACO Page 3

```
117          CALL DOTPRD(P2,UCYL,PB)
118      C
119      C***** TEST FOR INTERSECTION OF EDGES WITH ENDS OF CYLINDER
120      C
121          IF(IFLG1.EQ.0) GO TO 100
122          GO TO 200
123      C
124      100      CONTINUE
125      C
126          CALL CYLEND(PA,UCYL,P1,U2,Q1,Q2,INT1)
127          IF(INT1.EQ.1) GO TO 800
128      C
129          CALL CYLEND(PB,UCYL,P2,U2,Q1,Q2,INT2)
130          IF(INT2.EQ.1) GO TO 800
131      C
132      200      CONTINUE
133      C
134      C***** TEST IF EITHER ENDPOINT OF THE EDGE LIES INSIDE THE CYLINDER
135      C
136          CALL DIFF(Q1,P1,T3)
137          CALL XPROD2(T3,UN2,TEMP)
138          CALL VLEN(TEMP,LAMB2)
139          IF(LAMB2.GE.0..AND.LAMB2.LE.1.) GO TO 800
140      C
141          CALL DIFF(Q2,P1,T4)
142          CALL XPROD2(T4,UN2,TEMP)
143          CALL VLEN(TEMP,LAMB2)
144          IF(LAMB2.GE.0..AND.LAMB2.LE.1.) GO TO 800
145      C
146          GO TO 900
147      C
148      800      CONTINUE
149      C
150          INT=1
151      C
152      900      CONTINUE
153          RETURN
154          END
```

Listing of DET2X2.FOR at 14:44:07 on AUG 9, 1984 for CCid=DACD Page 1

```
1      SUBROUTINE DET2X2(V1,V2,DET)
2      C
3      C      THIS SUBROUTINE FINDS THE DETERMINANT OF A 2 X 2 MATRIX
4      C      OF THE FORM [V1,V2], WHERE V1(2),V2(2).
5      C
6      REAL V1(2),V2(2)
7      C
8      DET=V1(1)*V2(2)-V1(2)*V2(1)
9      RETURN
10     END
```

Listing of DET3X3.FOR at 14:44:07 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE DET3X3(V1,V2,V3,DET)
2
3      C      THIS SUBROUTINE FINDS THE DETERMINANT OF A 3 X 3 MATRIX
4      C      OF THE FORM [V1,V2,V3], V1(3),V2(3),V3(3).
5      C
6      REAL V1(3),V2(3),V3(3)
7      C
8      DET=V1(1)*V2(2)*V3(3)+V1(2)*V2(3)*V3(1)+V1(3)*V2(1)*V3(2)-
9      * V1(3)*V2(2)*V3(1)-V1(2)*V2(1)*V3(3)-V1(1)*V2(3)*V3(2)
10     C
11     RETURN
12     END
```

Listing of DIFF.FOR at 14:44:07 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE DIFF(V2,V1,DIF)
2      C
3      C      THIS SUBROUTINE PERFORMS THE VECTOR DIFFERENCE OPERATION
4      C      DIFF=V2-V1
5      C
6      REAL V1(3),V2(3),DIF(3)
7      C
8      DO 10 I=1,3
9          DIF(I)=V2(I)-V1(I)
10     CONTINUE
11     C
12     RETURN
13     END
```

Listing of DOTPRD.FOR at 14:44:28 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE DOTPRD(V1,V2,PROD)
2      C
3      C      THIS SUBROUTINE FINDS THE DOT PRODUCT OF TWO VECTORS,
4      C      V1(3),V2(3).
5      C
6      C      INPUTS: V1(3),V2(3)- TWO VECTORS
7      C
8      C      OUTPUTS: PROD- THE DOT PRODUCT PROD=V1.V2
9      C
10     C*****
11     REAL V1(3),V2(3)
12     C
13     PROD=V1(1)*V2(1)+V1(2)*V2(2)+V1(3)*V2(3)
14     C
15     RETURN
16     END
```

Listing of DRCYL.FOR at 14:44:29 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE DRCYL(A,B,K)
2      C
3      C      THIS SUBROUTINE DRAWS A CYLINDER AS AN OCTAHEDRAL APPROXIMATION.
4      C      THE CYLINDER IS DEFINED BY SPINE ENDPOINTS 'A','B' AND BY SPINE
5      C      DIRECTION VECTOR 'K'.
6      C
7      C      INPUT ARGUMENTS:          A(3),B(3)- CYLINDER SPINE ENDPOINTS.
8      C                               K(3)- SPINE DIRECTION VECTOR.
9      C
10     C      COMMON BLOCKS ACCESSED: /RADIUS/
11     C
12     C      SUBROUTINES CALLED:      ROTK
13     C                               TRANS3
14     C                               UNIT
15     C                               VLEN
16     C                               XPROD
17     C
18     C      'IGL' GRAPHICS SUBROUTINES:  CMCLOS
19     C                                     CMOPEN
20     C                                     DRAW3D
21     C                                     MOVE3D
22     C
23     C
24     C*****
25     C
26     C      REAL A(3),B(3),K(3),R(3),ROTK(4,4),RW(3),PA(8,3),PB(8,3)
27     C      REAL X(3),Y(3),RN(3)
28     C
29     C
30     C      DATA X/1.,0.,0./
31     C      DATA Y/0.,1.,0./
32     C      DATA DTHETA/0.7854/
33     C
34     C      COMMON /RADIUS/RADIUS
35     C
36     C      FIND A RADIUS VECTOR 'R' NORMAL TO THE SPINE 'K'.
37     C
38     C      CALL XPROD(X,K,R)
39     C      CALL VLEN(R,RLEN)
40     C      IF(RLEN.LT..00001) GO TO 100
41     C      GO TO 200
42     C      100 CONTINUE
43     C      CALL XPROD(Y,K,R)
44     C      200 CONTINUE
45     C
46     C      CALL UNIT(R,RN)
47     C
48     C      THETA=0.
49     C
50     C***** ROTATE THE RADIUS VECTOR ABOUT THE SPINE TO GENERATE POINTS
51     C***** ON THE CYLINDER.
52     C
53     C      DO 10 I=1,8
54     C          CALL ROTK(K,THETA,ROTK)
55     C          CALL TRANS3(ROTK,RN,RW)
56     C          DO 20 J=1,3
57     C              PA(I,J)=A(J)+RADIUS*RW(J)
58     C              PB(I,J)=B(J)+RADIUS*RW(J)

```

Listing of DRCYL.FOR at 14:44:29 on AUG 9, 1984 for CCid=DACO Page 2

```
59      20      CONTINUE
60          THETA=THETA+DTHETA
61      10      CONTINUE
62      C
63          CALL CMOPEN
64      C
65      C***** DRAW THE CYLINDER.
66      C
67          CALL MOVE3D(PA(8,1),PA(8,2),PA(8,3))
68          DO 30 I=1,8
69              CALL DRAW3D(PA(I,1),PA(I,2),PA(I,3))
70      30      CONTINUE
71      C
72          CALL MOVE3D(PB(8,1),PB(8,2),PB(8,3))
73          DO 40 I=1,8
74              CALL DRAW3D(PB(I,1),PB(I,2),PB(I,3))
75      40      CONTINUE
76      C
77          DO 50 I=1,8
78              CALL MOVE3D(PA(I,1),PA(I,2),PA(I,3))
79              CALL DRAW3D(PB(I,1),PB(I,2),PB(I,3))
80      50      CONTINUE
81      C
82          CALL CMCLOS
83      C
84          RETURN
85          END
```

Listing of DRPINT.FOR at 14:44:29 on AUG 9, 1984 for CCid=DAC0 Page 1

```

1      SUBROUTINE DRPINT(PINT)
2      C
3      C***** THIS SUBROUTINE DRAWS A POINT 'PINT' ON THE SCREEN AS A CROSS.
4      C
5      C      'IGL' GRAPHICS SUBROUTINES CALLED:      CMCLOS
6      C                                              CMOPEN
7      C                                              DRAW3D
8      C                                              MOVE3D
9      C
10     C*****
11     C
12     REAL PINT(3)
13     C
14     CALL CMOPEN
15     P1=PINT(1)-1.
16     P2=PINT(1)+1.
17     CALL MOVE3D(P1,PINT(2),PINT(3))
18     CALL DRAW3D(P2,PINT(2),PINT(3))
19     P1=PINT(2)-1.
20     P2=PINT(2)+1.
21     CALL MOVE3D(PINT(1),P1,PINT(3))
22     CALL DRAW3D(PINT(1),P2,PINT(3))
23     P1=PINT(3)-1.
24     P2=PINT(3)+1.
25     CALL MOVE3D(PINT(1),PINT(2),P1)
26     CALL DRAW3D(PINT(1),PINT(2),P2)
27     CALL CMCLOS
28     C
29     RETURN
30     END
```

Listing of DRVOL.FOR at 14:45:00 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE DRVOL
2      C
3      C      THIS SUBROUTINE DRAWS THE SWEPT VOLUME ON THE SCREEN.
4      C
5      C      COMMON BLOCKS ACCESSED: /CORNER/
6      C
7      C      'IGL' GRAPHICS SUBROUTINES CALLED:      CMCLOS
8      C                                              CMOPEN
9      C                                              DRAW3D
10     C                                              MOVE3D
11     C
12     C*****
13     C
14     C      COMMON /CORNER/P1(3),P2(3),P3(3),P4(3),P5(3),P6(3),P7(3),P8(3)
15     C
16     C
17     C      CALL CMOPEN
18     C
19     C***** DRAW SWEPT VOLUME.
20     C
21     C      CALL MOVE3D(P1(1),P1(2),P1(3))
22     C      CALL DRAW3D(P2(1),P2(2),P2(3))
23     C      CALL DRAW3D(P3(1),P3(2),P3(3))
24     C      CALL DRAW3D(P4(1),P4(2),P4(3))
25     C      CALL DRAW3D(P1(1),P1(2),P1(3))
26     C      CALL DRAW3D(P5(1),P5(2),P5(3))
27     C      CALL DRAW3D(P6(1),P6(2),P6(3))
28     C      CALL DRAW3D(P7(1),P7(2),P7(3))
29     C      CALL DRAW3D(P8(1),P8(2),P8(3))
30     C      CALL DRAW3D(P5(1),P5(2),P5(3))
31     C      CALL MOVE3D(P2(1),P2(2),P2(3))
32     C      CALL DRAW3D(P6(1),P6(2),P6(3))
33     C      CALL MOVE3D(P3(1),P3(2),P3(3))
34     C      CALL DRAW3D(P7(1),P7(2),P7(3))
35     C      CALL MOVE3D(P4(1),P4(2),P4(3))
36     C      CALL DRAW3D(P8(1),P8(2),P8(3))
37     C
38     C***** DRAW AXES ON SCREEN
39     C
40     C      CALL MOVE3D(0.,0.,0.)
41     C      CALL DRAW3D(100.,0.,0.)
42     C      CALL MOVE3D(0.,0.,0.)
43     C      CALL DRAW3D(0.,100.,0.)
44     C      CALL MOVE3D(0.,0.,0.)
45     C      CALL DRAW3D(0.,0.,100.)
46     C
47     C      CALL CMCLOS
48     C
49     C      RETURN
50     C      END
```

Listing of ECROSS.FOR at 14:45:01 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE ECROSS(NVERT,ICOUNT,UR,PINT,LISTE,NECRSS)
2      C
3      C      THIS SUBROUTINE COUNTS THE NUMBER OF TIMES A PROJECTED RAY
4      C      CROSSES FACE BOUNDARIES WHOSE ENDPOINTS ARE NOT ON THE RAY.
5      C
6      C      INPUT ARGUMENTS:
7      C          NVERT- INDEX OF FIRST VERTEX OF FACE.
8      C          ICOUNT- NUMBER OF VERTICES ON FACE.
9      C          UR- DIRECTION VECTOR OF RAY.
10     C          PINT- ORIGIN OF RAY.
11     C          LISTE- LIST OF EDGES WITH NEITHER VERTEX ON THE RAY.
12     C
13     C      OUTPUT ARGUMENTS:
14     C          NECRSS- NUMBER OF TIMES THE RAY CROSSES FACE BOUNDARIES AT
15     C                  EDGES.
16     C
17     C      COMMON BLOCKS ACCESSED: /DATA/
18     C
19     C      SUBROUTINES CALLED:      EDGINT
20     C
21     C*****
22     C
23     REAL UR(3),PINT(3),P1(3),P2(3)
24     INTEGER IEDGE(20),LISTE(10)
25     C
26     COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
27     C
28     NECRSS=0
29     C
30     C***** TEST ALL EDGES WHICH HAVE NEITHER VERTEX ON THE RAY.
31     C
32     DO 15 I=1,100
33     C
34     C***** FIND THE NEXT EDGE.
35     C
36     INDEX=LISTE(I)
37     IF(INDEX.EQ.0) GO TO 900
38     C
39     C***** FIND THE EDGE ENDPOINTS, 'P1','P2'.
40     C
41     DO 20 J=1,3
42     P1(J)=VERT(INDEX,J)
43     P2(J)=VERT(INDEX+1,J)
44     20 CONTINUE
45     C
46     C***** TEST IF THE RAY INTERSECTS THE EDGE.
47     C
48     CALL EDGINT(P1,P2,UR,PINT,INT)
49     IF(INT.EQ.1) NECRSS=NECRSS+1
50     C
51     15 CONTINUE
52     C
53     900 CONTINUE
54     C
55     RETURN
56     END

```

Listing of EDGCRS.FOR at 14:45:01 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE EDGCRS(UR,P1,P2,INT)
2      C
3      C      THIS SUBROUTINE FINDS IF A LINE SEGMENT AND A RAY INTERSECT.
4      C      THE LINE SEGMENT IS DEFINED BY ENDPOINTS P1(3),P2(3).
5      C      THE RAY STARTING POINT IS THE ORIGIN, WITH DIRECTION GIVEN BY
6      C      UR(3).
7      C
8      C      LET UE(3)=P2(3)-P1(3)
9      C
10     C      LINE SEGMENT:  R(3)=P1(3)+TLAMB1*UE(3),          O.LE.TLAMB1.LE.1
11     C      RAY:          R(3)=TLAMB2*UR(3),              O.LE.TLAMB2
12     C
13     C      COMBINING:
14     C                      TLAMB1*UE(3)-TLAMB2*UR(3)=-P1(3)
15     C
16     C*****
17     C
18     C      REAL P1(3),P2(3),UR(3),M1(2),M2(2),UE(3),M3(2)
19     C
20     C      INT=0
21     C
22     C***** FIND THE LINE DIRECTION VECTOR 'UE'.
23     C
24     C      DO 10 I=1,3
25     C          UE(I)=P2(I)-P1(I)
26     C      10 CONTINUE
27     C
28     C***** MAKE SURE THAT THE LINE SEGMENT LENGTH IS GREATER THAN ZERO.
29     C
30     C      UELEN=UE(1)**2+UE(2)**2+UE(3)**2
31     C      IF(UELEN.LE..00001) GO TO 500
32     C
33     C***** THERE ARE THREE EQUATIONS AND TWO UNKNOWNNS. FIND TWO EQUATIONS
34     C***** WHICH GIVE A NON-SINGULAR 2X2 MATRIX.
35     C
36     C      DO 20 I=1,3
37     C
38     C          J=I+1
39     C          IF(I.EQ.3) J=1
40     C
41     C          M1(1)=UE(I)
42     C          M1(2)=UE(J)
43     C          M2(1)=-UR(I)
44     C          M2(2)=-UR(J)
45     C
46     C      CALL DET2X2(M1,M2,DET)
47     C
48     C      IF(ABS(DET).GT..00001) GO TO 100
49     C
50     C      20 CONTINUE
51     C
52     C      100 CONTINUE
53     C
54     C***** IF THERE IS NO NON-SINGULAR MATRIX, NO SOLUTION EXISTS.
55     C
56     C      IF(ABS(DET).LE..00001) GO TO 500
57     C
58     C***** SOLVE THE TWO EQUATIONS FOR THE PARAMETERS 'TLAMB1','TLAMB2'.
```

Listing of EDGCRS.FOR at 14:45:01 on AUG 9, 1984 for CCId=DACO Page 2

```
59      C
60          M3(1)=-P1(I)
61          M3(2)=-P1(J)
62      C
63          CALL DET2X2(M3,M2,TEMP1)
64          CALL DET2X2(M1,M3,TEMP2)
65      C
66          TLAMB1=TEMP1/DET
67          TLAMB2=TEMP2/DET
68      C
69      C***** TEST IF THE INTERSECTION POINT LIES ON THE RAY AND ON THE
70      C***** LINE SEGMENT.
71      C
72          IF(TLAMB1.LT.O..OR.TLAMB1.GT.1.) GO TO 500
73          IF(TLAMB2.GT.O.) INT=1
74      C
75      500      CONTINUE
76              RETURN
77              END
```

Listing of EDGINT.FOR at 14:45:19 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE EDGINT(P1,P2,UR,PINT,INT)
2      C
3      C      THIS SUBROUTINE FINDS IF A LINE SEGMENT AND A RAY INTERSECT.
4      C
5      C      INPUT ARGUMENTS:
6      C          P1(3),P2(3)- END POINTS OF THE LINE SEGMENT
7      C          UR(3)- DIRECTION VECTOR OF THE RAY
8      C          PINT(3)- ORIGIN OF THE RAY
9      C
10     C      OUTPUT ARGUMENTS:
11     C          INT- 0=NO INTERSECTION
12     C              1=INTERSECTION
13     C
14     C      SUBROUTINES CALLED:      DET2X2
15     C
16     C
17     C      EQUATION OF LINE SEGMENT:      R=P1+TLAMB1*UE,      0.LE.TLAMB1.LE.1
18     C      EQUATION OF RAY:              R=PINT+TLAMB2*UR, TLAMB2.GE.0
19     C
20     C      COMBINING:
21     C
22     C          TLAMB1*UE-TLAMB2*UR=PINT-P1
23     C
24     C      WE GET THREE EQUATIONS IN TWO UNKNOWNNS.
25     C      SOLVE FOR PARAMETERS 'TLAMB1','TLAMB2' AND TEST THEIR VALUES.
26     C
27     C*****
28     C
29     C      REAL P1(3),P2(3),UR(3),PINT(3),M1(2),M2(2),UE(3),M3(2)
30     C
31     C      INT=0
32     C
33     C***** CALCULATE THE DIRECTION VECTOR 'UE' OF THE LINE SEGMENT.
34     C
35     C      DO 10 I=1,3
36     C          UE(I)=P2(I)-P1(I)
37     C      10 CONTINUE
38     C
39     C***** FIND TWO EQUATIONS OF THE THREE WHICH WILL GIVE A NON-SINGULAR
40     C***** SOLUTION.
41     C
42     C      DO 20 I=1,3
43     C          J=I+1
44     C          IF(I.EQ.3) J=1
45     C
46     C          M1(1)=UE(I)
47     C          M1(2)=UE(J)
48     C          M2(1)=-UR(I)
49     C          M2(2)=-UR(J)
50     C
51     C          CALL DET2X2(M1,M2,DET)
52     C
53     C          IF(DET.NE.0.) GO TO 100
54     C      20 CONTINUE
55     C
56     C***** IF NO NON-SINGULAR SOLUTION EXISTS, THE LINE AND RAY ARE PARALLEL
57     C***** AND THERE IS NO INTERSECTION.
58     C

```

Listing of EDGINT.FOR at 14:45:19 on AUG 9, 1984 for CCid=DAC0 Page 2

```
59          GO TO 500
60      C
61      100      CONTINUE
62      C
63      C***** FIND THE CORRESPONDING VALUES ON THE RIGHT HAND SIDE OF THE TWO
64      C***** EQUATIONS.
65      C
66          M3(1)=PINT(I)-P1(I)
67          M3(2)=PINT(J)-P1(J)
68      C
69      C***** SOLVE FOR THE PARAMETERS 'TLAMB1','TLAMB2'.
70      C
71          CALL DET2X2(M3,M2,TEMP1)
72          CALL DET2X2(M1,M3,TEMP2)
73      C
74          TLAMB1=TEMP1/DET
75          TLAMB2=TEMP2/DET
76      C
77      C***** TEST 'TLAMB1','TLAMB2'.
78      C
79          IF(TLAMB1.LT.O..OR.TLAMB1.GT.1.) GO TO 500
80          IF(TLAMB2.GT.O.) INT=1
81      C
82      500      CONTINUE
83          RETURN
84          END
```

Listing of ELIST.FOR at 14:45:20 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE ELIST(NVERT,ICOUNT,LISTV,LISTE)
2      C
3      C      THIS SUBROUTINE GENERATES A LIST OF EDGE STARTING VERTICES
4      C      OF EDGES FOR WHICH NEITHER VERTEX LIES ON THE TEST RAY.
5      C
6      C      INPUTS: NVERT- LOCATION OF FIRST VERTEX IN FACE VERTEX LIST.
7      C      ICOUNT- NUMBER OF FACE VERTICES.
8      C      LISTV- LIST OF FACE VERTICES WHICH LIE ON THE TEST RAY.
9      C
10     C      OUTPUTS: LISTE- LIST OF FACE VERTICES NOT ON TEST RAY.
11     C
12     C*****
13     C
14     C      INTEGER LISTV(10),LISTE(100)
15     C
16     C
17     C      INDEX=1
18     C
19     C***** FIND THE FIRST VERTEX IN LISTV.
20     C
21     C      IND=LISTV(1)-NVERT
22     C
23     C***** IF IND.LE.1, THERE ARE NO EDGES BEFORE LISTV(1)
24     C
25     C      IF(IND.LE.1) GO TO 150
26     C
27     C***** ELSE ENTER EDGES INTO LISTE.
28     C
29     C      DO 10 I=1,IND-1
30     C          LISTE(INDEX)=I+NVERT-1
31     C          INDEX=INDEX+1
32     C      10 CONTINUE
33     C
34     C      150 CONTINUE
35     C
36     C***** FIND THE VERTICES BETWEEN THE ENTRIES IN LISTV, AND ENTER THE
37     C***** EDGES INTO LISTE.
38     C
39     C      DO 20 I=1,100
40     C          MV1=LISTV(I)
41     C          MV2=LISTV(I+1)
42     C          IF(LISTV(I+1).EQ.0) GO TO 200
43     C          IF(MV2.EQ.NVERT) MV2=NVERT+ICOUNT-1
44     C
45     C          NCOUNT=MV2-MV1
46     C          DO 30 J=1,NCOUNT-2
47     C              LISTE(INDEX)=MV1+J
48     C              INDEX=INDEX+1
49     C          30 CONTINUE
50     C
51     C      20 CONTINUE
52     C
53     C***** FIND ANY EDGES AFTER THE LAST VERTEX IN LISTV
54     C
55     C      200 CONTINUE
56     C
57     C      LEFT=NVERT+ICOUNT-1-MV1
58     C      IF(LEFT.LT.2) GO TO 900
```

Listing of ELIST.FOR at 14:45:20 on AUG 9, 1984 for CCId=DAC0 Page 2

```
59      C
60          DO 40 K=1,LEFT-1
61              LISTE(INDEX)=MV1+K
62              INDEX=INDEX+1
63      40      CONTINUE
64      C
65      C***** TEST IF THE LAST VERTEX TO THE FIRST VERTEX IS A VALID EDGE.
66      C
67      900      CONTINUE
68              IF(LEFT.GT.O.AND.IND.NE.O) GO TO 950
69              GO TO 999
70      950      CONTINUE
71              LISTE(INDEX)=NVERT+ICOUNT-1
72      999      CONTINUE
73      C
74      RETURN
75      END
```

Listing of FACE.FOR at 14:45:20 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE FACE
2      C
3      C      THIS SUBROUTINE ALLOWS THE USER TO INTERACTIVELY SPECIFY AN
4      C      ARBITRARY NUMBER OF COPLANAR VERTICES DEFINING A POLYHEDRAL
5      C      FACE. THE FACE DATA IS STORED IN THE CORRECT FORMAT IN THE DATA
6      C      STRUCTURES 'VERT','FACES'.
7      C
8      C      COMMON BLOCKS ACCESSED: /DATA/
9      C
10     C      SUBROUTINES CALLED:      DOTPRD
11     C                               XPROD
12     C
13     C      'IGL' GRAPHICS SUBROUTINES:      CMCLOS
14     C                                           CMOPEN
15     C                                           DRAW3D
16     C                                           MOVE3D
17     C
18     C
19     C*****
20     C
21     REAL T1(3),T2(3),U(3),P1(3),PINT(3),P2(3),V1(3)
22     C
23     COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
24     C
25     C***** USER ENTERS THE NUMBER OF VERTICES.
26     C
27     WRITE(5,100)
28     100  FORMAT($,' ENTER NUMBER OF FACE VERTICES ')
29     READ(5,*) IFACE(1,2)
30     C
31     IFACE(1,1)=1
32     IOBJ(1,1)=1
33     IOBJ(1,2)=1
34     C
35     C***** USER ENTERS VERTEX COORDINATES.
36     C
37     DO 10 I=1,IFACE(1,2)
38     WRITE(5,101) I
39     101  FORMAT($,' ENTER VERTEX('',I2,'')(X,Y,Z) ')
40     READ(5,*) (VERT(I,J),J=1,3)
41     10   CONTINUE
42     C
43     DO 50 II=1,3
44     VERT(IFACE(1,2)+1,II)=VERT(1,II)
45     V1(II)=VERT(1,II)
46     T1(II)=VERT(2,II)-VERT(1,II)
47     T2(II)=VERT(3,II)-VERT(1,II)
48     50   CONTINUE
49     C
50     C***** CALCULATE FACE PLANE SURFACE NORMAL.
51     C
52     CALL XPROD(T1,T2,U)
53     C
54     DO 51 II=1,3
55     FACES(1,II)=U(II)/SQRT(U(1)**2+U(2)**2+U(3)**2)
56     51   CONTINUE
57     C
58     C***** CALCULATE PLANE DISTANCE FROM ORIGIN.

```

Listing of FACE.FOR at 14:45:20 on AUG 9, 1984 for CCid=DACD Page 2

```
59      C
60          CALL DOTPRD(V1,U,FACES(1,4))
61      C
62          CALL CMOPEN
63      C
64          NVERT=IFACE(1,2)
65      C
66      C***** DRAW RESULTING FACE POLYGON.
67      C
68          CALL MOVE3D(VERT(NVERT,1),VERT(NVERT,2),VERT(NVERT,3))
69      C
70          DO 30 I=1,NVERT
71              CALL DRAW3D(VERT(I,1),VERT(I,2),VERT(I,3))
72      30    CONTINUE
73      C
74          CALL CMCLOS
75      C
76      C
77          RETURN
78      END
```

Listing of FACINT.FOR at 14:45:50 on AUG 9, 1984 for CCId=DACD Page 1

```

1      C
2      RETURN
3      END
4      SUBROUTINE FACINT(NFACE,P1,P2,INT)
5      C
6      C      THIS SUBROUTINE TESTS IF A LINE SEGMENT INTERSECTS A FACE.
7      C
8      C      INPUT ARGUMENTS:
9      C          NFACE- INDEX IDENTIFYING FACE LOCATION IN FACE LIST.
10     C          P1(3),P2(3)- END POINTS OF LINE SEGMENT.
11     C
12     C      OUTPUT ARGUMENTS:
13     C          INT- 0=NO INTERSECTION FOUND
14     C              1=INTERSECTION FOUND
15     C
16     C      SUBROUTINES CALLED:      DRPINT
17     C                               INSIDE
18     C                               PLANE
19     C
20     C*****
21     C
22     C      REAL T1(3),T2(3),U(3),P1(3),PINT(3),P2(3),V1(3)
23     C
24     C      COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
25     C
26     C      INT=0
27     C
28     C***** TEST IF LINE SEGMENT INTERSECTS FACE PLANE.
29     C
30     C      CALL PLANE(NFACE,P1,P2,INT,PINT)
31     C
32     C      IF(INT.EQ.0) GO TO 900
33     C      IF(INT.EQ.1) GO TO 201
34     C
35     201 CONTINUE.
36     C
37     C***** TEST IF INTERSECTION POINT LIES INSIDE THE FACE BOUNDARIES.
38     C
39     C      CALL INSIDE(NFACE,PINT,INT)
40     C
41     C      IF(INT.EQ.0) GO TO 900
42     C
43     C***** DRAW THE INTERSECTION POINT ON THE SCREEN.
44     C
45     C      CALL DRPINT(PINT)
46     C
47     900 CONTINUE
48     C
49     C      RETURN
50     C      END
51     SUBROUTINE FACYL(NFACE,A,B,INT)
52     C
53     C      THIS SUBROUTINE TESTS IF EITHER END OF A CYLINDER INTERSECTS
54     C      A FACE.
55     C
56     C      INPUTS: NFACE- FACE IDENTIFIER.
57     C              A,B- ENDOPOINTS OF CYLINDER SPINE.
58     C
```

Listing of FACINT.FOR at 14:45:50 on AUG 9, 1984 for CCid=DACD Page 2

```
59      C      OUTPUTS: INT- 0=NO INTERSECTION, 1=INTERSECTION.
60      C
61      C      SUBROUTINES CALLED:  DIFF
62      C                               DOTPRD
63      C                               DRPINT
64      C                               FACINT
65      C                               XPROD
66      C
67      C*****
68      C
69      REAL A(3),B(3),ENDPT(2,3),PINT(3)
70      REAL TEMP(3),RK(3),Q(3),RAD(3),U(3),POINT(3)
71      C
72      COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
73      COMMON /RADIUS/RADIUS
74      C
75      INT=0
76      C
77      C*****  FIND THE FACE PLANE PARAMETERS U(I),P
78      C
79      DO 10 I=1,3
80      U(I)=FACES(NFACE,I)
81      10    CONTINUE
82      C
83      P=FACES(NFACE,4)
84      C
85      C*****  SET UP AN ENDPOINTS ARRAY.
86      C
87      DO 20 I=1,3
88      ENDPT(1,I)=A(I)
89      ENDPT(2,I)=B(I)
90      20    CONTINUE
91      C
92      CALL DIFF(B,A,RK)
93      C
94      DO 500 IND=1,2
95      C
96      DO 30 I=1,3
97      TEMP(I)=ENDPT(IND,I)
98      30    CONTINUE
99      C
100     CALL DOTPRD(TEMP,U,DOT)
101     DIST=DOT-P
102     IF(DIST.GT.RADIUS.OR.DIST.LT.O.) GO TO 900
103     C
104     CALL XPROD2(U,RK,Q)
105     CALL XPROD(Q,RK,RAD)
106     C
107     CALL DOTPRD(RAD,U,UDOT)
108     IF(ABS(UDOT).LT..000001) GO TO 900
109     C
110     RADLEN=DIST/UDOT
111     IF(RADLEN.GT.RADIUS) GO TO 900
112     C
113     DO 40 I=1,3
114     POINT(I)=TEMP(I)+RADIUS*RAD(I)
115     40    CONTINUE
116     C
```

Listing of FACINT.FOR at 14:45:50 on AUG 9, 1984 for CCid=DACD Page 3

```
117          CALL FACINT(NFACE,TEMP,POINT,INT,PINT)
118          IF(INT.EQ.1) GO TO 900
119      C
120      500      CONTINUE
121      C
122      900      CONTINUE
123          IF(INT.EQ.1) CALL DRPINT(PINT)
124      C
125          RETURN
126          END
```

Listing of FACYL.FOR at 14:45:52 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE FACYL(NFACE,A,B,INT)
2      C
3      C      THIS SUBROUTINE TESTS IF EITHER END OF A CYLINDER INTERSECTS
4      C      A FACE.
5      C
6      C      INPUTS: NFACE- FACE IDENTIFIER.
7      C      A,B- ENDPOINTS OF CYLINDER SPINE.
8      C
9      C      OUTPUTS: INT- 0=NO INTERSECTION, 1=INTERSECTION.
10     C
11     C      SUBROUTINES CALLED:  DIFF
12     C                        DOTPRD
13     C                        DRPINT
14     C                        FACINT
15     C                        XPROD
16     C
17     C*****
18     C
19     REAL A(3),B(3),ENDPT(2,3),PINT(3)
20     REAL TEMP(3),RK(3),Q(3),RAD(3),U(3),POINT(3)
21     C
22     COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
23     COMMON /RADIUS/RADIUS
24     C
25     INT=0
26     C
27     C*****  FIND THE FACE PLANE PARAMETERS U(I),P
28     C
29     DO 10 I=1,3
30         U(I)=FACES(NFACE,I)
31     10 CONTINUE
32     C
33     P=FACES(NFACE,4)
34     C
35     C*****  SET UP AN ENDPOINTS ARRAY.
36     C
37     DO 20 I=1,3
38         ENDPT(1,I)=A(I)
```

Listing of GENVOL.FOR at 14:45:52 on AUG 9, 1984 for CCId=DAC0 Page 1

```

1      SUBROUTINE GENVOL(A1,B1,A2,B2)
2      C
3      C      THIS SUBROUTINE FINDS THE PARAMETERS WHICH DESCRIBE THE SWEEP VOLUME.
4      C
5      C      INPUT ARGUMENTS: A1(3),B1(3)- INITIAL CYLINDER SPINE ENDPOINTS.
6      C                        A2(3),B2(3)- FINAL CYLINDER SPINE ENDPOINTS.
7      C
8      C      COMMON BLOCKS ACCESSED: /LINK/,/RADIUS/
9      C      COMMON BLOCKS MODIFIED: /VOLUME/,/SNORM/
10     C
11     C*****
12     C
13     REAL A1(3),B1(3),A2(3),B2(3)
14     REAL AN(3),BN(3),CN(3)
15     REAL LEN,RADIUS
16     C
17     COMMON /VOLUME/P(3),UO(3),U1(3),VO(3),V1(3)
18     COMMON /SNORM/DAN(3),DBN(3),DCN(3)
19     COMMON /LINK/H1(4,4),H2(4,4),LEN
20     COMMON /RADIUS/RADIUS
21     C
22     C***** CALCULATE VECTORS UO,U1,VO,V1
23     C
24     C
25     DO 10 I=1,3
26         UO(I)=A2(I)-A1(I)
27         U1(I)=B2(I)-B1(I)
28         VO(I)=B1(I)-A1(I)
29         V1(I)=B2(I)-A2(I)
30     10 CONTINUE
31     C
32     C
33     C***** CALCULATE THE UNIT SURFACE NORMAL
34     C
35     AN(1)=UO(2)*VO(3)-UO(3)*VO(2)
36     AN(2)=-UO(1)*VO(3)+UO(3)*VO(1)
37     AN(3)=UO(1)*VO(2)-UO(2)*VO(1)
38     C
39     BN(1)=UO(2)*(V1(3)-VO(3))-UO(3)*(V1(2)-VO(2))
40     BN(2)=-UO(1)*(V1(3)-VO(3))+UO(3)*(V1(1)-VO(1))
41     BN(3)=UO(1)*(V1(2)-VO(2))-UO(2)*(V1(1)-VO(1))
42     C
43     CN(1)=VO(3)*(V1(2)-VO(2))-VO(2)*(V1(3)-VO(3))
44     CN(2)=-VO(3)*(V1(1)-VO(1))+VO(1)*(V1(3)-VO(3))
45     CN(3)=VO(2)*(V1(1)-VO(1))-VO(1)*(V1(2)-VO(2))
46     C
47     C***** MULTIPLY THE UNIT SURFACE NORMAL BY THE RADIUS TO GET
48     C***** 'DAN','DBN','DCN'.
49     C
50     DO 30 I=1,3
51         DAN(I)=RADIUS*AN(I)
52         DBN(I)=RADIUS*BN(I)
53         DCN(I)=RADIUS*CN(I)
54     30 CONTINUE
55     C
56     C
57     RETURN
58     END

```

Listing of INITSV.FOR at 14:46:10 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE INITSV(FILNAM)
2      C
3      C***** THIS SUBROUTINE INITIALIZES THE LINK PARAMETERS.
4      C
5      C      INPUT PARAMETERS: FILNAM- NAME OF FILE CONTAINING LINK DIMENSIONS
6      C                          AND INITIAL AND FINAL POSITIONS.
7      C
8      C      COMMON BLOCKS INITIALIZED: /LINK/,/RADIUS/
9      C
10     C      SUBROUTINES CALLED:      OPEN
11     C                          SHUT
12     C
13     C
14     C      DEFINITION OF CONSTANTS AND VARIABLES.
15     C
16     C      H1,H2- HOMOGENEOUS TRANSFORMATIONS SPECIFYING THE INITIAL AND
17     C              FINAL POSITION OF THE LINK.
18     C      LEN,RADIUS- LINK LENGTH AND RADIUS.
19     C
20     C*****
21     C
22     C      BYTE FILNAM(20)
23     C      REAL LEN,RADIUS
24     C
25     C      COMMON /LINK/H1(4,4),H2(4,4),LEN
26     C      COMMON /RADIUS/RADIUS
27     C
28     C***** READ 'H1','H2','LEN','RADIUS' FROM THE FILE 'FILNAM'.
29     C
30     C      CALL OPEN(1,FILNAM,4,1)
31     C
32     C      READ(1,*)
33     C      DO 10 I=1,4
34     C          READ(1,*) (H1(I,J),J=1,4)
35     C10    CONTINUE
36     C      READ(1,*)
37     C      DO 20 I=1,4
38     C          READ(1,*) (H2(I,J),J=1,4)
39     C20    CONTINUE
40     C      READ(1,*)
41     C      READ(1,*) LEN,RADIUS
42     C
43     C      CALL SHUT(1)
44     C
45     C      RETURN
46     C      END
```

Listing of INPROJ.FOR at 14:46:11 on AUG 9, 1984 for CCId=DACD Page 1

```
1      SUBROUTINE INPROJ(P1,P2,P3,P4,INT)
2      C
3      REAL P1(3),P2(3),P3(3),P4(3)
4      C
5      INT=0
6      ICDUNT=0
7      C
8      CALL EDGCRS(P1,P2,P3,INT1)
9      IF (INT1.EQ.1) ICDUNT=ICDUNT+1
10     CALL EDGCRS(P1,P4,P3,INT1)
11     IF (INT1.EQ.1) ICDUNT=ICDUNT+1
12     C
13     ICASE=1
14     C
15     TOTAL1=0.
16     TOTAL2=0.
17     C
18     DO 10 I=1,3
19         DIFF1=(P1(I)-P2(I))**2
20         DIFF2=(P1(I)-P4(I))**2
21         TOTAL1=TOTAL1+DIFF1
22         TOTAL2=TOTAL2+DIFF2
23     10 CONTINUE
24     C
25     IF(TOTAL1.LT..00001) ICASE=2
26     IF(TOTAL2.LT..00001) ICASE=3
27     C
28     GO TO (100,200,300) ICASE
29     C
30     100 CONTINUE
31     CALL VRTCERS(P1,P2,P4,INT2)
32     IF (INT2.EQ.1) ICDUNT=ICDUNT+1
33     GO TO 400
34     C
35     200 CONTINUE
36     CALL VRTCERS(P1,P3,P4,INT2)
37     IF (INT2.EQ.1) ICDUNT=ICDUNT+1
38     GO TO 400
39     C
40     300 CONTINUE
41     CALL VRTCERS(P1,P2,P3,INT2)
42     IF (INT2.EQ.1) ICDUNT=ICDUNT+1
43     C
44     400 CONTINUE
45     C
46     ITEST=MOD(ICDUNT,2)
47     C
48     IF (ITEST.EQ.1) INT=1
49     C
50     RETURN
51     END
```

Listing of INSIDE.FOR at 14:46:11 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE INSIDE(NFACE,PINT,INT)
2      C
3      C      THIS SUBROUTINE TESTS IF AN INTERSECTION POINT 'PINT'
4      C      LIES INSIDE THE FACE 'NFACE'
5      C
6      C      INPUT ARGUMENTS:
7      C          NFACE- INDEX IDENTIFYING FACE IN FACE LIST.
8      C          PINT- INTERSECTION POINT LYING IN PLANE OF FACE.
9      C
10     C      OUTPUT ARGUMENTS:
11     C          INT- 0='PINT' NOT ON FACE.
12     C          1='PINT' ON FACE.
13     C
14     C      COMMON BLOCKS ACCESSED:
15     C          /DATA/VERT,FACES,IFACE,IOBJ
16     C
17     C      SUBROUTINES CALLED:      ECROSS
18     C                               ELIST
19     C                               VCROSS
20     C
21     C*****
22     C
23     REAL PINT(3),V(3),UR(3)
24     INTEGER LISTV(10),LISTE(100)
25     C
26     COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
27     C
28     NCROSS=0
29     INT=0
30     C
31     C***** FIND STARTING LOCATION 'NVERT' AND NUMBER OF VERTICES 'NCOUNT' IN
32     C***** VERTEX LIST.
33     C
34     NVERT=IFACE(NFACE,1)
35     NCOUNT=IFACE(NFACE,2)
36     C
37     C***** DEFINE A RAY 'UR' WITH ORIGIN AT 'PINT' AND CONTAINING THE FIRST
38     C***** VERTEX 'V' IN THE FACE VERTEX LIST.
39     C
40     DO 10 I=1,3
41         V(I)=VERT(NVERT,I)
42         UR(I)=V(I)-PINT(I)
43     10 CONTINUE
44     C
45     C***** FIND THE NUMBER OF TIMES THE RAY CROSSES FACE BOUNDARIES AT
46     C***** FACE VERTICES.
47     C
48     CALL VCROSS(NVERT,NCOUNT,UR,PINT,LISTV,NVCRSS)
49     C
50     C***** MAKE A LIST OF ALL THE FACE EDGES WHICH HAVE NEITHER
51     C***** VERTEX ON THE RAY.
52     C
53     CALL ELIST(NVERT,NCOUNT,LISTV,LISTE)
54     C
55     C***** FIND THE NUMBER OF EDGES IN THE LIST 'LISTE' CROSSED BY THE RAY.
56     C
57     CALL ECROSS(NVERT,NCOUNT,UR,PINT,LISTE,NECRSS)
58     C
```

Listing of INSIDE.FOR at 14:46:11 on AUG 9, 1984 for CCId=DACO Page 2

```
59      C***** THE TOTAL NUMBER OF BOUNDARY CROSSINGS IS THE SUM OF THE NUMBER OF
60      C***** CROSSINGS AT VERTICES AND THE NUMBER OF CROSSINGS AT EDGES.
61      C
62      NCROSS=NVCROSS+NECROSS
63      C
64      C***** IF THE NUMBER OF BOUNDARY CROSSINGS IS ODD, THE INTERSECTION POINT
65      C***** 'PINT' LIES ON THE FACE.
66      C
67      ITEST=MOD(NCROSS,2)
68      C
69      IF(ITEST.EQ.1) INT=1
70      C
71      RETURN
72      END
```

Listing of INVOL.FOR at 14:46:32 on AUG 9, 1984 for CCId=DACO Page 1

```

1      SUBROUTINE INVOL(NFACE,INT)
2      C
3      C      THIS SUBROUTINE CALCULATES WHETHER A GIVEN POINT LIES INSIDE
4      C      A PARAMETRIC SWEEP VOLUME. SINCE ANY POINT CAN BE USED, THE ROUTINE
5      C      USES THE FIRST VERTEX OF THE FACE 'NFACE'.
6      C
7      C      INPUT ARGUMENTS: NFACE- POINTER TO FACE LOCATION IN FACE LIST.
8      C
9      C      OUTPUT ARGUMENTS: INT- INTERSECTION FLAG, 0=NO INTERSECTION,
10     C      1=INTERSECTION.
11     C
12     C      SUBROUTINES CALLED:      NWTVOL
13     C      VLEN
14     C
15     C
16     C      LET THE SURFACE NORMAL BE N,
17     C
18     C      N=DAN+RLAMB(1)*DBN+RLAMB(2)*DCN
19     C
20     C      THE VOLUME EQUATION IS THEN
21     C
22     C      F(X)=RLAMB(1)*UO+RLAMB(2)*VO+RLAMB(3)*DAN+RLAMB(1)*RLAMB(2)*DELU+
23     C      RLAMB(1)*RLAMB(3)*DBN+RLAMB(2)*RLAMB(3)*DCN+G
24     C
25     C      0.LE.RLAMB(1).LE.1.
26     C      0.LE.RLAMB(2).LE.1.
27     C      -RADIUS/LENGTH(N).LE.RLAMB(3).LE.RADIUS/LENGTH(N)
28     C
29     C      WHERE DELU=U1-UO
30     C      G=P-VERT
31     C*****
32     C
33     C      REAL DELU(3),RLAMBO(3),RLAMB(3),G(3),N(3)
34     C
35     C      COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
36     C      COMMON /VOLUME/P(3),UO(3),U1(3),VO(3),V1(3)
37     C      COMMON /SNORM/DAN(3),DBN(3),DCN(3)
38     C      COMMON /RADIUS/RADIUS
39     C
40     C      INT=0
41     C
42     C***** CALCULATE CONSTANTS, INITIALIZE FIRST GUESS FOR SOLUTIONS RLAMB(3).
43     C
44     C      DO 10 I=1,3
45     C          DELU(I)=U1(I)-UO(I)
46     C          G(I)=P(I)-VERT(1,I)
47     C          RLAMB(I)=.5
48     C      10 CONTINUE
49     C
50     C***** SOLVE THE NON-LINEAR SYSTEM FOR RLAMB(3).
51     C***** THE ERROR LIMIT IS .01, AND THE MAXIMUM NUMBER OF ITERATIONS IS 10.
52     C
53     C      CALL NWTVOL(UO,VO,DAN,DELU,DBN,DCN,G,.01,10,RLAMBO,RLAMB,IFLAG)
54     C
55     C***** CALCULATE SURFACE NORMAL AT POINT DEFINED BY RLAMB(1),RLAMB(2).
56     C
57     C      DO 20 I=1,3
58     C          N(I)=DAN(I)+RLAMB(1)*DBN(I)+RLAMB(2)*DCN(I)

```

Listing of INVOL.FOR at 14:46:32 on AUG 9, 1984 for CCid=DACO Page 2

```
59      20      CONTINUE
60      C
61      CALL VLEN(N,RLEN)
62      C
63      C***** SCALE RLAMB(3)
64      C
65      RLAMB(3)=RLAMB(3)*RLEN/(2.*RADIUS)+.5
66      C
67      C***** TEST THE PARAMETERS RLAMB(I) AGAINST LIMITS TO DETERMINE
68      C***** IF POINT LIES INSIDE VOLUME.
69      C
70      DO 30 I=1,3
71          IF(RLAMB(I).LE.O.OR.RLAMB(I).GE.1.) GO TO 900
72      30      CONTINUE
73      INT=1
74      C
75      900      CONTINUE
76      C
77      RETURN
78      END
```

Listing of LINK.FOR at 14:46:32 on AUG 9, 1984 for CCid=DACO Page 1

```
1          SUBROUTINE LINK(A1,B1,A2,B2)
2      C
3      C      THIS SUBROUTINE FINDS THE COORDINATES OF THE LINK SPINE ENDPOINTS
4      C      AT THE BEGINNING AND END OF A MOTION.
5      C
6      C      INPUT ARGUMENTS: A1(3),B1(3)- SPINE ENDPOINTS IN INITIAL POSITION.
7      C                        A2(3),B2(3)- SPINE ENDPOINTS IN FINAL POSITION.
8      C
9      C      COMMON BLOCKS ACCESSED: /LINK/,/RADIUS/
10     C
11     C      SUBROUTINES CALLED:      TRANS3
12     C
13     C
14     C      DEFINITION OF CONSTANTS AND VARIABLES:
15     C
16     C      H1,H2- HOMOGENEOUS TRANSFORMATIONS FOR THE INITIAL AND FINAL POSITIONS.
17     C      A1,B1- SPINE ENDPOINTS IN INITIAL POSITION.
18     C      A2,B2- SPINE ENDPOINTS IN FINAL POSITION.
19     C
20     C*****
21     C
22     C      REAL A1(3),A2(3),B1(3),B2(3),TEMP(3)
23     C      REAL LEN,RADIUS
24     C
25     C      COMMON /LINK/H1(4,4),H2(4,4),LEN
26     C      COMMON /RADIUS/RADIUS
27     C
28     C      DATA TEMP/3*0./
29     C
30     C***** POINTS 'A1','A2' ARE AT THE ORIGINS OF THE HOMOGENEOUS COORDINATE
31     C***** FRAMES 'H1','H2'.
32     C
33     C      DO 10 I=1,3
34     C          A1(I)=H1(I,4)
35     C          A2(I)=H2(I,4)
36     C      10 CONTINUE
37     C
38     C      TEMP(1)=LEN
39     C
40     C***** TRANSLATE ALONG X-AXIS TO FIND 'B1','B2'
41     C
42     C      CALL TRANS3(H1,TEMP,B1)
43     C      CALL TRANS3(H2,TEMP,B2)
44     C
45     C
46     C      RETURN
47     C      END
```

Listing of MATDET.FOR at 14:46:32 on AUG 9, 1984 for CCid=DAC0 Page 1

```
1      SUBROUTINE MATDET(RMAT,DET)
2      C
3      C      THIS SUBROUTINE FINDS THE DETERMINANT OF A 3 X 3 MATRIX
4      C      OF THE FORM MATRIX(3,3)
5      C
6      C      INPUT ARGUMENT:
7      C          RMAT(3,3)- 3X3 MATRIX
8      C
9      C      OUTPUT ARGUMENT:
10     C          DET- DETERMINANT OF 'RMAT'.
11     C
12     C*****
13     C
14     C      REAL RMAT(3,3)
15     C
16     C      DET=RMAT(1,1)*(RMAT(2,2)*RMAT(3,3)-RMAT(2,3)*RMAT(3,2))-
17     C      *      RMAT(1,2)*(RMAT(2,1)*RMAT(3,3)-RMAT(2,3)*RMAT(3,1))+
18     C      *      RMAT(1,3)*(RMAT(2,1)*RMAT(3,2)-RMAT(2,2)*RMAT(3,1))
19     C
20     C      RETURN
21     C      END
```

Listing of NEWTON.FOR at 14:46:51 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE NEWTON(A,B,C,D,E,TOL,NUM,XO,X,IFLAG)
2      C
3      C      THIS SUBROUTINE USES NEWTONS METHOD FOR SOLVING A SYSTEM
4      C      OF NON-LINEAR EQUATIONS OF THE FORM:
5      C
6      C          F(X)=A*X(1)+B*X(2)+C*X(3)+D*X(1)*X(2)+E
7      C
8      C      INPUT ARGUMENTS:
9      C          A(3),B(3),C(3),D(3),E(3)- KNOWN COEFFICIENT VECTORS.
10     C          TOL- MAXIMUM ALLOWABLE ERROR.
11     C          NUM- MAXIMUM NUMBER OF ITERATIONS.
12     C          XO(3)- INITIAL GUESS FOR THE SOLUTION.
13     C
14     C      OUTPUT ARGUMENTS:
15     C          X(3)- APPROXIMATE SOLUTION.
16     C          IFLAG- 0=SOLUTION FOUND.
17     C                  1=NO SOLUTION, SINGULARITY EXISTS.
18     C
19     C      SUBROUTINES CALLED:      SOL3X3
20     C                               VLEN
21     C
22     C*****
23     C
24     C      REAL A(3),B(3),C(3),D(3),E(3),XO(3),X(3),FX(3),JAC(3,3),Y(3)
25     C
26     C      IFLAG=0
27     C
28     C***** INITIALIZE THE FIRST APPROXIMATION TO 'X' WITH THE VALUE OF 'XO'.
29     C
30     C      DO 1 I=1,3
31     C          X(I)=XO(I)
32     C      1 CONTINUE
33     C
34     C***** ITERATE THE SOLUTION A MAXIMUM OF 'NUM' TIMES.
35     C
36     C      DO 150 ICOUNT=1,NUM
37     C
38     C***** SOLVE FOR THE FUNCTION 'FX' FOR THE CURRENT VALUE OF 'X'.
39     C
40     C      DO 10 I=1,3
41     C          FX(I)=A(I)*X(1)+B(I)*X(2)+C(I)*X(3)+D(I)*X(1)*X(2)+E(I)
42     C      10 CONTINUE
43     C
44     C***** SOLVE THE JACOBIAN MATRIX.
45     C
46     C      DO 11 I=1,3
47     C          JAC(I,1)=A(I)+D(I)*X(2)
48     C          JAC(I,2)=B(I)+D(I)*X(1)
49     C          JAC(I,3)=C(I)
50     C      11 CONTINUE
51     C
52     C***** SOLVE FOR THE ERROR VECTOR 'Y'.
53     C
54     C      CALL SOL3X3(JAC,FX,Y,ISOLN)
55     C
56     C***** TEST IF A SOLUTION WAS FOUND.
57     C
58     C      IF(ISOLN.EQ.1) GO TO 950

```

Listing of NEWTON.FOR at 14:46:51 on AUG 9, 1984 for CCid=DACD Page 2

```
59      C
60      C***** ADD THE ERROR VECTOR 'Y' TO THE LAST VALUE OF 'X' TO GET THE NEXT
61      C***** APPROXIMATION TO 'X'.
62      C
63          DO 12 I=1,3
64              X(I)=X(I)+Y(I)
65      12      CONTINUE
66      C
67      C***** TEST IF THE MAGNITUDE OF THE ERROR IS LESS THAN 'TOL'.
68      C
69          CALL VLEN(Y,YLEN)
70          IF(YLEN.LT.TOL) GO TO 900
71      C
72      C***** TEST IF THE SOLUTION IS DIVERGING, INDICATING SINGULARITY.
73      C
74          CALL VLEN(X,XLEN)
75          IF(XLEN.GT.1000.) GO TO 950
76      C
77      150      CONTINUE
78      C
79      950      CONTINUE
80      C
81          IFLAG=1
82      C
83      900      CONTINUE
84      C
85          RETURN
86      END
```

Listing of NWTVOL.FOR at 14:46:51 on AUG 9, 1984 for CCId=DACO Page 1

```

1      SUBROUTINE NWTVOL(A,B,C,D,E,F,G,TOL,NUM,XO,X,IFLAG)
2      C
3      C      THIS SUBROUTINE USES NEWTONS METHOD FOR SOLVING A SYSTEM
4      C      OF NON-LINEAR EQUATIONS OF THE FORM:
5      C      F(X)=A*X(1)+B*X(2)+C*X(3)+D*X(1)*X(2)+E*X(1)*X(3)+
6      C      F*X(2)*X(3)+G
7      C
8      C      INPUTS: A(3),B(3),C(3),D(3),E(3),F(3),G(3)- CONSTANT COEFFICIENT
9      C      VECTORS.
10     C      TOL- MAXIMUM DESIRED ERROR.
11     C      NUM- NUMBER OF ITERATIONS ALLOWED.
12     C      XO- INITIAL GUESS FOR THE SOLUTION.
13     C
14     C      OUTPUTS: X- SOLUTION VECTOR.
15     C      IFLAG- 0-OK, 1-NO SOLUTION.
16     C
17     C      SUBROUTINES CALLED:      SOL3X3
18     C      VLEN
19     C
20     C*****
21     C
22     C      REAL A(3),B(3),C(3),D(3),E(3),XO(3),X(3),FX(3),JAC(3,3),Y(3)
23     C      REAL F(3),G(3)
24     C
25     C      IFLAG=0
26     C
27     C***** INITIALIZE THE FIRST GUESS FOR X(I).
28     C
29     C      DO 1 I=1,3
30     C      X(I)=XO(I)
31     C      CONTINUE
32     C
33     C***** ITERATE A MAXIMUM OF 'NUM' TIMES.
34     C
35     C      DO 150 ICOUNT=1,NUM
36     C
37     C***** CALCULATE CURRENT VALUE OF FUNCTION 'FX'.
38     C
39     C      DO 10 I=1,3
40     C      FX(I)=A(I)*X(1)+B(I)*X(2)+C(I)*X(3)+D(I)*X(1)*X(2)+
41     C      *      E(I)*X(1)*X(3)+F(I)*X(2)*X(3)+G(I)
42     C      CONTINUE
43     C
44     C***** CALCULATE THE JACOBIAN MATRIX.
45     C
46     C      DO 11 I=1,3
47     C      JAC(I,1)=A(I)+D(I)*X(2)+E(I)*X(3)
48     C      JAC(I,2)=B(I)+D(I)*X(1)+F(I)*X(3)
49     C      JAC(I,3)=C(I)+E(I)*X(1)+F(I)*X(2)
50     C      CONTINUE
51     C
52     C***** SOLVE FOR DIFFERENCE VECTOR 'Y'.
53     C
54     C      CALL SOL3X3(JAC,FX,Y,ISOLN)
55     C
56     C      IF(ISOLN.EQ.1) GO TO 950
57     C
58     C***** X(K)=X(K-1)+Y(K-1)

```

Listing of NWTVOL.FOR at 14:46:51 on AUG 9, 1984 for CCid=DAC0 Page 2

```
59      C
60          DO 12 I=1,3
61              X(I)=X(I)+Y(I)
62      12      CONTINUE
63      C
64      C***** TEST SIZE OF ERROR.
65      C
66          CALL VLEN(Y,YLEN)
67          IF(YLEN.LT.TOL) GO TO 900
68      C
69      C***** TEST IF SOLUTION IS DIVERGING.
70      C
71          CALL VLEN(X,XLEN)
72          IF(XLEN.GT.1000.) GO TO 950
73      C
74      150      CONTINUE
75      C
76      950      CONTINUE
77      C
78          IFLAG=1
79      C
80      900      CONTINUE
81      C
82          RETURN
83      END
```

Listing of ONRAY.FOR at 14:46:51 on AUG 9, 1984 for CCid=DAC0 Page 1

```

1      SUBROUTINE ONRAY(UR,PINT,V,ION)
2      C
3      C      THIS SUBROUTINE DETERMINES IF A POINT 'V' IS ON RAY DEFINED BY
4      C      POINT 'PINT' AND DIRECTION VECTOR 'UR'.
5      C
6      C      INPUT ARGUMENTS:
7      C          UR(3)- DIRECTION VECTOR OF RAY
8      C          PINT(3)- ORIGIN OF RAY
9      C          V(3)- POINT TO BE TESTED
10     C
11     C      OUTPUT ARGUMENTS:
12     C          ION- 0=NOT ON RAY
13     C              1=ON RAY
14     C
15     C      POINT 'V' IS ON THE RAY IF:
16     C
17     C           $TLAMB \cdot UR + PINT - V = 0$ 
18     C
19     C*****
20     C
21     REAL UR(3),PINT(3),V(3),TLAMB(3)
22     C
23     ION=0
24     C
25     K=0
26     C
27     C***** TEST EACH ONE OF THE THREE EQUATIONS.
28     C
29     DO 10 I=1,3
30     IF(I.EQ.3) J=1
31     C
32     C***** IF UR(I)=0, TLAMB IS ARBITRARY.
33     C
34     IF(UR(I).EQ.0.) GO TO 100
35     K=K+1
36     IF(K.GT.2) GO TO 200
37     C
38     C***** CALCULATE TLAMB FOR EQUATION I.
39     C
40     TLAMB(K)=(V(I)-PINT(I))/UR(I)
41     IF(K.EQ.1) GO TO 90
42     C
43     C***** TEST IF VALUE AGREES WITH OTHER CALCULATED VALUE.
44     C***** IF SO, 'V' LIES ON THE RAY.
45     C
46     IF(ABS(TLAMB(K)-TLAMB(K-1)).GT..00001) GO TO 400
47     90    CONTINUE
48     GO TO 10
49     C
50     100   CONTINUE
51     C
52     C***** TEST IF V(I)-PINT(I)=0
53     C
54     IF(V(I).NE.PINT(I)) GO TO 400
55     C
56     10    CONTINUE
57     C
58     200   CONTINUE

```

Listing of ONRAY.FOR at 14:46:51 on AUG 9, 1984 for CCId=DAC0 Page 2

```
59      C
60      C***** TEST RANGE OF TLAMB
61      C
62      IF(TLAMB(1).GT.O.) ION=1
63      GO TO 400
64      C
65      400      CONTINUE
66      C
67      RETURN
68      END
```

Listing of PLANE.FOR at 14:47:21 on AUG 9, 1984 for CCId=DACD Page 1

```
1      SUBROUTINE PLANE(NFACE,P1,P2,INT,PINT)
2      C
3      C      THIS SUBROUTINE TESTS FOR INTERSECTION OF A LINE SEGMENT
4      C      WITH A PLANE
5      C
6      C      INPUT ARGUMENTS:
7      C          NFACE- INDEX LOCATING FACE IN FACE LIST.
8      C          P1(3),P2(3)- END POINTS OF LINE SEGMENT.
9      C
10     C      OUTPUT ARGUMENTS:
11     C          INT- 0=NO INTERSECTION OF PLANE.
12     C          1=PLANE INTERSECTED BY LINE SEGMENT.
13     C          PINT(3)- INTERSECTION POINT.
14     C
15     C      COMMON BLOCKS ACCESSED:
16     C          /DATA/VERT,FACES,IFACE,IOBJ
17     C
18     C      SUBROUTINES CALLED:      DOTPRD
19     C
20     C      EQUATION OF PLANE:      R(3)*U(3)-P=0
21     C      EQUATION OF LINE:      RLAMB*U1(3)+P1=R(3)
22     C
23     C*****
24     C
25     C      REAL P1(3),P2(3),PINT(3),U(3),U1(3)
26     C
27     C      COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
28     C
29     C
30     C      INT=0
31     C
32     C***** FIND THE FACE PLANE SURFACE NORMAL 'U' AND DISTANCE FROM THE
33     C***** ORIGIN 'P'.
34     C
35     C      DO 10 I=1,3
36     C          U(I)=FACES(NFACE,I)
37     C      10 CONTINUE
38     C      P=FACES(NFACE,4)
39     C
40     C***** FIND THE DIRECTION VECTOR 'U1' OF THE LINE SEGMENT.
41     C
42     C      DO 15 I=1,3
43     C          U1(I)=P2(I)-P1(I)
44     C      15 CONTINUE
45     C
46     C***** SUBSTITUTE THE LINE ENDPOINTS INTO THE EQUATION OF THE PLANE.
47     C***** IF THE RESULTS ARE OPPOSITE IN SIGN, THE POINTS ARE ON OPPOSITE
48     C***** SIDES OF THE PLANE AND THE LINE SEGMENT INTERSECTS THE PLANE.
49     C
50     C      CALL DOTPRD(P1,U,P1U)
51     C      CALL DOTPRD(P2,U,P2U)
52     C
53     C      Z=(P1U-P)*(P2U-P)
54     C
55     C      IF(Z.GE.O) GO TO 800
56     C
57     C***** FIND THE VALUE OF THE PARAMETER 'RLAMB'.
58     C
```

Listing of PLANE.FOR at 14:47:21 on AUG 9, 1984 for CCid=DAC0 Page 2

```
59      RLAMB=(P-P1U)/(P2U-P1U)
60      C
61      C***** FIND THE INTERSECTION POINT 'PINT'.
62      C
63      DO 20 I=1,3
64          PINT(I)=P1(I)+RLAMB*U1(I)
65      20 CONTINUE
66      C
67      INT=1
68      C
69      800 CONTINUE
70      RETURN
71      END
```

Listing of ROTK.FOR at 14:47:21 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE ROTK(K,THETA,ROT)
2      C
3      C      THIS SUBROUTINE CALCULATES THE ROTATION MATRIX FOR A
4      C      ROTATION ABOUT A VECTOR 'K' BY AN ANGLE OF 'THETA'.
5      C
6      C      INPUT ARGUMENTS: K(3)- ROTATION AXIS VECTOR
7      C                        THETA- ROTATION ANGLE (RAD)
8      C
9      C      OUTPUT ARGUMENTS: ROT(4,4)- RESULTING HOMOGENEOUS ROTATION MATRIX.
10     C
11     C*****
12     C
13     REAL ROT(4,4),K(3)
14     C
15     C
16     RCOS=COS(THETA)
17     RSIN=SIN(THETA)
18     VERS=1.-RCOS
19     C
20     XX=K(1)*K(1)*VERS
21     YY=K(2)*K(2)*VERS
22     ZZ=K(3)*K(3)*VERS
23     XY=K(1)*K(2)*VERS
24     XZ=K(1)*K(3)*VERS
25     YZ=K(2)*K(3)*VERS
26     C
27     ROT(1,1)=XX+RCOS
28     ROT(1,2)=XY-K(3)*RSIN
29     ROT(1,3)=XZ+K(2)*RSIN
30     ROT(2,1)=XY+K(3)*RSIN
31     ROT(2,2)=YY+RCOS
32     ROT(2,3)=YZ-K(1)*RSIN
33     ROT(3,1)=XZ-K(2)*RSIN
34     ROT(3,2)=YZ+K(1)*RSIN
35     ROT(3,3)=ZZ+RCOS
36     ROT(4,4)=1.
37     C
38     RETURN
39     END
```

Listing of SOL3X3.FOR at 14:47:21 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE SOL3X3(RMAT,CONST,Y,ISOL)
2      C
3      C      THIS SUBROUTINE FINDS THE SOLUTION OF A SET OF LINEAR EQUATIONS OF
4      C      THE FOLLOWING FORM:
5      C          RMAT(3,3)*Y(3)+CONST(3)=0.
6      C
7      C      INPUT ARGUMENTS:
8      C          RMAT(3,3)- MATRIX OF COEFFICIENTS.
9      C          CONST(3)- VECTOR OF CONSTANTS.
10     C
11     C      OUTPUT ARGUMENTS:
12     C          Y(3)- SOLUTION TO THE SYSTEM OF EQUATIONS.
13     C          ISOL- 0=SOLUTION EXISTS.
14     C          1=NO SOLUTION EXISTS.
15     C
16     C      SUBROUTINES CALLED:    MATDET
17     C
18     C*****
19     C
20     REAL RMAT(3,3),CONST(3),Y(3),TEMP(3,3)
21     C
22     ISOL=0
23     C
24     C***** FIND THE DETERMINANT OF THE MATRIX 'RMAT' AND CHECK FOR SINGULARITY.
25     C
26     CALL MATDET(RMAT,DET)
27     C
28     IF(ABS(DET).LT..00001) GO TO 100
29     C
30     C***** SOLVE FOR 'Y'.
31     C
32     DO 10 I=1,3
33     DO 20 J=1,3
34     C
35         K1=I+1
36         K2=I+2
37         IF(K1.GE.4) K1=K1-3
38         IF(K2.GE.4) K2=K2-3
39     C
40         TEMP(J,I)=-CONST(J)
41         TEMP(J,K1)=RMAT(J,K1)
42         TEMP(J,K2)=RMAT(J,K2)
43     C
44     20    CONTINUE
45     C
46     CALL MATDET(TEMP,DTEMP)
47     Y(I)=DTEMP/DET
48     C
49     10    CONTINUE
50     C
51     GO TO 200
52     C
53     100   CONTINUE
54     ISOL=1
55     C
56     200   CONTINUE
57     C
58     RETURN

```

Listing of SOL3X3.FOR at 14:47:21 on AUG 9, 1984 for CCId=DACD Page 2

59

END

Listing of SOLVE.FOR at 14:47:22 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE SOLVE(P1,Q1,U1,U2,U,DIST,LAMB1,LAMB2)
2      C
3      C      THIS SUBROUTINE SOLVES FOR THE PARAMETERS LAMB1,LAMB2
4      C      DEFINING THE COMMON NORMAL POINTS ON TWO LINES.
5      C      LINE1:  R(3)=P1(3)+LAMB1*U1(3)
6      C      LINE2:  R(3)=Q1(3)+LAMB2*U2(3)
7      C
8      C*****
9      C
10     REAL P1(3),Q2(3),U1(3),U2(3),U(3),PINT(3),M1(2),M2(2),M3(2)
11     REAL C(3),QP(3)
12     REAL LAMB1,LAMB2
13     C
14     ICOUNT=0
15     C
16     C***** FIND TWO EQUATIONS WHICH GIVE A NON-ZERO DETERMINANT FOR
17     C***** THE DENOMINATOR
18     C
19     DO 20 I=1,3
20     J=I+1
21     K=I+2
22     IF(I.EQ.3) J=1
23     IF(I.GT.1) K=K-3
24     C
25     M1(1)=U1(I)
26     M1(2)=U1(J)
27     M2(1)=U2(I)
28     M2(2)=U2(J)
29     C
30     CALL DET2X2(M1,M2,DET)
31     C
32     ISAVE=I
33     JSAVE=J
34     KSAVE=K
35     C
36     C
37     IF(DET.NE.O.) GO TO 100
38     20    CONTINUE
39     C
40     100   CONTINUE
41     C
42     ICOUNT=ICOUNT+1
43     IF(ICOUNT.GT.2) GO TO 800
44     C
45     C***** CALCULATE THE VALUE OF THE RIGHT HAND SIDE OF THE EQUATION
46     C
47     CALL DIFF(Q1,P1,QP)
48     DO 10 I=1,3
49     C(I)=QP(I)-DIST*U(I)
50     10    CONTINUE
51     C
52     M3(1)=C(ISAVE)
53     M3(2)=C(JSAVE)
54     C
55     C***** SOLVE FOR THE UNKNOWNNS LAMB1,LAMB2
56     C
57     CALL DET2X2(M3,M2,TEMP1)
58     CALL DET2X2(M1,M3,TEMP2)
```

Listing of SOLVE.FOR at 14:47:22 on AUG 9, 1984 for CCId=DACO Page 2

```
59      C
60      LAMB1=TEMP1/DET
61      LAMB2=-TEMP2/DET
62      C
63      C***** TEST IF THIS IS A VALID SOLUTION
64      C
65      TEST=LAMB1*U1(KSAVE)-LAMB2*U2(KSAVE)
66      IF(ABS(TEST-C(K)).LT..00001) GO TO 900
67      C
68      C***** IF NOT, CHANGE THE SIGN OF THE VECTOR 'U' AND SOLVE AGAIN
69      C
70      DIST=-DIST
71      GO TO 100
72      C
73      800    CONTINUE
74      WRITE(5,*) 'ERROR - NO SOLUTION WAS FOUND IN SUBROUTINE SOLVE'
75      C
76      900    CONTINUE
77      C
78      RETURN
79      END
```

Listing of SURFIN.FOR at 14:47:40 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE SURFIN(P1,P2,P3,P4,E1,E2,INT)
2      C
3      C      THIS SUBROUTINE TESTS FOR INTERSECTION BETWEEN A STRAIGHT EDGE
4      C      DEFINED BY ENDPOINTS E1,E2, AND A PARAMETRIC SURFACE DEFINED
5      C      BY STRAIGHT BOUNDARIES AND CORNER POINTS P1,P2,P3,P4.
6      C
7      C      INPUT ARGUMENTS:
8      C          P1(3),P2(3),P3(3),P4(3)- CORNER POINTS OF A PARAMETRIC
9      C          RULED SURFACE.
10     C          E1(3),E2(3)- ENDPOINTS OF A STRAIGHT EDGE.
11     C
12     C      OUTPUT ARGUMENTS:
13     C          INT- 0=NO INTERSECTION
14     C          1=INTERSECTION
15     C
16     C      SUBROUTINES CALLED:      NEWTON
17     C
18     C      'IGL' GRAPHICS SUBROUTINES:      CMCLOS
19     C                                          CMOPEN
20     C                                          DRAW3D
21     C                                          MOVE3D
22     C
23     C      EQUATION OF THE SURFACE:
24     C           $R(3)=P1(3)+RLAMB1*UO(3)+RLAMB2*VO(3)+RLAMB1*RLAMB2*(U1(3)-UO(3))$ 
25     C
26     C      EQUATION OF THE EDGE:       $R(3)=E1(3)+RLAMB3*UE(3)$ 
27     C
28     C      COMBINING WE GET:
29     C
30     C       $RLAMB1*UO(3)+RLAMB2*VO(3)-RLAMB3*UE(3)+RLAMB1*RLAMB2*(U1(3)-UO(3)+$ 
31     C       $P1(3)-E1(3))=0$ 
32     C
33     C      THIS IS A NON-LINEAR SYSTEM OF EQUATIONS WHICH WE WILL SOLVE USING
34     C      NEWTONS METHOD.
35     C
36     C      C*****
37     C
38     C      REAL UO(3),VO(3),U1(3),UE(3),RLAMB(3)
39     C      REAL T1(3),T2(3),T3(3),T4(3),XO(3)
40     C      REAL P1(3),P2(3),P3(3),P4(3),V1(3),E1(3),E2(3)
41     C      REAL DELT(3),CONST(3)
42     C
43     C      DATA XO/3*.5/
44     C
45     C
46     C      INT=0
47     C
48     C      C***** DEFINE THE PARAMETRIC SURFACE BOUNDARY VECTORS 'VO','V1','UO','U1'.
49     C
50     C      DO 10 I=1,3
51     C          U1(I)=P3(I)-P4(I)
52     C          V1(I)=P3(I)-P2(I)
53     C          UO(I)=P2(I)-P1(I)
54     C          VO(I)=P4(I)-P1(I)
55     C          UE(I)=E1(I)-E2(I)
56     C          DELT(I)=U1(I)-UO(I)
57     C          CONST(I)=P1(I)-E1(I)
58     C

```

Listing of SURFIN.FOR at 14:47:40 on AUG 9, 1984 for CCId=DACD Page 2

```
59      10      CONTINUE
60      C
61      C***** SOLVE FOR THE PARAMETER VALUES AT THE INTERSECTION POINT USING
62      C***** NEWTONS METHOD FOR A SYSTEM OF NON-LINEAR EQUATIONS. THE MAXIMUM
63      C***** ERROR IS SET TO .01 AND THE MAXIMUM NUMBER OF ITERATIONS IS 10.
64      C
65      CALL NEWTON(UO,VO,UE,DELT,CONST,.01,10,XO,RLAMB,IFLAG)
66      IF(IFLAG.EQ.1) GO TO 900
67      C
68      C
69      C***** TEST THE VALUES OF THE PARAMETERS TO DETERMINE IF THE EDGE
70      C***** INTERSECTS THE SURFACE WITHIN ITS BOUNDARIES.
71      C
72      DO 35 I=1,3
73      IF(RLAMB(I).LT.(-0.001).OR.RLAMB(I).GT.(1.001)) GO TO 900
74      35      CONTINUE
75      C
76      C***** IF THERE IS AN INTERSECTION, INDICATE THE INTERSECTION POINT
77      C***** GRAPHICALLY BY DRAWING TWO CROSSED LINES ON THE SURFACE AT THAT POINT.
78      C
79      DO 60 I=1,3
80      T1(I)=P1(I)+RLAMB(1)*UO(I)
81      T2(I)=P1(I)+VO(I)+RLAMB(1)*U1(I)
82      T3(I)=P1(I)+RLAMB(2)*VO(I)
83      T4(I)=P1(I)+UO(I)+RLAMB(2)*V1(I)
84      60      CONTINUE
85      C
86      CALL CMOPEN
87      C
88      CALL MOVE3D(T1(1),T1(2),T1(3))
89      CALL DRAW3D(T2(1),T2(2),T2(3))
90      CALL MOVE3D(T3(1),T3(2),T3(3))
91      CALL DRAW3D(T4(1),T4(2),T4(3))
92      C
93      CALL CMCLOS
94      C
95      INT=1
96      C
97      900      CONTINUE
98      RETURN
99      END
```

Listing of SWPVOL.FOR at 14:47:40 on AUG 9, 1984 for CCid=DACO Page 1

```

1      SUBROUTINE SWPVOL
2      C
3      C      THIS SUBROUTINE DEFINES THE VOLUME SWEEPED OUT BY A MOVING
4      C      CYLINDER.
5      C
6      C      DEFINITION OF CONSTANTS AND VARIABLES:
7      C
8      C      A1,A2,B1,B2- CYLINDER SPINE ENDPOINTS
9      C      H1,H2- HOMOGENEOUS COORDINATE TRANSFORMATIONS SPECIFYING THE INITIAL
10     C      AND FINAL POSITION OF THE CYLINDER.
11     C      LEN,RADIUS- LENGTH AND RADIUS OF THE CYLINDER.
12     C      P,UO,U1,VO,V1- COEFFICIENTS OF PARAMETRIC SURFACE GENERATED BY THE
13     C      SPINE.
14     C      P1,P2,...P8- CORNER POINTS OF THE SWEEPED VOLUME.
15     C
16     C      SUBROUTINES CALLED:
17     C
18     C          CORNER
19     C          DRVOL
20     C          GENVOL
21     C          INITSV
22     C          LINK
23     C
24     C*****
25     C
26     C
27     REAL LEN,RADIUS
28     REAL HWORL(4,4)
29     BYTE FILNAM(20),OBJLST(20)
30     C
31     COMMON /LINK/H1(4,4),H2(4,4),LEN
32     COMMON /OBJLST/OBJLST
33     COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
34     COMMON /VOLUME/P(3),UO(3),U1(3),VO(3),V1(3)
35     COMMON /SNORM/DAN(3),DBN(3),DCN(3)
36     COMMON /CORNER/P1(3),P2(3),P3(3),P4(3),P5(3),P6(3),P7(3),P8(3)
37     COMMON /SPINES/A1(3),B1(3),A2(3),B2(3)
38     COMMON /VIEW/EYEPOS(3),VRP(3)
39     COMMON /RADIUS/RADIUS
40     C
41     DATA EYEPOS/1.,1.,1./
42     DATA VRP/0.,0.,0./
43     DATA OBJLST/'D','B','J','L','S','T','.',',','D','A','T',10*0/
44     DATA FILNAM/'F','I','L','N','A','M','.',',','D','A','T',10*0/
45     DATA HWORL/16*0./
46     C
47     HWORL(1,1)=1.
48     HWORL(2,2)=1.
49     HWORL(3,3)=1.
50     HWORL(4,4)=1.
51     C
52     C***** FIND THE SWEEPED VOLUME PARAMETERS
53     C
54     CALL INITSV(FILNAM)
55     C
56     C***** GET THE LINK PARAMETERS AND SPINE ENDPOINTS.
57     C
58     CALL LINK(A1,B1,A2,B2)

```

Listing of SWPVOL.FOR at 14:47:40 on AUG 9, 1984 for CCId=DAC0 Page 2

```
59      C
60      C***** FIND THE PARAMETRIC SURFACE GENERATED BY THE CYLINDER SPINE.
61      C
62      CALL GENVOL(A1,B1,A2,B2)
63      C
64      C***** FIND THE CORNER POINTS OF THE PARAMETRIC SWEPT VOLUME.
65      C
66      CALL CORNER
67      C
68      C***** DRAW THE SWEPT VOLUME.
69      C
70      CALL DRVOL
71      C
72      RETURN
73      END
```

Listing of TCROSS.FOR at 14:47:41 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE TCROSS(IV1,IV2,IV3,UR,PINT,INT)
2      C
3      C      THIS SUBROUTINE DETERMINES IF A RAY CONTAINING AT LEAST ONE
4      C      FACE VERTEX CROSSES THE FACE BOUNDARY AT THAT VERTEX. THIS IS
5      C      DONE BY DETERMINING IF THE ADJACENT VERTICES ON THE FACE LIE ON
6      C      OPPOSITE SIDES OF THE LINE CONTAINING THE RAY.
7      C
8      C      INPUT ARGUMENTS:
9      C          IV1,IV3- ADJACENT VERTEX INDEXES.
10     C          IV2- INDEX OF VERTEX CROSSED BY THE RAY.
11     C          UR- DIRECTION VECTOR OF THE RAY.
12     C          PINT- ORIGIN OF THE RAY.
13     C
14     C      OUTPUT ARGUMENTS:
15     C          INT- 0=NO FACE BOUNDARY CROSSED
16     C              1=FACE BOUNDARY CROSSED
17     C
18     C      COMMON BLOCK ACCESSED:
19     C          /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
20     C
21     C      SUBROUTINES CALLED:      DOTPRD
22     C                               XPROD
23     C
24     C*****
25     C
26     C      REAL PROD1(3),PROD2(3)
27     C      REAL UR(3),PINT(3),U1(3),U2(3)
28     C
29     C      COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
30     C
31     C      INT=0
32     C
33     C***** FIND THE DIRECTION VECTORS 'U1','U2' OF THE ADJACENT EDGES.
34     C
35     C      DO 10 I=1,3
36     C          U1(I)=VERT(IV1,I)-VERT(IV2,I)
37     C          U2(I)=VERT(IV3,I)-VERT(IV2,I)
38     C      10 CONTINUE
39     C
40     C***** FIND THE CROSS PRODUCTS OF THE ADJACENT EDGES AND THE RAY.
41     C
42     C      CALL XPROD(U1,UR,PROD1)
43     C      CALL XPROD(U2,UR,PROD2)
44     C
45     C***** TEST IF THE EDGES ARE ON OPPOSITE SIDES OF THE RAY.
46     C
47     C      CALL DOTPRD(PROD1,PROD2,DOT)
48     C      IF(DOT.LT.O.) INT=1
49     C
50     C      RETURN
51     C      END
```

Listing of TRANS3.FOR at 14:48:02 on AUG 9, 1984 for CCid=DAC0 Page 1

```

1      SUBROUTINE TRANS3(H,P1,P2)
2      C
3      C      THIS SUBROUTINE PERFORMS THE FOLLOWING HOMOGENEOUS
4      C      TRANSFORMATION.
5      C
6      C      P2(3)=H(4,4)*P1(3)
7      C
8      C      WHERE P1,P2 ARE 3X1 VECTORS AND H IS A 4X4 HOMOGENEOUS
9      C      COORDINATE TRANSFORMATION MATRIX.
10     C
11     C*****
12     C
13     REAL H(4,4),P1(3),P2(3)
14     C
15     DO 100 I=1,3
16         P2(I)=H(I,1)*P1(1)+H(I,2)*P1(2)+H(I,3)*P1(3)+H(I,4)
17     100 CONTINUE
18     C
19     RETURN
20     END
```

Listing of UNIT.FOR at 14:48:02 on AUG 9, 1984 for CCId=DAC0 Page 1

```

1      SUBROUTINE UNIT(VECT,UVECT)
2      C
3      C      THIS SUBROUTINE NORMALIZES A VECTOR.
4      C
5      C      INPUTS: VECT(3)- A VECTOR
6      C
7      C      OUTPUTS: UVECT(3)- VECT NORMALIZED.
8      C
9      C*****
10     C
11     REAL VECT(3),UVECT(3),LENGTH
12     C
13     LENGTH=SQRT(VECT(1)**2+VECT(2)**2+VECT(3)**2)
14     C
15     UVECT(1)=VECT(1)/LENGTH
16     UVECT(2)=VECT(2)/LENGTH
17     UVECT(3)=VECT(3)/LENGTH
18     C
19     RETURN
20     END
```

Listing of VCROSS.FOR at 14:48:28 on AUG 9, 1984 for CCId=DACO Page 1

```

1      SUBROUTINE VCROSS(NVERT,ICOUNT,UR,PINT,ILIST,NVCRSS)
2      C
3      C      THIS SUBROUTINE FINDS THE NUMBER OF TIMES A RAY CROSSES
4      C      THE POLYGONAL FACE BOUNDARY AT VERTICES.
5      C
6      C      INPUT ARGUMENTS:
7      C          NVERT- INDEX OF FIRST FACE VERTEX.
8      C          ICOUNT- NUMBER OF VERTICES ON FACE.
9      C          UR- DIRECTION VECTOR OF THE RAY.
10     C          PINT- ORIGIN OF RAY.
11     C
12     C      OUTPUT ARGUMENTS:
13     C          ILIST- LIST OF VERTICES ON RAY.
14     C          NVCRSS- NUMBER OF TIMES THE RAY CROSSES FACE BOUNDARIES AT
15     C          VERTICES.
16     C
17     C      COMMON BLOCKS ACCESSED: /DATA/
18     C
19     C      SUBROUTINES CALLED:      ONRAY
20     C                          TCROSS
21     C
22     C*****
23     C
24     C      REAL PINT(3),UR(3),V(3)
25     C      INTEGER ILIST(10),NVCRSS
26     C
27     C      COMMON /DATA/VERT(1000,3),FACES(100,4),IFACE(100,2),IOBJ(10,2)
28     C
29     C      NVCRSS=0
30     C      NUM=0
31     C
32     C***** TEST ALL VERTICES ON THE FACE.
33     C
34     C      DO 10 I=1,ICOUNT
35     C
36     C***** FIND THE NEXT VERTEX TO BE TESTED.
37     C
38     C      DO 15 J=1,3
39     C          V(J)=VERT(NVERT+I-1,J)
40     C      15 CONTINUE
41     C
42     C***** TEST IF THE VERTEX IS ON THE RAY.
43     C
44     C      CALL ONRAY(UR,PINT,V,ION)
45     C
46     C      IF(ION.EQ.0) GO TO 100
47     C
48     C***** UPDATE 'NUM' AND 'ILIST'.
49     C
50     C      NUM=NUM+1
51     C      ILIST(NUM)=NVERT+I-1
52     C
53     C      100 CONTINUE
54     C      10 CONTINUE
55     C
56     C
57     C***** TEST ALL VERTICES LYING ON RAY FOR FACE BOUNDARY CROSSINGS.
58     C***** ADJACENT VERTICES ARE THE CLOSEST VERTICES TO EITHER SIDE OF
```

Listing of VCROSS.FOR at 14:48:28 on AUG 9, 1984 for CCId=DACO Page 2

```
59      C***** THE CURRENT VERTEX WHICH DO NOT LIE ON THE RAY.
60      C
61      C      DO 20 I=1,NUM
62      C
63      C***** FIND CURRENT VERTEX.
64      C
65      C      M=ILIST(I)
66      C
67      C***** FIND ADJACENT VERTICES.
68      C
69      C      MV2=M+1
70      C      MV1=M-1
71      C      IF(I+1.GT.NUM) GO TO 600
72      C      GO TO 610
73      C
74      C***** LAST VERTEX IN LIST.
75      C
76      600      CONTINUE
77      C      IF(M.EQ.NVERT+ICOUNT-1) GO TO 620
78      C      GO TO 310
79      C
80      620      MV2=NVERT
81      C      IF(ILIST(1).EQ.NVERT) GO TO 500
82      C
83      610      CONTINUE
84      C      IF(ILIST(I+1).NE.M+1) GO TO 310
85      C
86      C      MV2=M+2
87      C      GO TO 350
88      C
89      310      CONTINUE
90      C      IF(ILIST(I-1).NE.M-1) GO TO 350
91      C
92      C      MV1=M-2
93      C
94      350      CONTINUE
95      C
96      C      IF(MV2.GT.NVERT+ICOUNT-1) GO TO 500
97      C      IF(MV1.LT.NVERT) GO TO 500
98      C
99      C***** TEST FOR CROSSING AT VERTEX.
100     C
101     C      CALL TCROSS(MV1,M,MV2,UR,PINT,INT)
102     C      IF(INT.EQ.1) NVCRSS=NVCRSS+1
103     C
104     C
105     500      CONTINUE
106     20      CONTINUE
107     C
108     C      IF(ILIST(1).NE.NVERT) GO TO 510
109     C      IF(ILIST(2).NE.NVERT+1) GO TO 511
110     C
111     C      MV2=NVERT+2
112     C      MV1=NVERT+ICOUNT-1
113     C      GO TO 550
114     C
115     511      CONTINUE
116     C      IF(ILIST(NUM).NE.NVERT+ICOUNT-1) GO TO 512
```

Listing of VCROSS.FOR at 14:48:28 on AUG 9, 1984 for CCId=DAC0 Page 3

```
117      C
118          MV2=NVERT+1
119          MV1=NVERT+ICOUNT-2
120          GO TO 550
121      C
122      512      CONTINUE
123      C
124          MV2=NVERT+1
125          MV1=NVERT+ICOUNT-1
126      C
127      550      CONTINUE
128      C
129      C***** TEST FOR CROSSING AT VERTEX.
130      C
131          CALL TCROSS(MV1,M,MV2,UR,PINT,INT)
132          IF(INT.EQ.1) NVCRSS=NVCRSS+1
133      C
134      510      CONTINUE
135      C
136          RETURN
137      END
```

Listing of VLEN.FOR at 14:48:29 on AUG 9, 1984 for CCid=DACO Page 1

```
1      SUBROUTINE VLEN(VECTOR,LENGTH)
2      C
3      C      THIS SUBROUTINE FINDS THE SCALAR LENGTH OF A VECTOR.
4      C
5      REAL LENGTH,VECTOR(3)
6      C
7      LENGTH=SQRT(VECTOR(1)**2+VECTOR(2)**2+VECTOR(3)**2)
8      C
9      RETURN
10     END
```

Listing of VRTCRS.FOR at 14:48:29 on AUG 9, 1984 for CCid=DACD Page 1

```
1      SUBROUTINE VRTCRS(UR,P1,P2,INT)
2      C
3      C      THIS SUBROUTINE DETERMINES IF A RAY CONTAINING AT LEAST ONE
4      C      FACE VERTEX CROSSES THE FACE BOUNDARY AT THAT VERTEX. THIS IS
5      C      DONE BY DETERMINING IF THE ADJACENT VERTICES ON THE FACE LIE ON
6      C      OPPOSITE SIDES OF THE LINE CONTAINING THE RAY.
7      C
8      C      UR(3)- RAY DIRECTION VECTOR.
9      C      P1(3),P2(3)- TWO ADJACENT VERTICES.
10     C
11     C*****
12     C
13     REAL UR(3),P1(3),P2(3),U1(3),U2(3),PROD1(3),PROD2(3)
14     C
15     INT=0
16     C
17     DO 10 I=1,3
18         U1(I)=P1(I)-UR(I)
19         U2(I)=P2(I)-UR(I)
20     10 CONTINUE
21     C
22     CALL XPROD2(U1,UR,PROD1)
23     CALL XPROD2(U2,UR,PROD2)
24     C
25     CALL DOTPRD(PROD1,PROD2,RESULT)
26     C
27     IF(RESULT.LT.O.) INT=1
28     C
29     RETURN
30     END
```

Listing of XPROD2.FOR at 14:50:37 on AUG 9, 1984 for CCId=DACO Page 1

```
1      SUBROUTINE XPROD2(A,B,C)
2      C
3      C      THIS SUBROUTINE CALCULATES VECTOR C = A X B (CROSS PRODUCT).
4      C
5      C      INPUTS:
6      C          A(3)=FIRST VECTOR
7      C          B(3)=SECOND VECTOR
8      C
9      C      OUTPUT:
10     C          C(3)=CROSS PRODUCT, A X B
11     C
12     C*****
13     C
14     C
15     C      REAL A(3), B(3), C(3), ABS
16     C
17     C      C(1)=A(2)*B(3)-A(3)*B(2)
18     C      C(2)=A(3)*B(1)-A(1)*B(3)
19     C      C(3)=A(1)*B(2)-A(2)*B(1)
20     C
21     C      RETURN
22     C      END
```