ANL-7633
RETURN TO ANL (IDAHO) LIBRARY.

7633 ANL-7633

THREE-DIMENSIONAL GRAPHICAL REPRESENTATION OF SURFACES BY COMPUTER (FAST REACTOR CONTAINMENT PROGRAM)

D. Basinger and J. Gvildys

U of C-AUA-USAEC

ARGONNE NATIONAL LABORATORY, ARGONNE, ILLINOIS

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Atomic Energy Commission, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

MEMBERS OF ARGONNE UNIVERSITIES ASSOCIATION

The University of Arizona
Carnegie-Mellon University
Case Western Reserve University
The University of Chicago
University of Cincinnati
Illinois Institute of Technology
University of Illinois
Indiana University
Iowa State University
The University of Iowa
Kansas State University
The University of Kansas
Loyola University
Marquette University
Michigan State University
The University of Michigan
University of Minnesota
University of Missouri
Northwestern University
University of Notre Dame
The Ohio State University
Ohio University
The Pennsylvania State University
Purdue University
Saint Louis University
Southern Illinois University
The University of Texas at Austin
Washington University
Wayne State University
The University of Wisconsin

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights.

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, Virginia 22151
Price: Printed Copy $3.00; Microfiche $0.95

ANL-7633
Reactor Technology

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, Illinois 60439

THREE-DIMENSIONAL GRAPHICAL REPRESENTATION OF SURFACES BY COMPUTER (FAST REACTOR CONTAINMENT PROGRAM)

by

D. Basinger and J. Gvildys

Reactor Analysis and Safety Division

November 1971

TABLE OF CONTENTS

	Page
ABSTRACT	5
I. INTRODUCTION	5
II. LIMITATIONS OF PLOT 3D	6
III. DESCRIPTION OF PLOT 3D PACKAGE	9
IV. USING PLOT 3D	12
V. CONCLUDING REMARKS	15
APPENDICES	16
A. Routines for System 360 - IBM 2280 Film Recorder Subroutine Plot 3D - Version 1	16
B. Routines for System 360 - Calcomp 780 Plotter Subroutine Plot 3D - Version 2	22
C. Routines for System 360 - IBM 2280 Film Recorder Subroutine Plot 3D - Version 3	28
D. Routines for System 360 - Calcomp 780 Plotter Subroutine Plot 3D - Version 4	41
ACKNOWLEDGMENT	53
BIBLIOGRAPHY	54

LIST OF FIGURES

No.	Title	Page
1.	Sample Output of PLOT 3D Version 2	7
2.	Sample Output of PLOT 3D Version 4, Computer Simulation of High-Energy Excursion.	8
3.	Sample Output of PLOT 3D Version 4	8
4.	Rotations of the Surface	9

THREE-DIMENSIONAL GRAPHICAL REPRESENTATION OF SURFACES BY COMPUTER (FAST REACTOR CONTAINMENT PROGRAM)

by

D. Basinger and J. Gvildys

ABSTRACT

Plot 3D is a package of FORTRAN IV Subprograms designed to draw three-dimensional surfaces from arrays of points (x,y,z). The surfaces can be drawn after arbitrary rotations about the three coordinate axes.

Four versions of Plot 3D are described. Output of Versions 1 and 3 is by film recorder. Output of Versions 2 and 4 is by Calcomp Plotter. Versions 3 and 4 do not draw lines which would be invisible to a viewer looking at an opaque surface, whereas Versions 1 and 2 draw every line on the surface.

Plot 3D is described along with its limitations. Sample output and listings of the subprograms are included.

I. INTRODUCTION

It can be difficult to visualize the behavior of complicated functions or the shapes of empirical surfaces corresponding to tabulated data or contour plots. The plotting of such surfaces is the purpose of PLOT 3D.

PLOT 3D is a computer program to plot any surface for which each coordinate pair (x,y) is associated with a unique z in the set of points (x,y,z). It uses matrix transformations of the points to generate different views of the surface after arbitrary rotations about the three coordinate axes. The views are not perspective.

This report describes completely the PLOT 3D package. Examples of the output are included, and the limitations of PLOT 3D are discussed.

PLOT 3D was written for use with the IBM System/360 and either the IBM 2280 Film Recorder or the Calcomp 780 Plotter at Argonne National Laboratory. The subprograms are explained, and there is a sample user's program. Possible modifications are mentioned.

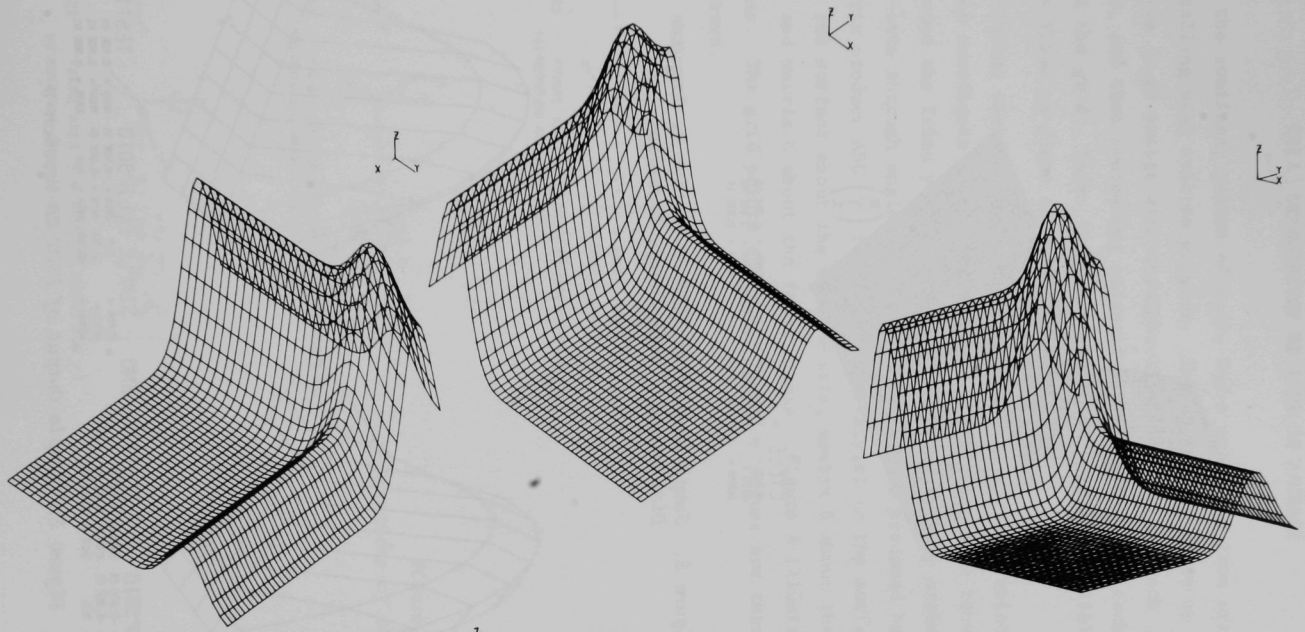
II. LIMITATIONS OF PLOT 3D

Plot 3D is designed to portray only those functions for which there may be derived for every pair (x,y) a unique z value. It cannot, for example, draw a sphere. The functions must be in rectangular coordinates. The arrays of (x,y) points must determine a grid which lies within a rectangle whose minimum and maximum values can be established.

Versions 1 and 2 draw every line on the surface, including those which would be invisible to an observer viewing an opaque surface. Consult Figs. 1 and 2 for examples. On a surface where there are many of these "invisible" lines, their appearance can cause the view to be confusing.

Versions 3 and 4 do not draw lines which would not be visible to an observer viewing an opaque surface. Consult Fig. 3 for an example. The masking routines greatly increase the running time of these subprograms. The magnitude of the time increase is dependent on the number of grid points in the arrays. On test runs an increase in the number of grid points by a factor of 4 increased the running time by a factor of 10.

The views may be difficult to study quantitatively. A careful choice of input parameters must be made to read values off the surface plots.



a.

Z=3 EXP (- (X+4) **2) +EXP (- (Y-4) **2)

ALPHA =	0.0	XMIN =	-0.50E 01	XMAX =	0.48E 01
BETA =	45.00	YMIN =	-0.30E 01	YMAX =	0.57E 01
GAMMA =	-45.00	ZMIN =	0.52E-21	ZMAX =	0.40E 01

b.

Z=3 EXP (- (X+4) **2) +EXP (- (Y-4) **2)

ALPHA =	0.0	XMIN =	-0.50E 01	XMAX =	0.48E 01
BETA =	45.00	YMIN =	-0.30E 01	YMAX =	0.57E 01
GAMMA =	45.00	ZMIN =	0.52E-21	ZMAX =	0.40E 01

c.

Z=3 EXP (- (X+4) **2) +EXP (- (Y-4) **2)

ALPHA =	0.0	XMIN =	-0.50E 01	XMAX =	0.48E 01
BETA =	15.00	YMIN =	-0.30E 01	YMAX =	0.57E 01
GAMMA =	45.00	ZMIN =	0.52E-21	ZMAX =	0.40E 01

Figure 1. Sample Output of PLOT 3D Version 2

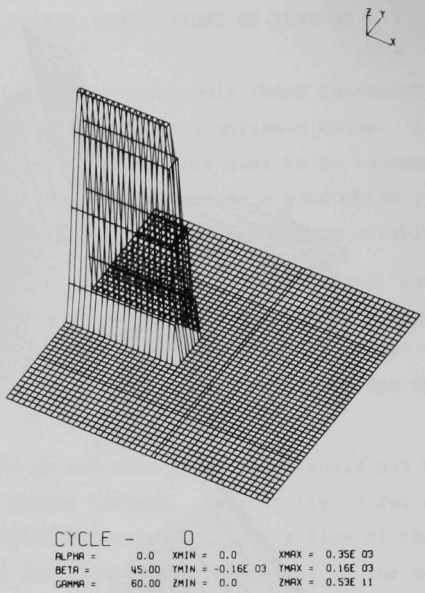


Figure 2. Sample Output of PLOT 3D Version 4, Computer Simulation of High-Energy Excursion

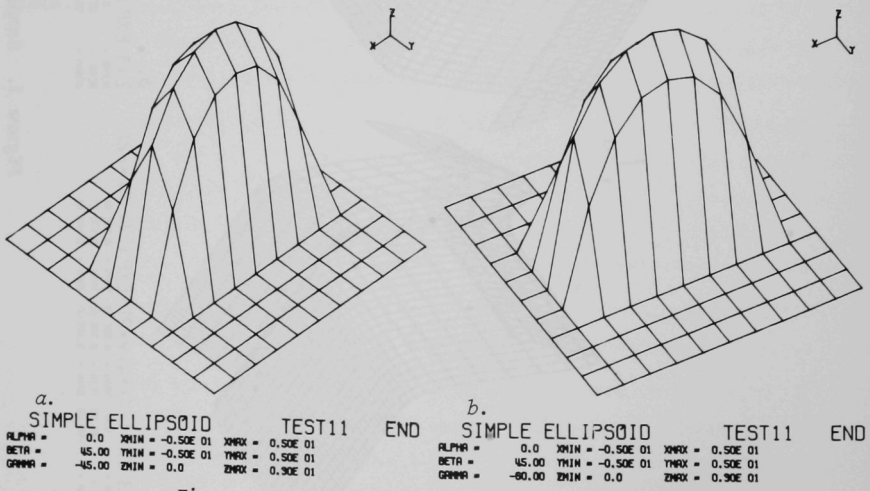


Figure 3. Sample Output of PLOT 3D Version 4

III. DESCRIPTION OF PLOT 3D PACKAGE

In the routines, values of x, y, and z are taken from arrays supplied in the calling user written program. The surface is drawn by connecting successive (y,z) points with straight-line segments for each x value on the grid, and then connecting successive (x,z) points for each fixed y value on the grid. These lines are then projected by parallel projection onto the fixed yz plane for plotting.

Different views of the surface are made by selected rotations about the fixed coordinate axes: first around the fixed z-axis through angle γ, then around the fixed y-axis through angle β and finally around the fixed x-axis through angle α. These rotations are produced by implementing the matrix product ABC $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ for each point (x,y,z) in the surface. Matrix A rotates the surface about the fixed x-axis, matrix B about the fixed y-axis, and matrix C about the fixed z-axis. Figure 4 illustrates these rotations. The grid points and their function values are then transformed and redrawn.

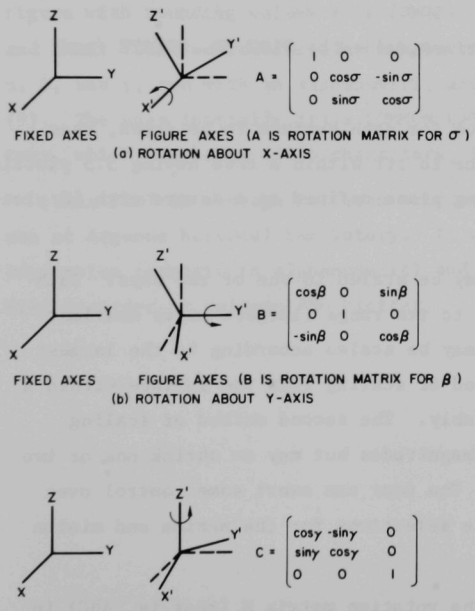


Figure 4.
Rotations of the Surfaces

The arrays supplied by the user along with the number of points along the length and the width of the xy grid determines the number of lines in each direction as well as the number of points on each line. Increasing the number of grid lines increases "fineness" of the mesh. A fine mesh provides shading of regions with small surface gradients. The more points that are used, the better is the representation of curvature, since straight-line segments are drawn between the points. However, the number of points used is limited by the computer storage capability and by the computation time.

The call to SUBROUTINE PLOT 3D initiates the Film Recorder or the Calcomp Plotter and calls subroutines to scale the values in the two-dimensional arrays specified in the PLOT 3D arguments, making use of maximum and minimum values supplied in COMMON/MAXES/.

ENTRY ROTATE to PLOT 3D calls subroutines which draw the surface after desired rotations about the three fixed axes are accomplished.

(ENTRY NOMORE in the PLOT 3D version for the Calcomp Plotter closes the Calcomp tape.)

A description of the subroutines called by PLOT 3D, ROTATE (and NOMORE) follows.

SUBROUTINE SCALE, called once for each surface to be drawn, uses COMMON/MAXES/ to scale the surface to fit within a cube having 5.5 plotting units on a side, with the plotting plane defined as a square with 10 plotting units on a side.

The variables x, y, and z may be scaled in one of two ways. Each variable may be scaled according to its range (largest minus smallest values), or all three variables may be scaled according to the largest of these ranges. The first method of scaling fits the surface within a cube and may distort it considerably. The second method of scaling preserves the original relative magnitudes but may so shrink one or two axes as to make them invisible. The user can exert some control over the scaling by making appropriate selections for the maxima and minima in COMMON/MAXES/.

SUBROUTINE TRNMAT sets up the rotation matrix M (that is, ABC) in terms of the supplied angles of rotation: α, β, and γ. TRNMAT is

called once for every rotated view, that is, for every call to ROTATE. The rotation matrix is placed in COMMON/MATRIX/TMAT (3,3).

SUBROUTINE PHI rotates the surface curves by multiplying the position vector of each point by the rotation matrix in COMMON/MATRIX/TMAT (3,3). It is called once for each rotated view.

SUBROUTINE AXIS, called once to initiate values and then once for each surface view, sets up, rotates, and draws the three figure axes. Drawing these axes helps to show the orientation of the surface with respect to the fixed coordinate axes. AXIS uses the rotation matrix in COMMON/MATRIX/TMAT (3,3) to rotate the figure axes in the same way that PHI rotates the surface curves. AXIS uses machine-dependent plotting routines to plot and label the axes.

SUBROUTINE DRAW, called once for each view plotted, draws the line segments connecting (y,z) grid points. DRAW uses plotting routines which are machine-dependent.

SUBROUTINE WRITE, called once for each view plotted, labels the figure with bounding values from COMMON/MAXES/XMIN, XMAX, YMIN, YMAX, ZMIN, and ZMAX. This subroutine also labels the figure with accumulated angles α, β, and γ, and with an alphanumeric array supplied in COMMON/LABEL/LAB (9). The user initially fills COMMON/LABEL/LAB (9) with the desired text, which can be up to 36 characters, including blanks.

This PLOT 3D package has been written for the IBM System 360 in use at Argonne National Laboratory. It uses subroutines for core-to-core conversion (numeric to alphanumeric) and for control of Argonne's IBM 2280 Film Recorder or Calcomp 780 Plotter.

IV. USING PLOT 3D

Preceding the call to PLOT 3D, the user must supply values to COMMON/MAXES/XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX, and if a label is desired, to COMMON/LABEL/LAB (9), where:

XMIN - less than or equal to the minimum value in array X.
XMAX - greater than or equal to the maximum value in array X.
YMIN - less than or equal to the minimum value in array Y.
YMAX - greater than or equal to the maximum value in array Y.
ZMIN - less than or equal to the minimum value in array Z.
ZMAX - greater than or equal to the maximum value in array Z.
LAB - one-dimensional array containing alphanumeric text if supplied by the user, blanks if not.

The first call must be CALL PLOT 3D (X, Y, Z, NX, NY, CUBE) where:

X - two-dimensional array containing the x-coordinates.
Y - two-dimensional array containing the y-coordinates.
Z - two-dimensional array containing the z-coordinates.
NX - the number of columns in the X, Y, Z arrays.
NY - the number of rows in the X, Y, Z, arrays.
CUBE - Logical variable. When CUBE =.TRUE. the three dimensions are scaled independently in order to fit the surface into a cube. When CUBE = .FALSE. the program does not fit the surface into a cube.

Views of the surface may now be made by the call CALL ROTATE (ALPHA, BETA, GAMMA, DRAWME) where:

ALPHA - Angle of rotation about the fixed X-axis (degrees).
BETA - Angle of rotation about the fixed Y-axis (degrees).
GAMMA - Angle of rotation about the fixed Z-axis (degrees).
DRAWME - LOGICAL variable set to .TRUE. if the particular view is to be plotted, or set to .FALSE. if the view is not to be plotted.

Positive angles are counterclockwise rotations, while negative angles are clockwise rotations. The surface is not returned to its original position after rotation.

(In the PLOT 3D version for the Calcomp 780 Plotter, the call CALL NOMORE after the last view has been plotted closes the Calcomp tape unit.)

A sample calling program follows:

```
      COMMON/LABEL/LAB(9)
      COMMON/MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
      DIMENSION X(50,30),Y(50,30),Z(50,30)
      LOGICAL CUBE,DRAWME
      READ333,LAB
  333 FORMAT(9A4)
      XA=-6.0
      DO 2 I=1,50
      YA=0.0
      DO 3 J=1,30
      X(I,J)=XA
      Y(I,J)=YA
      Z(I,J)=3.0*EXP(-(XA+4.0)**2)+EXP(-(YA-4.0)**2)
    3 YA=YA+0.3
    2 XA=XA+0.4
      NX=50
      NY=30
      XMIN=X(1,1)
      XMAX=X(1,1)
      YMIN=Y(1,1)
      YMAX=Y(1,1)
      ZMIN=Z(1,1)
      ZMAX=Z(1,1)
      DO4 I=1,NX
      DO4 J=1,NY
      XMIN=AMIN1(XMIN,X(I,J))
      XMAX=AMAX1(XMAX,X(I,J))
      YMIN=AMIN1(YMIN,Y(I,J))
      YMAX=AMAX1(YMAX,Y(I,J))
      ZMIN=AMIN1(ZMIN,Z(I,J))
      ZMAX=AMAX1(ZMAX,Z(I,J))
    4 CONTINUE
      CUBE=.TRUE.
      CALL PLOT3D(X,Y,Z,NX,NY,CUBE)
      DRAWME=.TRUE.
      CALL ROTATE(000.0,045.0,-45.0,DRAWME)
      DRAWME=.FALSE.
      CALL ROTATE(000.0,-45.0,000.0,DRAWME)
      DRAWME=.TRUE.
      CALL ROTATE(000.0,045.0,090.0,DRAWME)
      CALL ROTATE(000.0,015.0,000.0,DRAWME)
      CALL ROTATE(000.0,-45.0,000.0,DRAWME)
      CALL ROTATE(000.0,-60.0,000.0,DRAWME)
      DRAWME =.FALSE.
      CALL ROTATE(000.0,045.0,000.0,DRAWME)
      DRAWME=.TRUE.
      CALL ROTATE(000.0,060.0,015.0,DRAWME)
      CALL NOMORE
      END
```

Versions 1 and 3, for the IBM 2280 Film Recorder, require 7810 and 58225 bytes of core storage, exclusive of film recorder subroutines and arrays. Versions 2 and 4, for the Calcomp Plotter, require 6670 and 62435 bytes of core storage, exclusive of Calcomp Plotter routines and arrays.

V. CONCLUDING REMARKS

A package of FORTRAN subroutines has been designed to draw three-dimensional surfaces by means of the IBM 2280 Film Recorder or the Calcomp 780 Plotter. The package is readily adaptable to other hardware. It is intended for graphical representation of surfaces determined by single valued functions of two independent variables. Plots of complex surfaces may be difficult to interpret.

APPENDIX A

ROUTINES FOR SYSTEM 360--IBM 2280 FILM RECORDER

SUBROUTINE PLOT 3D--VERSION 1

Purpose

Draw and rotate three-dimensional surfaces with IBM 360/50-75 - 2280 Film Recorder.

Usage

```
COMMON/MAXES/XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX
COMMON/LABEL/LAB (9)
CALL PLOT 3D (X, Y, Z, NX, NY, CUBE)
CALL ROTATE (ALPHA, BETA, GAMMA, DRAWME)
```

Description of Parameters

XMIN - Less than or equal to the minimum value in array X

XMAX - Greater than or equal to the maximum value in array X

YMIN - Less than or equal to the minimum value in array Y

YMAX - Greater than or equal to the maximum value in array Y

ZMIN - Less than or equal to the minimum value in array Z

ZMAX - Greater than or equal to the maximum value in array Z

X - Two-dimensional array of X-coordinates

Y - Two-dimensional array of Y-coordinates

Z - Two-dimensional array of Z-coordinates

NX - Number of rows in X, Y, Z arrays

NY - Number of columns in X, Y, Z arrays

CUBE - LOGICAL variable set to .TRUE. if the three directions are to be scaled independently of each other and set to .FALSE. if the three directions are to be scaled dependently.

ALPHA- Angle of rotation about the X-axis, given in degrees

BETA - Angle of rotation about the Y-axis, given in degrees

GAMMA- Angle of rotation about the Z-axis, given in degrees

DRAWME- LOGICAL variable set to .TRUE. if the rotated view is to be drawn, and set to .FALSE. if this view is not to be drawn.

LAB - Array of BCD characters used to label the drawing. If label is desired, supply 9 or less four-character words in COMMON/LABEL/LAB (9).

Remarks

A call to PLOT 3D defines and scales the surface. Subsequent calls to ROTATE rotate it to new frames of reference. The surface is not returned to its original position after a rotation. The surface is rotated about the fixed XYZ axes in this order - first about the Z-axis, then about the Y-axis, and finally about the X-axis.

Care should be exercised in providing that the arrays X, Y, Z in the calling program have dimensions (NX, NY). Dimensions different from this may cause errors in the plots.

Subroutines Required

SCALE
TRNMAT
PHI
AXIS
DRAW
WRITE

In addition, PLOT 3D calls routines for the IBM 2280 Film Recorder. They follow:

FPLTM - Sets the vector and point-plotting mode

FMAREA - Defines the size of the film area onto which those coordinates specified as belonging to the user's coordinate system via the FDATM routine are to be mapped.

FXYTRN - Defines the positioning of the film area

FLIND - Sets the film line density

FLINW - Sets the film line width

FDATM - Sets the arithmetic mode and reference frame of the input X, Y values for vectors, points, and beam positioning.

FXYLIM - Sets the maximum and minimum values of the user's coordinate system.

FADV - Advances the film

FCHSZ - Sets the character size of the characters generated by the character generator.

FLNSG - Draws a line segment between two specified end points

FMOVE - Positions the beam at the specified coordinates

FLINE - Draws a line(s) from the current beam position to the specified beam position

FTEXT - Plots a string of characters

Finally, PLOT 3D calls an Argonne-developed routine which follows:

CONVO - Core-to-core conversion, internal machine representation to EBCDIC

A listing of the subroutines follows:

```
SUBROUTINE PLOT3D(SX,SY,SZ,NX,NY,CUBE)
LOGICAL CUBE
COMMON /MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
DIMENSION SX(NX,NY),SY(NX,NY),SZ(NX,NY)
CALL FPLTM(1)
CALL FDATM(3)
CALL FMAREA(2200,2200)
CALL FXYTRN(1000,1000)
CALL FLIND(1)
CALL FLINW(1)
CALL FDATM(2)
CALL FXYLIM(-5.0,-5.0,5.0,5.0)
CALL SCALE(SX,SY,SZ,NX,NY,CUBE)
CALL AXIS(0,.FALSE.)
ASUM=0.0
BSUM=0.0
CSUM=0.0
RETURN
ENTRY ROTATE(ALPHA,BETA,GAMMA,DRAWME)
LOGICAL DRAWME
CALL TRNMAT(ALPHA,BETA,GAMMA)
CALL PHI(SX,SY,SZ,NX,NY)
CALL AXIS(1,DRAWME)
ASUM=ASUM+ALPHA
BSUM=BSUM+BETA
CSUM=CSUM+GAMMA
IF(.NOT.DRAWME) RETURN
CALL DRAW(SX,SY,SZ,NX,NY)
CALL WRITE(ASUM,BSUM,CSUM)
CALL FADV(4)
RETURN
END

BLOCK DATA
COMMON/LABEL/LAB(9)
DATA LAB/'
END
```

```
      SUBROUTINE SCALE(SX,SY,SZ,NXPTS,NYPTS,CUBE)
      DIMENSION SX(NXPTS,NYPTS),SY(NXPTS,NYPTS),SZ(NXPTS,NYPTS)
      COMMON/MAXES/XMIN ,XMAX,YMIN,YMAX,ZMIN,ZMAX
      LOGICAL CUBE
      REAL MAXX,MAXY,MAXZ
      MAXX=(XMAX-XMIN)/5.5
      MAXY=(YMAX-YMIN)/5.5
      MAXZ=(ZMAX-ZMIN)/5.5
      IF(CUBE) GO TO 21
      MAXX=AMAX1(MAXX,MAXY,MAXZ)
      MAXY=MAXX
      MAXZ=MAXX
   21 DO3 I=1,NXPTS
      DO3 J=1,NYPTS
      SX(I,J)=(SX(I,J)-(XMAX + XMIN)/2.)/MAXX
      SY(I,J)=(SY(I,J)-(YMAX + YMIN)/2.)/MAXY
    3 SZ(I,J)=(SZ(I,J)-(ZMAX + ZMIN)/2.)/MAXZ
      RETURN
      END

      SUBROUTINE  TRNMAT(ALPHA,BETA,GAMMA)
      COMMON/MATRIX/TMAT(3,3)
      A=ALPHA/57.2957795
      B=BETA/57.2957795
      C=GAMMA/57.295795
      SINA=SIN(A)
      SINB=SIN(B)
      SINC=SIN(C)
      COSA=COS(A)
      COSB=COS(B)
      COSC=COS(C)
      TMAT(1,1)=COSB*COSC
      TMAT(1,2)=-COSB*SINC
      TMAT(1,3)=SINB
      TMAT(2,1)=COSA*SINC+SINA*SINB*COSC
      TMAT(2,2)=COSA*COSC-SINA*SINB*SINC
      TMAT(2,3)=-SINA*COSB
      TMAT(3,1)=SINA*SINC-COSA*SINB*COSC
      TMAT(3,2)=SINA*COSC+COSA*SINB*SINC
      TMAT(3,3)=COSA*COSB
      RETURN
      END

      SUBROUTINE PHI(X,Y,Z,NX,NY)
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      COMMON/MATRIX/TMAT(3,3)
      DO1 I=1,NX
      DO1 J=1,NY
      XP=TMAT(1,1)*X(I,J)+TMAT(1,2)*Y(I,J)+TMAT(1,3)*Z(I,J)
      YP=TMAT(2,1)*X(I,J)+TMAT(2,2)*Y(I,J)+TMAT (2,3)*Z(I,J)
      ZP=TMAT(3,1)*X(I,J)+TMAT(3,2)*Y(I,J)+TMAT(3,3)*Z(I,J)
      X(I,J)=XP
      Y(I,J)=YP
    1 Z(I,J)=ZP
      RETURN
      END
```

```
      SUBROUTINE AXIS(I,DRAWME)
      COMMON/MATRIX/TMAT(3,3)
      LOGICAL DRAWME
      DIMENSION XA(2),XB(2),XC(2),YA(2),YB(2),YC(2),X(4,1),Y(4,1),Z(4,1)
      IF(I.NE.0) GO TO 1
      X(1,1)=0.
      X(2,1)=.4
      X(3,1)=0.
      X(4,1)=0.
      Y(1,1)=0.
      Y(2,1)=0.
      Y(3,1)=.4
      Y(4,1)=0.
      Z(1,1)=0.
      Z(2,1)=0.
      Z(3,1)=0.
      Z(4,1)=.4
      RETURN
    1 DO22 I=1,4
      XP=TMAT(1,1)*X(I,1) + TMAT(1,2)*Y(I,1)+TMAT(1,3)*Z(I,1)
      YP=TMAT(2,1)*X(I,1)+TMAT(2,2)*Y(I,1)+TMAT(2,3)*Z(I,1)
      ZP=TMAT(3,1)*X(I,1)+TMAT(3,2)*Y(I,1)+TMAT(3,3)*Z(I,1)
      X(I,1)=XP
      Y(I,1)=YP
      Z(I,1)=ZP
   22 CONTINUE
      IF(DRAWME) GO TO 2
      RETURN
    2 XA(1)=Y(1,1)+4.5
      XB(1)=XA(1)
      XC(1)=XA(1)
      XA(2)=Y(2,1)+4.5
      XB(2)=Y(3,1)+4.5
      XC(2)=Y(4,1)+4.5
      YA(1)=Z(1,1)+3.0
      YB(1)=YA(1)
      YC(1)=YA(1)
      YA(2)=Z(2,1)+3.0
      YB(2)=Z(3,1)+3.0
      YC(2)=Z(4,1)+3.0
      CALL FCHSZ(1)
      CALL FDATM(2)
      CALL FLNSG(XA(1),YA(1),XA(2),YA(2))
      PX=XA(2)
      PY=YA(2)
      CALL FCHAR('X',PX,PY)
      CALL FLNSG(XB(1),YB(1),XB(2),YB(2))
      PX=XB(2)
      PY=YB(2)
      CALL FCHAR('Y',PX,PY)
      CALL FLNSG(XC(1),YC(1),XC(2),YC(2))
      PX=XC(2)
      PY=YC(2)
      CALL FCHAR('Z',PX,PY)
      RETURN
      END
```

```
      SUBROUTINE DRAW(X,Y,Z,NXPTS,NYPTS)
      DIMENSION X(NXPTS,NYPTS),Y(NXPTS,NYPTS),Z(NXPTS,NYPTS)
      DO 1 I=1,NXPTS
      CALL FMOVE(Y(I,1),Z(I,1))
      DO 2 J=2,NYPTS
      CALL FLINE(Y(I,J),Z(I,J))
    2 CONTINUE
    1 CONTINUE
      DO3 I=1,NYPTS
      CALL FMOVE(Y(1,I),Z(1,I))
      DO4  J= 2,NXPTS
      CALL FLINE(Y(J,I),Z(J,I))
    4 CONTINUE
    3 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE WRITE(A1,B1,C1)
      DIMENSION ALINE(15),BLINE(15),CLINE(15)
      COMMON/MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
      COMMON/LABEL/LAB(9)
      INTEGER ALINE,BLINE,CLINE,BLANK/'    '/
      SICK(XXX)=XXX-FLOAT(IFIX(XXX/360.0))*360.0
      A=SICK(A1)
      B=SICK(B1)
      C=SICK(C1)
      DO1 I=1,15
      ALINE(I)=BLANK
      BLINE(I)=BLANK
      CLINE(I)=BLANK
    1 CONTINUE
      CALL FCHSZ(2)
      CALL FDATM(3)
      CALL FTEXT(LAB,36,1,1100,920)
      CALL FTEXT(LAB,36,1,1100,920)
      CALL CONVO('("ALPHA = ",F10.2,"  XMIN =",E10.2,"  XMAX =",
     1E10.2)',ALINE,0,K,A,XMIN,XMAX)
      CALL FCHSZ(1)
      CALL FTEXT(ALINE,K,1,1100,840)
      CALL CONVO('("BETA =  ",F10.2,"  YMIN =",E10.2,"  YMAX =",
     1E10.2)',BLINE,0,L,B,YMIN,YMAX)
      CALL FTEXT(BLINE,L,1,1100,800)
      CALL CONVO('("GAMMA = ",F10.2,"  ZMIN =",E10.2,"  ZMAX =",
     1E10.2)',CLINE,0,M,C,ZMIN,ZMAX)
      CALL FTEXT(CLINE,M,1,1100,760)
      RETURN
      END
```

APPENDIX B

ROUTINES FOR SYSTEM 360--CALCOMP 780 PLOTTER

SUBROUTINE PLOT 3D--VERSION 2

Purpose

Draw and rotate three-dimensional surfaces with IBM 360/50-75 - Calcomp 780 Plotter.

Usage

```
COMMON/MAXES/MIN, XMAX, YMIN, YMAX, ZMIN, ZMAX
COMMON/LABEL/LAB (9)
CALL PLOT 3D (X, Y, Z, NX, NY, CUBE)
CALL ROTATE (ALPHA, BETA, GAMMA, DRAWME)
CALL NOMORE
```

Description of Parameters

XMIN - Less than or equal to the minimum value in array X

XMAX - Greater than or equal to the maximum value in array X

YMIN - Less than or equal to the minimum value in array Y

YMAX - Greater than or equal to the maximum value in array Y

ZMIN - Less than or equal to the minimum value in array Z

ZMAX - Greater than or equal to the maximum value in array Z

X - Two-dimensional array of X-coordinates

Y - Two-dimensional array of Y-coordinates

Z - Two-dimensional array of Z-coordinates

NX - Number of rows in X, Y, Z arrays

NY - Number of columns in X, Y, Z arrays

CUBE - LOGICAL variable set to .TRUE. if the three directions are to be scaled independently of each other and set to .FALSE. if the three directions are to be scaled dependently.

ALPHA- Angle of rotation about the X-axis, given in degrees

BETA - Angle of rotation about the Y-axis, given in degrees

GAMMA- Angle of rotation about the Z-axis, given in degrees

DRAWME- LOGICAL variable set to .TRUE. if the rotated view is to be drawn, and set to .FALSE. if this view is not to be drawn.

LAB - Array of BCD characters used to label the drawing. If label is desired, supply 9 or less four-character words in COMMON/LABEL/LAB (9).

Remarks

A call to PLOT 3D defines and scales the surface. Subsequent calls to ROTATE rotate it to new frames of reference. The surface is not returned to its original position after a rotation. The surface is rotated about the fixed XYZ axes in this order: first about the Z-axis, then about the Y-axis, and finally about the X-axis. Finally a call to NOMORE closes the Calcomp tape unit.

Care should be exercised in providing that the arrays X, Y, Z in the calling program have dimensions (NX, NY). Dimensions different from this may cause errors in the plots.

Subroutines Required

SCALE
TRNMAT
PHI
AXIS
DRAW
WRITE

In addition, PLOT 3D calls routines for the Calcomp 780 Plotter:

PLOTS - Allocates a work area to the PLOT routine
PLOT - Moves the pen to the specified page coordinates with pen lifed or lowered
SYMBOL - Draws a specified symbol
LINE - Draws a line or symbol through each successive data point

Finally, PLOT 3D calls an Argonne-developed routine which follows:

CONVO - Core-to-core conversion, internal machine representation to EBCDIC

A listing of the subroutines follows:

```
SUBROUTINE PLOT3D(SX,SY,SZ,NX,NY,CUBE)
LOGICAL CUBE
COMMON /MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
DIMENSION SX(NX,NY),SY(NX,NY),SZ(NX,NY)
COMMON/CALCMP/CCOMP(1000)
CALL PLOTS(CCOMP(1),4000)
CALL SCALE(SX,SY,SZ,NX,NY,CUBE)
CALL AXIS(0,.FALSE.)
ASUM=0.0
BSUM=0.0
CSUM=0.0
RETURN
ENTRY ROTATE(ALPHA,BETA,GAMMA,DRAWME)
LOGICAL DRAWME
CALL TRNMAT(ALPHA,BETA,GAMMA)
CALL PHI(SX,SY,SZ,NX,NY)
CALL AXIS(1,DRAWME)
ASUM=ASUM+ALPHA
BSUM=BSUM+BETA
CSUM=CSUM+GAMMA
IF(.NOT.DRAWME) RETURN
CALL PLOT(0.0,1.5,-3)
CALL DRAW(SX,SY,SZ,NX,NY)
CALL WRITE(ASUM,BSUM,CSUM)
CALL PLOT(10.0,-1.5,-3)
RETURN
ENTRY NOMORE
CALL PLOT(0.0,0.0,999)
RETURN
END


BLOCK DATA
COMMON/LABEL/LAB(9)
DATA LAB/'
END
```

```
      SUBROUTINE SCALE(SX,SY,SZ,NXPTS,NYPTS,CUBE)
      DIMENSION SX(NXPTS,NYPTS),SY(NXPTS,NYPTS),SZ(NXPTS,NYPTS)
      COMMON/MAXES/XMIN ,XMAX,YMIN,YMAX,ZMIN,ZMAX
      LOGICAL CUBE
      REAL MAXX,MAXY,MAXZ
      MAXX=(XMAX-XMIN)/5.5
      MAXY=(YMAX-YMIN)/5.5
      MAXZ=(ZMAX-ZMIN)/5.5
      IF(CUBE) GO TO 21
      MAXX=AMAX1(MAXX,MAXY,MAXZ)
      MAXY=MAXX
      MAXZ=MAXX
   21 DO3 I=1,NXPTS
      DO3 J=1,NYPTS
      SX(I,J)=(SX(I,J)-(XMAX + XMIN)/2.)/MAXX
      SY(I,J)=(SY(I,J)-(YMAX + YMIN)/2.)/MAXY
    3 SZ(I,J)=(SZ(I,J)-(ZMAX + ZMIN)/2.)/MAXZ
      RETURN
      END

      SUBROUTINE  TRNMAT(ALPHA,BETA,GAMMA)
      COMMON/MATRIX/TMAT(3,3)
      A=ALPHA/57.2957795
      B=BETA/57.2957795
      C=GAMMA/57.295795
      SINA=SIN(A)
      SINB=SIN(B)
      SINC=SIN(C)
      COSA=COS(A)
      COSB=COS(B)
      COSC=COS(C)
      TMAT(1,1)=COSB*COSC
      TMAT(1,2)=-COSB*SINC
      TMAT(1,3)=SINB
      TMAT(2,1)=COSA*SINC+SINA*SINB*COSC
      TMAT(2,2)=COSA*COSC-SINA*SINB*SINC
      TMAT(2,3)=-SINA*COSB
      TMAT(3,1)=SINA*SINC-COSA*SINB*COSC
      TMAT(3,2)=SINA*COSC+COSA*SINB*SINC
      TMAT(3,3)=COSA*COSB
      RETURN
      END

      SUBROUTINE PHI(X,Y,Z,NX,NY)
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      COMMON/MATRIX/TMAT(3,3)
      DO1 I=1,NX
      DO1 J=1,NY
      XP=TMAT(1,1)*X(I,J)+TMAT(1,2)*Y(I,J)+TMAT(1,3)*Z(I,J)
      YP=TMAT(2,1)*X(I,J)+TMAT(2,2)*Y(I,J)+TMAT (2,3)*Z(I,J)
      ZP=TMAT(3,1)*X(I,J)+TMAT(3,2)*Y(I,J)+TMAT(3,3)*Z(I,J)
      X(I,J)=XP
      Y(I,J)=YP
    1 Z(I,J)=ZP
      RETURN
      END
```

```
      SUBROUTINE AXIS(I,DRAWME)
      COMMON/MATRIX/TMAT(3,3)
      COMMON/CALCMP/CCOMP(1000)
      LOGICAL DRAWME
      DIMENSION XA(2),XB(2),XC(2),YA(2),YB(2),YC(2),X(4)  ,Y(4)  ,Z(4)
      IF(I.NE.0) GO TO 1
      X(1)  =0.
      X(2)  =.4
      X(3)  =0.
      X(4)  =0.
      Y(1)  =0.
      Y(2)  =0.
      Y(3)  =.4
      Y(4)  =0.
      Z(1)  =0.
      Z(2)  =0.
      Z(3)  =0.
      Z(4)  =.4
      RETURN
    1 DO22 I=1,4
      XP=TMAT(1,1)*X(I)   + TMAT(1,2)*Y(I)  +TMAT(1,3)*Z(I)
      YP=TMAT(2,1)*X(I)  +TMAT(2,2)*Y(I)  +TMAT(2,3)*Z(I)
      ZP=TMAT(3,1)*X(I)  +TMAT(3,2)*Y(I)  +TMAT(3,3)*Z(I)
      X(I)  =XP
      Y(I)  =YP
      Z(I)  =ZP
   22 CONTINUE
      IF(DRAWME) GO TO 2
      RETURN
    2 XA(1)=Y(1)  +7.0
      XB(1)=XA(1)
      XC(1)=XA(1)
      XA(2)=Y(2)  +7.0
      XB(2)=Y(3)  +7.0
      XC(2)=Y(4)  +7.0
      YA(1)=Z(1)  +7.0
      YB(1)=YA(1)
      YC(1)=YA(1)
      YA(2)=Z(2)  +7.0
      YB(2)=Z(3)  +7.0
      YC(2)=Z(4)  +7.0
      CALL PLOT(XA(1),YA(1),3)
      CALL PLOT(XA(2),YA(2),2)
      CALL SYMBOL(XA(2),YA(2),0.10,'X',0.0,1)
      CALL PLOT(XB(1),YB(1),3)
      CALL PLOT(XB(2),YB(2),2)
      CALL SYMBOL(XB(2),YB(2),0.10,'Y',0.0,1)
      CALL PLOT(XC(1),YC(1),3)
      CALL PLOT(XC(2),YC(2),2)
      CALL SYMBOL(XC(2),YC(2),0.10,'Z',0.0,1)
      RETURN
      END
```

```
      SUBROUTINE DRAW(X,Y,Z,NXPTS,NYPTS)
      DIMENSION X(NXPTS,NYPTS),Y(NXPTS,NYPTS),Z(NXPTS,NYPTS)
      COMMON/CALCMP/CCOMP(1000)
      CALL OFFSET(-5.0,1.25,-5.0,1.25)
      DO1 I=1,NXPTS
      CALL PLOT(Y(I,1),Z(I,1),13)
      DO2 J=2,NYPTS
      CALL PLOT(Y(I,J),Z(I,J),12)
    2 CONTINUE
    1 CONTINUE
      DO3 I=1,NYPTS
      CALL PLOT(Y(1,I),Z(1,I),13)
      DO4 J=2,NXPTS
      CALL PLOT(Y(J,I),Z(J,I),12)
    4 CONTINUE
    3 CONTINUE
      RETURN
      END


      SUBROUTINE WRITE(A1,B1,C1)
      DIMENSION ALINE(15),BLINE(15),CLINE(15)
      COMMON/MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
      COMMON/LABEL/LAB(9)
      COMMON/CALCMP/CCOMP(1000)
      INTEGER ALINE,BLINE,CLINE,BLANK/'    '/
      SICK(XXX)=XXX-FLOAT(IFIX(XXX/360.0))*360.0
      A=SICK(A1)
      B=SICK(B1)
      C=SICK(C1)
      DO1 I=1,15
      ALINE(I)=BLANK
      BLINE(I)=BLANK
      CLINE(I)=BLANK
    1 CONTINUE
      CALL SYMBOL(0.0,-.3,0.2,LAB(1),0.0,36)
      CALL CONVO('("ALPHA = ",F10.2,"  XMIN =",E10.2,"  XMAX =",
     1E10.2)',ALINE,0,K,A,XMIN,XMAX)
      CALL SYMBOL(0.0,-.5,0.1,ALINE(1),0.0,K)
      CALL CONVO('("BETA =  ",F10.2,"  YMIN =",E10.2,"  YMAX =",
     1E10.2)',BLINE,0,L,B,YMIN,YMAX)
      CALL SYMBOL(0.0,-.7,0.1,BLINE(1),0.0,L)
      CALL CONVO('("GAMMA = ",F10.2,"  ZMIN =",E10.2,"  ZMAX =",
     1E10.2)',CLINE,0,M,C,ZMIN,ZMAX)
      CALL SYMBOL(0.0,-.9,0.1,CLINE(1),0.0,M)
      RETURN
      END
```

APPENDIX C

ROUTINES FOR SYSTEM 360--IBM 2280 FILM RECORDER

SUBROUTINE PLOT 3D--VERSION 3

Purpose

Draw and rotate throw-dimensional surfaces with IBM 360/50-75 - 2280 Film Recorder. This version of PLOT 3D differs from Versions 1 and 2 in that "invisible" lines are not drawn.

Usage

```
COMMON/MAXES/XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX
COMMON/LABEL/LAB (9)
CALL PLOT 3D (X, Y, Z, NX, NY, CUBE)
CALL ROTATE (ALPHA, BETA, GAMMA, DRAWME)
```

Description of Parameters

XMIN - Less than or equal to the minimum value in array X

MXAX - Greater than or equal to the maximum value in array Y

YMIN - Less than or equal to the minimum value in array Y

YMAX - Greater than or equal to the maximum value in array Y

ZMIN - Less than or equal to the minimum value in array Z

ZMAX - Greater than or equal to the maximum value in array Z

X - Two-dimensional array of X-coordinates

Y - Two-dimensional array of Y-coordinates

Z - Two-dimensional array of Z-coordinates

NX - Number of rows in arrays X, Y, Z

NY - Number of columns in arrays X, Y, Z

CUBE - LOGICAL variable set to .TRUE. if the three directions are to be scaled independently of each other and set to .FALSE. if the three directions are to be scaled dependently.

ALPHA - Angle of rotation about the X-axis, given in degrees

BETA - Angle of rotation about the Y-axis, given in degrees

GAMMA - Angle of rotation about the Z-axis, given in degrees

DRAWME - LOGICAL variable set to .TRUE. if the rotated view is to be drawn, and set to .FALSE. if this view is not to be drawn.

LAB - Array of BCD characters used to label the drawing. If label is desired, supply 9 or less four-characters words in COMMON/LABEL/LAB (9).

Remarks

A call to PLOT 3D defines and scales the surface. Subsequent calls to ROTATE rotate it to new frames of reference. The surface is not returned to its original position after a rotation. The surface is rotated about the fixed XYZ axes in this order: first about the Z-axis, then about the Y-axis, and finally about the X-axis.

Care should be exercised in providing that the arrays X, Y, Z in the calling program have dimensions (NX, NY). Dimensions different from this may cause errors in the plots.

This version of PLOT 3D requires much more computer time than Versions 1 and 2. A grid of 121 points required nearly 0.5 min during test runs. A grid of 500 points required 5 min of computer time.

In general, a line segment is drawn between two "visible" endpoints and is not drawn between two "invisible" endpoints. If one endpoint is "visible" and the other endpoint is "invisible," the "visible" portion of the segment joining the points is drawn. No attempt is made to mask out an "invisible" part of a segment between two "visible" endpoints or to draw a "visible" part of a segment between two "invisible" endpoints.

Subroutines Required

SCALE
TRNMAT
PHI
AXIS
SEE
SCAN
TORF
ONEVIS
PLANE
WRITE

In addition, PLOT 3D calls routines for the IBM 2280 Film Recorder. They follow:

FPLTM - Sets the vector and point-plotting mode

FMAREA - Defines the size of the film area onto which those coordinates specified as belonging to the user's coordinate system via the FDATM routine are to be mapped.

FXYTRN - Defines the positioning of the film area

FLIND - Sets the film line density

FLINW - Sets the film line width

FDATM - Sets the arithmetic mode and reference frame of the input X, Y values for vectors, points, and beam positioning.

FXYLIM - Sets the maximum and minimum values of the user's coordinate system

FADV - Advances the film

FCHSZ - Sets the character size of the characters generated by the character generator

FLNSG - Draws a line segment between two specified endpoints

FMOVE - Positions the beam at the specified coordinates

FTEXT - Plots a string of characters

Finally, PLOT 3D calls an Argonne-developed routined which follows:

CONVO - Core-to-core conversion, internal machine representation to EBCDIC

Storage Requirements

This version of PLOT 3D requires about $E370_{16}$ bytes of storage, exclusive of Film Recorder subroutines and array storage.

A listing of the PLOT 3D Version 3 subroutines follow:

```
SUBROUTINE PLOT3D(SX,SY,SZ,NX,NY,CUBE)
LOGICAL CUBE
COMMON /MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
DIMENSION SX(NX,NY),SY(NX,NY),SZ(NX,NY)
CALL FPLTM(1)
CALL FDATM(3)
CALL FMAREA(2200,2200)
CALL FXYTRN(1000,1000)
CALL FLIND(1)
CALL FLINW(1)
CALL FDATM(2)
CALL FXYLIM(-5.0,-5.0,5.0,5.0)
CALL SCALE(SX,SY,SZ,NX,NY,CUBE)
CALL AXIS(0,.FALSE.)
ASUM=0.0
BSUM=0.0
CSUM=0.0
RETURN
ENTRY ROTATE(ALPHA,BETA,GAMMA,DRAWME)
LOGICAL DRAWME
CALL TRNMAT(ALPHA,BETA,GAMMA)
CALL PHI(SX,SY,SZ,NX,NY)
CALL AXIS(1,DRAWME)
ASUM=ASUM+ALPHA
BSUM=BSUM+BETA
CSUM=CSUM+GAMMA
IF(.NOT.DRAWME) RETURN
CALL SEE (SX,SY,SZ,NX,NY)
CALL WRITE(ASUM,BSUM,CSUM)
CALL FADV(4)
RETURN
END


BLOCK DATA
COMMON/LABEL/LAB(9)
DATA LAB/'
END
```

```
      SUBROUTINE SCALE(SX,SY,SZ,NXPTS,NYPTS,CUBE)
      DIMENSION SX(NXPTS,NYPTS),SY(NXPTS,NYPTS),SZ(NXPTS,NYPTS)
      COMMON/MAXES/XMIN ,XMAX,YMIN,YMAX,ZMIN,ZMAX
      LOGICAL CUBE
      REAL MAXX,MAXY,MAXZ
      MAXX=(XMAX-XMIN)/5.5
      MAXY=(YMAX-YMIN)/5.5
      MAXZ=(ZMAX-ZMIN)/5.5
      IF(CUBE) GO TO 21
      MAXX=AMAX1(MAXX,MAXY,MAXZ)
      MAXY=MAXX
      MAXZ=MAXX
   21 DO3 I=1,NXPTS
      DO3 J=1,NYPTS
      SX(I,J)=(SX(I,J)-(XMAX + XMIN)/2.)/MAXX
      SY(I,J)=(SY(I,J)-(YMAX + YMIN)/2.)/MAXY
    3 SZ(I,J)=(SZ(I,J)-(ZMAX + ZMIN)/2.)/MAXZ
      RETURN
      END

      SUBROUTINE  TRNMAT(ALPHA,BETA,GAMMA)
      COMMON/MATRIX/TMAT(3,3)
      A=ALPHA/57.2957795
      B=BETA/57.2957795
      C=GAMMA/57.295795
      SINA=SIN(A)
      SINB=SIN(B)
      SINC=SIN(C)
      COSA=COS(A)
      COSB=COS(B)
      COSC=COS(C)
      TMAT(1,1)=COSB*COSC
      TMAT(1,2)=-COSB*SINC
      TMAT(1,3)=SINB
      TMAT(2,1)=COSA*SINC+SINA*SINB*COSC
      TMAT(2,2)=COSA*COSC-SINA*SINB*SINC
      TMAT(2,3)=-SINA*COSB
      TMAT(3,1)=SINA*SINC-COSA*SINB*COSC
      TMAT(3,2)=SINA*COSC+COSA*SINB*SINC
      TMAT(3,3)=COSA*COSB
      RETURN
      END

      SUBROUTINE PHI(X,Y,Z,NX,NY)
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      COMMON/MATRIX/TMAT(3,3)
      DO1 I=1,NX
      DO1 J=1,NY
      XP=TMAT(1,1)*X(I,J)+TMAT(1,2)*Y(I,J)+TMAT(1,3)*Z(I,J)
      YP=TMAT(2,1)*X(I,J)+TMAT(2,2)*Y(I,J)+TMAT (2,3)*Z(I,J)
      ZP=TMAT(3,1)*X(I,J)+TMAT(3,2)*Y(I,J)+TMAT(3,3)*Z(I,J)
      X(I,J)=XP
      Y(I,J)=YP
    1 Z(I,J)=ZP
      RETURN
      END
```

```
      SUBROUTINE AXIS(I,DRAWME)
      COMMON/MATRIX/TMAT(3,3)
      LOGICAL DRAWME
      DIMENSION XA(2),XB(2),XC(2),YA(2),YB(2),YC(2),X(4,1),Y(4,1),Z(4,1)
      IF(I.NE.0) GO TO 1
      X(1,1)=0.
      X(2,1)=.4
      X(3,1)=0.
      X(4,1)=0.
      Y(1,1)=0.
      Y(2,1)=0.
      Y(3,1)=.4
      Y(4,1)=0.
      Z(1,1)=0.
      Z(2,1)=0.
      Z(3,1)=0.
      Z(4,1)=.4
      RETURN
    1 DO22 I=1,4
      XP=TMAT(1,1)*X(I,1) + TMAT(1,2)*Y(I,1)+TMAT(1,3)*Z(I,1)
      YP=TMAT(2,1)*X(I,1)+TMAT(2,2)*Y(I,1)+TMAT(2,3)*Z(I,1)
      ZP=TMAT(3,1)*X(I,1)+TMAT(3,2)*Y(I,1)+TMAT(3,3)*Z(I,1)
      X(I,1)=XP
      Y(I,1)=YP
      Z(I,1)=ZP
   22 CONTINUE
      IF(DRAWME) GO TO 2
      RETURN
    2 XA(1)=Y(1,1)+4.5
      XB(1)=XA(1)
      XC(1)=XA(1)
      XA(2)=Y(2,1)+4.5
      XB(2)=Y(3,1)+4.5
      XC(2)=Y(4,1)+4.5
      YA(1)=Z(1,1)+3.0
      YB(1)=YA(1)
      YC(1)=YA(1)
      YA(2)=Z(2,1)+3.0
      YB(2)=Z(3,1)+3.0
      YC(2)=Z(4,1)+3.0
      CALL FCHSZ(1)
      CALL FDATM(2)
      CALL FLNSG(XA(1),YA(1),XA(2),YA(2))
      PX=XA(2)
      PY=YA(2)
      CALL FCHAR('X',PX,PY)
      CALL FLNSG(XB(1),YB(1),XB(2),YB(2))
      PX=XB(2)
      PY=YB(2)
      CALL FCHAR('Y',PX,PY)
      CALL FLNSG(XC(1),YC(1),XC(2),YC(2))
      PX=XC(2)
      PY=YC(2)
      CALL FCHAR('Z',PX,PY)
      RETURN
```

```
      SUBROUTINE SEE (X,Y,Z,NX,NY)
C  THIS ROUTINE CHECKS EACH OF THE POINTS IN THE XYZ ARRAYS FOR
C  VISIBILITY. IT JOINS VISIBLE  PAIRS WITH SEGMENTS , PASSES OVER
C  INVISIBLE PAIRS WITHOUT PLOTTING  AND  INTERPOLATES A VISIBLE SEGMENT
C  BETWEEN PAIRS ONLY ONE OF  WHICH IS VISIBLE.
C  SUBROUTINE REQUIRES THE USE OF SUBROUTINE SCAN, FUNCTION TORF
C  AND SUBROUTINE ONEVIS AND SUBROUTINE PLANE
C
      LOGICAL SEEUM(100,100),TORF,Q,ANSWER
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      DIMENSION TRIX(5),TRIY(5),TRIZ(5)
      DO2 I=1,NX
      DO2 J=1,NY
      CALL SCAN(X(I,J),Y(I,J),Z(I,J),ANSWER,X,Y,Z,NX,NY)
      SEEUM(I,J)=ANSWER
    2 CONTINUE
C  AT THIS POINT IN EXECUTION ALL POINTS HAVE BEEN CHECKED FOR VISIBILI-
C  TY.    SEEUM(I,J) CONTAINS 'TRUE' FOR VISIBLE POINTS AND 'FALSE' FOR
C  INVISIBLE POINTS
C
C  THE PLOTTING ROUTINE FOLLOWS
C
   13 NYLONE=NY-1
      DO 14 I=1,NX
      DO 14 J=1,NYLONE
      IF(SEEUM(I,J))GO TO 401
      IF(SEEUM(I,J+1))GO TO 402
C
C  GO TO NEXT PAIR  IF BOTH POINTS INVISIBLE
C
      GO TO 14
  401 IF(SEEUM(I,J+1))GO TO 20
C
C  GO TO PLOT SEGMENT IF BOTH POINTS ARE VISIBLE
C
C
C  IF EXECUTION GETS HERE ONE POINT IS VISIBLE AND THE OTHER INVISIBLE
C  NOW INTERPOLATE BETWEEN THE POINTS TO FIND AN APPROXIMATE VISIBLE
C  ENDPOINT TO USE INSTEAD OF THE INVISIBLE POINT
C
C
      GO TO 15
  402 CONTINUE
      ISAVE=I
      JSAVE=J+1
      CALL ONEVIS(X(I,J+1),Y(I,J+1),Z(I,J+1),X(I,J),Y(I,J),Z(I,J),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
      GO TO 16
   15 ISAVE=I
      JSAVE=J
      CALL ONEVIS(X(I,J),Y(I,J),Z(I,J),X(I,J+1),Y(I,J+1),Z(I,J+1),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
   16 CALL FLNSG(Y(ISAVE,JSAVE),Z(ISAVE,JSAVE),YPLOT,ZPLOT)
      GO TO 14
   20 CALL FLNSG(Y(I,J),Z(I,J),Y(I,J+1),Z(I,J+1))
```

```
   14 CONTINUE
C
C  NOW PLOT SEGMENTS ALONG THE OTHER DIRECTION OF THE GRID.
C
      NXLONE= NX-1
      DO21 J=1,NY
      DO21 I=1,NXLONE
      IF(SEEUM(I,J))GO TO 403
      IF(SEEUM(I+1,J))GO TO 404
      GO TO 21
  403 IF(SEEUM(I+1,J))GO TO 26
      GO TO 22
  404 CONTINUE
      ISAVE=I+1
      JSAVE=J
      CALL ONEVIS(X(I+1,J),Y(I+1,J),Z(I+1,J),X(I,J),Y(I,J),Z(I,J),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
      GO TO 31
   22 ISAVE=I
      JSAVE=J
      CALL ONEVIS(X(I,J),Y(I,J),Z(I,J),X(I+1,J),Y(I+1,J),Z(I+1,J),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
   31 CALL FLNSG(Y(ISAVE,JSAVE),Z(ISAVE,JSAVE),YPLOT,ZPLOT)
      GO TO 21
   26 CALL FLNSG(Y(I,J),Z(I,J),Y(I+1,J),Z(I+1,J))
   21 CONTINUE
C
C  THE FOLLOWING ROUTINE PLOTS A SEGMENT BETWWEN OPPOSITE VISIBLE
C  VERTICES OF A QUADRILATERAL IF AT LEAST ONE OF THE OTHER VERTICES
C  IS INVISIBLE AND BEHIND THE PLANE OF THE OTHER THREE.
C
      DO355 I=1,NXLONE
      DO355 J=1,NYLONE
      IF(.NOT.SEEUM(I,J))GO TO 361
      IF(.NOT.SEEUM(I+1,J+1))GO TO 361
      IF(.NOT.SEEUM(I,J+1))GO TO 360
  361 IF(.NOT.SEEUM(I,J))GO TO 363
      IF(.NOT.SEEUM(I+1,J+1))GO TO 363
      IF(.NOT.SEEUM(I+1,J))GO TO 362
  363 IF(.NOT.SEEUM(I+1,J))GO TO 365
      IF(.NOT.SEEUM(I,J+1))GO TO 365
      IF(.NOT.SEEUM(I,J))GO TO 364
  365 IF(.NOT.SEEUM(I+1,J))GO TO 355
      IF(.NOT.SEEUM(I,J+1))GO TO 355
      IF(.NOT.SEEUM(I+1,J+1))GO TO 366
      GO TO 355
  360 TRIX(1)=X(I,J)
      TRIY(1)=Y(I,J)
      TRIZ(1)=Z(I,J)
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIZ(2)=Z(I,J+1)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIZ(3)=Z(I+1,J+1)
```

```
      CALL PLANE(TRIX,TRIY,TRIZ,X(I+1,J),Y(I+1,J),Z(I+1,J),ANSWER)
      IF(ANSWER)GO TO 356
      GO TO 361
  362 TRIX(1)=X(I,J)
      TRIY(1)=Y(I,J)
      TRIZ(1)=Z(I,J)
      TRIX(2)=X(I+1,J)
      TRIY(2)=Y(I+1,J)
      TRIZ(2)=Z(I+1,J)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIZ(3)=Z(I+1,J+1)
      CALL PLANE(TRIX,TRIY,TRIZ,X(I,J+1),Y(I,J+1),Z(I,J+1),ANSWER)
      IF(ANSWER)GO TO 356
      GO TO 363
  364 TRIX(1)=X(I+1,J)
      TRIY(1)=Y(I+1,J)
      TRIZ(1)=Z(I+1,J)
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIZ(2)=Z(I,J+1)
      TRIX(3)=X(I,J)
      TRIY(3)=Y(I,J)
      TRIZ(3)=Z(I,J)
      CALL PLANE(TRIX,TRIY,TRIZ,X(I+1,J+1),Y(I+1,J+1),Z(I+1,J+1),ANSWER)
      IF(ANSWER)GO TO 357
      GO TO 365
  366 TRIX(1)=X(I+1,J)
      TRIY(1)=Y(I+1,J)
      TRIZ(1)=Z(I+1,J)
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIZ(2)=Z(I,J+1)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIZ(3)=Z(I+1,J+1)
      CALL PLANE(TRIX,TRIY,TRIZ,X(I,J),Y(I,J),Z(I,J),ANSWER)
      IF(ANSWER)GO TO 357
      GO TO 355
  356 CALL FLNSG(Y(I,J),Z(I,J),Y(I+1,J+1),Z(I+1,J+1))
      GO TO 355
  357 CALL FLNSG(Y(I+1,J),Z(I+1,J),Y(I,J+1),Z(I,J+1))
  355 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SCAN(XP,YP,ZP,ANSWER,X,Y,Z,NX,NY)
C  THIS ROUTINE TAKES A POINT AND DETERMINES ITS VISIBILITY.  IF VISIBLE
C    1ANSWER IS FILLED WITH 'TRUE', OTHERWISE WITH 'FALSE'.
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      DIMENSION TRIX(5),TRIY(5),TRIZ(5)
      LOGICAL ANSWER,YES/.TRUE./,NOTYES/.FALSE./,Q,TORF
      K=1
      L=1
   10 LL=1
      DO3 M=1,2
      MM=L-1+M
      TRIX(M)=X(K,MM)
      TRIY(M)=Y(K,MM)
      TRIZ(M)=Z(K,MM)
      TRIX(M+3)=TRIX(M)
      TRIY(M+3)=TRIY(M)
    3 TRIZ(M+3)=TRIZ(M)
      TRIX(3)=X(K+1,L)
      TRIY(3)=Y(K+1,L)
      TRIZ(3)=Z(K+1,L)
    8 DO4 M=1,3
      IF(XP.NE.TRIX(M))GO TO 4
      IF(YP.NE.TRIY(M))GO TO 4
      IF(ZP.EQ.TRIZ(M))GO TO 6
C
C  IF THE ABOVE CONDITION HOLDS THE POINT IS A VERTEX OF THE TRIANGLE
    4 CONTINUE
      DO5 M=1,3
      Q=TORF(TRIY(M),TRIZ(M),TRIY(M+1), TRIZ(M+1),TRIY(M+2),TRIZ(M+2),
     1YP,ZP)
      IF(.NOT.Q)GO TO 6
    5 CONTINUE
C  IF EXECUTION GETS TO HERE THE POINT IS IN THE TRIANGLE ON THE YZ
C  PLANE.  NEXT CHECK FOR BEING IN FRONT OF THE TRIANGLE IN SPACE.
C
      IF(XP.LE.TRIX(1))GO TO 222
      IF(XP.LE.TRIX(2))GO TO 222
      IF(XP.GT.TRIX(3))GO TO 6
  222 CONTINUE
      IF(XP.GE.TRIX(1))GO TO 223
      IF(XP.GE.TRIX(2))GO TO 223
      IF(XP.LT.TRIX(3))GO TO 22
  223 CONTINUE
C  IF EXECUTION GETS TO HERE POINT IS NEAR TRIANGLE. NEXT CHECK FOR
C  BEING IN FRONT OR  BEHIND.
      CALL PLANE(TRIX,TRIY,TRIZ,XP,YP,ZP,ANSWER)
      IF(ANSWER) GO TO 6
C
C  IF EXECUTION GETS TO HERE THE POINT IS INVISIBLE
   22 ANSWER=NOTYES
      RETURN
C  FOLLOWING IS A ROUTINE TO PASS ON TO THE NEXT TRIANGLE
    6 GO TO (30,9),LL
   30 LL=2
```

```
      MM=K-1+M
      TRIX(M)=X(MM,L+1)
      TRIY(M)=Y(MM,L+1)
      TRIZ(M)=Z(MM,L+1)
      TRIX(M+3)=TRIX(M)
      TRIY(M+3)=TRIY(M)
    7 TRIZ(M+3)=TRIZ(M)
      GO TO 8
C   SET K AND L  FOR NEXT PAIR OF TRIANGLES
    9 IF((L+1).GE.NY)GO TO 11
      L=L+1
      GO TO 10
   11 IF((K+1).GE.NX)GO TO 12
C   FINISHED WITH ALL TRIANGLES
      K=K+1
      L=1
      GO TO 10
C   THE POINT HAS BEEN CHECKED WITH THE LAST TRIANGLE, THE POINT IS
C   VISIBLE.
   12 ANSWER= YES
      RETURN
      END


      LOGICAL FUNCTION TORF(X1,Y1,X2,Y2,X3,Y3,XP,YP)
C   THIS ROUTINE SUPPLIES TRUE  IF POINT(XP,YP) AND (X3,Y3) SAME SIDE OF
C   LINE (X1,Y1),(X2,Y2). FALSE IF ON OPPOSITE SIDES
      TEE(A1,A2,A3)=(A3-A1)/(A2-A1)
      IF(X2.EQ.X1)GO TO 3
      IF(Y2.EQ.Y1)GO TO 5
      TX1=TEE(X1,X2,X3)
      TY1=TEE(Y1,Y2,Y3)
      TXP=TEE(X1,X2,XP)
      TYP=TEE(Y1,Y2,YP)
      IF(ABS(TYP-TXP).LE.1.0E-5)GO TO 4
      IF((TY1-TX1).LE.1.0E-5)GO TO 22
      IF((TYP-TXP).GT.1.0E-5)GO TO 2
   22 IF((TY1-TX1).GE.-1.0E-5)GO TO 4
      IF((TYP-TXP).LT.-1.0E-5)GO TO 2
    4 TORF=.FALSE.
      RETURN
    2 TORF=.TRUE.
      RETURN
    3 IF((XP-X1)*(X3-X1).LT.0.0)GO TO 4
      GO TO 2
    5 IF((YP-Y1)*(Y3-Y1).LT.0.0)GO TO 4
      GO TO 2
      END
```

```
      SUBROUTINE ONEVIS(XVIS,YVIS,ZVIS,XINVIS,YINVIS,ZINVIS,YPLOT,ZPLOT,
     1X,Y,Z,NX,NY)
C  THIS ROUTINE INTERPOLATES BETWEEN A VISIBLE POINT AND AN INVISIBLE
C  POINT TO FIND THE ENDPOINTS OF A VISIBLE SEGMENT.
      LOGICAL VIS
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      DIST=SQRT((XVIS-XINVIS)**2+(YVIS-YINVIS)**2+(ZVIS-ZINVIS)**2)
      XV=XVIS
      YV=YVIS
      ZV=ZVIS
      XINV=XINVIS
      YINV=YINVIS
      ZINV=ZINVIS
      DO17 KK=1,7
      XMID=(XV  +XINV  )/2.0
      YMID=(YV  +YINV  )/2.0
      ZMID=(ZV  +ZINV  )/2.0
C  HERE CALL SCAN TO CHECK VISIBILITY OF MIDPOINT
      CALL SCAN(XMID,YMID,ZMID,VIS,X,Y,Z,NX,NY)
      IF(VIS)GO TO 19
      XINV  = XMID
      YINV  = YMID
      ZINV  = ZMID
      GO TO 18
   19 XV  = XMID
      YV  = YMID
      ZV  = ZMID
   18 IF((SQRT((XV-XINV)**2+(YV-YINV)**2+(ZV-ZINV)**2)).LE.(.01*DIST))
     1GO TO 22
   17 CONTINUE
   22 YPLOT=YMID
      ZPLOT=ZMID
      RETURN
      END


      SUBROUTINE PLANE(ARRAYX,ARRAYY,ARRAYZ       ,XP,YP,ZP,FRONT)
      LOGICAL FRONT
      DIMENSION ARRAY(3,3),SAVE(3),ARRAYX(3),ARRAYY(3),ARRAYZ(3)
      DO333 I=1,3
      ARRAY(I,1)=ARRAYX(I)
      ARRAY(I,2)=ARRAYY(I)
      ARRAY(I,3)=ARRAYZ(I)
  333 CONTINUE
      DO2 J=1,3
      ARRAY(2,J)=ARRAY(1,J)-ARRAY(2,J)
      ARRAY(3,J)=ARRAY(1,J)-ARRAY(3,J)
    2 CONTINUE
      IF(ARRAY(2,3).EQ.0.0)GO TO 3
    7 DO4 J=1,3
      ARRAY(3,J)=ARRAY(3,3)*ARRAY(2,J)-ARRAY(2,3)*ARRAY(3,J)
```

```
 4 CONTINUE
   B=-ARRAY(3,1)/ARRAY(3,2)
   C=-(ARRAY(2,1)+B*ARRAY(2,2))/ARRAY(2,3)
   D=-(ARRAY(1,1)+B*ARRAY(1,2)+C*ARRAY(1,3))
   XPLANE=-(B*YP+C*ZP+D)
   IF(XP.GT.(XPLANE+1.0E-5))GO TO 5
   FRONT=.FALSE.
   RETURN
 5 FRONT=.TRUE.
   RETURN
 3 DO6 J=1,3
   SAVE(J)=ARRAY(2,J)
   ARRAY(2,J)=ARRAY(3,J)
   ARRAY(3,J)=SAVE(J)
 6 CONTINUE
   GO TO 7
   END


   SUBROUTINE WRITE(A1,B1,C1)
   DIMENSION ALINE(15),BLINE(15),CLINE(15)
   COMMON/MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
   COMMON/LABEL/LAB(9)
   INTEGER ALINE,BLINE,CLINE,BLANK/'    '/
   SICK(XXX)=XXX-FLOAT(IFIX(XXX/360.0))*360.0
   A=SICK(A1)
   B=SICK(B1)
   C=SICK(C1)
   DO1 I=1,15
   ALINE(I)=BLANK
   BLINE(I)=BLANK
   CLINE(I)=BLANK
 1 CONTINUE
   CALL FCHSZ(2)
   CALL FDATM(3)
   CALL FTEXT(LAB,36,1,1100,920)
   CALL FTEXT(LAB,36,1,1100,920)
   CALL CONVO('("ALPHA = ",F10.2,"  XMIN =",E10.2,"  XMAX =",
  1E10.2)',ALINE,0,K,A,XMIN,XMAX)
   CALL FCHSZ(1)
   CALL FTEXT(ALINE,K,1,1100,840)
   CALL CONVO('("BETA =  ",F10.2,"  YMIN =",E10.2,"  YMAX =",
  1E10.2)',BLINE,0,L,B,YMIN,YMAX)
   CALL FTEXT(BLINE,L,1,1100,800)
   CALL CONVO('("GAMMA = ",F10.2,"  ZMIN =",E10.2,"  ZMAX =",
  1E10.2)',CLINE,0,M,C,ZMIN,ZMAX)
   CALL FTEXT(CLINE,M,1,1100,760)
   RETURN
   END
```

APPENDIX D

ROUTINES FOR SYSTEM 360--CALCOMP 780 PLOTTER

SUBROUTINE PLOT 3D--VERSION 4

Purpose

Draw and rotate three-dimensional surfaces with IBM 360/50-75-Calcomp 780 Plotter. This version of PLOT 3D differs from Versions 1 and 2 in that "invisible" line segments are not drawn.

Usage

```
COMMON/MAXES/XMIN, XMAX, YMIN, YMAX, ZMIN, ZMAX
COMMON/LABEL/LAB (9)
CALL PLOT 3D (X, Y, Z, NX, NY, CUBE)
CALL ROTATE (ALPHA, BETA, GAMMA, DRAWME)
CALL NOMORE
```

Description of Parameters

XMIN - Less than or equal to the minimum value in array X

XMAX - Greater than or equal to the maximum value in array X

YMIN - Less than or equal to the minimum value in array Y

YMAX - Greater than or equal to the maximum value in array Y

ZMIN - Less than or equal to the minimum value in array Z

ZMAX - Greater than or equal to the maximum value in array Z

X - Two-dimensional array of X-coordinates

Y - Two-dimensional array of Y-coordinates

Z - Two-dimensional array of Z-coordinates

NX - Number of rows in arrays X, Y, Z, Limit 100

NY - Number of columns in arrays X, Y, Z, limit 100

CUBE - LOGICAL variable set to .TRUE. if the three directions are to be scaled independently and set to .FALSE. if the three directions are to be scaled dependently.

ALPHA - Angle of rotation about the X-axis, given in degrees

BETA - Angle of rotation about the Y-axis, given in degrees

GAMMA - Angle of rotation about the Z-axis, given in degrees

DRAWME - LOGICAL variable set to .TRUE. if the rotated view is to be drawn, and set to .FALSE. if this view is not to be drawn.

LAB - Array of BCD characters used to label the drawing. If label is desired, supply 9 or less four-character words in COMMON/LABEL/LAB (9).

Remarks

A call to PLOT 3D defines and scales the surface. Subsequent calls to ROTATE rotate it to new frames of references. The surface is not returned to its original position after a rotation. The surface is rotated about the fixed XYZ axes in this order: first about the Z-axis, then about the Y-axis, and finally, about the X-axis. Finally, a call to NOMORE closes the Calcomp tape unit.

Care should be exercised in providing that the arrays X, Y, Z in the calling program have dimensions (NX, NY). Dimensions different from this may cause errors in the plots.

This version of PLOT 3D requires much more computer time than Versions 1 and 2. During test runs a grid of 121 points required nearly 0.5 min. A grid of 500 points required 5 min of computer time.

In general, a line segment is drawn between two "invisible" endpoints and is not drawn between two "invisible" endpoints. If one endpoint is "visible" and the other endpoint is "invisible," the "visible" portion of the segment joining the points is drawn. No attempt is made to mask out an "invisible" part of a segment between two "visible" endpoints, or to draw a "visible" part of a segment between two "invisible" endpoints.

Subroutines Required

SCALE
TRNMAT
PHI
AXIS
SEE
SCAN
TORF
ONEVIS
PLANE
WRITE

In addition, PLOT 3D calls routines for the Calcomp 780 Plotter. They follow:

SYMBOL - Plots a string of alphanumeric characters

PLOTS - Allocates a work region to the PLOT routine

PLOT - Generates the increments necessary to move pen from current position to specified position

OFFSET - Enters factors to be used by the PLOT routine

Storage Requirements

This version of PLOT 3D requires about $F3E0_{16}$ bytes of storage, exclusive of Calcomp plotter routines and array storage.

A listing of the PLOT 3D Version 4 subroutines follows:

```
SUBROUTINE PLOT3D(SX,SY,SZ,NX,NY,CUBE)
LOGICAL CUBE
COMMON /MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
DIMENSION SX(NX,NY),SY(NX,NY),SZ(NX,NY)
COMMON/CALCMP/CCOMP(1000)
CALL PLOTS(CCOMP(1),4000)
CALL SCALE(SX,SY,SZ,NX,NY,CUBE)
CALL AXIS(0,.FALSE.)
ASUM=0.0
BSUM=0.0
CSUM=0.0
RETURN
ENTRY ROTATE(ALPHA,BETA,GAMMA,DRAWME)
LOGICAL DRAWME
CALL TRNMAT(ALPHA,BETA,GAMMA)
CALL PHI(SX,SY,SZ,NX,NY)
CALL AXIS(1,DRAWME)
ASUM=ASUM+ALPHA
BSUM=BSUM+BETA
CSUM=CSUM+GAMMA
IF(.NOT.DRAWME) RETURN
CALL PLOT(0.0,1.5,-3)
CALL SEE (SX,SY,SZ,NX,NY)
CALL WRITE(ASUM,BSUM,CSUM)
CALL PLOT(10.0,-1.5,-3)
RETURN
ENTRY NOMORE
CALL PLOT(0.0,0.0,999)
RETURN
END

BLOCK DATA
COMMON/LABEL/LAB(9)
DATA LAB/'
END
```

```
      SUBROUTINE SCALE(SX,SY,SZ,NXPTS,NYPTS,CUBE)
      DIMENSION SX(NXPTS,NYPTS),SY(NXPTS,NYPTS),SZ(NXPTS,NYPTS)
      COMMON/MAXES/XMIN ,XMAX,YMIN,YMAX,ZMIN,ZMAX
      LOGICAL CUBE
      REAL MAXX,MAXY,MAXZ
      MAXX=(XMAX-XMIN)/5.5
      MAXY=(YMAX-YMIN)/5.5
      MAXZ=(ZMAX-ZMIN)/5.5
      IF(CUBE) GO TO 21
      MAXX=AMAX1(MAXX,MAXY,MAXZ)
      MAXY=MAXX
      MAXZ=MAXX
   21 DO3 I=1,NXPTS
      DO3 J=1,NYPTS
      SX(I,J)=(SX(I,J)-(XMAX + XMIN)/2.)/MAXX
      SY(I,J)=(SY(I,J)-(YMAX + YMIN)/2.)/MAXY
    3 SZ(I,J)=(SZ(I,J)-(ZMAX + ZMIN)/2.)/MAXZ
      RETURN
      END

      SUBROUTINE  TRNMAT(ALPHA,BETA,GAMMA)
      COMMON/MATRIX/TMAT(3,3)
      A=ALPHA/57.2957795
      B=BETA/57.2957795
      C=GAMMA/57.295795
      SINA=SIN(A)
      SINB=SIN(B)
      SINC=SIN(C)
      COSA=COS(A)
      COSB=COS(B)
      COSC=COS(C)
      TMAT(1,1)=COSB*COSC
      TMAT(1,2)=-COSB*SINC
      TMAT(1,3)=SINB
      TMAT(2,1)=COSA*SINC+SINA*SINB*COSC
      TMAT(2,2)=COSA*COSC-SINA*SINB*SINC
      TMAT(2,3)=-SINA*COSB
      TMAT(3,1)=SINA*SINC-COSA*SINB*COSC
      TMAT(3,2)=SINA*COSC+COSA*SINB*SINC
      TMAT(3,3)=COSA*COSB
      RETURN
      END

      SUBROUTINE PHI(X,Y,Z,NX,NY)
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      COMMON/MATRIX/TMAT(3,3)
      DO1 I=1,NX
      DO1 J=1,NY
      XP=TMAT(1,1)*X(I,J)+TMAT(1,2)*Y(I,J)+TMAT(1,3)*Z(I,J)
      YP=TMAT(2,1)*X(I,J)+TMAT(2,2)*Y(I,J)+TMAT (2,3)*Z(I,J)
      ZP=TMAT(3,1)*X(I,J)+TMAT(3,2)*Y(I,J)+TMAT(3,3)*Z(I,J)
      X(I,J)=XP
      Y(I,J)=YP
    1 Z(I,J)=ZP
      RETURN
      END
```

```
      SUBROUTINE AXIS(I,DRAWME)
      COMMON/MATRIX/TMAT(3,3)
      COMMON/CALCMP/CCOMP(1000)
      LOGICAL DRAWME
      DIMENSION XA(2),XB(2),XC(2),YA(2),YB(2),YC(2),X(4)  ,Y(4)  ,Z(4)
      IF(I.NE.0) GO TO 1
      X(1)  =0.
      X(2)  =.4
      X(3)  =0.
      X(4)  =0.
      Y(1)  =0.
      Y(2)  =0.
      Y(3)  =.4
      Y(4)  =0.
      Z(1)  =0.
      Z(2)  =0.
      Z(3)  =0.
      Z(4)  =.4
      RETURN
    1 DO22 I=1,4
      XP=TMAT(1,1)*X(I)   + TMAT(1,2)*Y(I)  +TMAT(1,3)*Z(I)
      YP=TMAT(2,1)*X(I)  +TMAT(2,2)*Y(I)  +TMAT(2,3)*Z(I)
      ZP=TMAT(3,1)*X(I)  +TMAT(3,2)*Y(I)  +TMAT(3,3)*Z(I)
      X(I)  =XP
      Y(I)  =YP
      Z(I)  =ZP
   22 CONTINUE
      IF(DRAWME) GO TO 2
      RETURN
    2 XA(1)=Y(1)  +7.0
      XB(1)=XA(1)
      XC(1)=XA(1)
      XA(2)=Y(2)  +7.0
      XB(2)=Y(3)  +7.0
      XC(2)=Y(4)  +7.0
      YA(1)=Z(1)  +7.0
      YB(1)=YA(1)
      YC(1)=YA(1)
      YA(2)=Z(2)  +7.0
      YB(2)=Z(3)  +7.0
      YC(2)=Z(4)  +7.0
      CALL PLOT(XA(1),YA(1),3)
      CALL PLOT(XA(2),YA(2),2)
      CALL SYMBOL(XA(2),YA(2),0.10,'X',0.0,1)
      CALL PLOT(XB(1),YB(1),3)
      CALL PLOT(XB(2),YB(2),2)
      CALL SYMBOL(XB(2),YB(2),0.10,'Y',0.0,1)
      CALL PLOT(XC(1),YC(1),3)
      CALL PLOT(XC(2),YC(2),2)
      CALL SYMBOL(XC(2),YC(2),0.10,'Z',0.0,1)
      RETURN
      END
```

```
      SUBROUTINE SEE (X,Y,Z,NX,NY)
C  THIS ROUTINE CHECKS EACH OF THE POINTS IN THE XYZ ARRAYS FOR
C  VISIBILITY. IT JOINS VISIBLE  PAIRS WITH SEGMENTS , PASSES OVER
C  INVISIBLE PAIRS WITHOUT PLOTTING  AND  INTERPOLATES A VISIBLE SEGMENT
C  BETWEEN PAIRS ONLY ONE OF  WHICH IS VISIBLE.
C  SUBROUTINE REQUIRES THE USE OF SUBROUTINE SCAN, FUNCTION TORF
C  AND SUBROUTINE ONEVIS AND SUBROUTINE PLANE
C
      LOGICAL SEEUM(100,100),TORF,Q,ANSWER
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      COMMON/CALCMP/CCOMP(1000)
      DIMENSION TRIX(5),TRIY(5),TRIZ(5)
      DO2 I=1,NX
      DO2 J=1,NY
      CALL SCAN(X(I,J),Y(I,J),Z(I,J),ANSWER,X,Y,Z,NX,NY)
      SEEUM(I,J)=ANSWER
    2 CONTINUE
C  AT THIS POINT IN EXECUTION ALL POINTS HAVE BEEN CHECKED FOR VISIBILI-
C  TY.    SEEUM(I,J) CONTAINS 'TRUE' FOR VISIBLE POINTS AND 'FALSE' FOR
C  INVISIBLE POINTS
C
      CALL OFFSET(-5.,1.25,-5.,1.25)
C  THE PLOTTING ROUTINE FOLLOWS
C
   13 NYLONE=NY-1
      DO 14 I=1,NX
      DO 14 J=1,NYLONE
      IF(SEEUM(I,J))GO TO 401
      IF(SEEUM(I,J+1))GO TO 402
C
C  GO TO NEXT PAIR  IF BOTH POINTS INVISIBLE
C
      GO TO 14
  401 IF(SEEUM(I,J+1))GO TO 20
C
C  GO TO PLOT SEGMENT IF BOTH POINTS ARE VISIBLE
C
C
C  IF EXECUTION GETS HERE ONE POINT IS VISIBLE AND THE OTHER INVISIBLE
C
C  NOW INTERPOLATE BETWEEN THE POINTS TO FIND AN APPROXIMATE VISIBLE
C  ENDPOINT TO USE INSTEAD OF THE INVISIBLE POINT
C
      GO TO 15
  402 CONTINUE
      ISAVE=I
      JSAVE=J+1
      CALL ONEVIS(X(I,J+1),Y(I,J+1),Z(I,J+1),X(I,J),Y(I,J),Z(I,J),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
      GO TO 16
   15 ISAVE=I
      JSAVE=J
      CALL ONEVIS(X(I,J),Y(I,J),Z(I,J),X(I,J+1),Y(I,J+1),Z(I,J+1),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
   16 CALL PLOT(Y(ISAVE,JSAVE),Z(ISAVE,JSAVE),13)
```

```
      CALL PLOT(YPLOT,ZPLOT,12)
      GO TO 14
   20 CALL PLOT(Y(I,J),Z(I,J),13)
      CALL PLOT(Y(I,J+1),Z(I,J+1),12)
   14 CONTINUE
C
C  NOW PLOT SEGMENTS ALONG THE OTHER DIRECTION OF THE GRID.
C
      NXLONE= NX-1
      DO21 J=1,NY
      DO21 I=1,NXLONE
      IF(SEEUM(I,J))GO TO 403
      IF(SEEUM(I+1,J))GO TO 404
      GO TO 21
  403 IF(SEEUM(I+1,J))GO TO 26
      GO TO 22
  404 CONTINUE
      ISAVE=I+1
      JSAVE=J
      CALL ONEVIS(X(I+1,J),Y(I+1,J),Z(I+1,J),X(I,J),Y(I,J),Z(I,J),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
      GO TO 31
   22 ISAVE=I
      JSAVE=J
      CALL ONEVIS(X(I,J),Y(I,J),Z(I,J),X(I+1,J),Y(I+1,J),Z(I+1,J),YPLOT,
     1ZPLOT,X,Y,Z,NX,NY)
   31 CALL PLOT(Y(ISAVE,JSAVE),Z(ISAVE,JSAVE),13)
      CALL PLOT(YPLOT,ZPLOT,12)
      GO TO 21
   26 CALL PLOT(Y(I,J),Z(I,J),13)
      CALL PLOT(Y(I+1,J),Z(I+1,J),12)
   21 CONTINUE
C
C  THE FOLLOWING ROUTINE PLOTS A SEGMENT BETWWEN OPPOSITE VISIBLE
C  VERTICES OF A QUADRILATERAL IF AT LEAST ONE OF THE OTHER VERTICES
C  IS INVISIBLE AND BEHIND THE PLANE OF THE OTHER THREE.
C
      DO355 I=1,NXLONE
      DO355 J=1,NYLONE
      IF(.NOT.SEEUM(I,J))GO TO 361
      IF(.NOT.SEEUM(I+1,J+1))GO TO 361
      IF(.NOT.SEEUM(I,J+1))GO TO 360
  361 IF(.NOT.SEEUM(I,J))GO TO 363
      IF(.NOT.SEEUM(I+1,J+1))GO TO 363
      IF(.NOT.SEEUM(I+1,J))GO TO 362
  363 IF(.NOT.SEEUM(I+1,J))GO TO 365
      IF(.NOT.SEEUM(I,J+1))GO TO 365
      IF(.NOT.SEEUM(I,J))GO TO 364
  365 IF(.NOT.SEEUM(I+1,J))GO TO 355
      IF(.NOT.SEEUM(I,J+1))GO TO 355
      IF(.NOT.SEEUM(I+1,J+1))GO TO 366
      GO TO 355
  360 TRIX(1)=X(I,J)
      TRIY(1)=Y(I,J)
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIX(4)=X(I+1,J)
      TRIY(4)=Y(I+1,J)
      CALL PLOT(TRIX(1),TRIY(1),13)
      CALL PLOT(TRIX(2),TRIY(2),12)
      CALL PLOT(TRIX(3),TRIY(3),12)
      CALL PLOT(TRIX(4),TRIY(4),12)
      CALL PLOT(TRIX(1),TRIY(1),12)
  355 CONTINUE
C
C  THE FOLLOWING ROUTINE PLOTS A SEGMENT BETWWEN OPPOSITE VISIBLE
C  VERTICES OF A QUADRILATERAL IF AT LEAST ONE OF THE OTHER VERTICES
C  IS INVISIBLE AND BEHIND THE PLANE OF THE OTHER THREE.
C
      DO355 I=1,NXLONE
      DO355 J=1,NYLONE
      IF(.NOT.SEEUM(I,J))GO TO 361
      IF(.NOT.SEEUM(I+1,J+1))GO TO 361
      IF(.NOT.SEEUM(I,J+1))GO TO 360
  361 IF(.NOT.SEEUM(I,J))GO TO 363
      IF(.NOT.SEEUM(I+1,J+1))GO TO 363
      IF(.NOT.SEEUM(I+1,J))GO TO 362
  363 IF(.NOT.SEEUM(I+1,J))GO TO 365
      IF(.NOT.SEEUM(I,J+1))GO TO 365
      IF(.NOT.SEEUM(I,J))GO TO 364
  365 IF(.NOT.SEEUM(I+1,J))GO TO 355
      IF(.NOT.SEEUM(I,J+1))GO TO 355
      IF(.NOT.SEEUM(I+1,J+1))GO TO 366
      GO TO 355
  360 TRIX(1)=X(I,J)
      TRIY(1)=Y(I,J)
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIX(4)=X(I+1,J)
      TRIY(4)=Y(I+1,J)
      CALL PLOT(TRIX(1),TRIY(1),13)
      CALL PLOT(TRIX(2),TRIY(2),12)
      CALL PLOT(TRIX(3),TRIY(3),12)
      CALL PLOT(TRIX(4),TRIY(4),12)
      CALL PLOT(TRIX(1),TRIY(1),12)
  355 CONTINUE
```

```
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIZ(2)=Z(I,J+1)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIZ(3)=Z(I+1,J+1)
      CALL PLANE(TRIX,TRIY,TRIZ,X(I+1,J),Y(I+1,J),Z(I+1,J),ANSWER)
      IF(ANSWER)GO TO 356
      GO TO 361
  362 TRIX(1)=X(I,J)
      TRIY(1)=Y(I,J)
      TRIZ(1)=Z(I,J)
      TRIX(2)=X(I+1,J)
      TRIY(2)=Y(I+1,J)
      TRIZ(2)=Z(I+1,J)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIZ(3)=Z(I+1,J+1)
      CALL PLANE(TRIX,TRIY,TRIZ,X(I,J+1),Y(I,J+1),Z(I,J+1),ANSWER)
      IF(ANSWER)GO TO 356
      GO TO 363
  364 TRIX(1)=X(I+1,J)
      TRIY(1)=Y(I+1,J)
      TRIZ(1)=Z(I+1,J)
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIZ(2)=Z(I,J+1)
      TRIX(3)=X(I,J)
      TRIY(3)=Y(I,J)
      TRIZ(3)=Z(I,J)
      CALL PLANE(TRIX,TRIY,TRIZ,X(I+1,J+1),Y(I+1,J+1),Z(I+1,J+1),ANSWER)
      IF(ANSWER)GO TO 357
      GO TO 365
  366 TRIX(1)=X(I+1,J)
      TRIY(1)=Y(I+1,J)
      TRIZ(1)=Z(I+1,J)
      TRIX(2)=X(I,J+1)
      TRIY(2)=Y(I,J+1)
      TRIZ(2)=Z(I,J+1)
      TRIX(3)=X(I+1,J+1)
      TRIY(3)=Y(I+1,J+1)
      TRIZ(3)=Z(I+1,J+1)
      CALL PLANE(TRIX,TRIY,TRIZ,X(I,J),Y(I,J),Z(I,J),ANSWER)
      IF(ANSWER)GO TO 357
      GO TO 355
  356 CALL PLOT(Y(I,J),Z(I,J),13)
      CALL PLOT(Y(I+1,J+1),Z(I+1,J+1),12)
      GO TO 355
  357 CALL PLOT(Y(I+1,J),Z(I+1,J),13)
      CALL PLOT(Y(I,J+1),Z(I,J+1),12)
  355 CONTINUE
      CALL OFFSET(00.0,1.0,00.0,1.0)
      RETURN
      END
```

```
      SUBROUTINE SCAN(XP,YP,ZP,ANSWER,X,Y,Z,NX,NY)
C  THIS ROUTINE TAKES A POINT AND DETERMINES ITS VISIBILITY.  IF VISIBLE
C    1ANSWER IS FILLED WITH 'TRUE', OTHERWISE WITH 'FALSE'.
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      DIMENSION TRIX(5),TRIY(5),TRIZ(5)
      LOGICAL ANSWER,YES/.TRUE./,NOTYES/.FALSE./,Q,TORF
      K=1
      L=1
   10 LL=1
      DO3 M=1,2
      MM=L-1+M
      TRIX(M)=X(K,MM)
      TRIY(M)=Y(K,MM)
      TRIZ(M)=Z(K,MM)
      TRIX(M+3)=TRIX(M)
      TRIY(M+3)=TRIY(M)
    3 TRIZ(M+3)=TRIZ(M)
      TRIX(3)=X(K+1,L)
      TRIY(3)=Y(K+1,L)
      TRIZ(3)=Z(K+1,L)
    8 DO4 M=1,3
      IF(XP.NE.TRIX(M))GO TO 4
      IF(YP.NE.TRIY(M))GO TO 4
      IF(ZP.EQ.TRIZ(M))GO TO 6
C
C  IF THE ABOVE CONDITION HOLDS THE POINT IS A VERTEX OF THE TRIANGLE
    4 CONTINUE
      DO5 M=1,3
      Q=TORF(TRIY(M),TRIZ(M),TRIY(M+1), TRIZ(M+1),TRIY(M+2),TRIZ(M+2),
     1YP,ZP)
      IF(.NOT.Q)GO TO 6
    5 CONTINUE
C  IF EXECUTION GETS TO HERE THE POINT IS IN THE TRIANGLE ON THE YZ
C  PLANE.  NEXT CHECK FOR BEING IN FRONT OF THE TRIANGLE IN SPACE.
C
      IF(XP.LE.TRIX(1))GO TO 222
      IF(XP.LE.TRIX(2))GO TO 222
      IF(XP.GT.TRIX(3))GO TO 6
  222 CONTINUE
      IF(XP.GE.TRIX(1))GO TO 223
      IF(XP.GE.TRIX(2))GO TO 223
      IF(XP.LT.TRIX(3))GO TO 22
  223 CONTINUE
C  IF EXECUTION GETS TO HERE POINT IS NEAR TRIANGLE. NEXT CHECK FOR
C  BEING IN FRONT OR  BEHIND.
      CALL PLANE(TRIX,TRIY,TRIZ,XP,YP,ZP,ANSWER)
      IF(ANSWER) GO TO 6
C
C  IF EXECUTION GETS TO HERE THE POINT IS INVISIBLE
   22 ANSWER=NOTYES
      RETURN
C  FOLLOWING IS A ROUTINE TO PASS ON TO THE NEXT TRIANGLE
    6 GO TO (30,9),LL
   30 LL=2
      DO7 M=1,2
```

```
      MM=K-1+M
      TRIX(M)=X(MM,L+1)
      TRIY(M)=Y(MM,L+1)
      TRIZ(M)=Z(MM,L+1)
      TRIX(M+3)=TRIX(M)
      TRIY(M+3)=TRIY(M)
    7 TRIZ(M+3)=TRIZ(M)
      GO TO 8
C  SET K AND L  FOR NEXT PAIR OF TRIANGLES
    9 IF((L+1).GE.NY)GO TO 11
      L=L+1
      GO TO 10
   11 IF((K+1).GE.NX)GO TO 12
C  FINISHED WITH ALL TRIANGLES
      K=K+1
      L=1
      GO TO 10
C  THE POINT HAS BEEN CHECKED WITH THE LAST TRIANGLE, THE POINT IS
C  VISIBLE.
   12 ANSWER= YES
      RETURN
      END
```

```
      LOGICAL FUNCTION TORF(X1,Y1,X2,Y2,X3,Y3,XP,YP)
C  THIS ROUTINE SUPPLIES TRUE  IF POINT(XP,YP) AND (X3,Y3) SAME SIDE OF
C  LINE (X1,Y1),(X2,Y2). FALSE IF ON OPPOSITE SIDES
      TEE(A1,A2,A3)=(A3-A1)/(A2-A1)
      IF(X2.EQ.X1)GO TO 3
      IF(Y2.EQ.Y1)GO TO 5
      TX1=TEE(X1,X2,X3)
      TY1=TEE(Y1,Y2,Y3)
      TXP=TEE(X1,X2,XP)
      TYP=TEE(Y1,Y2,YP)
      IF(ABS(TYP-TXP).LE.1.0E-5)GO TO 4
      IF((TY1-TX1).LE.1.0E-5)GO TO 22
      IF((TYP-TXP).GT.1.0E-5)GO TO 2
   22 IF((TY1-TX1).GE.-1.0E-5)GO TO 4
      IF((TYP-TXP).LT.-1.0E-5)GO TO 2
    4 TORF=.FALSE.
      RETURN
    2 TORF=.TRUE.
      RETURN
    3 IF((XP-X1)*(X3-X1).LT.0.0)GO TO 4
      GO TO 2
    5 IF((YP-Y1)*(Y3-Y1).LT.0.0)GO TO 4
      GO TO 2
      END
```

```
      SUBROUTINE ONEVIS(XVIS,YVIS,ZVIS,XINVIS,YINVIS,ZINVIS,YPLOT,ZPLOT,
     1X,Y,Z,NX,NY)
C  THIS ROUTINE INTERPOLATES BETWEEN A VISIBLE POINT AND AN INVISIBLE
C  POINT TO FIND THE ENDPOINTS OF A VISIBLE SEGMENT.
      DIMENSION X(NX,NY),Y(NX,NY),Z(NX,NY)
      LOGICAL VIS
      DIST=SQRT((XVIS-XINVIS)**2+(YVIS-YINVIS)**2+(ZVIS-ZINVIS)**2)
      XV=XVIS
      YV=YVIS
      ZV=ZVIS
      XINV=XINVIS
      YINV=YINVIS
      ZINV=ZINVIS
      DO17 KK=1,7
      XMID=(XV  +XINV  )/2.0
      YMID=(YV  +YINV  )/2.0
      ZMID=(ZV  +ZINV  )/2.0
C  HERE CALL SCAN TO CHECK VISIBILITY OF MIDPOINT
      CALL SCAN(XMID,YMID,ZMID,VIS,X,Y,Z,NX,NY)
      IF(VIS)GO TO 19
      XINV  = XMID
      YINV  = YMID
      ZINV  = ZMID
      GO TO 18
   19 XV  = XMID
      YV  = YMID
      ZV  = ZMID
   18 IF((SQRT((XV-XINV)**2+(YV-YINV)**2+(ZV-ZINV)**2)).LE.(.01*DIST))
     1GO TO 22
   17 CONTINUE
   22 YPLOT=YMID
      ZPLOT=ZMID
      RETURN
      END
```

```
      SUBROUTINE PLANE(ARRAYX,ARRAYY,ARRAYZ        ,XP,YP,ZP,FRONT)
      LOGICAL FRONT
      DIMENSION ARRAY(3,3),SAVE(3),ARRAYX(3),ARRAYY(3),ARRAYZ(3)
      DO333 I=1,3
      ARRAY(I,1)=ARRAYX(I)
      ARRAY(I,2)=ARRAYY(I)
      ARRAY(I,3)=ARRAYZ(I)
  333 CONTINUE
      DO2 J=1,3
      ARRAY(2,J)=ARRAY(1,J)-ARRAY(2,J)
      ARRAY(3,J)=ARRAY(1,J)-ARRAY(3,J)
    2 CONTINUE
      IF(ARRAY(2,3).EQ.0.0)GO TO 3
    7 DO4 J=1,3
      ARRAY(3,J)=ARRAY(3,3)*ARRAY(2,J)-ARRAY(2,3)*ARRAY(3,J)
```

```
 4 CONTINUE
   B=-ARRAY(3,1)/ARRAY(3,2)
   C=-(ARRAY(2,1)+B*ARRAY(2,2))/ARRAY(2,3)
   D=-(ARRAY(1,1)+B*ARRAY(1,2)+C*ARRAY(1,3))
   XPLANE=-(B*YP+C*ZP+D)
   IF(XP.GT.(XPLANE+1.0E-5))GO TO 5
   FRONT=.FALSE.
   RETURN
 5 FRONT=.TRUE.
   RETURN
 3 DO6 J=1,3
   SAVE(J)=ARRAY(2,J)
   ARRAY(2,J)=ARRAY(3,J)
   ARRAY(3,J)=SAVE(J)
 6 CONTINUE
   GO TO 7
   END



   SUBROUTINE WRITE(A1,B1,C1)
   DIMENSION ALINE(15),BLINE(15),CLINE(15)
   COMMON/MAXES/XMIN,XMAX,YMIN,YMAX,ZMIN,ZMAX
   COMMON/LABEL/LAB(9)
   COMMON/CALCMP/CCOMP(1000)
   INTEGER ALINE,BLINE,CLINE,BLANK/'    '/
   SICK(XXX)=XXX-FLOAT(IFIX(XXX/360.0))*360.0
   A=SICK(A1)
   B=SICK(B1)
   C=SICK(C1)
   DO1 I=1,15
   ALINE(I)=BLANK
   BLINE(I)=BLANK
   CLINE(I)=BLANK
 1 CONTINUE
   CALL SYMBOL(0.0,-.3,0.2,LAB(1),0.0,36)
   CALL CONVO('("ALPHA = ",F10.2,"  XMIN =",E10.2,"  XMAX =",
  1E10.2)',ALINE,0,K,A,XMIN,XMAX)
   CALL SYMBOL(0.0,-.5,0.1,ALINE(1),0.0,K)
   CALL CONVO('("BETA =  ",F10.2,"  YMIN =",E10.2,"  YMAX =",
  1E10.2)',BLINE,0,L,B,YMIN,YMAX)
   CALL SYMBOL(0.0,-.7,0.1,BLINE(1),0.0,L)
   CALL CONVO('("GAMMA = ",F10.2,"  ZMIN =",E10.2,"  ZMAX =",
  1E10.2)',CLINE,0,M,C,ZMIN,ZMAX)
   CALL SYMBOL(0.0,-.9,0.1,CLINE(1),0.0,M)
   RETURN
   END
```

ACKNOWLEDGMENT

This report and the subroutine packages described therein are, in part, a revision of "PLOT 3D - A Package of FORTRAN Subprograms to Draw Three-Dimensional Surfaces," by R. Bruce Canright, Jr., and Paul Swigert of the Lewis Research Center, Cleveland, Ohio, NASA TM-X-1598. This publication is available through the National Technical Information Service, Springfield, Virginia 22151.

BIBLIOGRAPHY

1. R. Bruce Canwright, Jr. and Paul Swigert, *Plot 3D - A Package of FORTRAN Sub-programs to Draw Three-dimensional Surfaces*, NASA TM-X-1598, Clearinghouse for Federal Scientific and Technical Information, Springfield, Virginia.

2. Ruth A. Weiss, *BE VISION, A Package of IBM 7090 FORTRAN Programs to Draw Orthographic Views of Combinations of Plane and Quatric Surfaces*, J. Assoc. Comp. Mach., *13*, 194-204 (April 1966).

3. Henry Margenau and George M. Murphy, *The Mathematics of Physics and Chemistry*, Second Ed., D. Van Nostrand Co., Inc., (1956) 565-570.

4. D. Carson, *A Film-plotting Subroutine Package (FSP) for the IBM 2280 Film Recorder*, Technical Memorandum No. 167, Argonne National Laboratory.

5. Ronald F. Krupp, *S/360 Programming Techniques for the Calcomp 780*, Technical Memorandum No. 130, Argonne National Laboratory.

ARGONNE NATIONAL LAB WEST
3 4444 00008226 3

This material goes in the back of ANL-76-33

Cont. Do not Foot

Na Fires → 7 mr/hr at 820 ft from site stack
total 22 mrem (3.2 Rem)
these are less than DOE OSER limits

Cover Gas Rel. One subass of noble gases
0.8 s 2 h 8 hr
1.15 mR 1.47 R 1.78 R at 820 ft from stack
(< 10 CFR100 of 25 Rem W.B)

Earthquake - Closure prevents release
- activity release not identified

Spent fuel Handle One subass - 1 MW
100% noble gas, 50% of Iodine
at 8 hr from shutdown (25% for light water reactors)
No iodine:
0.8 s 2 h 8 h
146 mrem 431 Rem 519 Rem } typical
at 820 ft from site stack
at 5000 m = 43 mr 130 Rem 155 Vs 300 allow

∴ Quoting at 5000 ft would reduce limits considerably, as well as reducing the Iodine to 10%

Calc values at 5000 m
ratio ~ $1.5\times10^{-5} / 5\times10^{-5}$ = 0.3 see above

Design Basis release
100% noble gases, 25% halogens,
~~1% solid fission products~~, Not released to envir.
1% plutonium

Containment isolates for this one
2 hr at 600 meter — 30 d at 5 km
100 Rem (Thyroid) — 17 Rem (Thyroid)
(Thyroid = 300 Rem limit)
Whole body dose not limit.

2 hr at 5000 m — 30 d at 5 km
3.8 Rem — 17 Rem Thyroid

Did not analyze non ~~leak~~ closure case.

If iodine reduced to 1%
assuming direct reduction of dose,
the ~~new~~ 1% doses would be

Containment isolates
2 hr at 600 m: — 30 d at 5000 m:
~4 rem — ~700 m Rem
1/25 X
Not worthwhile to isolate
2 hr at 5000 m:
150 m Rem

to estimate OBA if it does not isolate, use simple
outrageously cases
1 MW x 62.5 → reactor x $\frac{1}{50}$ for 1% Iodine →

→ 5000 ft:	0.8 s	2 hr	50 hr	Thyroid doses
	180 m Rem ✓	540 Rem ✓	650 Rem ✓	
→ 5000 ft m	55 Rem (See 0.3 factor calc. on page 1)	160 Rem	195 Rem	: OK - below 10 CFR 100

If: reduce by 1% on iodine/
→ Not: If held at 25% or 50%

ANALYSIS

CONDITIONS

App. A: Sodium Fire — No isolation, Elevated release → Very low dose at pt of max. conc.

$\chi/Q = 5\times10^{-5}$ now (200 ft

$\chi \cong 1.5\times10^{-4}$ → result OK, but needs re-evaluation

above assumes no isolation. With isolation, the ~~χ/Q~~ shortened stack would have no impact

---

Acc. from Cover Gas release — No isol. Elev. release } moderate doses, well within accident limits

But Rad monitors would isolate, so dose → 1.15 mRem

DBA — Isolation RQD: dose OK, ~~although~~ ~~also~~ assumes ground level release with Bldg isolated

---

fire — Isolates & leaks → ground level release

Earthquake → No Act. release

4×10^5 SFT^3 (of Air)

$4 \times 10^5 \times .21 = 8.29 \times 10^4$ FT^3 of O_2

$\frac{8.29 \times 10^4}{359} = 231$ lb-moles of O_2

$2Na + \frac{1}{2} O_2 \rightarrow Na_2O$

ratio of $\frac{Na}{O_2} = \frac{2}{\frac{1}{2}} = \frac{4}{1}$

231 moles of $O_2 \times \frac{4}{1} = 924$ moles of Na

$924 \times 23 = 21250$ lbs (approx.) for EBR-II of Na

ARGONNE
NATIONAL
LABORATORY

INTRA-LABORATORY MEMO

J. I. SACKETT

MAR 17 1990

COPIES SENT TO:

D. W. Cissel
S. Fryer
~~M. R. Lenio~~
R. L. Parks
~~W. [illegible]~~
~~H. P. Planchon~~
~~B. N. Smith~~
V. N. Thompson
L. C. Walters
R. KING
Action:
File:

Pete
Could we use this in our work modified to establish a EBR II source term for containment testing?
Jim

March 1, 1990

To: Y. I. Chang, J. Marchaterre, L. Walters
J. Sackett, D. Wade, J. Battles

From: D. R. Pedersen DRP RAS

Subject: Notes from GE Meeting with NRC on Containment, February 27, 1990

1. GE still has GEMS in their reference ALMR design.

2. GE attempted to stress the mitigation capability of the PRISM design.

3. J. Wilson (NRC), Ed Throm (NRC) and G. Van Tuyle (BNL) attended the meeting. Wilson and Throm are the replacements for T. King and R. Landry.

4. NRC may push for further definition of the IFR Phase III program plan.

5. Figures 1 and 2 represent GE's new containment design.

6. GE has developed a proposed source term (containment evaluation source term) for the ALMR

   100% Nobles
   0.1% Iodine, Bromine
   0.1% Cesium Rubedium
   0.01% other (fuel)
   1700# Na

   a. The amount of Na is equivalent to the amount of oxygen available such that all the Na burns with no remaining oxygen.

   b. This source term is the source available in upper containment for the purposes of containment evaluation.

7. The sodium in the source term is assumed to remain within the reactor vessel and burn as a pool fire. The sodium never is allowed to burn as a spray fire. This is a major weakness. The failure that allows the breach in the head is likely also to eject Na and cause a sodium spray fire.

8. GE intends to use the CONTAIN code for the containment evaluation. Sandia will be releasing a new LMR version of CONTAIN LMR in about a month.

9. Ed Throm and G. Van Tuyle have requested copies of the Metal Fuel Handbook.

   Edward D. Throm
   USNRC
   Mail Stop NL/S-314
   Washington, DC 20555

10. T. King attended the meeting for a while. Mentioned conditional probability of containment failure as something NRC, and ACRS is considering.

11. Figure 3 represents an outline for the proposed GE amendment. GE/DOE would like the NRC to prepare a final Safety Evaluation Report by September 1990. Reason given (to support FY92 budget cycle).

12. Wilson stated that the anticipated date for issuance of revised final SER is November 1990 including commissioner approval.

13. Wilson indicates it may be possible to send letter report, Beckjord (NRC) to Young (GE) to meet the earlier requested date of September 1990 by DOE. Bill Morris (J. Wilson's boss) came into the meeting at 2:00 p.m. A brief presentation was made on containment mitigation of the ALMR. (Morris feels that unless the letter has ACRS review it may be hollow. Morris, Throm and Wilson will be evaluating schedule for the review and the mode of NRC response after receipt of the amendment.)

14. Morris asked about the time scale for ANL to analyze the energetics with metal fuel (answer: FY95); What code will you use (answer: SAS4A); Do you anticipate use of SIMMER (answer: no).

15. Wilson stated that the revised key issues document, SECY 203, should be sent to the commissioners in late spring.

16. Future anticipated meetings:

Technical Review (including GEMS)	Hanford	March 20-22, 1990
Containment Evaluation	Rockville	April __, 1990

mm

cc: RAS Section Managers
A. J. Goldman
L. W. Deitrich
H. P. Planchon
R. W. Seidensticker
RAS Files: 47403-34, A15

Containment Boundary During Operation

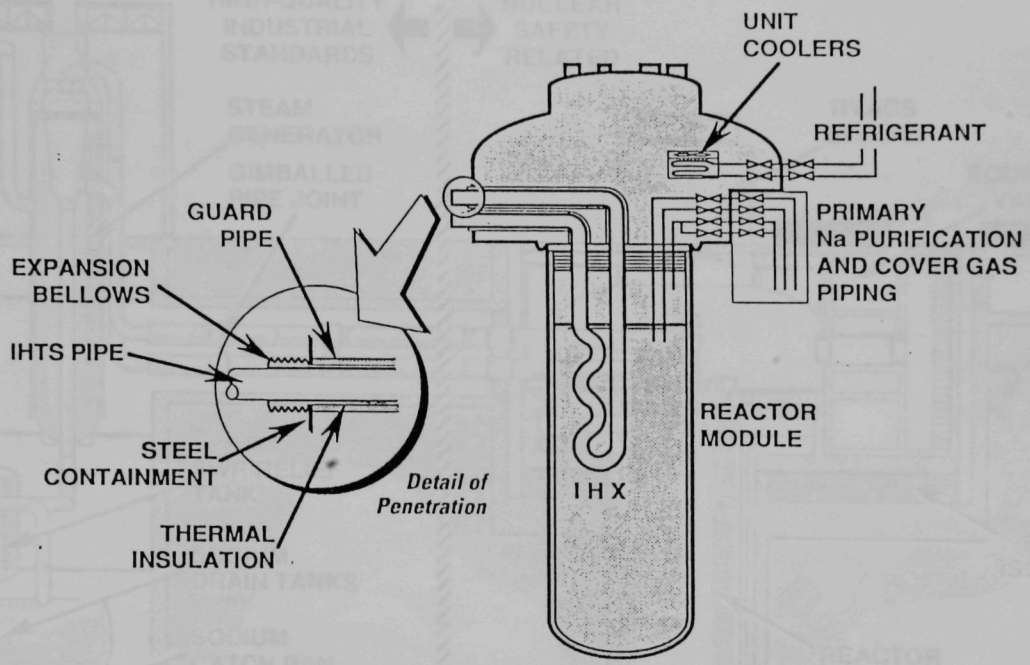


Figure 1

90-072-02

Reactor Steam Supply System

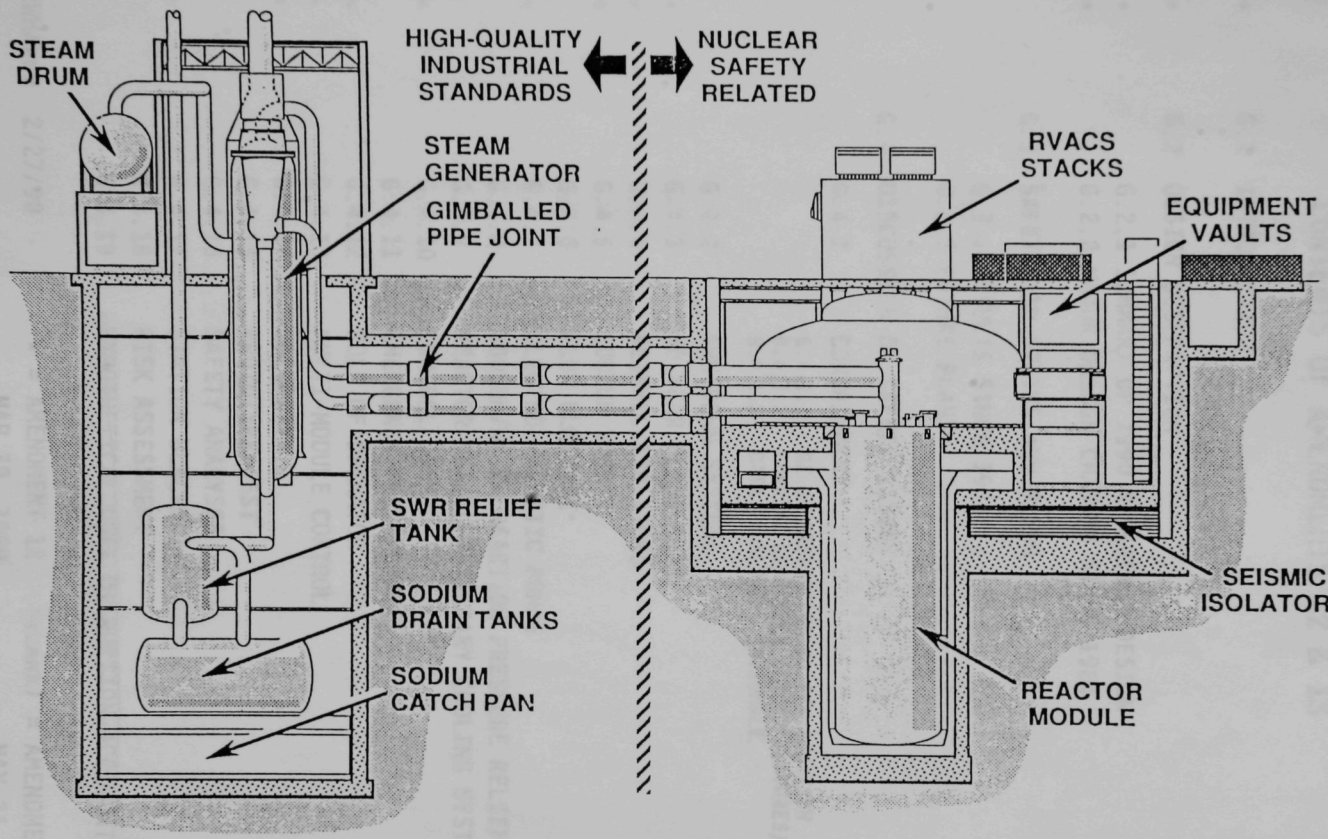


Figure 2

90-063-01

Figure 3

OUTLINE OF APPENDIX G
CONTENTS OF AMENDMENTS 12 & 13

* G.1 INTRODUCTION

* G.2 DESIGN DESCRIPTION

* G.2.1 SUMMARY OF 1990 REFERENCE DESIGN

* G.2.2 MAJOR DESIGN CHANGES SINCE 1986

G.3 SAFETY R&D STATUS AND PLANS

G.3.1 RESULTS SINCE 1986

G.3.2 FUTURE PLANS

G.4 DISCUSSION OF SAFETY ISSUES

G.4.1 CONTAINMENT

G.4.1.1 GE UNDERSTANDING OF SER POSITION
G.4.1.2 CURRENT REFERENCE DESIGN FEATURES/APPROACH
G.4.1.3 DATA, ANALYSES, RATIONALE

G.4.2 SHUTDOWN SYSTEMS

* G.4.3 60 YEAR PLANT LIFE

* G.4.4 SEISMIC ISOLATORS

* G.4.5 SODIUM VOID

G.4.6 FLOW BLOCKAGE

* G.4.7 ELECTRO-MAGNETIC PUMPS

G.4.8 SODIUM/WATER REACTION PRESSURE RELIEF SYSTEM

G.4.9 REACTOR VESSEL AUXILIARY COOLING SYSTEM

* G.4.10 CONTROL ROOM

G.4.11 EMERGENCY PREPAREDNESS

* G.4.12 ROLE OF OPERATOR

* G.4.13 MULTI-MODULE CONTROL

* G.4.14 SECURITY

G.4.15 PROTOTYPE TEST

G.4.16 SAFETY ANALYSES

* G.4.17 STATION BLACKOUT

G.4.18 RISK ASSESSMENT

G.4.19 ~~HYPOTHETICAL CORE DISRUPTION ACCIDENT~~
MITIGATION CAPABILITY

RWHardy 2/27/90

* = AMENDMENT 12 MAR 30, 1990

(BLANK) = AMENDMENT 13 MAY 31, 1990

Fig 3