A Mini-Computer Based General Purpose Interactive Computer Graphics Software System

1981

Victor M. Haddock
University of Central Florida
A MINI-COMPUTER BASED GENERAL PURPOSE INTERACTIVE COMPUTER GRAPHICS SOFTWARE SYSTEM

BY

VICTOR M. HADDOCK, JR.
B.S.E.E., University of Puerto Rico, 1976

RESEARCH REPORT

Submitted in partial fulfillment of the requirements for the Master of Science degree in the Graduate Studies Program of the College of Engineering University of Central Florida Orlando, Florida

Fall Term
1981
ABSTRACT

This research paper presents the description of a Mini-Computer Based General Purpose Interactive Computer Graphics Software System; which has been installed at the University of Central Florida, College of Engineering PDP 11/34 computer system.

This system is a simplified version of MOVIE.BYU. MOVIE.BYU is a General Purpose Interactive Computer Graphics Software System that requires a computer with a word length of at least 32-bits. The MOVIE.BYU system was developed at the Brigham Young University, Utah.

Extensive modification was required in order to satisfy the PDP 11/34 FORTRAN IV compiler. In addition, the program routines have been overlayed in order to accommodate the programs in the PDP 11/34 memory.

The system has been exercised and hard-copy of the results are presented in order to demonstrate its capabilities.
# TABLE OF CONTENTS

LIST OF FIGURES ........................................ v

Chapter

I. INTRODUCTION ........................................... 1
   Background ................................................. 1
   Objectives of this Project ................................. 4
   General Description of the Computer Graphics System .... 5

II. DATA MANAGEMENT ....................................... 9
   Data Structure ............................................. 9
   Data Generation ........................................... 12

III. SCENE SELECTION AND MANIPULATION .................. 13
   Coordinates Systems ..................................... 13
   Rigid Body Motion ....................................... 15
   Perspective ............................................... 23

IV. DISPLAY GENERATION ................................... 30
   Hidden Lines and Surfaces ............................... 32
   Window of View Clipping ................................ 37
   Drawing the Edges on the TEKTRONIX .................... 49

V. UTILITY PROGRAM COMMANDS AND CAPABILITIES ........ 51

VI. MOVIE PROGRAM COMMANDS AND CAPABILITIES ........... 63
   General Commands ........................................ 64
   Scene Manipulation Commands ............................. 67
   Animation Commands ...................................... 70

VII. RESULTS AND CONCLUSION ............................. 73

iii
TABLE OF CONTENTS (continued)

Results .................................................. 73
Conclusion ................................................. 74

APPENDIX A: PROGRAMS HIGH-LEVEL FLOW CHARTS ........ 78
APPENDIX B: PROGRAMS LISTINGS .......................... 130
APPENDIX C: PROGRAMS OVERLAY STRUCTURES .......... 279
APPENDIX D: COMPUTER GRAPHICS PACKAGE USER MANUAL . 282
APPENDIX E: RESULTS ..................................... 331
LITERATURE CITED ......................................... 402
BIBLIOGRAPHY ............................................... 403
LIST OF FIGURES

1. Geometry File Data Structure .................. 11
2. MOVIE Coordinate Systems and Pyramid of Vision .................. 16
3. Translation of the Origin .................. 18
4. Rotation of a Point About the X-Axis .......... 18
5. Rotation About the Y-Axis .................. 20
6. Rotation About the Z-Axis .................. 20
8. MOVIE Display Generation Procedure .......... 31
10. TEKTRONIX 4051 Screen Layout .................. 39
11. Similar Triangles Used for Clipping Against the Left Edge .................. 42
12. Similar Triangles Used for Clipping Against the Right Edge .................. 44
13. Similar Triangles Used for Clipping Against the Bottom Edge .................. 46
14. Similar Triangles Used for Clipping Against the Top Edge .................. 48
15 Thru 29. Program UTILITY Results , Appendix E 332
30 Thru 45. Program MOVIE Results , Appendix E ...... 363
Like many of the new technologies of recent years, computer graphics were first developed for U. S. military applications. In the mid and late fifties, the SAGE (Semi-Automatic Ground Environment) air defense system used a display system to give visual indication of the position of aircraft (Pollack 1980).

During the 1950's, little progress was made in the field of interactive computer graphics, simply because the computers of that period were not suitable for interactive use. These computers basically were "number crunchers" used by physicists and missile designers to perform lengthy calculations (Newman and Sproull 1979).

In the early 1960's, Ivan Sutherland, in a federally funded project at the Massachusetts Institute of Technology (MIT) Lincoln Labs showed how a computer could be used for interactive design of line drawings using a simple cathode-ray-tube (CRT) display and a few auxiliary controls. This project, which became his Ph.D. thesis,
was entitled **Sketchpad: A Man Machine Graphical Communication System.** His brilliant work proved to many that interactive computer graphics was a viable, useful and exciting field of research (Pollack 1980).

By the mid 1960's, large computer graphics research projects were underway at MIT, General Motors, Bell Telephone Laboratories and Lockheed Aircraft, the golden age of computer graphics has begun (Newman and Sproull 1979).

Computer graphics at this time required the largest and most expensive computers, and only universities, the government and large industries, like the ones mentioned above, could afford the high price tag. Several computed-aided design (CAD) systems for automobile, aircraft or missile design came about independently during the 1960's. These systems allowed the engineer to alter design parameters and then see the changes in design displayed on the screen. Data on how the new parameters would affect other areas, such as cost, strength, and aerodynamics, were also immediately obtained (Pollack 1980).

Fortunately, as it has many other technological innovations, time favored computer graphics. The cost of computer and display equipment kept dropping year after
year, while improving performance at the same time. Computer operating systems were improved, and the ability to cope with the complex software necessary for computer graphics became more sophisticated. Impressive progress was made in the development of algorithms for generating pictures, especially those intended to represent views of three dimensional objects. The progress, though slow was sufficient so that at the end of the 1970's, computer graphics finally became accepted as an effective powerful, and economical sound tool of the engineer, scientist, designer, manager, illustrator, and even the artist.

Microprocessor technology and the drastic price reduction of computer memory have reduced the cost of computer graphics equipment up to the point that it has made its way into many american homes, for basically the same cost of a sophisticated stereo system. Video games represents the first major use of computer graphics in the home (Newman and Sproull 1979).

Today interactive computer graphics affect our lives in a number of direct and indirect ways. Pilots spend much of the training time in a flight simulator rather than in a real aircraft. The flight simulator is a mockup of the aircraft cockpit and computer graphics
provides an out-the-window display of what the pilot will see in a real flight. Computer graphics technology in this way has enhance the realism of training for all kinds of pilots. Teachers will have available computer graphics systems to help them to present abstract ideas and concepts. Computer graphics helps doctors to provide better health care. With new body scanners, medical investigators are using computer graphics to view complex parts inside the human body. These are only a few example of how interactive computer graphics is affecting our life. It is clear that the field of computer graphics will keep helping us to improve our life style.

Objectives of this Project

The objective of this project is to develop a General Purpose Interactive Mini-Computer Based Computer Graphics System.

This system has been developed by modifying MOVIE.BYU which is a general purpose computer graphics software system that requires a computer with a word length of at least 32 bits. MOVIE.BYU was developed at the Brigham Young University and it has been distributed since 1976. The computer graphics system runs on the University
of Central Florida, College of Engineering Digital Equipment Corporation PDP 11/34 under the RT-11 operating system. The graphics output terminal device used is a TEKTRONIX 4051 with the data communication option.

Extensive modification was required to the source programs to satisfy the PDP 11/34 FORTRAN IV compiler. In addition, the system routines have been overlayed, in order to accommodate the programs in the available 28K words PDP 11/34 memory. This overlay structure is described in Appendix B.

General Description of the Computer Graphics System

This Mini-Computer Based System differs considerably from MOVIE.BYU. Even though the routines are overlayed, still there is a memory limitation in order to implement the Watkin's Hidden Line/Surfaces algorithm. The Watkin's algorithm provides the capability to remove from display line/surfaces hidden by other objects. Only the poor's man algorithm has been implemented. The poor's man algorithm only provides the capability to remove from display objects line/surfaces hidden by other surfaces of the object itself. If two or more objects are to be displayed, lines or surfaces from an object hidden by
another object, will not be removed.

In addition the only graphic output device available was the TEKTRONIX 4051, which is a line drawing device. Therefore the system as presently configured is not be able to generate color models in a continuous tone device.

The computer graphics system contains two programs, UTILITY which is used to generate or modify data files that describe the models, and MOVIE that is used to manipulate and display the models generated by UTILITY.

This section describe in general the capabilities of these two programs. Chapters V and VI presents a detailed description of the programs capabilities. In addition a user manual is included in APPENDIX D.

UTILITY is a data generation and/or editing program which allows the user to produce and/or edit models of two or three dimension. The data structures used by UTILITY to describe the models is presented in Chapter II. Two files are used to store models data. One file contains the data that describes the geometry of the model, and the other one contains displacement data for the model nodes.

The program has model generation capabilities based upon three generalized primitives: hexahedrons,
ellipsoids and volume of revolution. The special and general cases of these primitives allows a wide variety of models to be easily created. The program also has the capability of creating symmetrical additions to any already described model. Old files could be read for the purpose of modification and/or to merge two or more models data files into one file.

UTILITY is controlled by four-letter commands entered by the user from the key board. Once a command is specified the program will request from the user any information required to complete the command.

MOVIE is the program for display and manipulation of the models. MOVIE is also controlled by four-letter commands. The user may interact with MOVIE through twenty five (25) commands. Amongs the commands there are commands to specify the files where the models data is stored, commands to perform translations and rotations of the models in any direction and commands to display the model with or without hidden-lines removed.

One of the most important features of MOVIE is that its commands could be applied to the entire model or only to parts which are subsets of the model. This capability give a great amount of flexibilty for the manipulation of the model.
Another very useful feature of the MOVIE program is the animation command. The animation command allows the user to specify a sequence of displays to be executed. The user specifies parameters like total number of frames to be displayed, total translation of the model, total rotations in all directions and the scheme to be used to perform the parameters changes that will define each frame. This animation capability is what ease the generation of movies. For example an animation sequence could be specified, and then a movie could be produced unattended. Every time a frame is completed the program will ring the terminal bell. The bell will then signal a camera to take that frame picture. At the end of the animation sequence the "movie" has been produced.
II. DATA MANAGEMENT

Data Structure

The data in which the UTILITY and MOVIE programs operates resides in two files. These are the geometry file and the displacement file. The geometry file contains the information required to describe the geometry of the model, this include the control variables, the parts array, the coordinate array and the connectivity array. The control variable give the number of parts on the model, the number of coordinates (nodes) and the number of elements in the model.

The coordinates array contains the cartesian coordinates of the nodes stored sequentially by node number. The model geometry is described by polygon elements which vertices are the nodes defined in the coordinate array. The connectivity array is used to define each polygon element nodes. The elements are stored in the connectivity array sequentially by element number.

The system allows the user to define subsets of the
models by grouping elements into parts. This subsets are defined in the parts array. The parts array is a two dimensional array that contains the upper and lower number of the continuous elements that form the parts.

The parts array allows the user to group elements into parts by function, material or any other criteria. An important advantage of the models defined by parts is that the parts can be manipulated independently by the MOVIE program.

Figure 1 contains a pictorial representation of the geometry file data structure.

The displacement file is used to define desired displacement of the model nodes. The displacement data is stored in an array like the coordinate array.

The system has the capability to handle a model described by a maximum of 250 nodes, a maximum of 250 polygons elements and a maximum of 20 parts.
CONTROL VARIABLES

<table>
<thead>
<tr>
<th>NUMBER OF PARTS (NP)</th>
<th>NUMBER OF NODES (NJ)</th>
<th>NUMBER OF ELEMENTS (NPT)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

COORDINATE ARRAY

<table>
<thead>
<tr>
<th>NODE #</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CONNECTIVITY ARRAY

<table>
<thead>
<tr>
<th>ELEMENT NO.</th>
<th>NODE NO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

PARTS ARRAY

<table>
<thead>
<tr>
<th>PART NO.</th>
<th>LOWER ELEMENT</th>
<th>UPPER ELEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1. Geometry File Data Structure
Data Generation

UTILITY provides an interactive capability for entering model geometry and displacement information. When this feature is used, properly formatted files are automatically generated for later used by MOVIE. Any other program could be used to generate the data provided that the resulting file would have the same structures as the files produced by UTILITY.

Chapter V presents a complete description of the commands available in program UTILITY to generate the model geometry and displacement files.
III. SCENE SELECTION AND MANIPULATION

Coordinate Systems

Before discussing the mathematics of scene manipulation, it is necessary to define the coordinate systems used by the system.

Five coordinate systems are used. The first is fixed at the eye of the observer with the x-axis horizontal and the y-axis vertical. The z-axis points away from the observer toward the display screen. This is the eye coordinate system. It is a left handed system. Coordinates in this system are denoted by a subscript e.

The second coordinate system is the screen coordinate system. This system has its origin at the center of the display device with the x-axis horizontal and the y-axis vertical. The minimum and maximum values allowable in x and y are minus one (-1) and one (1). This is a two dimensional system. Coordinates in this system are denoted by a subscript s.

The third system is the general coordinate system.
This system is right handed. The x-axis is horizontal, the y-axis vertical, and the z-axis points towards the observer along the z-axis of the eye coordinate system. The general coordinates will not be subscribed.

The fourth coordinate system is the global coordinate system, which also is a right handed system. All the models are defined on the global coordinate system. When no rigid body movements has been made on the model the global and general coordinate systems will coincide. When a rotation or translation is performed on the model the coordinates are transformed from the global to the general coordinate system. Then in order to display the model, a transformation is made to the eye coordinate system, then perspective transformation is applied to generate the display coordinates. The global coordinate system is always parallel to the general coordinate system and it will be subscribed using the letter g.

The fifth coordinate system used is the local coordinate system. Each part of the model has a local coordinate system. The local coordinate system is a right handed system and it is parallel to the general coordinate system. The local coordinate system will be subscribed with the letter l. The origin location for
the local coordinate system of each part is entered interactively during execution of program MOVIE. The local coordinate system is the system used to perform local parts rotations.

Figure 2 shows the relation between the five coordinate system used. The global and local coordinate are shown as if some rigid body movement have been performed.

**Rigid Body Motion**

The two rigid body movements available to the user are translation of the model origin and rotation of the model about one of the coordinate axes.

To translate the origin of the model, the coordinates of the new origin are added to the coordinates of all points as Figure 3 shows. This translation may be written as:

\[
\begin{align*}
x' &= x + tx \\
y' &= y + ty \\
z' &= z + tz
\end{align*}
\]
Figure 2. MOVIE Coordinate System and Pyramid of Vision
In the above equations, \( tx, ty, tz \) are the \( x, y, z \) coordinates of the old origin in the new system and the prime indicates the translated coordinates. This operation may also be written in matrix form as:

\[
\begin{align*}
\hat{X}' &= \hat{X} + \hat{T} \\
X' &= \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \\
X &= \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\
T &= \begin{bmatrix} tx \\ ty \\ tz \end{bmatrix}
\end{align*}
\]

where \( X' \) is the matrix of the translated coordinates,

\( x' \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

\( x \) is the original coordinates matrix,

When a point is rotated about an axis, the transformation is dependent upon the axis of rotation and the angle. For example, a rotation about the \( x \)-axis through angle \( \theta \), as illustrated in figure 4 will result
Figure 3. Translation of the Origin

\[ \hat{X}' = \hat{X} + \hat{T} \]

Figure 4. Rotation of a Point About the X-Axis
in the following relationships.

\[ x' = x \]
\[ y' = r \cdot \cos (A + B) \]
\[ z' = r \cdot \sin (A + B) \]

The second two equation may also be written:

\[ y' = r \cdot \cos A \cdot \cos B - r \cdot \sin A \cdot \sin B \]
\[ z' = r \cdot \sin A \cdot \cos B + r \cdot \cos A \cdot \sin B \]

Realizing that:

\[ y = r \cdot \cos A \]
\[ z = r \cdot \sin A \]

The following substitutions may be made:

\[ x' = x \]
\[ y' = y \cdot \cos B - z \cdot \sin B \]
\[ z' = y \cdot \sin B + z \cdot \cos B \]

These equations have matrix equivalent given below:

\[ X' = R_x \cdot X \]

where:

\[
R_x = \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos B & -\sin B \\
0 & \sin B & \cos B
\end{bmatrix}
\]

Similar rotation matrices may be derived for rotations about the y and z axes. Figures 5 and 6 illustrate these rotations through angle B. From Figure 5 the transformation caused by a rotation about the y-axis
Figure 5. Rotation About the Y-Axis

Figure 6. Rotation About the Z-Axis
is:

\[
\begin{align*}
    x' &= r \cdot \cos (A - B) \\
    y' &= y \\
    z' &= r \cdot \sin (A - B)
\end{align*}
\]

After expanding the equations as before and substituting \( x \) and \( z \) for \( r \cdot \cos A \) and \( r \cdot \sin A \), the following \( y \) rotation matrix results.

\[
\begin{bmatrix}
    \cos B & 0 & \sin B \\
    0 & 1 & 0 \\
    -\sin B & 0 & \cos B
\end{bmatrix}
\]

From Figure 6 the transformations are:

\[
\begin{align*}
    x' &= r \cdot \cos (A + B) \\
    y' &= r \cdot \sin (A + B) \\
    z' &= z
\end{align*}
\]

Again, these equations may be reduced to a \( z \) rotation matrix given below:

\[
\begin{bmatrix}
    \cos B & -\sin B & 0 \\
    \sin B & \cos B & 0 \\
    0 & 0 & 1
\end{bmatrix}
\]

These rotation matrices may be multiplied together
to form a single 3 by 3 matrix that defines several combined rotations. For example, a rotation about x, one about y, another about x and one about z may be written as:

\[
\begin{align*}
X_i &= R_{x} \cdot X \\
X_{ii} &= R_{y} \cdot X_i \\
X_{iii} &= R_{x} \cdot X_{ii} \\
X_{iv} &= R_{z} \cdot X_{iii}
\end{align*}
\]

Combining these four equations by substituting the first in the second, the second in the third, etc. results in the following equation:

\[
X_{iv} = R_{z} \cdot R_{x} \cdot R_{y} \cdot R_{x} \cdot X
\]

This may then be written as:

\[
X_{iv} = R \cdot x
\]

where:

\[
R = R_{z} \cdot R_{x} \cdot R_{y} \cdot R_{x}.
\]

Before continuing, the point should be made that the sequence in which rotations are taken is important. A rotation about x followed by one about y is not the same as rotations about y and then about x.

The transformation described above may be used globally to translate or rotate the entire model or locally to translate or rotate only one or more parts selected by the user. The local translation and rotation feature allow the user to fully use a parts array that
0 uses multiple definitions of a single part.

**Perspective**

After the model has been transformed on the general coordinate system it is necessary to prepare it for display. This is to transform the model from a three dimensional description to a two dimensional description.

The pyramid of vision in Figure 7 is defined by the angle between the top and bottom or between left and right surfaces. The screen coordinate system is placed along the z-axis of the eye coordinates so that the minimum and maximum values where the axes pierce the pyramid of vision are minus one (-1) and plus one (+1). This distance (ds), as seen in Figure 7 along the y-z plane through the pyramid is:

\[ ds = \frac{1}{\tan \left( \frac{\theta}{2} \right)} = \cot \left( \frac{\theta}{2} \right) \]

The distance from the eye coordinates to the general coordinates is the distance d. This value is arbitrary and determines the size of the model displayed on the screen.
Figure 7. X-Z Plane Through Pyramid of Vision
Perspective is a necessary part of a life-like three dimensional display. An orthographic projection would simply ignore all z-coordinates and plot all equivalent x,y coordinate pairs on top of one another. Therefore, a perspective projection must include the z-coordinate as it transforms the three dimensional eye coordinate to the two dimensional screen.

Similar triangles will be used to formulate a perspective transformation. The transformation will be described using eye and screen coordinates. First, the general coordinates need to be converted to eye coordinates. Figure 7 show that the following equations accomplish this goal.

\[
\begin{align*}
xe &= x \\
ye &= y \\
ze &= d - z
\end{align*}
\] (1)

The similar triangles, defined for point 1 by lines a, b, and c and a', b', and c' in Figure 7, show that the following relationship holds true.

\[
a / b = a' / b'
\]

In terms of eye and screen coordinates, this equation may be written as:

\[
xe / ze = xs / ds
\]

After substituting the value of ds determined
earlier, and solving for the screen coordinates of this
point, the x screen coordinate, and by similar
development, the y screen coordinate are:

\[
xs = \frac{xe}{ze \cdot \tan(\theta/2)} = \frac{xe \cdot ds}{ze} \\
ys = \frac{ye}{ze \cdot \tan(\theta/2)} = \frac{ye \cdot ds}{ze}
\] (2)

A negative ze coordinate will cause the point to be
inverted on the screen. The model should always be
positioned far enough away so that no negative ze
coordinate occur. If the screen coordinate of a point,
such as point 3 in Figure 7, is less than minus one or
greater than plus one, a clipping process must take place
to limit the line or surface to the segment within the
pyramid of vision.

Careful study of Equations 1 and 2 will show that
the perspective calculations of screen coordinates is
dependent upon both angle of view (\(\theta\)) and the distance
(ds) to the general coordinate system origin from the
observer eye.

For example if the distance to the origin is held
constant and the angle of view is increased, the model
will appear smaller on the screen and the effect of
perspective will diminish. To prove this, consider the
following: if the angle of view is increased, the distance $ds$ is decreased since,

$$ds = \frac{1}{\tan \left( \theta/2 \right)}$$

and $\theta' > \theta$, then $\tan \left( \theta'/2 \right) > \tan \left( \theta/2 \right)$ and therefore $ds' < ds$.

Now Equation 2: since $ze = d - z$, $ze$ will remain constant for every point then,

$$xs' = xe \cdot ds' / ze \quad \text{and} \quad ys' = ye \cdot ds' / ze$$

and since $ds' < ds$, $xs' < xs$ and $ys' < ys$, this means that the model will appear smaller on the screen.

Now it has to be proved that the perspective effect will be less. To prove this it must be shown that $\Delta xs' < \Delta xs$ and $\Delta ys' < \Delta ys$, when the angle of view is increased. Let's first show that $\Delta xs$ and $\Delta ys$ are proportional to $ds$. Consider any two points of a model $P_1 = (x_1, y_1, z_2)$ and $P_2 = (x_2, y_2, z_2)$, then

$$xsl = x_1 \cdot ds / ze_1 \quad \text{and} \quad xs2 = x_2 \cdot ds / ze_2$$

since,

$$\Delta xs = xs1 - xs2$$

we have,

$$\Delta xs = ds \cdot (x_1 - x_2) \cdot (ze_2 - ze_1) / (ze_1 \cdot ze_2) \quad (3)$$

therefore for a constant $d$ and the same two points,

$$\Delta xs \sim ds,$$

and by the same procedure it can be shown that,

$$\Delta ys \sim ds.$$
Now if \( ds' < ds \), then for the same model two points
\[
\Delta xs' < \Delta xs \quad \text{and} \quad \Delta ys' < \Delta ys.
\]

A similar effect occurs when the angle of view is held constant and the distance to the origin is increased. To show that the model will appear smaller on the screen consider the following:

\[
xs = xe \cdot ds / ze \quad \text{and} \quad xs' = xe \cdot ds / ze',
\]

since \( ze = d - z \) and \( ze' = d' - z \),

\[
ze' > ze,
\]

and therefore \( xs' < xs \) and \( ys' < ys \).

To prove that the effect of perspective will diminish lets consider Equation 3. In this case \( ds \) is constant so lets say that \( k = ds' \cdot (x_1 - x_2) \), and then the equation becomes,

\[
\Delta xs = k \cdot (ze_2 - ze_1) / (ze_1 \cdot ze_2),
\]

then since \( ze = d - z \),

\[
\Delta xs = k \cdot [(d - z_2) - (d - z_1)] / [(d - z_1)(d - z_2)],
\]

and finally,

\[
\Delta xs = k \cdot (z_1 - z_2) / (d_2 - d \cdot z_1 \cdot z_2 + z_1 \cdot z_2).
\]

Now if \( d' > d \), then it can be easily seen that,

\[
\Delta xs' < \Delta xs \quad \text{and} \quad \Delta ys' < ys.
\]

It can be shown that the opposite will occur if the distance, \( d \) is help constant and \( \theta \) is decreased or if \( \theta \) is help constant and \( d \) is decreased.
By carefully examining the above relations it can be deduced that by decreasing the angle of view and increasing the distance to the model origin by the proper amounts, the effect of perspective could be diminished without making the model smaller on the screen.
IV. DISPLAY GENERATION

When command DRAW or VIEW is entered by the user the model is displayed on the TEKTRONIX 4051 screen. Before the model is displayed all the manipulation commands previously specified are applied to the model. The procedure used by MOVIE to display the model, is to process each model element through the sequence presented in Figure 8.

The first step is to apply any displacement and global translation to the element nodes. The second step is to apply the local translation/rotations and the global rotation transformations. Then, the third step is to submit the element to the hidden line/surface algorithm, if the specified command is VIEW. Finally if the element is visible, the perspective transformation, transformation to the TEKTRONIX coordinates, and the clipping procedure are performed. During the clipping procedure any edge or portion of it which lies inside the pyramid of vision is drawn on the screen. When every element of the model has been through this loop the model
Figure 8. MOVIE Display Generation Procedure
display is completed.

The model manipulation transformations were presented in Chapter 3. The following sections of this chapter describes the remaining process applied to the model elements in the display generation procedure.

Hidden Lines and Surfaces

When command VIEW is entered by the user the hidden lines/surfaces algorithm is applied to every model element during the display generation procedure. The algorithm will determine if the element is visible or not. Now lets explain how the algorithm works.

First the algorithm requires that the element nodes be defined in a consitance clockwise or counter-clockwise order as the object is seen from the outside. Then after the model has been manipulated and if the nodes of an element appear to the observer in the same order as it was defined the element is visible. On the other hand if the order is the opposite, the element most be seen from the inside of the model and therefore it is not visible. Now, the problem is to mathematically determine the direction of the element nodes as seen from the observer point.
For explanation purposes let's assume that we have a model in which the element nodes have been ordered in the clockwise direction. In this case the polygon elements normals obtained by using the nodes coordinates transformed to the eye coordinate system, will appear pointing toward the observer.

In order to calculate the polygon elements normals let's refer to vector algebra. First, note that the vector cross product of two vectors yields a vector perpendicular to the two vectors. The direction of the resulting vector is obtained by using the right or left hand rule. In this case the left hand rule applies since the eye coordinate system is a left handed system.

Three consecutive edges of an element define two vectors, if the two edges node coordinates are subtracted from one another. The geometric interpretation is that both vectors define a plane which is parallel to the element plane and passing through the origin. Therefore any vector perpendicular to this plane is also perpendicular to the element plane, which will be the element normal.

Refering to Figure 9 where the eye coordinates of node 1, 2 and 3 are \((x_1, y_1, z_1)\), \((x_2, y_2, z_2)\) and \((x_3, y_3, z_3)\) respectively. The two vectors parallel to the element
will be:

\[ \hat{v} = (x_2-x_1, y_2-y_1, z_2-z_1) = (x_v, y_v, z_v) \]

\[ \hat{w} = (x_3-x_1, y_3-y_1, z_2-z_1) = (x_w, y_w, z_w). \]

Now the two vector cross product to obtain the normal will be:

\[ \hat{n} = (x_n, y_n, z_n) \]

where,

\[ x_n = (y_v \cdot z_w - z_x \cdot y_w) \]
\[ y_n = (x_w \cdot z_v - x_v \cdot z_w) \]
\[ z_n = (x_v \cdot y_w - x_w \cdot y_v). \]

Since nodes 1, 2 and 3 were ordered in a clockwise order, the normal vector will point toward the observer.

In order to mathematically determine if the vector is pointing toward the observer or not, the scalar product between the normal vector and a vector \( \vec{l} \) from the origin to node 1 is obtained. It should be noted that the scalar product is proportional to the cosine of the angle \( \theta \) between these two vectors, where \( 0 < \theta < 180 \).

The scalar product \( d \) is:

\[ d = \hat{n} \cdot \vec{l} = (x_n \cdot x_1) + (y_n \cdot y_1) + (z_n \cdot z_1) = ||\hat{n}|| ||\vec{l}|| \cos \theta. \]
Figure 9. (a) Element Normal Vector for Element Nodes Appearing In Clockwise Direction

(b) Element Normal Vector for Element Nodes Appearing in Counter-Clockwise Direction
If \( d \) is negative then, \( \theta > 90^\circ \) and therefore the normal vector is pointing toward the observer, as shown in Figure 9a. If \( d \) is positive then \( \theta < 90^\circ \) and therefore the normal vector is pointing away from the observer, as shown in Figure 9b.

In conclusion if \( d \) is negative the nodes appear in the clockwise direction and the element is visible, otherwise the nodes appear in the counter clockwise direction and the element is not visible.

When the element nodes are ordered in a counter-clockwise direction as seen from the outside, then if \( d \) is negative the element is not visible and if \( d \) is positive the element is visible.

In MOVIE the user has the capability of "informing" the program, in which order the elements nodes of every part has been specified.

**Window of View Clipping**

The MOVIE program model manipulation capabilities allows the user to view the model from different angles and from different distances. Sometimes it happens that when a particular view of a model is to be displayed portions of it may fall outside the observer field of
view. One can think of the display screen as a window. If the model is too close, close portions of it may not be seen. If the observer moves closer to the window or the model is moved further out from the window more of the model could be seen. The process of eliminating parts of a model which lie outside the observer field of view is known as "windowing" (Sproull and Sutherland 1980).

The method used in MOVIE to perform the windowing process is clipping, which is the process of dividing each element of the model into its visible and invisible portions; this will allow the invisible portions to be discarded. When clipping is used, the display is given only valid visible information with the portions of the model outside the display already eliminated.

As presented in Chapter 3 the perspective transformation (Equation 2) is:

\[
xs = xe / ze \cdot \tan (\theta/2) \\
ys = ye / ze \cdot \tan (\theta/2).
\]

As explained in Chapter 3 the screen coordinate system is placed along the z-axis of the eye coordinate so that the origin is at the center of the screen and the minimum and maximum values where the axes pierce the pyramid of vision are minus one (-1) and plus one (+1).

Before the clipping procedure is applied to the
model the nodes coordinates most be transformed to the TEKTRONIX 4051 coordinate system. The TEKTRONIX 4051 coordinate system has the origin at the lower left corner and has a resolution of 1024 in the X direction and 780 in the Y direction. In order to present a square display the same resolution is used in both X and Y directions. Therefore the maximum allowed resolution is 780. The user has the option of specifying a resolution lower than 780. In which case the model will be displayed inside a smaller square.

In order to explain the equation used to perform the transformation to the TEKTRONIX coordinates, lets refer to Figure 10, where RES is the display resolution and R=RES/2. When xs and ys are multiplied by R, the result is coordinate values with origin at the center of the screen and with range of -R to R in the screen edges. Now in order to transform to the TEKTRONIX coordinate system R is added and the result will be coordinates values with origin at the lower left corner of the display and edges values from 0 to RES. Therefore the equations to perform the transformation to the TEKTRONIX 4051 coordinates system are:
Figure 10. TEKTRONIX 4051 Screen Layout
\[ x_{st} = R + x_s \cdot R \]
\[ y_{st} = R + y_s \cdot R. \]

Now that the coordinates has been transformed to the TEKTRONIX coordinate system we are ready to perform the model clipping. The way MOVIE perform the clipping procedure is by clipping every polygon element of the model. Furthermore the polygon elements are clipped by clipping every edge of the elements.

To perform the edges clipping each element is clipped against the four display edges. After the edge clipping process has been completed the resulting edge is drawn on the display.

To explain the edges clipping algorithm lets consider an edge for which the nodes display coordinates are \((x_l, y_l)\) and \((x_2, y_2)\).

**Clipping left edge:**

If \(x_l\) and \(x_2\) are both greater than zero no clipping against the left edge is necessary since both edge nodes lie to the right of the display left edge. Therefore the next step is to perform clipping against the right edge.

If \(x_l\) and \(x_2\) are both less than zero no clipping is
necessary since the edge lies to the left of the display left edge. This means that the edge is not visible, therefore the edge does not have to be displayed.

Now if only one of $x_1$ or $x_2$ is less than zero only one of the edge nodes is outside the display and clipping must be performed to calculate the coordinates of the intersection of the edge and the display left edge.

If $x_1 < 0$ and $x_2 > 0$, then node 1 is outside the display, therefore the new $x_1 = 0$ and the new $y_1$ must be calculated. Using similar triangles as shown in Figure 11,

$$(y_2 - y_1)/(y - y_1) = (x_2 - x_1)/-x_1$$

where $y$ is the new $y_1$ and therefore,

$$\text{new } y_1 = [(y_1 - y_2)(x_1)/(x_2 - x_1)] + y_1.$$

If $x_2 < 0$ and $x_1 > 0$, then node 2 is outside the display, therefore the new $x_2 = 0$ and the

$$\text{new } y_2 = [(y_2 - y_1)(x_2)/(x_1 - x_2)] + y_2.$$
Figure 11. Similar Triangles Used for Clipping Against the Left Edge
Clipping right edge:

If \( x_1 \) and \( x_2 \) are both less than \( \text{RES} \) no clipping against the right edge is necessary since both edge nodes lie to the left of the display right edge. Therefore the next step is to perform clipping against the bottom edge.

If \( x_1 \) and \( x_2 \) are both greater than \( \text{RES} \) no clipping is necessary since the edge lies to the right of the display right edge. This means that the edge is not visible, therefore the edge does not have to be displayed.

Now if only one of \( x_1 \) or \( x_2 \) is greater than \( \text{RES} \) only one of the edge nodes is outside the display and clipping must be performed to calculate the coordinates of the intersection of the edge and the display right edge.

If \( x_1 > \text{RES} \) and \( x_2 < \text{RES} \), then node 1 is outside the display, therefore the new \( x_1 = \text{RES} \) and the new \( y_1 \) must be calculated. Using similar triangles as shown in Figure 12,

\[
\frac{y_1 - y_2}{y - y_2} = \frac{x_1 - x_2}{\text{RES} - x_2}
\]
where $y$ is the new $y_1$ and therefore,
new $y_1 = [(y_1 - y_2)(\text{RES} - x_2)/(x_1 - x_2)] + y_2$.

If $x_2 > \text{RES}$ and $x_1 < \text{RES}$, then node 2 is outside the display, therefore the new $x_2 = \text{RES}$ and the new $y_2 = [(y_2 - y_1)(\text{RES} - x_1)/(x_2 - x_1)] + y_1$.

---

Figure 12. Similar Triangles Used for Clipping Against the Right Edge
Clipping bottom edge:

If \( y_1 \) and \( y_2 \) are both greater than zero no clipping against the left edge is necessary since both edge nodes lie above the display bottom edge. Therefore the next step is to perform clipping against the top edge.

If \( y_1 \) and \( y_2 \) are both less than zero no clipping is necessary since the edge lies below the display bottom edge. This means that the edge is not visible, therefore the edge does not have to be displayed.

Now if only one of \( y_1 \) or \( y_2 \) is less than zero only one of the edge nodes is outside the display and clipping must be performed to calculate the coordinates of the intersection of the edge and the display bottom edge.

If \( y_1 < 0 \) and \( y_2 > 0 \), then node 1 is outside the display, therefore the new \( y_1 = 0 \) and the new \( x_1 \) must be calculated. Using similar triangles as shown in Figure 13,

\[
\frac{x_2 - x_1}{x - x_1} = \frac{y_2 - y_1}{-y_1}
\]
where $x$ is the new $x_1$ and therefore,

$$\text{new } x_1 = \left[\frac{(x_1 - x_2)(y_1)}{(y_2 - y_1)}\right] + x_1.$$  

If $y_2 < 0$ and $y_1 > 0$, then node 2 is outside the display, therefore the new $y_2 = 0$ and the new $x_2 = \left[\frac{(x_2 - x_1)(y_2)}{(y_1 - y_2)}\right] + x_2$.

Figure 13. Similar Triangles Used for Clipping Against the Bottom Edge
Clipping top edge:

If \( y_1 \) and \( y_2 \) are both less than \( RES \) no clipping against the top edge is necessary since both edge nodes lie below the display top edge. Therefore this will be the end of the clipping procedure.

If \( y_1 \) and \( y_2 \) are both greater than \( RES \) no clipping is necessary since the edge lies above the display top edge. This means that the edge is not visible therefore the edge does not have to be displayed.

Now if only one of \( y_1 \) or \( y_2 \) is greater than \( RES \) only one of the edge nodes is outside the display and clipping must be performed to calculate the coordinates of the intersection of the edge and the display top edge.

If \( y_1 > RES \) and \( y_2 < RES \), then node 1 is outside the display, therefore the new \( y_1 = RES \) and the new \( x_1 \) must be calculated.

Using similar triangles as shown in Figure 14,

\[
\frac{x_1 - x_2}{x - x_2} = \frac{y_1 - y_2}{RES - y_2}
\]

where \( x \) is the new \( x_1 \) and therefore,

\[
\text{new } x_1 = \left[\frac{(x_1 - x_2)(RES - y_2)}{(y_1 - y_2)}\right] + x_2.
\]
If \( y_2 > \text{RES} \) and \( y_1 < \text{RES} \), then node 2 is outside the display, therefore the new \( y_2 = \text{RES} \) and the new \( x_2 = \frac{(x_2 - x_1)(\text{RES} - y_1)/(y_2 - y_1)}{x_1 - x_2} + x_1 \).

**Figure 14.** Similar Triangles Used for Clipping Against the Top Edge
At this point the clipped edge is drawn in the display, unless it has been found non-visible in which case this step will be skipped.

**Drawing the Edges on the TEKTRONIX**

In order to display the model edges, each edge node coordinates information must be transmitted from the PDP 11/34 to the TEKTRONIX 4051. The first step of this process is to set the TEKTRONIX in the Graph submode. This done by transmitting the ASCII control character GS (decimal 29) from the PDP 11/34 to the TEKTRONIX communication interface. When in the Graph submode, the TEKTRONIX display respond to ASCII characters by drawing straight lines in its high resolution storage screen.

Each node coordinate pair must be encoded as four ASCII characters. The first two characters represent the Y coordinate and the second two characters represent the x coordinate. Subroutine TKPLOT (see Appendix B) perform the encoding and then transmit the resulting four ASCII characters to the TEKTRONIX.

The first four ASCII characters sent immediately after the GS control character moves the graphic point to
the specified location. This point is interpreted as the starting point of the edge. The next four ASCII characters sent to the display are interpreted as the ending of the edge and a line is immediately drawn to that point on the screen. This process is repeated for every edge of the model until the model has been completely drawn.
V. UTILITY PROGRAM COMMANDS AND CAPABILITIES

UTILITY has three command levels. The program is entered at level one. If an option exist after a level one command, the program enters command level two. Finally, the program enters command level three when an option exist at level two. A carriage return after a lower level command prompt will transfer control to a higher command level. For example, a carriage return after level three command prompt will transfer to level two, and a carriage return after level two prompt will transfer control to level one.

The level one commands are GEOMetry, DISPlacement, CLEAn, SYMMetry, ORDER, MERGe, MAKE, HELP and EXIT. The level two commands for commands GEOMetry and DISPlacement are READ, WRITE, PRINT, CHANge and EXIT. The level three commands for these level two commands are GROUp, ELEMent, COORDinates, SHIFT, MOVE and EXIT.

These commands will now be described in terms of the action they initiate. The first four letters of each command are capitalized. This is its four letter key
word. Level one commands are followed by level two
commands, separated by a hyphen, and these by level three
commands, where applicable.

GEOMetry - READ
DISPlacement - READ

The two READ commands read the geometry or
displacement files, after requesting the file name from
the user. The files are assumed to reside in the system
disk although any storage device may be used, by
specifying it as part of the file name.

GEOMetry - WRITe
DISPlacement - WRITe

The two write commands write the geometry or
displacement files to disk (or the specified device)
after requesting the file name from the user.

GEOMetry - PRINT - GROUP
GEOMetry - PRINT - COORDinate
GEOMetry - PRINT - ELEMENT

The GEOMetry - PRINT commands list on the user's
terminal the element groupings (internal parts array),
coordinate, and element node numbers for the geometry
read or created by the program. For coordinates and elements, the program first requests the range of nodal coordinates or elements numbers the user wishes to look at.

**DISPlacement - PRINT**

This command lists on the user's terminal the values of the displacements after requesting from the user the range of nodes numbers.

**GEOMetry - CHANge - GROUp**

This command allows the user to change the internal parts array definition. The parts array read and written by UTILITY conforms to the definition given in Chapter II. The internal parts array used by UTILITY is a list of the total number of elements in a part. It is the difference between the upper and lower element limits of part plus one. The sum of the internal parts array equals the number of elements in the model.

The information requested by this command is the number of elements groups and the number of elements in each group.

**GEOMetry - CHANge - ELEMent**
This command allows the user to add, delete or replace elements from the connectivity array. To add or delete an element, the element group number and the element node number are entered. When adding elements, the number of elements in the group is increased by one and the element is entered at the end of that part in the connectivity array after pushing all the elements from the end of that part to the end of the array down one. When deleting elements, the elements numbers are matched with all elements in the part until a match is found. The number of elements in the group is then reduced by one and all elements after the deleted element are moved up one location. To replace an element the group number and the old and new nodes numbers must be entered. The program simply matches the old nodes numbers in the connectivity array, and when found the old node numbers are replaced with the new node numbers.

GEOMetry - CHANge - COORDinate

This command allows the user to change the number of coordinates in the coordinate array and overwrite any existing coordinate or add additional coordinates by entering the node number and coordinates of the desired
node.

GEOMETRY - CHANGE - MOVE

This command allows the user to move elements from one position in the connectivity array to another. The algorithm used simply takes the elements to be moved and place them in a temporary array. The connectivity array is now restructured by moving elements downward or upward according to the direction of the move. The data requested is the number of elements to be moved, the starting element number and which element the elements to be moved should follow. If the starting element number of those to be move is greater than the number of the one it is to follow, the shift will be downward for the elements from the one after the later location to one before the former location. The opposite holds if the starting element number to be moved is less than the element number of the one it is follow. Those elements stored in the temporary array are finally placed in their new position in the connectivity array to complete the MOVE command.

GEOMETRY - CHANGE - SHIFT

This command allows the user to shift the location
of the origin for his coordinate system. The command has two important uses. First, it may be used to position the model for a subsequent symmetry operation and second, it may be used to achieve consistent coordinate systems for multiple data file in anticipation that these files will be merged.

GEOMetry - CHANge - SCALE

This command allows the user to multiply the model node coordinates by a scale factor.

CLEAn

The CLEAn command allows the user to reduce data files which contain more than one node at the same location.

DISPlacement - CHANge

This command allows the user to overwrite and add values to the displacement file in the same way as the command sequence GEOMetry - CHANge - COORDinate.

DISPlacement - TRANsform

The transform command allows the user to build a displacement file which is the difference in the
coordinates, by components, of two geometry files.

**SYMMetry**

This command allows the user to create symmetrical additions to a model. For example, a full sphere may be formed from an eight of the model defined in the first quadrant.

After the user selects a symmetry plane about which the model will be reflected, the program forms a symmetry map of the node numbers in the new half. Nodes on the symmetry plane are not repeated.

New coordinates, together with displacements are formed by duplicating old values except those on the symmetry plane as indicated by the symmetry map. Furthermore, the coordinate in the direction perpendicular to the symmetry plane has its numerical sign reversed.

**ORDER**

This command invokes the ordering option which attempts to organize elements nodes consistently in a clockwise or counterclockwise direction. The algorithm presupposes that the all the parts elements are to be ordered in the order in which the first element nodes are
ordered.

MERGe

The MERGe command allows the user to append geometry and displacement file to those previously read in. Any number of files may be added. If no displacement file is specified by the user, the corresponding locations of the displacement array are set to zero. A blank geometry file name will terminate this command.

HELP

This command will print on the user's terminal a list of options available at that command level. The help message will also be printed when a command is not recognized.

EXIT

This command allows the user to exit program
UTILITY.

MAKE

This command allows the user to generate a wide variety of models using subroutines which are based on three primitives. These primitives are: hexahedrons,
ellipsoids, and volumes of revolution. All special cases of these objects are generated using the general algorithms, but input requests are limited to required non-redundant data as soon as a special circumstance has been recognized.

The MAKE command allows the user to generate two classes of objects. SHEETS which refers to those objects which are quasi two-dimensional. They may take on curvature and warp out of a plane, but they have no thickness when viewed on edge. VOLUMES, on the other hand, may be thought of as solid objects which indeed occupy some volume of space.

MAKE - SHEETS - QUADrilateral

This command is used to generate a surface as a whole (not just one element) described by four corner points, whether planar or warped. The program request from the user the coordinated of the four corner points. Then the program request the number of elements desired between the first and second corner points and between the third and fourth corner points. The program will then calculate the model node coordinates and will generate the connectivity array entries for the model elements.
MAKE - SHEEts - PARAllelogram

This command is used to generate a planar surface (again, not just a single element) which has parallel sides. The information requested will be the same as the quadrilateral except that only three corner points needs to be specified.

MAKE - SHEEts - SHELL

This command is used to generate the surface or any portion of an ellipsoid. The information required to generate the shell is as follow: ellipsoid's center coordinates, the radii in the x, y and z directions, starting and ending latitude and longitude, and the number of elements desired in the latitude and longitude directions. The program then will generate all the coordinates and the elements connectivity array.

MAKE - SHEEts - REVolution

This command is used to generate a surface formed by revolving a curve around and translating along the y-axis. Example include cones, cylinders and spirals. The information requested by the program is as follow: the number of nodes in the curve to be revolved, the
nodes coordinates and the angle of revolution. This command also allows the user to incrementally translate the curve and to change the radius distance to the y-axis while the curve is being revolved. Once the curve revolution has been performed the program allows the user to translate or rotate the object to a desired position.

MAKE - VOLUME - HEXAhedrons

This command is used to generate a volume which can be described with eight corner points and any quadrilatereal surface (warped or planar). The method used to describe the hexahedrons is by specifying a front and back faces which are quadrilaterals. The logic used to describe each of the faces is the same used in the QUADrilateral and PARAllelogram commands. After the front face has been described, the program asks the user if the hexahedron is prismatic (hexahedron with constant cross section) or not. If it is only a fifth point is required to describe the prismatic hexahedron, otherwise the back face quadrilateral must be specified.

MAKE - VOLUME - ELLIpsoid

This command is used to generate thick-shelled volume whose boundaries are described by ellipsoids. All
or any portion of this object may be generated. Furthermore the outer and inner ellipsoids of a thick-shelled volume need not to be similar. Special cases of an ellipsoid are, spheres which have the same radius in all three coordinate directions, spheroids which has the same radius in only two coordinate directions, and ellipsoids which has different radius in all three coordinate directions. The information required to generate the thick-shelled volume is the same of the MAKE - SHEETs - SHELL command, but in this case the radii for two ellipsoids are required, the outer and inner ellipsoids.

MAKE - VOLUME - REVOLUTION

This command is used to generate a model by revolving and/or translating a curve around the y-axis. The input information is the same of the MAKE - SHEETs - REVOLUTION. The only difference is that elements are generated between the last and first user specified nodes, thus closing the loop, for a volume of revolution.
VI. MOVIE PROGRAM COMMANDS AND CAPABILITIES

Program MOVIE is an interactive program that allows the manipulation of a model in space before it is displayed on the graphics output device. The sequence of manipulations and display may be repeated again and again until the user is satisfied with the display.

MOVIE, like UTILITY, acts as an interrogative program when it begins. After the minimum information necessary for the display of a picture is entered in response to the program's questions, the program becomes interactive. The user must enter at least the first four letters of a command for it to be recognized. If a command is not recognized, the unrecognized command is printed on the terminal and the user is given the option of looking at a list of available commands.

Program MOVIE commands are divided in three categories, general commands, scene manipulation commands and animation commands. The description of these commands is presented below.
General Commands

This section presents a description of commands of general nature. These commands are used to define scope parameters, produce a summary of the models data, read models data files, invoke the hidden line/surface algorithm, display the picture, list the commands and exit the program.

SCOPe

The SCOPe command allows the user to select from a number of options with respect to the coordinate triad. The coordinate triad is a coordinate axes drawn in the left lower corner of the display. The options from which the user may select are the following: the user may select to display or suppress displayment of the coordinate triad, the user may set the coordinate triad to follow the model rotation or to stay unchanged and show the original orientation of the model.

In addition the SCOPe command allows the user to specify the display frame resolution. The display frame resolution will determine the side of the frame.
SUMMARY

The SUMMARY command is a subset of the CENTER command described in the next section. It finds the maximum values of the translated coordinates, the displacements, and the scalar functions. These values are then printed on the user's terminal.

READ

The READ command lets the user read new data from disk or other storage media. The initial data files are requested from the user during program initialization.

FAST

The FAST command allows the user to specify which parts of the model are to be submitted to the poor man's hidden line/surface algorithm. The user specifies if the direction of the elements nodes are defined in a clockwise or counterclockwise order.

DRAW

The DRAW command displays a picture of the model as modified by all previous commands without using the poor man's hidden line/surface algorithm.
VIEW

The VIEW command displays a picture of the model as modified by all previous commands. The poor man's algorithm is used to eliminate hidden line or surface as the case may be.

SHRink

The SHRink command allows the display of separated elements. The node coordinates are moved toward the center of the element. A shrink value of zero will produce no separation, while a maximum value of 1.0 will cause the elements to disappear.

TITLE

The TITLE command allows the user to print a one line title at the top of the display frame.

NODE

The NODE command allows the user to display the object node numbers.

POLY

The POLY command allows the user to display the object polyon numbers.
HELP

The HELP command lists all available commands on the user's terminal.

EXIT

The EXIT command lets the user stop execution of the program in a controlled manner.

Scene Manipulation Commands

The scene manipulation commands allow the user to perform the features described in Chapters III and IV. The features include selective parts display, pyramid of vision definition, global and local translation and rotations.

PARTS

The PARTS command allow the user to select the parts for display. A one-dimensional display array is loaded with a one if the part is to be displayed and a zero if the part is not to be displayed.

FAST
The FAST command allows the user to specify which parts of the model are to be submitted to the hidden lines/surfaces algorithm. This command is also used to specify the order in which the objects parts elements nodes has been ordered, clockwise or counter-clockwise.

IMMUne

The IMMUne command allows the user to make parts temporarily IMMUne to global rotations.

FIELD

The FIELDd commands gets its name from field of view. It allows the user to select the angle of view and front and back clipping planes to define the pyramid of vision.

DISTance

The DISTance command allows the user to set the distance to the general coordinate system from the eye. Varying this distance, as explained earlier, alters the size and perspective of the model.

TRANslate

The TRANslate is the global translation command. The user indicates the coordinates of the origin of the
general coordinate system in the global coordinate system. The negative of the new origin is added to the global coordinates since the origin of the general coordinate system is defined in terms of them.

**CENTER**

The CENTER command provides a SUMMARY of the coordinates and displacements as described previously. It also sets the global translation matrix to the average of the minimum and maximum x, y, and z coordinate values thus translating the origin to the model center.

**EXPLode**

EXPLode is the local translation command. It allows the user to define an explosion pattern for every part. The user also sets a scale factor which, when multiplied by the pattern, gives the local translation. The explosion pattern may be thought of as the vector direction in which the part will move and the scale factor as a scalar multiplier of this vector.

**ROTAte**

The ROTAte command allows the user to perform global rotations of the model about the general coordinate axis.
Every time the ROTate command is given, the rotation matrix is updated by that rotation. When a DRAW or VIEW command is given, the coordinates are transformed as they are displayed.

PIVOt

The PIVOt command gives the user local rotation capabilities. A local rotation matrix is defined for every part. These are modified in a manner similar to the ROTate command. In addition to defining the rotation for each part, the user must define the center of a coordinate system parallel to the global coordinates about which the rotations will take place for each part. This is a relative origin since the part is first translated to it, then rotated about it, and finally translated back.

RESTore

This command zeroes the global translation matrix and initialize the global and the local rotation matrices to the identity matrix. The effect of these operations is to kill all previous translation and rotations except for the local translation which can be reset by setting its scale factor to zero.
The SCALE command allows the user to select the scale factor for the nodes displacements.

The SHIFT command allows the user to move the viewing window in the plane of the display device.

Animation Commands

Along with the capabilities described in the previous sections of this chapter, the ability to define a sequence of pictures and then have the computer display them, updating them as it does, is very desirable. To this end, several of the commands spoken of in previous section may be automated for a specified number of frames. All global and local translations and rotations may be animated together with distance to the origin.

For example, suppose the user wishes to look at an object by revolving it about the y-axis. Using the ROTATE command, he would have to give a command every time he wished a new view. Using the animation capabilities, the user need only to specify the total
rotation and the number of frames over which the rotation will take place.

One application of the animation feature is the production of "movies". The picture sequence may be filmed for playback in real time. This is certainly the most exciting application. Other applications, such as the one described in the previous paragraph, show the usefulness of animation.

ANIMation

This is the command that allows the user to select animation options. The user specifies the number of frames and sets the change in global translation, global and local rotation (local translation) scale factor and distance to the origin. The user may optionally specify a displacement scalar factor which provides a linearly or harmonic change in displacements. The user has the option of viewing the sequence or not. If the user only wishes to see the last picture, he may by so indicating. The program, when given a DRAW or VIEW command, will process the sequence but does not display under this feature. After the sequence is finished, the user may look at the last picture by giving another VIEW or DRAW command.
LINEar

This command along with the transient data option in the animate command allows the user to linearly interpolate between two displacement files. Specifically the LINEar command is used to specify and read the two displacement files. Then when the transient option is set on the animate command the displacement scale factor will be increased from zero to full value over the specified number of frames which will achieve a linear interpolation between the two displacement files.
VII. RESULTS AND CONCLUSION

Results

To demonstrate the capabilities of program UTILITY and MOVIE, a number of sample runs were performed and the results are presented in Appendix E. This runs included examples from the referenced MOVIE.BYU documentation as well as a number of geometric figures of the author's own design.

UTILITY model generation capabilities (MAKE command) is demonstrated by presenting a sample for every model generation primitive. First a hard-copy of the UTILITY program run used to generate the model is presented. Then one or more model sample displays, generated using program MOVIE, are presented.

In order to demonstrate the other UTILITY commands capabilities a Bullet shaped model generation is presented. This Bullet model is used to demonstrated program MOVIE model manipulation and display capabilities.

Program MOVIE capabilities are presented in the same
maner as program UTILITY.

**Conclusion**

This paper presents the description of a Mini-Computer Based General Purpose Interactive Computer Graphics Software System which has been developed and installed at the University of Central Florida, College of Engineering PDP 11/34 computer system.

As shown by the results presented in Appendix E, the system provides the capabilities to generate the computer data structure necessary to describe the geometry of an object and the necessary functions required to manipulate and display the object.

Figures 15 to 29 demonstrate the capability of the system, through program UTILITY, to generate objects based on various primitives: hexahedrons, ellipsoids and surfaces and volumes of revolution.

A Bullet shaped model generated by UTILITY, is included to demonstrate its ability to make symmetrical additions to the objects. This Bullet shaped model is used in figures 30 to 45 demonstrate program MOVIE capabilities to manipulate and display the objects.

Programs UTILITY and MOVIE, as described in this
paper and as demonstrated by the results, provide a Mini-Computer Based General Purpose Interactive Computer Graphic Software System, therefore satisfying the objectives of this project.

Although the project objectives have been satisfied there is always room for enhancements. One of the limitations of the system is that the only output devices it can handle are vector drawing devices. When a continuous tone hardware display device becomes available, the system could be modified to provide the necessary functions to display objects in a color continuous tone display. The majority of the required routine to do this are included in MOVIE.BYU. They can be modified and incorporated since the system structure was kept the same.

Another limitation of the system is that the hidden lines/surfaces algorithm implemented, only eliminate hidden lines/surfaces hidden by parts of an object itself. This is, the procedure cannot eliminate lines/surfaces hidden by other objects on the scene. In order to provide this capability a more complicated algorithm must be implemented.

The Watkin's hidden lines/surfaces algorithm is included in MOVIE.BYU and it will provide the capability
described above. This procedure as written is too big to be installed on the PDP 11/34, and a considerable amount of modification will have to be done in order to accommodate it into the actual system.
APPENDIX A

PROGRAMS HIGH-LEVEL FLOW CHARTS
UTILITY
Gets user commands and calls the appropriate routine to process cmd.

CMD
Reads commands from user's terminal

CMD "GEOM" Y
GEOM
Reads, writes, prints and changes Geometry file.

N

CMD "DISP" Y
DISP
Reads, writes, prints and changes Displacement files

N

CMD "SYMM" Y
SYMM
Creates symmetrical additions to the Geom. and Displacement files

N

CMD "ORDER" Y
ORDER
Orders polygon elements nodes in consistent cw or ccw order

N
80

CMD "MERGE" Y
MERGE Merges two or more Geom. Disp. files in to one file
N

CMD "MAKE" Y
MAKE Generates model based on primitives: Hexahedrons, Ellipsoids and Revolution
N

CMD "CLEAN" Y
CLEAN Consolidates nearby nodes
N

CMD "HELP" Y
HELP Prints HELP messages on user's terminal
N
100
CMD
Prints command prompt and reads cmd

BEGIN

CMD "EXIT"

STOP PROGRAM

RETURN

Y

N
CLEAN
Consolidates nearby nodes.

BEGIN

N multiple files

Y

RDDISP
Gets Disp. file name and then reads it.

RDGEOM
Gets Geom. file name and then reads it.

RETURN
CONNAR
Calculates the connectivity array of hexahedrons or ellipsoids.

BEGIN

NODEP
Finds the number of nodes in a panel element

LOAD
Loads node number into their proper loc. in the connectivity array

RETURN
COORDP
Calculates the nodal coordinates of a hexahedron element.

BEGIN

MESCOR
Interpolates between two points to obtain the coordinates of an intermediate point.

RETURN
DISP
Reads, writes, prints and changes Displacement files

BEGIN

CMD
Read second level command

CMD
"CR"

Y
RETURN

N

CMD
"READ"

Y

RDDISP
Reads Displacement file

N

CMD
"WRITE"

Y

WRDISP
Writes Displacement file to output file

N

B
CMD "CHANGE"

Change nodes displacements by node number

Y

IS Node # 0

Y

200

N

Is Node # max# of nodes

Y

OVER Output error mess when Disp. array is exceeded

N

CMD "PRINT"

Prints specified nodes disp. on user's terminal

Y

200

N

Is node # 0

N

C
C

CMD

N

HELP
Output HELP message

Read two Geom. files set
Disp. array to difference
of the two

OPEN
Gets Geometry files
names and then
open them

200
ELLIP
Generates Geom. file for shell and volume ellipsoids

BEGIN

Read ellipsoid parameters from user's terminal

MESHDV
Reads number of elemental divisions and check maxes

INTERS
Calculates the coordinates of a node on ellipsoid on lat. + long

N
SHELL?

Y

CONNAR
Calculates the connectivity array of a volume ellipsoid element

RETURN
GEOM
Reads, writes, prints, change, translate and scale the Geom. file.

BEGIN

CMD
Reads second level command

CMD "CR"

Y
RETURN

N

CMD "READ"

Y

RDGEOM
Reads Geom. file

N

CMD "WRITE"

Y

WRGEOM
Writes GEOM. file

N

D
HELP
Types HELP message if command not recognized

400
CMD
Read third level command

CMD "CR"
Y
N

CMD "MOVE"
Y
N

CMD "GROUP"
N

CMD "COOR"
Y
N

CMD "SHIFT"
Y
N

CMD "SCALE"
Y
N

MOVE
Relocates connectivity array elements

Perform desired change to the parts array

Perform desired changes to the coordinates array

Translate model coordinates to specified origin

Multiply coordinates by scale factor

HELP
Types HELP message if command not recognized
CMD
Get third level command

CMD "CR"
Y 300

CMD "GROUP"
Y
Print parts element boundaries on user's terminal
N

CMD "ELEM"
Y
Print elements connectivity array on user's terminal
N

CMD "COORD"
Y
Print nodes coordinates on the user's terminal
N

HELP
Types HELP message if command not recognized
HEXGEN
Generates Geom. file for hexahedrons

BEGIN

CORNPT
Reads generates and changes the hexahedrons corner points

MESHDV
Reads the number of elemental division and check maxes

COORDP
Generates hexahedrons coordinates array

CONNAR
Calculates the connectivity array for the hexahedron

RETURN
MAKE
Model generation routine based on primitives: hexahedrons, ellipsoid and rev.

BEGIN

CMD
Get second level command

CMD "CR"
Y
RETURN

N

CMD "SHELL"
Y
G

N

CMD "VOLUME"
Y
H

N
CLEAN
Consolidates nearby nodes

CMD
Get third level command

CMD "CR"

CMD "PARA" or "QUAD"

CMD "ELLIP"

CMD "REVO"

Y

400

QUADGN
Generates Geom. file for quadrilateral surfaces.

ELLIP
Generates Geom. file for ellipsoids surfaces and volumes

RELOLV
Generates Geom. file for surfaces and volume of revolution
CLEAN
Consolidates nearby nodes

CMD
Read third level command

CMD "CR"

Y

CMD "HEXA"

Y

CMD "ELLIP"

Y

CMD "REVO"

N

HEXGEN
Generates Geom. file for hexahedrons

ELLIP
Generates Geom. file for surface and volume ellipsoids

REVOLV
Generates Geom. file for volume of revolution
MERGE
Merge Geom. and/or Disp. files with other Geom. and/or Disp files

BEGIN

RDGEOM
Reads Geom. file to be merged

RDDISP
Reads Disp. file to be merged

RETURN
MESHDV
Reads number of elemental division and check maxes

BEGIN

Read the elemental divisions and calculate new #of parts, #of elements and # of nodes

IS
# of parts, # of elem or # of nodes exec.

Y
OVER
Output error mess. and stop program

N

RETURN
RDDDISP
Gets displacement file name and then reads the site

BEGIN

OPEN
Request file name and open specified file

END
RDGEOM
Gets Geometry file name and then reads it

BEGIN

OPEN
Request file name and open specified file

RETURN
REVOLV
Generates Geometry file for surface and volume of revolution

BEGIN

Get volume or surface of revolution parameters

is

# of part, elements & coordinates

Y

REVCOR calculates coordinates for volume of revolution

OVER
Output error message and stops program

N

translate rotate

Y

TRAROT Translate or rotate the model

N

RETURN
QUADGN
Generates Geometry file for quadrilateral surfaces

BEGIN

CORNPT
Reads, generates and change quadrilateral corner points

MESHDV
Reads the number of elemental divisions and check maxes

COORDP
Generates quadrilateral coordinate array

CONNAR
Calculates the connectivity array for quadrilaterals

RETURN
SYMM
Creates symmetrical additions to the geometry and displacement file

BEGIN

get plane of symmetry

is

\# of part elements, node \( \geq \) max

Y

OVER
Output user message and stop the program

N

Multiple files

Y

RDDISP
Reads displacement file

N

WRDISP
Writes displacement file

RETURN
WRDISP
 Writes displacement array to output file

BEGIN

OPEN
 Gets displacement output file name and then open it

RETURN
WRGEOM
Writes Geometry file array to output file

BEGIN

OPEN
Gets Geometry output file name and then open it

RETURN
MOVIE
Computer Graphic Software
System models manipulation
and display program

READ
Reads Geometry and
Displacement files

CLEAR
Clear arrays and
other variables

INIT
Initialize flags
parameters and other
arrays

Read user command
READ
Reads Geometry and displacement files

EXIT
STOP PROGRAM

ROTA
Gets axis and angle of rotation. Generates Global rotation transformation matrix

REST
Clears Rotation and translation transformations

TRAN
Translate origin to specified coordinate
SCAL
Reads displacement scale factor from user's terminal

SCOP
Scope parameters

DIST
To specify distance from observer to model coordinate origin

FIEL
To specify angle of view

SE PART
To specify which models part are to be displayed
To specify parts local translations (Explode)

SUMCEN
Gives Geom and Disp files data summary mis and max etc.

TITLE
To specify the frame title, to be typed

ANIMAT
To specify the animation sequency plus parameters and conditions

VIDERA
Performs the display generation procedure
To specify the parts to be submitted to the hidden line/surface algorithm and parts elements nodes direction CC or CW.

To specify local rotation about relative origin. Calculate local rotation transformation matrix.

Type the available commands on user's terminal.

To specify model shrinking factor.

To enable/disable node numbering.
HELP
Type available commands on user's terminal when command not recognized

POLY
To enable/disable polygon numbering -

IMMU
To specify which model part to be immune to global rotations

TRAN
To translate the origin to specified coordinates
BGNFRM
Perform frame initialization
sets scale factor clears
screen and draws a box around the picture area

BEGIN

PL RAS
Clears TEKTRONIX screen and sets Graph submode

MOVABS
Moves graphic point to specified coordinate

DRWABS
Draws a line from current position to specified coordinates

RETURN
CAORRO
Draws coordinates triad on frame lower left corner

BEGIN

PLTLIN
Draw line segment, defined by two end point, on TEKTRONIX scope

RETURN
COORD
Performs local and global transformation and shrinking on polygon elements, also calculates element normals.

BEGIN

POINTS
Set up element coordinates array and perform local & global transformation.

Shrink Element

Y

SHRINK
Moves polygon nodes toward center of polygon.

N

is hidden line specified

Y

NORMAL
calculates polygon normals.

N

RETURN
DRAW
Calculates edges nodes
TEKTRONIX coordinate does
correction and then plots
the edge on the scope

BEGIN

PLTLIN
Draw line segment on TEKTRONIX
scope

RETURN
DRWABS
Draw line segment on TEKTRONIX scope

RETURN

TKPLOT
Translate point coordinate to TEKTRONIX, ASCII Codes and transmit Bytes to TEKTRONIX

RETURN
INIT
Initialize flags parameters and other arrays

BEGIN

REST
Clears rotation and translation transformation

SUMCEN
Gives Geom, and Disp files data summary
Min and Max, etc.

RETURN
MOVABS
Moves graphic point to specified coordinate location on screen

BEGIN

TKPLOT
Translate point coordinate to TEKTRONIX, ASCII Codes and transmit Bytes to TEKTRONIX

RETURN
PART
Performs display generation procedure for array elements of a part

BEGIN

CONEL
Gets the polygon element connectivity array

COORD
Gets element node coordinates and calculates normals

hidden lines

VISIP
Test for polygon element visibility

Y

N

POLY
Sends polygon element edges for display

RETURN

VISIT
Test for visibility of subdivider triangles
PIVOT
To set object part local rotation about a relative origin

BEGIN

ROTAT
Calculate the part local transformation matrix

RETURN
PLTLIN
Draws line segment on TEKTRONIX scope

BEGIN

MOVABS
Moves graphic point to specified coordinate

DRWABS
Draws a line from current position to specified coordinate

END
POINTS
Perform initiation, translation and displacement transformation on a polygon element

BEGIN

MULTDD
Multiplies coordinates by local transformation matrix

MULTDC
Multiplies coordinate by local transformation matrix

RETURN
POLY
Sends polygon elements edges for simple line drawings on the TEKTRONIX scope

BEGIN

DRAW
Calculate TEKTRONIX screen coordinates, perform clipping on the edge and then plots the edge

RETURN
POLNUM
Sends visible elements numbers for display

BEGIN

OUTNUM
Calculate element number position in screen and then types the number on the screen

RETURN
READ
Reads Geom-file

BEGIN

OPEN
Gets files names and then opens the file for I/O

LASTC
Sets parts element limits & determines if polygon's center point is specified

SEPART
Sets the object parts to be displayed on the next DRAW or VIEW command

RETURN
ROTA
To specify global rotation

BEGIN

ROTAT
Calculate new global rotation
transformation matrix

RETURN
TKPLOT
Translate real coordinates to TEKTRONIX code and transmit the code to the TEKTRONIX

BEGIN

TTOUT
Assemble routine that transmit a Byte from the PDP 11/34 to the TEKTRONIX 4051

RETURN
VIEDRA
Performs the display generation procedures for the object when the DRAW or VIEW command is entered

BEGIN

Animation

Y

ROTAT
Calculates frame global and local transformation matrixes

N

BGNFRM
Performs frame initialization sets scale factor, clears screen and draws a box around the picture area

COARRO
Draw coordinates triad

PART
Perform the display generation procedure for every element of a part

F
ENDFRM
Ends frame and return scope to Alpha Numeric sub mode

F

Node number

Y

NODNUM
Sends visible node, number for display

N

POLNUM
Sends visible polygon, number for display

Polygon number

Y

N

RETURN
APPENDIX B

PROGRAMS LISTINGS
Program UTILITY Listing

NOTE: For variables definition refer to MOVIE.BYU listings.
PROGRAM UTILITY
INTEGER OUTPUT
REAL LIT

DIMENSION NPL(2,20),X(3,250),IP(9,250),U(3,250)
DIMENSION JNK(9,250),NPX(2,20),LIT(13)
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEVII/ INPUT,OUTPUT
COMMON/FUNC/ SX
COMMON/JUNK/ JNK
COMMON/MAXI/ NMAX,NJMAX,NPTMAX

SET FOLLOWING PARAMETERS CONSISTENT WITH THE DIMENSIONED ARRAYS

DATA NMAX/20/,NJMAX/250/,NPTMAX/250/,NODE/9/
DATA LIT/1HS,4HGEOM,4HDISP,4HFUNC,4HSYMM
1,4HORDE,4HHELP,1H?,1H,4HMerg,4HMAKE,1HA,4HCLEA/

TYPE TITLE FOR USER INFORMATION

TYPE 30

INITIALIZE CONTROL VARIABLES AND REQUEST DATA TYPE

NJ=0
NP=0
NPT=0

IF CORE IS NOT CLEARED TO ZERO WHEN THIS PROGRAM IS RUN,
YOU MAY WISH TO CLEAR THE FOLLOWING ARRAYS AT THIS LOCATION:
NPL, NPX, X, IP, U, S, SX, AND JNK

REQUEST COMMAND AND PROCEED
CMD IS THE COMMAND WORD

10 WORD=CMD(1)

LIT(2) = "GEOM"

IF(WORD.EQ.LIT(2)) CALL GEOM(NPL,NPX,X,IP,NODE)

LIT(3) = "DISP"

IF(WORD.EQ.LIT(3)) CALL DISP(U,JNK)
LIT(5) = "SYMM"
IF(WORD.EQ.LIT(5)) CALL SYMM(NPL, X, IP, U, S, NODE)
LIT(6) = "ORDE"
IF(WORD.EQ.LIT(6)) CALL ORDER(NPL, IP, JNK, NODE)
LIT(10) = "MERG"
20 IF(WORD.EQ.LIT(10)) CALL MERGE(NPL, X, IP, JNK, U, S, NODE)
    I2=(NPTMAX/NODE)+2
LIT(11) = "MAKE"
IF(WORD.EQ.LIT(11)) CALL MAKE(NPL, X, IP, NODE, U, S, JNK $, JNK(1, I2))
LIT(7) = "HELP", LIT(8) = "?", LIT(9) = ""
LIT(13) = "CLEA"
    I1=0
    IF(WORD.EQ.LIT(13)) CALL CLEAN(IP, X, U, S, NPL, JNK, JNK $, JNK(1, I2), NODE, I1)
        IF(WORD.EQ.LIT(7), OR. WORD.EQ.LIT(8), OR. WORD.EQ.LIT( $9))
            CALL HELP(1)
    GO TO 10
C
30 FORMAT(44H <MOVIE SYSTEM UTILITY AND MODEL GENERATI
$ON>)
40 FORMAT(A1)
END
SUBROUTINE CLEAN(IP,X,U,S,NPL,IG,INEW,NODE,IFLAG)
INTEGER OUTPUT
REAL LIT

C SEE POSSIBLE INCOMPATIBILITIES (5)

DIMENSION IP(9,250),X(3,250),U(3,250),IG(250),INEW($250),LIT(1)
DIMENSION NPL(2,20)
COMMON/DEVI/ INPUT,OUTPUT
COMMON/CNTL/ NP,NJ,NPT
DATA LIT/1HY/

IDELE=0
NODE1=NODE-1
NODE2=NODE-2
TEST=0.0
IF(IFLAG,EQ.1) GO TO 10
TYPE 410,
ACCEPT 370, TEST
10 TEST2=TEST*TEST
DO 20 I=1,NJ
20 INEW(I)=0
NJ1=NJ-1
NP1=0
NP2=0
DO 90 N=1,NP

C CREATE LIST OF NODES IN GROUP

C
DO 30 I=1,NJ
30 IG(I)=0
NP1=NP2+1
NP2=NP2+NPL(1,N)
DO 50 J=NP1,NP2
   DO 40 I=1,NODE1
      I1=IP(I,J)
      IF(I1.EQ.0) GO TO 50
40 IG(I1)=1
50 CONTINUE

C TEST FOR AND CAPTURE CLOSE NODES

C
DO 80 I=1,NJ1
   IF(IG(I),EQ.0.OR.INEW(I),NE.0) GO TO 80
   I1=I+1
   X1=X(1,I)
   X2=Y(2,I)
   X3=X(3,I)
   DO 70 J=I1,NJ


IF(IG(J) .EQ. 0) GO TO 70
IF(INEW(J) .EQ. 0) GO TO 60
IF(INEW(J) .LT. I) GO TO 70
50
XT=ABS(X(1,J)-X1)
IF(XT .GT. TEST) GO TO 70
YT=ABS(X(2,J)-X2)
IF(YT .GT. TEST) GO TO 70
ZT=ABS(X(3,J)-X3)
IF(ZT .GT. TEST) GO TO 70
XT=XT*XT+YT*YT+ZT*ZT
IF(XT .GT. TEST2) GO TO 70
INEW(J)=I
70 CONTINUE
80 CONTINUE
90 CONTINUE

C FORM LIST OF REDUCED NODE NUMBERS

NNJ=0
DO 110 I=1,NJ
   IF(INEW(I) .NE. 0) GO TO 100
   NNJ=NNJ+1
   INEW(I)=NNJ
   GO TO 110
100 I1=INEW(I)
   INEW(I)=INEW(I1)
110 CONTINUE
IF(NNJ .EQ. NJ) GO TO 150

C RENUMBER CONNECTIVITY ARRAY

DO 140 J=1,NPT
   DO 120 I=1,NODE1
      I1=IP(I,J)
      IF(I1 .EQ. 0) GO TO 130
      IP(I,J)=INEW(I1)
120 CONTINUE
130 I1=IP(NODE,J)
   IF(I1 .NE. 0) IP(NODE,J)=INEW(I1)
140 CONTINUE

C REDUCE EDGES IN CONNECTIVITY ARRAY

150 DO 220 J=1,NPT
   DO 210 I=1,NODE1
160   I1=IP(I,J)
   IF(I1 .EQ. 0) GO TO 220
   I2=I+1
   IF(I2 .EQ. NODE) GO TO 170
   I3=IP(I2,J)
210 CONTINUE
220 CONTINUE
IF(I3.NE.0) GO TO 180

170 I2=1
I3=IP(I2,J)

180 IF(I1.NE.I3) GO TO 210.
IF(I.EQ.NODE1) GO TO 200
DO 190 K=I,NODE2
190 IP(K,J)=IP(K+1,J)

200 IP(NODE1,J)=0
IDELE=IDELE+1

210 CONTINUE

220 CONTINUE
NDELETE=NJ-NNJ
IF(NDELETE.EQ.0) GO TO 330

C REDUCE COORDINATE ARRAY

C J1=0
DO 240 J=1,NJ
IF(INFW(J).LE.J1) GO TO 240
J1=J1+1
DO 230 I=1,3
230 X(I,J1)=X(I,J)

240 CONTINUE
IF(IFLAG.EQ.1) GO TO 320

C MULTIPLE DISP. AND SCAL. FUNC. FILES?

C TYPE 380,
ACCEPT 340, ANS
MULTI=.FALSE.

C LIT(1) = "Y"

C IF(ANS.EQ.LIT(1)) MULTI=.TRUE.
NFFILES=1
IF(.NOT.MULTI) GO TO 250

C REDUCE DISP. ARRAY

C TYPE 390,
ACCEPT 360, NFFILES
250 DO 280 K=1,NFILES
IF(MULTI) CALL RDDISP(I)
J1=0
DO 270 J=1,NJ
IF(INEW(J).LE.J1) GO TO 270
J1=J1+1
DO 260 I=1,3
260 U(I,J1)=U(I,J)
270 CONTINUE
IF(MULTI) CALL WRDISP(U,NNU)
230 CONTINUE
320 NNU=NNU
330 IF(IFLAG.EQ.1) RETURN
       TYPE 350, NODEL, IDELE
       RETURN
C
340 FORMAT(A1)
350 FORMAT(10H+<DELETED:,I3,10H NODES AND,I3,7H EDGES>/$)
C
SEE POSSIBLE INCOMPATIBILITIES (1)
C
360 FORMAT(I)
370 FORMAT(F10.0)
C
SEE POSSIBLE INCOMPATIBILITIES (2)
C
380 FORMAT(19H+<MULTIPLE FILES?> $)
390 FORMAT(32H+<NUMBER OF DISPLACEMENT FILES> $)
410 FORMAT(31H+<NODE CONSOLIDATION DISTANCE> $)
END
FUNCTION CMD(INDX)
INTEGER OUTPUT
REAL LIT
DIMENSION CHR(3), LIT(1)
COMMON/DEVI/ INPUT, OUTPUT
DATA CHR/5H+>, ,5H+>>> ,5H+>>> /
DATA LIT/4HEXIT/

TYPE COMMAND PROMPT

TYPE 20, CHR(INDX)

ACCEPT COMMAND

ACCEPT 10, CMD

LIT(1) = "EXIT"

IF(CMD, EQ, LIT(1)) CALL EXIT
RETURN

10 FORMAT(A4)

SEE POSSIBLE INCOMPATIBILITIES (2)

20 FORMAT(A5, $)
END
SUBROUTINE CONNAR(IP,SIDE,NODE)
DIMENSION NN(8),IP(9,250),SIDE(6)
COMMON/CNTL/ NP,NJ,NPT
COMMON/MESH/ NE1,NE2,NE3,NNP,NNJ,NNPT

GENERATE CONNECTIVITY ARRAY

NEXT=NPT+1

GET CONNECTIVITY OF ELEMENTS ON FRONT FACE

IF(SIDE(1).LT.0.) GO TO 20
K=1
DO 10 JJ=1,NE2
   J=JJ
   DO 10 II=1,NE1
      I=II
      CALL NODEP(NN,I,J,K)
10 CALL LOAD(NN,IP,NEXT,1,2,3,4,NODE)

GET CONNECTIVITY OF ELEMENTS ON BACK FACE

20 IF(SIDE(2).LT.0.) GO TO 40
   K=NE3
   DO 30 JJ=1,NE2
      J=JJ
      DO 30 II=1,NE1
         I=II
         CALL NODEP(NN,I,J,K)
30 CALL LOAD(NN,IP,NEXT,6,5,8,7,NODE)

GET CONNECTIVITY OF ELEMENTS ON FIRST SIDE

40 IF(SIDE(3).LT.0.) GO TO 60
   I=1
   DO 50 JJ=1,NE2
      J=JJ
      DO 50 KK=1,NE3
         K=KK
         CALL NODEP(NN,I,J,K)
50 CALL LOAD(NN,IP,NEXT,5,1,4,8,NODE)

GET CONNECTIVITY OF ELEMENTS ON SECOND SIDE

60 IF(SIDE(4).LT.0.) GO TO 80
   I=NE1
   DO 70 JJ=1,NE2
      J=JJ
      DO 70 KK=1,NE3
         K=KK
90 CONTINUE

END
CALL NODEP(NN,I,J,K)
70 CALL LOAD(NN,IP,NEXT,2,6,7,3,NODE)

C
C GET CONNECTIVITY OF ELEMENTS ON BOTTOM
C
90 IF(SIDE(5).LT.0.) GO TO 100
   J=1
   DO 90 KK=1,NE3
      K=KK
      DO 90 II=1,NE1
         I=II
         CALL NODEP(NN,I,J,K)
   90 CALL LOAD(NN,IP,NEXT,5,6,2,1,NODE)

C
C GET CONNECTIVITY OF ELEMENTS ON TOP
C
100 IF(SIDE(6).LT.0.) GO TO 120
   J=NE2
   DO 110 KK=1,NE3
      K=KK
      DO 110 II=1,NE1
         I=II
         CALL NODEP(NN,I,J,K)
   110 CALL LOAD(NN,IP,NEXT,4,3,7,8,NODE)
120 CONTINUE
RETURN
END
SUBROUTINE CONNSH(NE1,NE2,IP,NODE)
DIMENSION IP(9,250)
COMMON/CNTL/ NP,NJ,NPT
C
DO 10 J=1,NE2
   DO 10 I=1,NE1
      I1=NPT+NE1*(J-1)+I
      IP(1,I1)=NJ+(J-1)*(NE1+1)+I
      IP(2,I1)=IP(1,I1)+1
      IP(3,I1)=IP(2,I1)+NE1+1
      IP(4,I1)=IP(3,I1)-1
   DO 10 K=5,NODE
10   IP(K,I1)=0
RETURN
END
SUBROUTINE COORDP(Y,X)
REAL MESCOR
DIMENSION Y(3,8),X(3,250),Z(3,4)
COMMON/CMTL/ NP,NJ,NPT
COMMON/MESH/ NE1,NE2,NE3,NNP,NNJ,NNPT

DO 90 L=1,NE3+1
   L1=L

CALCULATE COORDINATES OF FOUR CORNERS OF FACE

   DO 10 K=1,4
      DO 10 J=1,3
         Z(J,K)=MESCOR(Y(J,K),Y(J,K+4),NE3,L1)
   10

CHECK FOR FRONT FACE OF HEXAHEDRON

   IF(L.GT.1) GO TO 60

CALCULATE COORDINATES OF THE NODES ON TOP AND BOTTOM OF FACE

   IZ=0
   DO 20 T=1,NE1+1
      I1=NJ+IZ+I
      I2=T1+(NE1+1)*NE2
      I3=I
      DO 20 J=1,3
         X(J,I1)=MESCOR(Z(J,1),Z(J,2),NE1,I3)
      20      X(J,I2)=MESCOR(Z(J,4),Z(J,3),NE1,I3)

CALCULATE COORDINATES OF NODES ON SIDES OF FACE

   IF(NE2.LT.2) GO TO 90
   DO 40 I=2,NE2
      I1=NJ+IZ+(NE1+1)*I
      I2=I1-NE1
      I3=I
      DO 40 J=1,3
         X(J,I1)=MESCOR(Z(J,2),Z(J,3),NE2,I3)
      40      X(J,I2)=MESCOR(Z(J,1),Z(J,4),NE2,I3)

CALCULATE COORDINATES OF NODES ON INTERIOR OF FACE

   IF(NE1.LT.2) GO TO 90
   DO 50 I=2,NE2
      DO 50 K=2,NE1
         I1=NJ+IZ+(I-1)*(NE1+1)+K
         I2=I1-K+1
         I3=I2+NE1
      50   }
K1 = K
DO 50 J = 1, 3
50 X(J, I1) = MESCOR(X(J, I2), X(J, I3), NE1, K1)
GO TO 90
60 IZ = (NE1 + 1) * (NE2 + 1) + 2 * (L - 2) * (NE1 + NE2)
CHECK FOR BACK FACE OF HEXAHEDRON
IF (L .EQ. NE3 + 1) GO TO 20
GET COORDINATES OF NODES ON TOP AND BOTTOM OF HEXAHEDRON'S INTERIOR
DO 70 I = 1, NE1 + 1
   I1 = NJ + IZ + I
   I2 = I1 + NE1 + NE2 + NE2 - 1
   I3 = T
   DO 70 J = 1, 3
      Y(J, I1) = MESCOR(Z(J, 1), Z(J, 2), NE1, I3)
    70 X(J, I2) = MESCOR(Z(J, 4), Z(J, 3), NE1, I3)
   IF (NE2 .LT. 2) GO TO 90
CALCULATE COORDINATES OF NODES ON SIDES OF HEXAHEDRON'S INTERIOR
DO 80 I = 2, NE2
   I1 = NJ + IZ + NE1 + I + I - 1
   I2 = I1 - 1
   I3 = T
   DO 80 J = 1, 3
      X(J, I1) = MESCOR(Z(J, 2), Z(J, 3), NE2, I3)
    80 X(J, I2) = MESCOR(Z(J, 1), Z(J, 4), NE2, I3)
CONTINUE
RETURN
END
SUBROUTINE CORNPT(IN,Y)
INTEGER OUTPUT
REAL LIT
DIMENSION Y(3,8),LIT(1)
COMMON/DEV/ INPUT,OUTPUT
DATA LIT/IHY/

REQUEST CORNER POINT INFORMATION

KS=1

CHECK FOR BACK FACE

IF(IN.LE.0) KS=5
IF(IN.EQ.0) GO TO 50
INA=IN
IF(INA.LT.0) INA=1*INA
KS1=KS+1
KS2=KS+2

INPUT 3 CORNER POINTS FOR A PARALLELOGRAM

IF(INA.NE.3) GO TO 10
TYPE 90, KS,KS1,KS2
GO TO 20
10 KS3=KS+3

INPUT 4 CORNER POINTS FOR A QUADRILATERAL

TYPE 100, KS,KS1,KS2,KS3

READ COORDINATES OF THE CORNER POINTS

20 DO 30 I=KS,KS+INA-1
   TYPE 180,
30 ACCEPT 160, (Y(J,I),J=1,3)
   IF(INA.NE.3) GO TO 50

GENERATE FOURTH CORNER POINT OF PARALLELOGRAM

35 DO 40 J=1,3
40 Y(J,KS+3)=Y(J,KS)+Y(J,KS+2)-Y(J,KS+1)

CHECK CORNER POINTS AND MAKE CORRECTIONS

50 TYPE 120,
   DO 60 I=KS,KS+3
60 TYPE 170, I,(Y(J,I),J=1,3)
   TYPE 190,
   ACCEPT 140, ANS
LIT(1) = "Y"

IF(ANS.NF.LIT(1)) RETURN

**** CORRECTION ARE REQUIRED CHECK TO SEE WHAT

TYPE 130,
70 TYPE 180,
  ACCEPT 175, K1,X1,X2,X3
  IF(K1.EQ.0) GO TO 85
  IF(K1.LT.0 .OR. K1.GT.8) GO TO 80
  Y(1,K1)=X1
  Y(2,K1)=X2
  Y(3,K1)=X3
  GO TO 70
80 TYPE 150,
  GO TO 70

**** IF THIS A PARALELOGRAM RE - CALCULATE THE FOURTH CORNER POINT

85 IF(INA.GE.4) GO TO 50
  GO TO 35

90 FORMAT(1H/, ' < ENTER 3 COOR.(X,Y,Z) OF THE PARALLELOGRAM SEQUENTIAL
  $1/, ' CORNER POINTS > ( '"I3','I3','I3','I3')')
100 FORMAT(1H/, ' < ENTER 4 COOR(X,Y,Z) OF THE QUADRILATERAL
  $AL SEQUENTIAL
  $1/, ' CORNER POINTS > ( '"I3','I3','I3','I3','I3','I3')')
120 FORMAT( ' < THESE ARE THE ENTERED CORNER POINTS AND CL
  $ORDINATES>
130 FORMAT(1H/, ' < ENTER CORRECTION (NODE# X,Y,Z) ENTER
  $NODE=0
  $ TO FINISH CORRECTION LOOP>
140 FORMAT(A1)
150 FORMAT(34H+<INVALID CORNER POINT, TRY AGAIN>/)

SEE POSSIBLE INCOMPATIBILITIES (1)

160 FORMAT(3F10.0)
170 FORMAT(I,3F8.2)
175 FORMAT(I,3F10.0)
SEE POSSIBLE INCOMPATIBILITIES (2)

190 FORMAT(/6H>>> $)
190 FORMAT(/.16H+<CORRECTIONS?> $)
END
SUBROUTINE CROSS(L, I, J)
REAL L
DIMENSION L(3,3)

DX = L(2, I) * L(3, J) - L(3, I) * L(2, J)
DY = L(1, I) * L(3, J) + L(3, I) * L(1, J)
DZ = L(1, I) * L(2, J) - L(2, I) * L(1, J)
L(1, J) = DX
L(2, J) = DY
L(3, J) = DZ
RETURN
END
SUBROUTINE DISP(U,XJNK)
INTEGER OUTPUT
REAL LIT
DIMENSION U(3,250),LIT(6),XJNK(3,250)
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEV/ INPUT,OUTPUT
COMMON/MAXI/ NPMAX,NJMAX,NPTMAX
DATA LIT/1H ,4HREAD,4HWRLT,4HCHAN,4HPRIN,4HTRAN/
REQUEST COMMAND WORD, RETURN IF BLANK

10 WORD=CMD(2)
   LIT(1) = " "
   IF(WORD.EQ.LIT(1)) RETURN
   READ DISPLACEMENT FILE
   LIT(2) = "READ"
   IF(WORD.NE.LIT(2)) GO TO 20
   CALL RODISP(U)
   GO TO 10
   WRITE DISPLACEMENT FILE
   LIT(3) = "WRIT"
   20 IF(WORD.NE.LIT(3)) GO TO 30
   CALL WR_DISP(U,NJ)
   GO TO 10
   CHANGE DISPLACEMENTS
   LIT(4) = "CHAN"
   30 IF(WORD.NE.LIT(4)) GO TO 50
   TYPE 200,
   40 TYPE 260,
   ACCEPT 250, I,X1,X2,X3
   IF(I.EQ.0) GO TO 10
   IF(I.GT.NJMAX) CALL OVER(2)
   U(1,I)=X1
   U(2,I)=X2
   U(3,I)=X3
   GO TO 40
   PRINT DISPLACEMENTS ON TTY
LIT(5) = "PRIN"

IF(WOORD.NE.LIT(5)) GO TO 70

TYPE 270,
ACCEPT 240, I1, I2
IF(I1.EQ.0) GO TO 10
IF(I1.LT.0.0) I1=0
IF(I1.GT.NJ) GO TO 60
IF(I2.LT.I1) I2=I1
IF(I2.GT.NJ) I2=NJ
TYPE 280, (J, (U(I,J), I=1,3), J=I1,I2)
GO TO 60

READ TWO GEOMETRY FILES AND SET THE DISPLACEMENTS EQUAL TO THE DIFFERENCE IN THE COORDINATES

LIT(6) = "TRAN"

IF(WOORD.NE.LIT(6)) GO TO 190
NJ1=0
NJ2=0
CALL OPEN(5HGEOM1, IUNIT, IREAD, IERR)
IF(IFRR) 90, 100, 90
READ(IUNIT, 220) M1, NJ1, N2, N3
READ(IUNIT, 220) ((K, I=1,2), J=1,N1)
READ(IUNIT, 230) ((XJNK(I,J), I=1,3), J=1,NJ1)
CALL OPEN(5HGEOM2, IUNIT, IREAD, IERR)
IF(IFRR) 100, 120, 110
READ(IUNIT, 220) M1, NJ2, N2, N3
IF(NJ1.NE.0 .AND. NJ1.NE.NJ2) GO TO 180
READ(IUNIT, 220) ((K, I=1,2), J=1,N1)
READ(IUNIT, 230) ((U(I,J), I=1,3), J=1,NJ2)
IF(NJ1.EQ.0 .AND. NJ2.EQ.0) RETURN
NJ=NJ1
IF(NJ.LT.NJ2) NJ=NJ2
IF(NJ1.NE.0) GO TO 140
DO 130 J=1,NJ
   DO 130 I=1,3
130 XJNK(I,J)=0.0
IF(NJ2.NE.0) GO TO 160
DO 150 J=1,NJ
   DO 150 I=1,3
150 U(I,J)=0.0
DO 170 J=1,NJ
   DO 170 I=1,3
170 U(I,J)=XJNK(I,J)-U(I,J)
GO TO 10
TYPE 210,
GO TO 10
C TYPE HELP MESSAGE IF COMMAND NOT RECOGNIZED

190 CALL HELP(8)
   GO TO 10

200 FORMAT(33H+N ODE NUMBER AND DISPLACEMENTS> /)
210 FORMAT(37H+W ARNING: TRANSFORM COMMAND ABORT/>)
220 FORMAT(16I5)
230 FORMAT(6E12.5)

SEE POSSIBLE INCOMPATIBILITIES (1)

240 FORMAT(21)
250 FORMAT(7.3E5.0)

SEE POSSIBLE INCOMPATIBILITIES (2)

260 FORMAT(6H+>>>> $)
270 FORMAT(14H+NODE I1/I2> $)

SEE POSSIBLE INCOMPATIBILITIES (3)

280 FORMAT(1H+,I4,1P3E12.4/)  END
SUBROUTINE ELLIP(NPL, X, IPE, NTYPE, SHELL, NODE)
INTEGER OUTPUT
REAL MESCOR, LIT
LOGICAL SHELL

C SEE POSSIBLE INCOMPATIBILITIES (5)
C
DIMENSION XI(3),XO(3),C(3),X(3,250),NPL(2,20),
1IP(9,250),SIDE(6)
COMMON/DEVI/ INPUT,OUTPUT
COMMON/CNTL/ NP,NJ,NPT
COMMON/MESH/ NE1,NE2,NE3,NNP,NNJ,NNPT

C READ COORDINATES OF THE ELLIPSOID'S CENTER
C
TYPE 150,
ACCEPT 140, C(1),C(2),C(3)

C READ OUTER RADIUS OF THE ELLIPSOID
C
TYPE 160,
ACCEPT 140, AO,BO,CO
IF(BO.EQ.0.0) BO=AO
IF(C0.EQ.0.0) CO=BO

C CHECK TO SEE IF SHELL ONLY
C
IF(SHELL) GO TO 10

C READ INNER RADIUS OF ELLIPSOID
C
TYPE 180,
ACCEPT 140, AI,BI,CI
IF(BI.EQ.0.0) BI=AI
IF(CI.EQ.0.0) CI=BI

C READ STARTING AND ENDING LATITUDE
C
10 TYPE 190,
ACCEPT 140, XLATB,XLATE

C READ STARTING AND ENDING LONGITUDE
C
TYPE 170,
ACCEPT 140, XLONGB,XLONGE
ND=4
IF(SHELL) ND=5

C GET ELEMENTAL DIVISIONS AND CHECK MAXES
CALL MESHNV(ND, NTYP

GENERATE COORDINATE ARRAY

SET INITIALIZATION VALUES

DELONG=(XLONGE-XLONGB)/NE3
DELAT=(XLATE-XLATB)/NE2
XLONG=XLONGB-DELONG
NEXT=NJ
DO 80 I=1, NE3+1
  XLAT=XLATB-DELAT
  XLONG=XLONG+DELONG
  DO 80 J=1, NE2+1
    XLAT=XLAT+DELAT

GET INTERSECTION WITH OUTER ELLIPSOID

CALL INTERS(XLAT, XLONG, AO, BO, CO, XO(1), XO(2), X

$O(3)) IF(SHELL) GO TO 60

GET INTERSECTION WITH INNER ELLIPSOID

CALL INTERS(XLAT, XLONG, AI, BI, CI, XI(1), XI(2), X

$I(3)) IF(NTYPE.EQ.8) GO TO 20

CHECK TO SEE IF ON AN OUTER SURFACE

IF(J.EQ.1 OR J.EQ.NE2+1) GO TO 20
IF(I.EQ.1 AND I.NE.NE3+1) GO TO 40
20 DO 30 K=1, NE1+1
   K1=K
   NEXT=NEXT+1

INTERPOLATE BETWEEN INNER AND OUTER INTERSECTIONS

DO 30 M=1,3
  30 X(M,NEXT)=MESCOR(XI(M),XO(M),NE1,K1)+C(M)
GO TO 80

LOAD ONLY OUTER AND INNER INTERSECTIONS WHERE INTERIOR NODES ARE NOT NEEDED

40 DO 50 M=1,3
   50 X(M,NEXT+1)=XI(M)+C(M)
   NEXT=NEXT+2
GO TO 80

GET NODAL COORDINATES FOR SHELL

60    NEXT=NJ+(NE2+1)*I+1-J
    DO 70 M=1,3
70    X(M,NEXT)=XO(M)+C(M)
80    CONTINUE

GENERATE CONNECTIVITY ARRAY

IF(SHELL) GO TO 120

CONNECTIVITY FOR PANEL ELEMENTS

DO 90 I=1,8
90    SIDE(1)=0.0
   ALONG=ABS(XLONGE-XLONGB)
   IF(ALONG.NE.360.) GO TO 100
   SIDE(1)=-1
   SIDE(2)=-1
100   IF(ALI.EQ.0..AND.BI.EQ.0..AND.CI.EQ.0.) SIDE(3)=-1
   IF(XLATB.EQ.-90..OR.XLATB.EQ.90.) SIDE(5)=-1
   IF(XLATF.EQ.-90..OR.XLATF.EQ.90.) SIDE(6)=-1
   CALL CONNAR(IP,SIDE,NODE)
   GO TO 130

CONNECTIVITY FOR SHELL

120 CALL CONNSH(NE2,NE3,IP,NODE)

GENERATE PARTS ARRAY

130    NNPT=NNPT+(SIDE(1)+SIDE(2))*NE1*NE2+(SIDE(3)+SIDE(4)+
   (SIDE(5)+SIDE(6)))*NE1*NE3
   NPT1=NNPT-NPT
   NPL(1,NNP)=NNPT-NPT
   NPL(2,NNP)=0

UPDATE NP, NJ AND NPT

NP=NNP
NJ=NNJ
NPT=NNPT
RETURN

SEE POSSIBLE INCOMPATIBILITIES (1)

140 FORMAT(3F10.0)
C SEE POSSIBLE INCOMPATIBILITIES (2)
C
150 FORMAT(49H<COORDINATES (X,Y,Z) OF THE ELLIPSOID'S
$CENTER> $)
160 FORMAT(44H<OUTER RADII IN THE X, Y AND Z DIRECTION
$S> $)
170 FORMAT(31H<STARTING AND ENDING LONGITUDE,/
, 142H (MEASURED IN DEGREES FROM THE XY PLANE)> $)
180 FORMAT(44H<INNER RADII IN THE X, Y AND Z DIRECTION
$S> $)
190 FORMAT(30H<STARTING AND ENDING LATITUDE,/
, 142H (MEASURED IN DEGREES FROM THE XZ PLANE)> $)
END
SUBROUTINE GEOM(NPL,NPX,X,IP,NODE)
INTEGER OUTPUT
REAL LIT

C
C
C
C
C
C

DIMENSION NPL(2,20),X(3,250),IP(9,250),NPX(2,20)
DIMENSION II(20),JP(250),LPN(250),LIT(15),JJ(20)
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEV/ INPUT,OUTPUT
COMMON/FUNC/ II,LPN
COMMON/JUNK/ JP
COMMON/MAXI/ NPMAX,NJMAX,NPTMAX
DATA LIT/1H,4HREAD,4HWRIT,4HCHAN,4HMOVE,4HGROUP,
        1,4HCOPY,1HY,4HELEM,1HA,1HD,4HPRIN,4HSHIFT,4HSCALE,1HR/$
        NODE1=NODE-1

C
C
C
C

10 WORD=CMD(2)
C
   LIT(1) = " "
C
   IF(WORD.EQ.LIT(1)) RETURN
C
C
C
C
C
C

READ GEOMETRY FILE
C
   LIT(2) = "READ"
C
   IF(WORD.NE.LIT(2)) GO TO 20
   CALL RDGFOM(NPL,X,IP,JP,0,0,0,NODE)
   GO TO 10
C
C
C
C
C

WRITE GEOMETRY FILE
C
   LIT(3) = "WRIT"
C
   IF(WORD.NE.LIT(3)) GO TO 30
   CALL WRFOM(NPL,NPX,X,IP,JP,NODE)
   GO TO 10
C
C
C
C
C

CHANGE GEOMETRY FILE
C
   LIT(4) = "CHAN"
C
   IF(WORD.NE.LIT(4)) GO TO 550
   40 WORD=CMD(3)
C
   LIT(1) = " "
C
C IF(WORD.EQ.LIT(1)) GO TO 10
C
C MOVE ELEMENTS
C
C LIT(5) = "MOVE"
C
IF(WORD.NE.LIT(5)) GO TO 100
TYPE 680,
50 TYPE 850,
ACCEPT 830, I1, I2, I3
IF(I1.EQ.0) GO TO 40
IF(I2.EQ.1) GO TO 60
I5=I2-1
CALL MOVE(IP, JP, 1, I5, 1, node)
60 I5=I1+I2-1
J4=NPT-I1+1
CALL MOVE(IP, JP, I2, I5, J4, NODE)
I4=I1+I2
IF(I4.GT.NPT) GO TO 70
CALL MOVE(IP, JP, I4, NPT, I2, NODE)
70 IF(I3.EQ.0) GO TO 80
CALL MOVE(JP, IP, 1, I3, 1, node)
80 I4=I3+1
I5=NPT-I1
IF(I5.LT.I4) GO TO 90
J4=I1+I3+1
CALL MOVE(JP, IP, I4, I5, J4, NODE)
90 I4=NPT-I1+1
J4=I3+1
IF(I2.LT.I3) J4=J4-I1
CALL MOVE(JP, IP, I4, NPT, J4, NODE)
GO TO 50
C
C PART GROUPS
C
C LIT(6) = "GROUP"
C
100 IF(WORD.NE.LIT(6)) GO TO 190
NP1=NP
TYPE 870,
ACCEPT 830, NP
IF(NP.GT.NPMAX) CALL OVER(1)
TYPE 840,
ACCEPT 670, ANS
C
C LIT(8) = "Y"
C
IF(ANS.NE.LIT(8)) GO TO 170
NPT=0
DO 110 I=1,NP1
   NPL(2,I)=NPT+NPL(1,I)
   NPL(1,I)=NPT+1
110 NPT=NPL(2,I)
   TYPE 770,
   NPT=0
   DO 130 J=1,NP
      TYPE 860, J
      ACCEPT 830, (II(I),I=1,10)
   DO 120 I=1,10
      II=II(I)
      IF(II.EQ.0) GO TO 130
      NPT=NPT+1
      CALL MOVE(IP,JP,NPL(1,I1),NPL(2,I1),NPT,NODE)
120 NPT=NPT+NPL(2,I1)-NPL(1,I1)
130 LPN(J)=NPT
   CALL MOVE(IP,JP,NPL(1,I1),NPT,NODE)
140 NPL(1,1)=LPN(1)
   NPL(2,1)=0
   DO 150 J=2,NP
      NPL(1,J)=LPN(J)-LPN(J-1)
150 NPL(2,J)=0
      IF(NP.GE.NP1) GO TO 40
      II=NP+1
   DO 160 J=II,NP1
      NPL(1,J)=0
160 NPL(2,J)=0
      GO TO 40
170 TYPE 650,
   ACCEPT 830, (NPL(1,I),I=1,NP)
   NPT=0
   DO 180 I=1,NP
180 NPT=NPT+NPL(1,I)
      IF(NPT.GT.NPTMAX) CALL OVER(3)
      GO TO 40
C
C COORDINATES
C
C LIT(7) = "COOR"
C
C 190 IF(WORD.NE.LIT(7)) GO TO 220
   TYPE 880,
   ACCEPT 670, ANS
C
C LIT(8) = "Y"
C
C IF(ANS.NE.LIT(8)) GO TO 200
   TYPE 890,
   ACCEPT 830, NJ
   IF(NJ.GT.NJMAX) CALL OVER(2)
200 TYPE 780,
210 TYPE 850,
   ACCEPT 810, I, X1, X2, X3
IF(I.EQ.0) GO TO 40
IF(I.GT.NJMAX) CALL OVERC2)
IF(I.GT.NJ) NJ=I
X(1,I)=X1
X(2,I)=X2
X(3,I)=X3
GO TO 210
C
C LIT(13) = "SHIF"
C
220 IF(WORD.NE.LIT(13)) GO TO 240
   TYPE 900,
   ACCEPT 820, XC, YC, ZC
   DO 230 I=1,NJ
       X(1,I)=X(1,I)-XC
       X(2,I)=X(2,I)-YC
   230 X(3,I)=X(3,I)-ZC
   GO TO 40
C
C LIT(14) = "SCAL"
C
240 IF(WORD.NE.LIT(14)) GO TO 260
   TYPE 950,
   ACCEPT 830, XC
   DO 250 J=1,NJ
       DO 250 I=1,3
   250 X(I,J)=XC+X(I,J)
   GO TO 40
C
C ELEMENTS
C
C LIT(9) = "ELEM"
C
260 IF(WORD.NE.LIT(9)) GO TO 540
   TYPE 940,
   ACCEPT 830, NN
   IF(NN.EQ.0) GO TO 40
270 TYPE 910,
   ACCEPT 670, WORD
C
C LIT(10) = "A"
C
   IF(WORD.EQ.LIT(10)) GO TO 280
C
C LIT(11) = "D"
C
   IF(WORD.EQ.LIT(11)) GO TO 370
LIT(15) = "R"

IF(WORD.EQ.LIT(15)) GO TO 450
GO TO 270

ADD ELEMENTS

280 TYPE 690, NN
290 TYPE 850,
DO 300 I=NN,NODE1
300 II(I)=0
ACCEPT 830, J1,(II(I),I=1,NN),J4
IF(J1.EQ.0) GO TO 40
IF(J1.GT.NPMAX) GO TO 360
IF(J1.LT.0) GO TO 360
NP1=1
DO 310 I=1,J1
310 NP1=NP1+NPL(1,I)
IF(NP1.GT.NPMax) CALL OVER(C)
IF(NP1.GT.NPT) GO TO 340
J3=NPT+1
DO 330 J=NP1,NPT
   J2=J3-1
   DO 320 I=1,NODE
320   IP(I,J3)=IP(I,J2)
330 J3=J2
340 DO 350 I=1,NODE1
350   IP(I,NP1)=II(I)
   IP(NODE,NP1)=J4
   NPT=NPT+1
   NPL(1,J1)=NPL(1,J1)+1
   GO TO 290
360 TYPE 720,
   GO TO 290

DELETE ELEMENTS

370 TYPE 700, NN
380 TYPE 850,
ACCEPT 830, J1,(II(I),I=1,NN)
IF(J1.EQ.0) GO TO 40
IF(J1.GT.NPMax) GO TO 420
IF(J1.LT.0,0) GO TO 420
NP2=0
DO 390 I=1,J1
390 NP2=NP2+NPL(1,I)
   NP1=NP2-NPL(1,J1)+1
   DO 410 J=NP1,NP2
      J2=J+1
DO 400 I=1,NN
400 IF(IP(T,J).NE.II(I)) GO TO 410
   GO TO 430
410 CONTINUE
   TYPE 730,
   GO TO 380
420 TYPE 740,
   GO TO 380
430 DO 440 I=J2,NPT
   J3=I-1
   DO 440 J=1,NODE
440 TP(J,J3)=IP(J,I)
   NPT=NPT-1
   NPL(I,J1)=NPL(I,J1)-1
   GO TO 380

REPLACE ELEMENTS

450 TYPE 660, NN, NN
460 TYPE 850,
   ACCEPT 830, J1, (II(I), I=1, NN), (JJ(I), I=1, NN), J4
   IF(J1.EQ.0) GO TO 40
   IF(J1.GT."NMAX") GO TO 500
   IF(J1.LT.0) GO TO 500
   NP2=0
   DO 470 I=1,J1
470 NP2=NP2+NPL(I,I)
   NP1=NP2-NPL(I,J1)+1
   DO 490 J=NP1,NP2
      J2=J+1
      DO 480 I=1,NN
480 IF(IP(T,J).NE.II(I)) GO TO 490
      GO TO 510
490 CONTINUE
   TYPE 730,
   GO TO 460
500 TYPE 740,
   GO TO 460
510 DO 520 I=NN,NODE1
520 IF(I,J)=0
   DO 530 I=1,NN
530 IF(I,J)=JJ(I)
   IF(NODE,J)=J4
   GO TO 460

TYPE HELP MESSAGE IF COMMAND NOT RECOGNIZED

540 CALL HELP(4)
   GO TO 40
PRINT DATA ON TTY

LIT(12) = "PRIN"

550 IF(WORD.NE.LIT(12)) GO TO 640
560 WORD=CMD(3)

LIT(1) = ""

IF(WORD.EQ.LIT(1)) GO TO 10

PART GROUPS

LIT(6) = "GROUP"

IF(WORD.NE.LIT(6)) GO TO 570
TYPE 750, (NPL(1,I),I=1,NP)
TYPE 710,
GO TO 560

COORDINATES

LIT(7) = "COORD"

570 IF(WORD.NE.LIT(7)) GO TO 590
580 TYPE 920,
ACCEPT 830, I1,I2
IF(I1.LE.0) GO TO 560
IF(I1.GT.MJ) GO TO 580
IF(I2.LT.I1) I2=I1
IF(I2.GT.MJ) I2=MJ
TYPE 790,
TYPE 960, (J,(X(I,J),I=1,3),J=I1,I2)
GO TO 580

ELEMENTS

LIT(9) = "ELEM"

590 IF(WORD.NE.LIT(9)) GO TO 630
600 TYPE 930,
ACCEPT 830, I1,I2
IF(I1.LE.0) GO TO 560
IF(I1.GT.MPT) GO TO 600
IF(I2.LT.I1) I2=I1
IF(I2.GT.MPT) I2=MPT
TYPE 800,
DO 620 J=I1,I2
   DO 610 I=1,NODE1
      IF(IP(I,J),NE,0) GO TO 610
NN=I-1
GO TO 620
610 CONTINUE
NN=NODE1
620 TYPE 760, J, (IP(I,J),I=1,NN);IP(NODE,J)
TYPE 710,
GO TO 600
C
TYPE HELP MESSAGE IF COMMAND NOT RECOGNIZED
C
630 CALL HELP(3)
GO TO 560
C
TYPE HELP MESSAGE IF COMMAND NOT RECOGNIZED
C
640 CALL HELP(2)
GO TO 10
C
650 FORMAT(31H<NUMBER OF ELEMENTS IN GROUPS>)
660 FORMAT(8H<GROUP;I3,11H OLD NODES;I3;
125H NEW NODES; CENTRAL NODE>)
670 FORMAT(A1)
680 FORMAT(30H<MOVE I1 ELEMENTS STARTING AT;
125H I2 TO FOLLOW ELEMENT I3>)
690 FORMAT(8H<GROUP;I3,21H NODES; CENTRAL NODE>)
700 FORMAT(8H<GROUP;I3,7H NODES>)
710 FORMAT(1H )
720 FORMAT(23H<ILLEGAL GROUP NUMBER> )
730 FORMAT(30H<WARNING: ELEMENT NOT THERE>)
740 FORMAT(16H<NO SUCH GROUP>)
750 FORMAT(21H<ELEMENT GROUP LIST>/(1X,10I4))
760 FORMAT(1H ,1515)
770 FORMAT(40H<NEW GROUP NUMBER = OLD GROUP NUMBERS> /$
780 FORMAT(31H<NODE NUMBER AND COORDINATES> )
790 FORMAT(5H<NODE;4X,7HX-COOR,...,5X,7HY-COOR,...,5X,7HZ-COO
$R.>)
800 FORMAT(44H<ELEM. NODE1 NODE2 ... CENTER NODE (OR ZE
$RO))
C
SEE POSSIBLE INCOMPATIBILITIES (1)
C
810 FORMAT(1,3F10.0)
820 FORMAT(3F10.0)
830 FORMAT(20I)
C
SEE POSSIBLE INCOMPATIBILITIES (2)
C
840 FORMAT(23H<REORDER OLD GROUPS?> $)
850 FORMAT(6H<<<< $)
163

860 FORMAT(8H+<GROUP> ,I3,5H = > $)
870 FORMAT(2AH+<NUMBER OF ELEMENT GROUPS> $)
890 FORMAT(27H+<CHANGE NUMBER OF NODES?> $)
890 FORMAT(19H+<NUMBER OF NODES> $)
900 FORMAT(24H+<NEW ORIGIN (X, Y, Z)> $)
910 FORMAT(27H+<ADD, DELETE, OR REPLACE> $)
920 FORMAT(14H+<NODE I1/I2> $)
930 FORMAT(17H+<ELEMENT I1/I2> $)
940 FORMAT(19H+<ELEMENT I1/I2> $)
950 FORMAT(27H+<COORDINATE SCALE FACTOR> $)

C SEE POSSIBLE INCOMPATIBILITIES (3)
C
960 FORMAT(1H+,I4,1P3E12.4/)
END
SUBROUTINE HELP(INDX)
INTEGER OUTPUT
COMMON/DEVIC/ INPUT,OUTPUT
C
C JUMP TO APPROPRIATE HELP MESSAGE
C
GO TO (10,20,30,40,50,60,70,80),INDX
C
LEVEL ONE HELP MESSAGE
C
10 TYPE 90,
RETURN
C
LEVEL TWO HELP MESSAGE (ALL BUT MAKE AND DISP)
C
20 TYPE 100,
RETURN
C
LEVEL THREE HELP MESSAGE (ALL BUT GEOM-CHAN AND MAKE)
C
30 TYPE 110,
RETURN
C
LEVEL THREE HELP MESSAGE (GEOM-CHAN ONLY)
C
40 TYPE 120,
RETURN
C
LEVEL TWO HELP MESSAGE (MAKE ONLY)
C
50 TYPE 130,
RETURN
C
LEVEL THREE HELP MESSAGE (MAKE-SHEETS ONLY)
C
60 TYPE 140,
RETURN
C
LEVEL THREE HELP MESSAGE (MAKE-VOLUMES ONLY)
C
70 TYPE 150,
RETURN
C
LEVEL TWO HELP MESSAGE (DISP ONLY)
C
80 TYPE 160,
RETURN
C
90 FORMAT(44H<GEOMETRY, DISPLACEMENT, FUNCTION, SYMME
STRY,/)
142H ORDER, MAKE, MERGE, HELP, CLEAN, OR EXIT>
100 FORMAT(38H<READ, WRITE, CHANGE, PRINT, OR EXIT>/)
110 FORMAT(38H<GROUP, ELEMENT, COORDINATE, OR EXIT>/)
120 FORMAT(49H<GROUP, ELEMENT, COORDINATE, MOVE, SHIFT $, SCALE,$ 19H OR EXIT>/)
130 FORMAT(26H<SHEETS, VOLUMES OR EXIT>/)
140 FORMAT(38H<PARALLELOGRAM, QUADRILATERAL, SHELL,, 120H REVOLUTION OR EXIT>/)
150 FORMAT(44H<HEXAHEDRON, ELLIPSOID, REVOLUTION OR EXIT$IT>/)
160 FORMAT(49H<READ, WRITE, CHANGE, PRINT, TRANSFORM,$ OR EXIT>/)
END
SUBROUTINE HEXGEN(NPL, X, IP, NTYPE, NODE)
INTEGER OUTPUT
REAL LIT, MESCOR

SEE POSSIBLE INCOMPATIBILITIES (5)

DIMENSION NPL(2,20), X(3,250), IP(9,250), Y(3,8)
1, LIT(1), NN(8), SIDE(6)
COMMON/DEVI/ INPUT, OUTPUT
COMMON/CNTL/ NP, NJ, NPT
COMMON/MESH/ NE1, NE2, NE3, NNP, NNJ, NNPT
DATA LIT/1HY/

REQUEST FRONT FACE INFORMATION

TYPE 70,
TYPE 120,
ACCEPT 90, ANS
IN=3

LIT(1) = "Y"

IF(ANS.NF,LIT(1)) IN=4
CALL CORMPT(IN,Y)

REQUEST BACK FACE INFORMATION

TYPE 130,
ACCEPT 90, ANS

LIT(1) = "Y"

IF(ANS.NF,LIT(1)) GO TO 20

GENERATE CORNER POINTS OF PRISMATIC HEXAHEDRON

TYPE 110,
ACCEPT 100, (Y(J,5), J=1,3)
DO 10 I=6,8
   DO 10 J=1,3
10 Y(J,I)=Y(J,I-4)+Y(J,5)-Y(J,1)
   IN=0
   CALL CORMPT(IN,Y)
   GO TO 30

READ BACK FACE CORNER POINTS

20 TYPE 80,
TYPE 120,
ACCEPT 90, ANS
IN=-3
LIT(1) = "Y"
IF(ANS,NF,LIT(1)) IN=-4
CALL CORNPT(IN,Y)

GENERATE MESH AND CHECK NP_MAX, NJ_MAX AND NPT_MAX

30 ND=3
CALL MESHDV(ND,NTYPE)

PANEL GEOMETRY

GENERATE COORDINATE ARRAY
CALL COORDP(Y,X)

GENERATE CONNECTIVITY ARRAY

DO 40 I=1,8
40 SIDE(I)=0.0
CALL CONNAR(IP,SIDE,NODE)
GO TO 60

GENERATE PARTS ARRAY

60 NPL(1,NNP)=NNPT-NPT
NPL(2,NNP)=0

UPDATE NP, NJ AND NPT

NP=NNP
NJ=NNJ
NPT=NNPT
RETURN

70 FORMAT(25H+FRONT FACE INFORMATION/>)
80 FORMAT(24H+BACK FACE INFORMATION/>)
90 FORMAT(A1)

SFE POSSIBLE INCOMPATIBILITIES (1)
100 FORMAT(3F10.0)

SEE POSSIBLE INCOMPATIBILITIES (2)
110 FORMAT(41H+COORDINATES (X,Y,Z) OF CORNER POINT 5>
$$)
120 FORMAT(33H+IS THIS FACE A PARALLELOGRAM?> $$)
130 FORMAT(32H+"IS THE HEXAHEDRON PRISMATIC?" $)
END
SUBROUTINE INTERS(XLAT, XLONG, A, B, C, X, Y, Z)

XLATR = XLAT * 3.1416 / 180.
XLONGR = XLONG * 3.1416 / 180.

GET SINES AND COSINES OF LATITUDE AND LONGITUDE

COLAT  = COS(XLATR)
COLONG = COS(XLONGR)
SILAT  = SIN(XLATR)
SILONG = SIN(XLONGR)
A2 = A * A
B2 = B * B
C2 = C * C

CHECKS TO AVOID DIVIDING BY ZERO

IF (A .EQ. 0 . . AND. B .EQ. 0 . . AND. C .EQ. 0.) GO TO 40
ALAT = ABS(XLAT)
IF (ALAT .EQ. 90 . . OR. ALAT .EQ. 270 . . ) GO TO 10
YN = SILAT / COLAT
ALONG = ABS(XLONG)
IF (ALONG .EQ. 90 . . OR. ALONG .EQ. 270 . . ) GO TO 20

CALCULATE COORDINATES AT INTERSECTION POINT

XM = SILONG / COLONG
C1 = COLONG * COLONG
Y = SQRT(X2)
Y = X * XM
Z = ABS(Y)
Z = ABS(Z)
GO TO 30

COORDINATES WHEN LATITUDE = 90 DEGREES

10 X = 0.0
Y = B
Z = 0.0
GO TO 30

COORDINATES WHEN LONGITUDE = 90 DEGREES

Z = SQRT(Z2)
Y = Z * XM
X = 0.0
Y = ABS(Y)
ASSIGN PROPER SIGN TO COORDINATES

C

30 IF(COLONG.LT.0.0) X=-X
   IF(SILAT.LT.0.0) Y=-Y
   IF(SILONG.GT.0.0) Z=-Z
   RETURN
40 X=0.
   Y=0.
   Z=0.
   RETURN
END
SUBROUTINE LOAD(MN,IP,NEXT,I1,I2,I3,I4,NODE)
DIMENSION MN(250),IP(9,250)

C
IP(1,NEXT)=MN(I1)
IP(2,NEXT)=MN(I2)
IP(3,NEXT)=MN(I3)
IP(4,NEXT)=MN(I4)
DO 10 I=5,NODE

10 IP(I,NEXT)=0
NEXT=NEXT+1
RETURN
END
SUBROUTINE MAKE(NPL,X,IP,NODE,U,S,IG,INEW)
INTEGER OUTPUT
LOGICAL LOOP,SHELL
REAL LIT

SEE POSSIBLE INCOMPATIBILITIES (5)

DIMENSION NPL(2,20),X(3,250),IP(9,250),IG(250),INEW(250)

DIMENSION LIT(10)
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEVI/ INPUT,OUTPUT
DATA LIT/'1H', '4HSHEE', '4HVOLU', '4HPARA', '4HQUAD', '4HELLI',
'1', '4HHEXA', '4HREVO', '4HSHEL', '1HY'/

CHECK FOR SOLID DATA
NTYPE=4

C*** ASK FOR SHEET OR VOLUME OF REVOlUTION

10 TYPE 15
15 FORMAT('1H',*')< WILL THIS BE A "SHEET OR VOLUME " TY
$PE >')
WORD=CMD(2)

C*** RETURN IF ANSWER IS BLANK

IF(WORD.EQ.LIT(1))RETURN

C*** IF THE ANSWER IS NOT SHEET OR VOLUME ASK AGAIN

IF(WORD.EQ.LIT(3))GO TO 90
IF(WORD.NE.LIT(2))GO TO 10

C*** NOW ASK FOR PARALLELOGRAM, QUADRILATERAL, SHELL OR

REVOLUTION

20 TYPE 25
25 FORMAT('1H',*'< PARA, QUAD, SHELL OR REVOL >')
WORD=CMD(3)
C LIT(1)=" ",-" 
C IF(WORD.EQ.LIT(1)) GO TO 10
C LIT(4)="PARA"
C 30 IF(WORD.NE.LIT(4)) GO TO 40
IN=3
CALL QUADGN(NPL,X,IP,IN,NODE)
GO TO 150
C LIT(5)="QUAD"
C 40 IF(WORD.NE.LIT(5)) GO TO 50
IN=4
CALL QUADGN(NPL,X,IP,IN,NODE)
GO TO 150
C LIT(9) = "SHEL"
C 50 IF(WORD.NE.LIT(9)) GO TO 60
SHELL=.TRUE.
CALL ELLIP(NPL,X,IP,NTYPE,SHELL,NODE)
GO TO 150
C LIT(8) = "REVO"
C 60 IF(WORD.NE.LIT(8)) GO TO 70
LOOP=.FALSE.
CALL REVOLV(NPL,X,IP,LOOP,NODE)
GO TO 150
C CREPEAT IF COMMAND NOT RECOGNIZED
C 70 GO TO 20
C CREATE VOLUME GEOMETRY
C LIT(3) = "VOLU"
C 90 TYPE 95
95 FORMAT(1H 0, ' < ENTER (HEXA, ELLI OR REVO) >')
WORD=CMD(3)
C LIT(1) = " ",-" 
C IF(WORD.EQ.LIT(1)) GO TO 10
LIT(7) = "HEXA"

IF(WORD.NE.LIT(7)) GO TO 100
CALL HEXGEN(NPL,X,IP,NTYPE,NODE)
GO TO 160

LIT(6) = "ELLI"

100 IF(WORD.NE.LIT(6)) GO TO 110
SHELL=.FALSE.
CALL FLLIP(NPL,X,IP,NTYPE,SHELL,NODE)
GO TO 160

LIT(8) = "REVO"

110 IF(WORD.NE.LIT(8)) GO TO 130
120 LOOP=.TRUE.
CALL REVOLV(NPL,X,IP,LOOP,NODE)
GO TO 160

ASK AGAIN IF COMMAND NOT RECOGNIZED

130 GO TO 90
150 CALL CLEAN(IP,X,U,S,NPL,IG,INEW,NODE,1)
TYPE 190, NP,NJ,NPT
GO TO 20
160 CALL CLEAN(IP,X,U,S,NPL,IG,INEW,NODE,1)
TYPE 190, NP,NJ,NPT
GO TO 90

170 FORMAT(4AH<ERROR: SOLID DATA NOT POSSIBLE IN MAKE-SHEETS>)
180 FORMAT(32H<ERROR: SOLID DATA NOT POSSIBLE, 128H IN MAKE-VOLUMES-REVOLUTION>)
190 FORMAT(18H<CURRENT TOTALS: ,I3,7H PARTS,, I4,7H NODES,, I4,10H ELEMENTS>)
200 FORMAT(A1)

SEE POSSIBLE INCOMPATIBILITIES (2)

END
SUBROUTINE MERGE(NPL,X,IP,JP,U,S,NODE)
INTEGER OUTPUT
C
SEE POSSIBLE INCOMPATIBILITIES (5)
C
DIMENSION NPL(2,20),X(3,250),IP(9,250),U(3,250),JP(250)
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEV/ INPUT,OUTPUT
COMMON/ERROR/ IERR
C
SET POINTERS TO ONE PAST END OF CURRENT FILES
C
10 NP1=NP+1
NJ1=NJ+1
NPT1=NPT+1
NP2=NP
NJ2=NJ
NPT2=NPT
C
READ GEOMETRY FILE
C
CALL RDGEOM(NPL(1,NP1),X(1,NJ1),IP(1,NPT1),JP(1,NP2,NJ2,NPT2,NODE))
C
IF NO FILE READ, THEN PREPARE TO EXIT
C
IGEO=0
IF(IERR.EQ.0) GO TO 70
IGEO=-1
C
READ DISPLACEMENT FILE
C
CALL RDDISP(U(1,NJ1))
NJ2=NJ1+NJ-1
C
IF NO FILE READ, THEN ZERO DISPLACEMENTS
C
IF(IERR.NE.0) GO TO 50
DO 20 J=NJ1,NJ2
   DO 20 I=1,3
20   U(I,J)=0.0
C
RENUMBER ELEMENT NODES
C
50 NPT2=NPT+NPT1-1
NJ1=NJ1-1
DO 60 J=NPT1,NPT2
   DO 60 I=1,NODE
60 IF(IP(I,J).NE.0) IP(I,J)=IP(J,J)+NJ1
NP2 = NP1 + NP - 1
C
C   RESET CONTROL VARIABLES
C
70   NP = NP2
     MJ = NJ2
     NPT = NPT2
     IF (IGEO .NE. 0) GO TO 10
     RETURN
     END
REAL FUNCTION MESCOR(XB,XE,NR,I)
A=I
B=NR

C AVOID DIVIDING BY ZERO
C
IF(NR.EQ.0) B=1.
WT=(A-1.)/B
MESCOR=(1.-WT)*XB+WT*X
RETURN
END
SUBROUTINE MESHDV(ND, NTYPE)
INTEGER OUTPUT
COMMON/CNTL/ MP, NJ, NPT
COMMON/DEV/ INPUT, OUTPUT
COMMON/MAI/ NPMAX, NJMAX, NPTMAX
COMMON/MESH/ NE1, NE2, NE3, NNP, NNJ, NNPT

C
C
C**** INITIALIZE THE NUMBER OF ELEMENTS
C
C
NE1=0
NE2=0
NE3=0
NC=ND-1
GO TO (10, 20, 30, 40), NC

C
C
READ ELEMENTAL DIVISIONS FOR QUADRILATERAL
C
10 TYPE 90,
ACCEPT 80, NE1, NE2
GO TO 50

C
C
READ ELEMENTAL DIVISIONS FOR HEXAHEDRON
C
20 TYPE 100,
ACCEPT 80, NE1, NE2, NE3
GO TO 50

C
C
READ ELEMENTAL DIVISIONS FOR ELLIPSOID VOLUME
C
30 TYPE 110,
ACCEPT 80, NE1, NE2, NE3
GO TO 50

C
C
READ ELEMENTAL DIVISIONS FOR ELLIPSOID SHELL
C
40 TYPE 120,
ACCEPT 80, NE2, NE3
C
C
IF NOTHING READ, SET Defaults
C
50 IF(NE1.LT.1) NE1=1
   IF(NE2.LT.1) NE2=1
   IF(NE3.LT.1) NE3=1
   IF(ND.EQ.5) NE1=0
IF(ND.EQ.2) NE3=0

CHECK NPMAK, NJMAX AND NPTMAX

IF(ND.EQ.4) ND=3
NNP=NP+1
IF(NNP.GT.NPMAK) CALL OVER(1)
NNJ=NNJ+(NE1+1)*(NE2+1)*(NE3+1)
IF(NTYPE.NE.4.OR.ND.NE.3) GO TO 60
NNJ=NNJ-(NE1-1)*(NE2-1)*(NE3-1)
NPT=NPT+2*(NE1*NE2*NE3+NE1*NE3)
GO TO 70
60 NPT=NPT+NE1*NE2*NE3
IF(ND.EQ.3) NPT=NPT+NE1*NE2*NE3
70 IF(NNJ.GT.NJMAX) CALL OVER(2)
IF(NPT.GT.NPTMAX) CALL OVER(3)
RETURN

SEE POSSIBLE INCOMPATIBILITIES (1)

80 FORMAT(3I)

SEE POSSIBLE INCOMPATIBILITIES (2)

90 FORMAT(43H<NUMBER OF ELEMENTS BETWEEN CORNER POINT $S:,
  134H 1 AND 2 (I1), AND 1 AND 4 (I2)> $)
100 FORMAT(43H<NUMBER OF ELEMENTS BETWEEN CORNER POINT $S:,
  14AH 1 AND 2 (I1), 1 AND 4 (I2), AND 1 AND 5 (I3)> $)
110 FORMAT(35H<NUMBER OF ELEMENTS: RADially (I1),/$
  143H IN LATITUDE (I2), AND IN LONGITUDE (I3)> $)
120 FORMAT(21H<NUMBER OF ELEMENTS:/$
  143H IN LATITUDE (I1), AND IN LONGITUDE (I2)> $)
END
SUBROUTINE MOVE(IP, IQ, L, M, N, NODE)
DIMENSION IP(9, 250), IQ(9, 250)
C
MOVE ELEMENTS L THRU M OF IP TO IQ STARTING AT N+1
C
J1 = N - L
DO 10 J = L, M
   J2 = J + J1
   DO 10 I = 1, NODE
  10 IQ(I, J2) = IP(I, J)
C
RETURN
END
SUBROUTINE NODEN(N,I,J,K)
COMMON/CNTL/ MP,NJ,NPT
COMMON/MESH/ NE1,NE2,NE3
CALCULATE NUMBER OF NODES ALONG EACH AXIS
ME1=ME1+1
ME2=ME2+1
ME3=ME3+1
NC=NJ
SEE IF ON FRONT FACE OF HEXAHEDRON
IF(K.NE.1) GO TO 20
10 N=NC+ME1*(J-1)+I
RETURN
20 NC=NJ+ME1*ME2+2*(K-2)*(NE1+NE2)
SEE IF ON BACK FACE OF HEXAHEDRON
IF(K.EQ.ME3) GO TO 10
SEE IF ON LEFT SIDE OF HEXAHEDRON
IF(J.NE.1) GO TO 30
N=NC+I
RETURN
SEE IF ON RIGHT SIDE OF HEXAHEDRON
30 IF(J.NE.ME2) GO TO 40
N=NC+2*(ME1+ME2)-ME1+1
RETURN
SEE IF ON BOTTOM OF HEXAHEDRON
40 IF(I.NE.1) GO TO 50
N=NC+ME1+J+J-3
RETURN
SEE IF ON TOP OF HEXAHEDRON
50 IF(I.NE.ME1) RETURN
N=NC+ME1+J+J-2
RETURN
END
SUBROUTINE NODEP(NN,I,J,K)
DIMENSION NN(A)

J1=I+1
J2=J+1
K1=K+1
CALL NODEP(NN(1),I,J,K)
CALL NODEP(NN(2),I1,J,K)
CALL NODEP(NN(3),I1,J1,K)
CALL NODEP(NN(4),I,J1,K)
CALL NODEP(NN(5),I,J,K1)
CALL NODEP(NN(6),I1,J,K1)
CALL NODEP(NN(7),I1,J1,K1)
CALL NODEP(NN(8),I,J1,K1)
RETURN
END
SUBROUTINE OPEN(FILEID, IUNIT, IOP, IERROR)
INTEGER OUTPUT
LOGICAL*1 XNAME(20)
DOUBLE PRECISION LIT(2)
COMMON/DEV1/ INPUT, OUTPUT
DATA LIT/*'READ', 'WRITE'*/

C IERROR=0
C C LIT(1) = "READ"
C C OPTYPE=LIT(1)
C C LIT(2) = "WRITE"
C
IF(IOP.LT.0) OPTYPE=LIT(2)
TYPE 20, OPTYPE, FILEID
ACCEPT 11, NCNT, (XNAME(I), I=1, NCNT)
IF(NCNT.EQ.0) RETURN
IERRO=-1
C
UNIT=10
XNAME(NCNT+1)=0
CLOSE(UNIT=IUNIT, ERR=4)
IF(IOP.LE.0) OPEN(UNIT=IUNIT, NAME=XNAME, TYPE='NEW', $ERR=5)
IF(IOP.GT.0) OPEN(UNIT=IUNIT, NAME=XNAME, TYPE='OLD', $ERR=5)
IERRO=1
RETURN
5 TYPE 6, (XNAME(I), I=1, NCNT)
IERRO=-1
RETURN
6 FORMAT(' OPEN FAILURE ON ',20A1)
C C
11 FORMAT(0,20A1)
C C
20 FORMAT(' < ENTER 'A5,1X,A4,' FILE >', $)
END
SUBROUTINE ORDER(NPL, IP, JP, NODE)
INTEGER OUTPUT

SEE POSSIBLE INCOMPATIBILITIES (5)

DIMENSION NPL(2,20), IP(9,250), JP(250)
COMMON/CNTL/ NP, NJ, NPT
COMMON/DEVI/ INPUT, OUTPUT

Determine number of edges in each element

NODE1=NODE-1
DO 20 I=1,NPT
   DO 10 J=1,NODE1
      J1=IP(J,I)
      IF(J1.EQ.0) GO TO 20
   10   JP(I)=J
20  CONTINUE

Process each part individually

M2=0
DO 120 N=1,NP
   M1=M2+1
   MP=M2+NPL(1,N)
   M3=M1+1

Compare the Nth thru last element with those already ordered

DO 110 M=M3,M2
   L1=M-1
   L2=L1+M1
   IPT=M
   JH=JP(IPT)

Search the previously ordered polygons backwards

30  DO 80 L3=M1,L1
    L=L2-L3

Search for a corresponding node number then check line segment forward and backward

   KH=JP(L)
   DO 70 K=1,KH
      KL=IP(K,L)
   70  DO 60 J=1,JH
       IF(IP(J,IPT).NE.KL) GO TO 60
J1=J+1
K1=K+1
IF(J1.GT.JH) J1=1
IF(K1.GT.KH) K1=K1
J2=J-1
IF(J2.LT.1) J2=JH
K2=K-1
IF(K2.LT.1) K2=KH
IF(IP(J1,IPT).EQ.IP(K1,L).OR.
IP(J2,IPT).EQ.IP(K2,L)) GO TO 40
IF(IP(J2,IPT).NE.IP(K1,L).AND.
IP(J1,IPT).NE.IP(K2,L)) GO TO 80
GO TO 90

C REVERSE POLYGON NODES IF NOT CONSISTENT WITH PROCESSED DATA
C
40
N1=1
N2=JH
ITEMP=IP(N1,IPT)
IP(N1,IPT)=IP(N2,IPT)
IP(N2,IPT)=ITEMP
N1=N1+1
N2=N2-1
IF(N1.LT.N2) GO TO 50
GO TO 90

70 CONTINUE
90 CONTINUE

C IF CURRENT POLYGON DOES NOT MATCH ANY PREVIOUS POLYGON,
C MOVE POINTER TO NEXT POLYGON AND TRY AGAIN
C
C IPT=IPT+1
JH=JP(IPT)
IF(IPT.LE.M2) GO TO 30
C
C IF POINTER IS GREATER THAN THE NUMBER OF ELEMENTS,
C THEN
C THIS POLYGON HAS NO NEIGHBOR
C
C TYPE 130, M
C GO TO 110
C
C IF THE LAST ORDERED POLYGON IS NOT THE SAME AS THE CURRENT
C POLYGON, THEN EXCHANGE THEM
C
90 IF(IPT.EQ.M) GO TO 110
DO 100 I=1, NODE
   ITEMP=IP(I,M)
   TP(I,M)=IP(I,IPT)
100   TP(I,IPT)=ITEMP
110   CONTINUE
120   CONTINUE
   RETURN
130 FORMAT(18H<WARNING: ELEMENT, IS, 18H HAS NO NEIGHBOR $3$/)
END
SUBROUTINE OVER(INDX)
INTEGER OUTPUT
COMMON/DEV1/ INPUT, OUTPUT
COMMON/MAX1/ NPMAX, NJMAX, NPTMAX

C C JUMP TO APPROPRIATE ERROR MESSAGE
C C GO TO (10, 20, 30), INDX
C C ERROR: NP .GT. NPMAX
C
10 TYPE 40, NPMAX
STOP
C C ERROR: NJ .GT. NJMAX
C
20 TYPE 50, NJMAX
STOP
C C ERROR: NPTMAX .GT. NPTMAX
C
30 TYPE 60, NPTMAX
STOP
C
40 FORMAT(35H+<ERROR: ATTEMPT TO EXCEED NPMAX OF, I5, 1H $$$>/)
50 FORMAT(35H+<ERROR: ATTEMPT TO EXCEED NJMAX OF, I5, 1H $$$>/)
60 FORMAT(36H+<ERROR: ATTEMPT TO EXCEED NPTMAX OF, I5, 1
$H$$>/)
END
SUBROUTINE QUADGN(NPL, X, IP, IN, NODE)
DIMENSION NPL(2, 20), X(3, 250), IP(9, 250), Y(3, A)
COMMON/CNTRL/ NP, NJ, NPT
COMMON/MESH/ NE1, NE2, NE3, NNP, NNJ, NNPT

REQUEST COORD. INFORMATION

CALL CORNPT(IN, Y)

GENERATE MESH AND CHECK NPMAX, NJMAX AND NP MAX

ND=2
CALL MESHDV (ND, 4)

GENERATE COORDINATE ARRAY

CALL COORDP(Y, X)

GENERATE CONNECTIVITY ARRAY

CALL CONNSH(NE1, NE2, IP, NODE)

GENERATE PARTS ARRAY

NPL(1, NNP)=NE1*NE2
NPL(2, NNP)=0

CHANGE NP, NJ AND NPT TO NEW VALUES

NP=NNP
NJ=NNJ
NPT=NNPT
RETURN
END
SUBROUTINE RODISP(U)
INTEGER OUTPUT
DIMENSION U(3,250)
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEVI/ INPUT,OUTPUT
COMMON/ERROR/ IERR
DATA IREAD/1/

C REQUEST FILE INFORMATION
C
10 CALL OPEN(5HDISP.,IUNIT,IREAD,IERR)
   IF(IERR) 10,30,20
C
C READ DISPLACEMENTS
C
20 READ(IUNIT,40) ((U(I,J),I=1,3),J=1,NJ)
30 RETURN
C
40 FORMAT(6F12.5)
END
SUBROUTINE ROGEOM(NPL,X,IP,JP,NP2,NJ2,NPT2,NODE)
INTEGER OUTPUT
REAL LIT
C
C SEE POSSIBLE INCOMPATIBILITIES (5)
C
DIMENSION NPL(2,20),X(3,250),IP(9,250),LIT(1),JP(25)
$0
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEVI/ INPUT,OUTPUT
COMMON/ERR0/ IERR
COMMON/MAXI/ NMAX,NJMAX,NPTMAX
DATA IREAD/1/
DATA LIT/1HY/
C
REQUEST FILE INFORMATION
C
10 CALL OPEN(SHGEOM,,IUNIT,IREAD,IERR)
IF(IERR) 10,120,20
C
READ GEOMETRY
C
20 READ(IUNIT,180) NP,NJ,NPT,NEDGE
IF(NP+NPT2,GT,NP2) CALL OVER(1)
IF(NP2=NPT2,GT,NPTMAX) CALL OVER(2)
IF(NPT+NPT2,GT,NPTMAX) CALL OVER(3)
READ(IUNIT,180) ((NPL(I,J),I=1,2),J=1,NP)
READ(IUNIT,170) ((X(I,J),I=1,3),J=1,NJ)
READ(IUNIT,180) (JP(I),I=1,NEDGE)
C
EXPAND IP ARRAY FROM MOVIE-79 FORMAT
C
IROW=0
ICOL=0
I=0
30 ICOL=ICOL+1
DO 40 J=1,NODE
40 IP(J,ICOL)=0
50 IROW=IROW+1
I=I+1
IF(I,GT,NEDGE) GO TO 70
IF(JP(I),LT.0) GO TO 60
IP(IROW,ICOL)=JP(I)
GO TO 50
60 IP(IROW,ICOL)=-JP(I)
IROW=0
I1=I+1
IF(I1,GT,NEDGE) GO TO 70
IF(JP(I1),LT.0) GO TO 30
IP(NODE,ICOL)=-JP(I1)
I=I+1
GO TO 30
70 TYPE 130, ((NPL(I,J),I=1,2),J=1,NP)
   TYPE 200,
   NPT1=0
   DO 80 J=1,NP
       NPL(1,J)=NPL(2,J)-NPL(1,J)+1
       NPL(2,J)=0
   80 NPT1=NPT1+NPL(1,J)
   IF(NPT1,NE,NPT) GO TO 90
   TYPE 140, (NPL(1,J),J=1,NP)
   TYPE 200,
   TYPE 230,
   ACCEPT 190, ANS
C
C   LIT(1) = "Y"
C
C   IF(ANS,NE,LIT(1)) GO TO 100
   RETURN
C
C   REQUEST PART GROUPINGS
C
90 TYPE 150,
100 TYPE 220,
   ACCEPT 210, NP
   IF(NP,GT,NPMAX) CALL OVER(1)
   DO 110 J=1,NP
110 NPL(2,J)=0
   TYPE 160,
   ACCEPT 210, (NPL(1,I),I=1,NP)
120 RETURN
C
130 FORMAT(14H+<PART LIMITS>/(1X,10I5))
140 FORMAT(36H+<NUMBER OF ELEMENTS IN PART GROUPS>/(1X,
       $10I5))
150 FORMAT(31H+<MULTIPLY DEFINED PART GROUPS>$/)
160 FORMAT(31H+<NUMBER OF ELEMENTS IN GROUPS>$/)
170 FORMAT(6F12.5)
180 FORMAT(16I5)
190 FORMAT(A1)
200 FORMAT(1H )
C
C   SEE POSSIBLE INCOMPATIBILITIES (1)
C
210 FORMAT(20T)
C
C   SEE POSSIBLE INCOMPATIBILITIES (2)
C
220 FORMAT(25H+<NUMBER OF PART GROUPS> $)
230 FORMAT(7H+<OK?> $)
SUBROUTINE REVCOR(NN, X)
DIMENSION X(3,250)
COMMON/REVDO/ NPNTS,NX,DELANG,DELTRA,DELRAD

C
CALCULATE INITIAL ANGLE OF NODE

Y=X(2,NN)
IF(X(1,NN).EQ.0.0) GO TO 10
4=X(3,NN)/X(1,NN)
ANG=ATAN(4)
IF(X(1,NN).LT.0.0) ANG=ANG+3.1416
GO TO 20

10 ANG=1.571
IF(X(3,NN).LT.0.0) ANG=-ANG
IF(X(3,NN).NE.0.0) GO TO 20
RAD=0.0
GO TO 30

C
CALCULATE RADIUS OF REVOLUTION

20 RAD=SQR(T(X(1,NN)*X(1,NN)+Y(3,NN)*X(3,NN))
30 CONTINUE

C
CALCULATE COORDINATES OF NODES

DO 40 I=1,NX-1

NN=NN+NPNTS
ANG=ANG-DELANG
XI=I
X(1,NN)=COS(ANG)*(RAD+DELRAD*XI)
X(2,NN)=Y+DELTRA*XI

40 Y(3,NN)=SIN(ANG)*(RAD+DELRAD*XI)
RETURN
END
SUBROUTINE REVOLV(NPL,X,IP,LOOP,NODE)
INTEGER OUTPUT
REAL LIT
LOGICAL LOOP,CLOSE

SEE POSSIBLE INCOMPATIBILITIES (5)

DIMENSION X(3,250),IP(9,250),NPL(2,20),LIT(3)
COMMON/DEV/ INPUT,OUTPUT
COMMON/CNTL/ MP,NP,NJ,NPT
COMMON/MAXI/ NPMAX,NJMAX,NPTMAX
COMMON/REV/ NPNTS,NX,DELANG,DELTRA,DELRAD
DATA LIT/1HY,5HCRV,5HLOO /

REQUEST INFORMATION ABOUT VOLUME OF REVOLUTION
CLOSE=.FALSE.

READ IN COORDINATES OF NODES ON CURVE(LOOP)

LIT(2) = "CURVE"
LIT(3) = "LOOP"

IF(LOOP) LIT(2)=LIT(3)
TYPE 140, LIT(2)
ACCEPT 120, NPNTS,DEGREV
TYPE 100,
DO 10 I=NJ+1,NJ+NPNTS
  TYPE 150,
10 ACCEPT 130, (X(J,I),J=1,3)

READ TRANSLATION ALONG Y AXIS AND CHANGE IN RADIUS

TYPE 170,
ACCEPT 130, TRANSY,CHARAD
IF(DEGREV.EQ.360..AND.,TRANSY.EQ.0.0..AND.,
  CHARAD.EQ.0.0.) CLOSE=.TRUE.

READ NUMBER OF CIRCUMFERENTIAL ELEMENTS

TYPE 160,
ACCEPT 120, NUMDIV

CHECK NPMAX, NJMAX AND NPTMAX

NMP=NP+1
IF(NMP.GT.NPMAX) CALL OVER(1)
NX=NUMDIV+1
IF(CLOSE) NX=NX-1
NNJ=NJ+NPNTS*NX
IF (NNJ .GT. NJMAX) CALL OVER(2)
NY=NPNTS-1
IF (LOOP) NY=NY+1
NNPT=NPT+NY*NUMDIV
IF (NNPT .GT. NPTMAX) CALL OVER(3)

CALCULATE INCREMENTAL CHANGE IN REVOLUTION ANGLE, TRANSLATION AND RADIUS OF REVOLUTION

DELANG=DEGREV*3.1416/(180.*NUMDIV)
DELTRA=TRANSY/NUMDIV
DELRAD=CHARAD/NUMDIV

GENERATE COORDINATE ARRAY

DO 20 I=NJ+1,NJ+NPNTS
   NN=I
   CALL REVCOR(NN,X)
20 CONTINUE

GENERATE CONNECTIVITY ARRAY

I1=NPT

DO 90 J=1,NUMDIV
   DO 50 I=1,NPNTS-1
      I1=I1+1
      DO 30 I2=5,NODE
         IP(I2,I1)=0
      30
      IP(1,I1)=NJ+(J-1)*NPNTS+I
      IP(2,I1)=IP(1,I1)+1
      IF(J.EQ.NUMDIV.AND.CLOSE.EQ..TRUE.) GO TO 40
      IP(3,I1)=IP(2,I1)+NPNTS
      IP(4,I1)=IP(3,I1)-1
      GO TO 50
   40
   IP(3,I1)=NJ+I+1
   IP(4,I1)=NJ+I
   CONTINUE
   IF (.NOT. LOOP) GO TO 90
   I1=I1+1
   DO 70 I2=5,NODE
   70
   IP(I2,I1)=0
   IP(1,I1)=NJ+J*NPNTS
   IP(2,I1)=IP(1,I1)-NPNTS+1
   IF(J.EQ.NUMDIV.AND.CLOSE.EQ..TRUE.) GO TO 80
   IP(3,I1)=IP(2,I1)+1
   IP(4,I1)=IP(1,I1)+NPNTS
   GO TO 90
   80
   IP(3,I1)=NJ+1
   IP(4,I1)=NJ+NPNTS
   90 CONTINUE
GENERATE PARTS ARRAY

NPL(1, NNP) = NNPT - NPT
NPL(2, NNP) = 0

TRANSLATE AND/OR ROTATE THE OBJECT

TYPE 180,
ACCEPT 110, ANS

LIT(1) = "Y"

IF(ANS.EQ.LIT(1)) CALL TRAROT(X, NNJ)

UPDATE NP, NJ AND NPT

NP = NNP
NJ = NNJ
NPT = NNPT
RETURN

100 FORMAT(31H<COORD.S (X,Y,Z) OF EACH NODE>/)
110 FORMAT(A1)

SEE POSSIBLE INCOMPATIBILITIES (1)

120 FORMAT(I,F10.0)
130 FORMAT(3F10.0)

SEE POSSIBLE INCOMPATIBILITIES (2)

140 FORMAT(25H<NUMBER OF NODES ON THE ,A5,
123H, ANGLE OF REVOLUTION> $)
150 FORMAT(6H+>>> $)
160 FORMAT(3H+<NUMBER OF CIRCUMFERENTIAL ELEMENTS> $)
170 FORMAT(53H+<TOTAL: TRANSLATION ALONG Y AXIS,
1 CHANGE IN RADIUS> $)
180 FORMAT(37H+<WOULD YOU LIKE TO TRANSLATE AND/OR>,
120H ROTATE THIS PART?> $)

END
SUBROUTINE SYMM(NPL,X,IP,U,S,NODE)
INTEGER OUTPUT
REAL LIT
LOGICAL MULTI

SEE POSSIBLE INCOMPATIBILITIES (5)

DIMENSION NPL(2,20),X(3,250),IP(9,250),U(3,250),IX($250),LIT(5)
COMMON/CNTL/ NP,NJ,NPT
COMMON/DEV1/ INPUT,OUTPUT
COMMON/JUNK/ IX
COMMON/MAXI/ NMAX,NJMAX,NPTMAX
DATA LIT/1H ,2HXY,2HXZ,2HYZ,1HY/

REQUEST SYMMETRY PLANE, RETURN IF BLANK

10 TYPE 200,
ACCEPT 170, SYMP

LIT(1) = " "

IF(SYMP.EQ.LIT(1)) RETURN
ISYM=0

LIT(2) = "XY"

IF(SYMP.EQ.LIT(2)) ISYM=3
LIT(3) = "XZ"

IF(SYMP.EQ.LIT(3)) ISYM=2
LIT(4) = "YZ"

IF(SYMP.EQ.LIT(4)) ISYM=1
IF(ISYM.EQ.0) GO TO 10

MULTIPLE DISPLACEMENT AND SCALAR FUNCTION FILES?

TYPE 210,
ACCEPT 180, ANS
MULTI=.FALSE.

LIT(5) = "Y"

IF(ANS.EQ.LIT(5)) MULTI=.TRUE.

ARRAY IX GETS SYMMETRY MAPPING OF NODE NUMBERS
IF(NP+NP_GT.NPMAX) CALL OVER(1)
IF(NPT+NP*GT.NPTMAX) CALL OVER(3)
DO 20 I=1,NP
  K=I+NP
20 MPl(I,K)=MPl(I,I)
K=0
DO 30 I=1,NJ
  IX(I)=I+NJ-K
  IF(X(I*SYM*J).NE.0.0) GO TO 30
  IX(I)=I
  K=K+1
30 CONTINUE
NNJ=NJ+NJ-K
IF(NNJ*GT.NJMAX) CALL OVER(2)

USE ARRAY IX TO FORM SYMMETRY ELEMENTS

NODE1=NODE-1
DO 70 J=1,NPT
  J1=J+NPT
  DO 40 I=1,NODE
40   IP(I,J1)=0
   I1=IP(NODE,J)
   IF(I1_NE.0) IP(NODE,J1)=IX(I1)
   N1=1
   N2=0
  DO 50 I=1,NODE1
      I1=IP(I,J)
      IF(I1_EQ.0) GO TO 60
      IP(I,J1)=IX(I1)
50   N2=I
60   I=IP(N1,J1)
   IP(N1,J1)=IP(N2,J1)
   IP(N2,J1)=I
   N1=N1+1
   N2=N2-1
   IF(N1,LT.N2) GO TO 60
70 CONTINUE

USE ARRAY IX TO FORM SYMMETRY COORDINATES

DO 90 J=1,NJ
  J1=IX(J)
  IF(J1.LE.NJ) GO TO 90
  DO 80 I=1,3
80   X(I,J1)=X(I,J)
   X(ISYM,J1)=-X(ISYM,J1)
90 CONTINUE

MFILES=1
IF(.NOT.MULI) GO TO 100
USE ARRAY IX TO FORM SYMMETRY DISPLACEMENTS

TYPE 220,
ACCEPT 190., NFILES
100 DO 130 N=1,NFILES
   IF(MULTI) CALL RDDISP(U)
   DO 120 J=1,NJ
      J1=IX(J)
      IF(J1.LE.NJ) GO TO 120
   DO 110 I=1,3
110   U(I,J1)=U(I,J)
     U(ISYM,J1)=-U(ISYM,J1)
120   CONTINUE
   IF(MULTI) CALL WRDISP(U,NNJ)
130 CONTINUE
   IF(.NOT.*MULTI) GO TO 140

CALCULATE NEW VALUES FOR NP, NJ, AND NPT

140 NP=NP+NP
   MJ=NNJ
   NPT=NPT+NPT
RETURN

170 FORMAT(A2)
180 FORMAT(A1)

SEE POSSIBLE INCOMPATIBILITIES (1)

190 FORMAT(I)

SEE POSSIBLE INCOMPATIBILITIES (2)

200 FORMAT(24H+<WHICH SYMMETRY PLANE> $)
210 FORMAT(12H+<MULTIPLE FILES?> $)
220 FORMAT(32H+<NUMBER OF DISPLACEMENT FILES> $)
END
SUBROUTINE TRAROT(X,NNJ)
INTEGER OUTPUT
REAL L
DIMENSION X(3,250),L(3,3),Y(3,3)
COMMON/DFVI/ INPUT,OUTPUT
COMMON/CNTL/ NP,N,J,NPT

C REQUEST INFORMATION ABOUT NEW COORDINATES

C TYPE 80,
ACCEPT 70, Y(1,1),Y(2,1),Y(3,1)
TYPE 90,
ACCEPT 70, Y(1,2),Y(2,2),Y(3,2)
TYPE 100,
ACCEPT 70, Y(1,3),Y(2,3),Y(3,3)

C CALCULATE TRANSFORMATION MATRIX

DO 10 I=1,3
   L(I,1)=Y(I,2)-Y(I,1)
10   L(I,3)=Y(I,3)-Y(I,1)
CALL CROSSX(L,1,3)
NORM1=DIST(L(1,1),L(2,1),L(3,1))
NORM2=DIST(L(1,3),L(2,3),L(3,3))
DO 20 I=1,3
   L(I,1)=L(I,1)/NORM1
20   L(I,3)=L(I,3)/NORM2
DO 30 I=1,3
30   L(I,2)=L(I,1)
CALL CROSSX(L,3,2)

C TRANSLATE AND/OR ROTATE

DO 60 I=NJ+1,NNJ
   DO 40 J=1,3
40   Y(J,3)=L(J,1)*X(1,I)+L(J,2)*X(2,I)+L(J,3)*X(3,I)
   +Y(J,1)
   DO 50 K=1,3
50   X(K,I)=Y(K,3)
60 CONTINUE
RETURN

C SEE POSSIBLE INCOMPATIBILITIES (1)

70 FORMAT(3F10.0)

C SEE POSSIBLE INCOMPATIBILITIES (2)

80 FORMAT(14H<NEW ORIGIN> $)
90 FORMAT(26H<POINT ALONG NEW X AXIS> $)
100 FORMAT(25H"POINT IN NEW XY PLANE" $)
END
SUBROUTINE WRDISP(U,NJ)
INTEGER OUTPUT
DIMENSION U(3,250)
COMMON/DEV/ INPUT,OUTPUT
DATA IWRITE/-1/

REQUEST FILE INFORMATION

10 CALL OPEN(5,DISP.,IUNIT,IWRITE,IERROR)
IF(IERROR) 10,30,20

WRITE DISPLACEMENTS

20 WRITE(IUNIT,40) ((U(I,J),I=1,3),J=1,NJ)
30 RETURN

SEE POSSIBLE INCOMPATIBILITIES (3)

40 FORMAT(6F12.5)
END
SUBROUTINE WRGEOM(NPX, NPL, X, IP, JP, NODE)
INTEGER OUTPUT
REAL LIT
DIMENSION NPL(2,20), NPX(2,20), X(3,250), IP(2250), LIT(1), JP(2250)
COMMON/CNTL/ NP, NJ, NPT
COMMON/DEV1/ INPUT, OUTPUT
COMMON/MAXI/ NMAX, NJMAX, NJMAX
DATA IWRITE/-1/
DATA LIT/1HY/
C REQUEST FILE INFORMATION
C 10 CALL OPEN(5HGEOM.,IUNIT,IWRITE,IERRO)
IF(IERROR) 10,80,20
C REQUEST ELEMENT LIMITS FOR PARTS LIST
C 20 NPT1=0
DO 30 J=1, NP
   NPL(2,J)=NPT1+NPX(1,J)
   NPL(1,J)=NPT1+1
30 NPT1=NPL(2,J)
TYPE 90, ((NPL(I,J),I=1,2),J=1,NP)
TYPE 150,
ACCEPT 110, ANS
IF(ANS.EQ.LIT(1)) GO TO 40
TYPE 140,
ACCEPT 130, NP
IF(NP.GT.NMAX) CALL OVER(1)
TYPE 100,
ACCEPT 130, ((NPL(I,J),I=1,2),J=1,NP)
C REDUCE IP ARRAY TO MATCH MOVIE-79 FORMAT
C 40 NODE1=NODE-1
NI=0
NEDGE=0
DO 70 I=1,NPT
   DO 50 J=1, NODE1
      NI=NI+1
      IF(IP(NI).EQ.0) GO TO 60
      NEDGE=NEDGE+1
50 JP(NEDGE)=IP(NI)
60 JP(NEDGE)=JP(NEDGE)
   NI=I*NODE
   IF(IP(NI).EQ.0) GO TO 70
   NEDGE=NEDGE+1
   JP(NEDGE)=-IP(NI)
70 CONTINUE
WRITE GEOMETRY FILE

WRITE (IUNIT,120) NP,NJ,NPT,NEDGE
WRITE (IUNIT,120) ((NPL(I,J)*I=1,2)*J=1,NP)
WRITE (IUNIT,160) ((X(I,J)*I=1,3)*J=1,NJ)
WRITE (IUNIT,120) (JP(I),I=1,NEDGE)
30 RETURN

90 FORMAT(26H+<ELEMENT LIMITS OF PARTS>/(1X,10I5))
100 FORMAT(26H+<ELEMENT LIMITS OF PARTS>/*)
110 FORMAT(A1)
120 FORMAT(16I5)

SEE POSSIBLE INCOMPATIBILITIES (1)

130 FORMAT(20I)

SEE POSSIBLE INCOMPATIBILITIES (2)

140 FORMAT(19H+<NUMBER OF PARTS>$)
150 FORMAT(7H <OK?> $)

SEE POSSIBLE INCOMPATIBILITIES (3)

160 FORMAT(6F12.5)
END
Program MOVIE Listings

NOTE: For variable description refer to MOVIE.BYU listings
PROGRAM MOVIE
COMMON/BAUD/IBAUD
COMMON/DEVI/ INPUT, OUTPUT, ERROR
COMMON/DLPA/NPOL, NSMAX
COMMON/PRO/ NP, NJ, CMD, IREAD, SKALE, NPT
DIMENSION LIT(36), IIP(4, 250)
INTEGER OUTPUT, ERROR
REAL LIT
DATA LIT/4HREAD, 4HEXIT, 4HROTA, 4HREST, 4HTRAN, 4HSCAL
1, 4HFLAT, 4HSMOO, 4HUNIF, 4HSCOP, 4HDEV
2, 4HDIFF, 4HDIST, 4HFIEL, 4HPART, 4HEXPL, 4HSUMM, 4HCENT, 4
$HTITL
3, 4HFRIN, 4HCOLO, 4HANIM, 4HVIEW, 4HDRAW, 4HFAST, 4HPIVO, 4
$HHELP
4, 4HLINE, 4HCONT, 4HSHRI, 4HLIGH, 4HFEAT, 4HNODE, 4HPOLY, 4
$HIMM
5, 4HSHIF/

C
C MUST BE CHANGED IF NSMAX DIMENSIONING IS ALTERED
C
C DATA NSMAX/8/
C
C IBAUD=240
TYPE 390,
C
C NPOL SHOULD BE SET EQUAL TO NPMAX
C
C NPOL=20
CALL READ
CALL CLEAR
CALL INIT
C
C READ INPUT COMMAND STRING FOR PROCESSING
C
10 TYPE 410,
ACCEPT 380, CMD
C
IF(CMD.NE.LIT(1)) GO TO 20
CALL READ
GO TO 10
C
20 IF(CMD.NE.LIT(2)) GO TO 30
CALL EXIT
C
30 IF(CMD.NE.LIT(3)) GO TO 40
CALL ROTA
GO TO 10
C
40 IF(CMD.NE.LIT(4)) GO TO 50
LIT(1) = "READ"
LIT(2) = "EXIT"
LIT(3) = "ROTA"
LIT(4) = "REST"
CALL REST
GO TO 10

C 50 IF(CMD.NF.LIT(5)) GO TO 60
  CALL TRAN
  GO TO 10

C 60 IF(CMD.NF.LIT(6)) GO TO 100
  CALL SCAL
  GO TO 10

C 100 IF(CMD.NF.LIT(10)) GO TO 110
  CALL SCAL
  GO TO 10

C 110 IF(CMD.NF.LIT(11)) GO TO 130
  CALL SCAL
  GO TO 10

C 130 IF(CMD.NF.LIT(13)) GO TO 140
  CALL DIST
  GO TO 10

C 140 IF(CMD.NF.LIT(14)) GO TO 150
  CALL FIEL
  GO TO 10

C 150 IF(CMD.NF.LIT(15)) GO TO 160
  ICODE=0
  CALL SEPART(ICODE)
  GO TO 10

C 160 IF(CMD.NF.LIT(16)) GO TO 170
  CALL EXPL
  GO TO 10

C 170 IF(CMD.NF.LIT(17)) GO TO 180
  CALL SUMCEN
  GO TO 10

C 180 IF(CMD.NF.LIT(18)) GO TO 190
  CALL SUMCEN
  GO TO 10

C 190 IF(CMD.NF.LIT(19)) GO TO 220
  CALL TITLE
  GO TO 10

C 220 IF(CMD.NF.LIT(22)) GO TO 230

LIT(5) = "TRAN"
LIT(6) = "SCAL"
LIT(7) = "FLAT"
LIT(10) = "SCAL"
LIT(11) = "DEVI"
LIT(13) = "DIST"
LIT(14) = "FIEL"
LIT(15) = "PART"
LIT(16) = "EXPL"
LIT(17) = "SUMM"
LIT(18) = "CENT"
LIT(19) = "TITL"
LIT(22) = "ANIM"
CALL ANIMAT
GO TO 10

C 230 IF(CMD.NE.LIT(23)) GO TO 240
CALL VIEDRA
GO TO 10

C 240 IF(CMD.NE.LIT(24)) GO TO 250
CALL VIEDRA
GO TO 10

C 250 IF(CMD.NE.LIT(25)) GO TO 260
CALL FAST
GO TO 10

C 260 IF(CMD.NE.LIT(26)) GO TO 270
CALL PIVOT
GO TO 10

C 270 IF(CMD.NE.LIT(27)) GO TO 300
CALL HELP
GO TO 10

C 300 IF(CMD.NE.LIT(30)) GO TO 330
CALL SHRKR
GO TO 10

C 330 IF(CMD.NE.LIT(33)) GO TO 340
CALL POLMOD
GO TO 10

C 340 IF(CMD.NE.LIT(34)) GO TO 350
CALL POLMOD
GO TO 10

C 350 IF(CMD.NE.LIT(35)) GO TO 360
CALL IMMUNE
GO TO 10

C 360 IF(CMD.NE.LIT(36)) GO TO 370
CALL TRAN
GO TO 10

370 CALL HELP
GO TO 10

380 FORMAT(A4)
390 FORMAT(1H , ' *** UCF *** EMCS DEPT. PDP 11/34 ***', $/1H
$ ' *** COMPUTER GRAPHICS SYSTEM ***')
400 FORMAT(20I4)
C     SEE POSSIBLE INCOMPATIBILITIES (2)
C
410 FORMAT(5H+>> ,$)
END
SUBROUTINE ANIMAT
COMMON/ANTI/ DA, DT, DR, CPF, IFR1, IFR2, DDOZ, DDELT, IPM,
$NFRAME,
1 SFDEL, ANISHR
COMMON/DELSH/ DELTA, SHRK
COMMON/ANICR/ICON, HARCOP
COMMON/DEVIA/OUTPUT, ERROR
COMMON/LOGI/ISMOO, IPOOR, IMIX, DIRC, ISPEC, ISMA, LINEA
SR,
1 IHLR, IFRING
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
DIMENSION DA(3, 20)
DIMENSION DT(3), DR(3)
LOGICAL ISMA, LINEAR, IPM, ICON, HARCOP
INTEGER OUTPUT
REAL LIT
DATA LIT/1HY/

TYPE 130,
ACCEPT 110, NFRAME
IF(NFRAME .EQ. 0) RETURN
DO 10 J=1, NP
DO 10 I=1, 3
10 DA(I, J) = 0.0
IFR1 = 1
IFR2 = NFRAME
TYPE 140,
ACCEPT 110, I1, I2
IF(I1 .LE. 0) GO TO 20
IFR1 = I1
IFR2 = I2
IF(IFR2 .LT. IFR1) IFR2 = IFR1
20 TYPE 230,
ACCEPT 80, ANS
ISMA = .FALSE.

LIT = "Y"

IF(ANS .EQ. LIT) ISMA = .TRUE.
TYPE 150,
ACCEPT 80, ANS
LINEAR = .FALSE.

LIT = "Y"

IF(ANS .EQ. LIT) LINEAR = .TRUE.
CPF = 0.0
IF(LINEAR) GO TO 30
IF(SKALE .EQ. 0.0) GO TO 30
TYPE 160,
ACCEPT 100, CPF

30 TYPE 170,
ACCEPT 100, DT(1), DT(2), DT(3)
TYPE 180,
ACCEPT 100, DR(1), DR(2), DR(3)
TYPE 90,
40 TYPE 250,
ACCEPT 120, I1, I2, X1, X2, X3
IF(I1.EQ.0) GO TO 60
DO 50 I=I1, I2
   DA(1, I) = X1
   DA(2, I) = X2
50 DA(3, I) = X3
GO TO 40
60 TYPE 190,
ACCEPT 100, DDOZ
70 TYPE 240,
ACCEPT 100, ANISHR
ANISHR = ANISHR + SHRK
IF (ANISHR .LT. 0.0 .OR. ANISHR .GE. 1.0) GO TO 70
TYPE 200,
ACCEPT 100, SFDEL
XFRAME = NFRAME
SFDEL = SFDEL / XFRAME
TYPE 210,
ACCEPT 100, DDELTA
TYPE 220,
ACCEPT 80, ANS
IPM = .FALSE.

LIT = "Y"

IF (ANS.EQ.LIT) IPM = .TRUE.

TYPE 300
ACCEPT 80, ANS
ICON = .FALSE.
IF (ANS.EQ. 'Y') ICON = .TRUE.

TYPE 310
ACCEPT 80, ANS
HARCO = .FALSE.
IF (ANS.EQ. 'Y') HARCO = .TRUE.
RETURN

30 FORMAT(A1)
90 FORMAT(3H+<PIVOT PARTS I1/I2, DX, DY, DX>/)

SEE POSSIBLE INCOMPATIBILITIES (1)

100 FORMAT(3F10.0)
110 FORMAT(1G1)
120 FORMAT(2I, 3F10.0)
SEE POSSIBLE INCOMPATIBILITIES (2)

130 FORMAT (20H <NUMBER OF FRAMES> ,$)
140 FORMAT (21H <SEND FRAMES I1/I2> ,$)
150 FORMAT (19H <TRANSIENT DATA?> ,$)
160 FORMAT (26H <VIBRATION CYCLES/FRAME> ,$)
170 FORMAT (39H <TOTAL TRANSLATION CHANGE (DX, DY, DZ)> ,$)
180 FORMAT (36H <TOTAL ROTATION CHANGE (DX, DY, DZ)> ,$)
190 FORMAT (33H <DISTANCE TO ORIGIN (DELTA)> ,$)
200 FORMAT (37H <DISPLACEMENT SCALE FACTOR (DELTA)> ,$)
210 FORMAT (33H <POSITION SCALE FACTOR (DELTA)> ,$)
220 FORMAT (13H <PICTURES?> ,$)
230 FORMAT (21H <SHRINK FACTOR (DELTA)> ,$)
240 FORMAT (5H >>> ,$)
300 FORMAT (1H ,'<DO YOU WANT CONTINUOUS OPERATION ? > ',$)
310 FORMAT (1H ,'<DO YOU WANT A HARD-COPY OF EVERY FRAME $ ? > ',$)
END
SUBROUTINE BGNFRM
INTEGER OUTPUT,ERROR
REAL LIT
LOGICAL IFIRST, MOSAIC
DIMENSION LIT(1)
COMMON/ILINE/ILINE,NSIZE
COMMON/MOSAIC/MOSAIC
COMMON/BAUD/IBAUD
COMMON/DEV1/ INPUT, OUTPUT, ERROR
COMMON/INTENS/ IPH, IPL, IPB, IFX, IMRES, URES
COMMON/TEK/XBAR,YBAR, SF, RES
COMMON/PLTTER/ PLTSIZ, XLAST, YLAST, XDISP
COMMON/OFORIO/ CONTRS, IDVICE, IBAD, SHOSHR, LBLSPC
DATA IFIRST/.TRUE./
DATA LIT /1HY /

IF(IDVICE.GT.0) RETURN
RES=IFX-1

TEKTRONIX SCOPE IS PICTURE DEVICE

PLTSIZ=URES/RES

CALL PLEAS

NEXT FIVE COMMANDS DRAW A BOX AROUND PICTURE AREA

MOVABS AND DRWABS ARE PLOT-10 ROUTINES WHICH MOVE A
AND DRAG THE

PEN TO THE GIVEN LOCATION, RESPECTIVELY

XDISP=RES-URES
CALL MOVABS(XDISP, URES)
CALL DRWABS(RES, URES)
CALL DRWABS(RES, 0.)
CALL DRWABS(XDISP, 0.)
CALL DRWABS(XDISP, URES)
XLAST=XDISP
YLAST=URES
RETURN

END
SUBROUTINE CLEAR
COMMON/XAN/ DA, DT, DR, CPF, IFR1, IFR2, DDOZ, DDELTA, IPM,
$NFMT, 1 . SFOEL
COMMON/CONLV/ CONHI, CONLO, NCONLV, CLEVEL
COMMON/COUU/ NN
COMMON/CONT/ CONT
COMMON/COOR/ XP
COMMON/DISP/ JCOL
COMMON/FYSL/ FUN, YY, SLINR
COMMON/NE/ NFR
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
COMMON/REST2/ XX
COMMON/ROT/ NONROT
COMMON/SPUX/ SPEC, SPEC1, UX
COMMON/TRIS/ IKILL, XT
COMMON/VRB/ UFRING, DRC, FRING
COMMON/VSTI/ NNN, IVIS
COMMON/XX/ XN
DIMENSION U(3, 250), X(3, 250)
DIMENSION DA(3, 20), NONROT(20), XX(3, 20)
DIMENSION NN(9), NNN(8), XN(3, 9), XP(3, 9)
DIMENSION DT(3), DR(3), FUN(3), YY(3)
LOGICAL NONROT

DATA BELOW WILL NEED TO BE CHANGED IF DIMENSIONS ARE ALTERED

DATA NJMAX/250/, NPMAX/20/, NSMAX/8/

XT = 0.0
SKALE = 0.0
DO 20 I = 1, NPMAX
   DO 20 J = 1, 3
      DA(J, I) = 0.0
      DT(J) = 0.0
      DR(J) = 0.0
      XX(J, I) = 0.0
20   FUN(J) = 0.0
DO 30 I = 1, NSMAX
   NN(I) = 0
30   NNN(I) = 0
   M1 = NSMAX + 1
   DO 40 I = 1, M1
      DO 40 J = 1, 3
         XN(J, I) = 0.0
40    XP(J, I) = 0.0
   DO 60 I = 1, NPMAX
      NONROT(I) = .FALSE.
60   CONTINUE
SUBROUTINE COARRO
COMMON/PRO/NP,NJ,CMD,IC,IREAD,SKALE,NPT
COMMON/RESLO/XO,DC
COMMON/ROTM/ NONROT
COMMON/SCMR/ DOZ,FIOLD,FIELD,TANAL,RES
COMMON/TRIS/ IKILL,XT
DIMENSION DC(3,3,21),NONROT(20)
DIMENSION AE(3,3),X0(3)
LOGICAL NONROT

C
IF(IKILL.EQ.1) RETURN

C
SET LOCATION OF AXIS VECTORS AND THEIR ORIGIN

C
ACX=RES/10.,0
A1=RES/18.,0
DO 10 I=1,3
   DO 10 J=1,3
10   AE(I,J)=0.,0
    AE(1,1)=A1
    AE(2,2)=A1
    AE(3,3)=A1
II=NP+1
DO 20 I=1,3
    X1=AE(1,I)
    X2=AE(2,I)
    X3=AE(3,I)
C
APPLY ROTATION TRANSFORMATION

C
IF(IKILL.EQ.2)GO TO 15
   AE(1,I)=DC(1,1,II)*X1+DC(2,1,II)*X2+DC(3,1,II)*X3
   AE(2,I)=DC(1,2,II)*X1+DC(2,2,II)*X2+DC(3,2,II)*X3
   AE(3,I)=DC(1,3,II)*X1+DC(2,3,II)*X2+DC(3,3,II)*X3
15 CONTINUE
   AA1=AE(1,I)+ACX
   AA2=AE(2,I)+ACX
C
PLOT AXIS LINES
   CALL PLTLIN(ACX,ACX,AA1,AA2)
C
SET LOCATION OF LETTERS X,Y,Z CENTERS

C
AE(1,I)=1.5*AE(1,I)+ACX
20 AE(2,I)=1.5*AE(2,I)+ACX
SET SIZE OF LETTERS

R = RES / 72.0

LETTER X POINTS

X1X = AE(1, 1) - R
X1Y = AE(2, 1) + R
X2X = AE(1, 1) + R
X2Y = X1Y
X3X = X2X
X3Y = AE(2, 1) - R
X4X = X1X
X4Y = X3Y

CALL PLTLIN(X1X, X1Y, X3X, X3Y)
CALL PLTLIN(X2X, X2Y, X4X, X4Y)

LETTER Y POINTS

Y1X = AE(1, 2) - R
Y1Y = AE(2, 2) + R
Y2X = AE(1, 2) + R
Y2Y = Y1Y
Y3X = AE(1, 2)
Y3Y = AE(2, 2) - R1

CALL PLTLIN(Y1X, Y1Y, Y3X, AE(2, 2))
CALL PLTLIN(Y2X, Y2Y, Y3X, AE(2, 2))
CALL PLTLIN(Y3X, Y3Y, Y3X, AE(2, 2))

LETTER Z POINTS

Z1X = AE(1, 3) - R
Z1Y = AE(2, 3) + R
Z2X = AE(1, 3) + R
Z2Y = Z1Y
Z3X = Z2X
Z3Y = AE(2, 3) - R
Z4X = Z1X
Z4Y = Z3Y

CALL PLTLIN(Z1X, Z1Y, Z2X, Z2Y)
CALL PLTLIN(Z2X, Z2Y, Z4X, Z4Y)
CALL PLTLIN(Z4X, Z4Y, Z3X, Z3Y)
RETURN
END
SUBROUTINE CONEL
COMMON/CIP/ IP
COMMON/CONN/ NN
COMMON/DEV/ INPUT, OUTPUT, ERROR
COMMON/LAI/ JLAST, IFLAG
COMMON/LINES/LINELM
COMMON/NEME/ NEDGE, MEDGE, IPART
COMMON/OLPA/ NPOL, NSMAX
DIMENSION IP(1000)
DIMENSION NN(9)
INTEGER ERROR

C
LINELM=0
NCOUNT=0
DO 20 K=1, NSMAX
   JLAST=JLAST+1
   NCOUNT=NCOUNT+1
   NN(K)=IP(JLAST)
   IF(NN(K).GT.0) GO TO 20
   IF(NCOUNT.EQ.2) LINELM=1
   NN(K)=-NN(K)
10  NEDGE=K
   MEDGE=MEDGE+1
   IF(IP(JLAST+1).GE.0) RETURN
   JLAST=JLAST+1
   NN(MEDGE)=-IP(JLAST)
   RETURN
20  CONTINUE
TYPE 30, NSMAX
30 FORMAT(424<ERROR: MAXIMUM NUMBER OF EDGES EXCEEDED $>, I3/)
CALL EXIT
END
SUBROUTINE COORD(ICEN)
COMMON/CENT/ NCENPT
COMMON/CONN/ NN
COMMON/CONT/ CONT
COMMON/COOR/ XP
COMMON/DLISH/ DELTA,SHRK
COMMON/LAIF/ JLAST,IFLAG
COMMON/LINES/LINELM
COMMON/LOGI/ISMOTH, IPOOR, IMIX, DIRC, ISPEC, ISMA, LINEA
$R$
1 IHLR, IFRING
COMMON/NAMES/ MEDGE, MEDGE, IPART
COMMON/OFORT/ CONTS, IDVICE, IBAD, SHOSHR, LBLSPC
COMMON/XXN/ XN
DIMENSION NCENPT(250)
DIMENSION CONT(9), NN(9), XN(3,9), XP(3,9)
LOGICAL IPOOR, IHLR
C
CALL POINTS(ICEN)
XEDGE=MEDGE
C
CENTER POINT COORDINATE CALCULATED = AVERAGE OF
OTHER NODES (UNLESS SPECIFIED IN DATA)
C
IF(NCENPT(ICEN).EQ.1) GO TO 40
DO 10 I=1,3
10 XP(I,MEDGE)=0.0
DO 20 I=1,MEDGE
DO 20 J=1,3
20 XP(J,MEDGE)=XP(J,MEDGE)+XP(J,I)
DO 30 I=1,3
30 XP(I,MEDGE)=XP(I,MEDGE)/XEDGE
C
CALL SHRINK IF SPECIFIED
C
40 IF(SHRK.EQ.0.0) GO TO 50
CALL SHRINK
50 IF(LINELM.EQ.1) RETURN
IF(IPOOR) GO TO 60
C
IF SIMPLE LINE DRAWING AND NO POOR MANS - NO NORMAL
LS CALCULATED
C
IF(.NOT.IHLR) RETURN
C
CALL NORMAL
C
IF COORD CALLED FROM NORAV, CENTER POINT INFO NOT CALCUOR
C
IF(IFLAG.EQ.0) RETURN
x1 = 0.0
x2 = 0.0
x3 = 0.0
x4 = 0.0

C CENTER POINT NORMAL AND CONTOUR VALUE = AVERAGE OF THE OTHER NODES

IF (NCENPT(ICEEN) .EQ. 1) GO TO 100
DO 80 II = 1, NEDGE
   I = II
   X2 = X2 + XN(1, I)
   X3 = X3 + XN(2, I)
   X4 = X4 + XN(3, I)
80 X1 = X1 + CONT(I)
   CONT(MEDGE) = X1/XEDGE
90 X1 = SQRT(X2*X2 + X3*X3 + X4*X4)
   XN(1, MEDGE) = X2/X1
   XN(2, MEDGE) = X3/X1
   XN(3, MEDGE) = X4/X1
C
RETURN
100 DO 110 II = 1, NEDGE
   I = II
   X2 = X2 + XN(1, I)
   X3 = X3 + XN(2, I)
110 X4 = X4 + XN(3, I)
   GO TO 90
END
SUBROUTINE DIST
COMMON/DEV1/INPUT, OUTPUT, ERROR
COMMON/SOMR/ DOZ, FIELD1, FIELD, TANAL, RES
INTEGER OUTPUT

TYPE 20, DOZ
ACCEPT 10, DOZ
RETURN

SEE POSSIBLE INCOMPATIBILITIES (1)

10 FORMAT(F10.0)

SEE POSSIBLE INCOMPATIBILITIES (2)

20 FORMAT(13H+TO ORIGIN (.F8.2,3H) $)
END
SUBROUTINE DRAW(I1, I2)
COMMONT/CONN/ NN
COMMONT/COORD/ XP
COMMONT/LABEL/NUM,NNUM, PNUM
COMMONT/SCMR/ DOZ,FIELD, FIELD, TANAL, RES
COMMONT/XNO/ SCOORD
DIMENSION SCOORD(3, 250)
DIMENSION NN(9), XP(3, 9)
LOGICAL NUM
C
R = 0.5*RES
D = R / ((DOZ - XP(3, I1)) * TANAL)
U1 = R + XP(1, I1) * D
V1 = R + XP(2, I1) * D
U2 = R + XP(1, I2) * D
V2 = R + XP(2, I2) * D
U3 = U1
V3 = V1
U4 = U2
V4 = V2
C
CLIP LEFT EDGE
C
IF(U1.GE.0. AND. U2.GE.0.) GO TO 20
IF(U1.LT.0. AND. U2.LT.0.) RETURN
IF(U1.GT.0.) GO TO 10
V1 = (V1 - V2) * U1/(U2 - U1) + V1
U1 = 0.
GO TO 20
10 V2 = (V2 - V1) * U2/(U1 - U2) + V2
U2 = 0.
C
CLIP RIGHT EDGE
C
20 IF(U1.LE.RES. AND. U2.LE.RES) GO TO 40
IF(U1.GT.RES. AND. U2.GT.RES) RETURN
IF(U1.GT.RES) GO TO 30
V2 = (V2 - V1) * (RES - U1)/(U2 - U1) + V1
U2 = RES
GO TO 40
30 V1 = (V1 - V2) * (RES - U2)/(U1 - U2) + V2
U1 = RES
C
CLIP BOTTOM EDGE
C
40 IF(V1.GE.0. AND. V2.GE.0.) GO TO 60
IF(V1.LT.0. AND. V2.LT.0.) RETURN
IF(V1.GT.0.) GO TO 50
U1 = (U1 - U2) * V1/(V2 - V1) + U1
V1=0.
GO TO 60
50 U2=(U2-U1)*(V2/(V1-V2)+U2
V2=0.

CLIP TOP EDGE

60 IF(V1.LE.RES.AND.V2.LE.RES) GO TO 80
   IF(V1.GT.RES.AND.V2.GT.RES) RETURN
   IF(V1.GT.RES) GO TO 70
   U2=(U2-U1)*(RES-V1)/(V2-V1)+U1
   V2=RES
   GO TO 80
70 U1=(U1-U2)*(RES-V2)/(V1-V2)+U2
   V1=RES
80 CALL PLTLIN(U1,V1,U2,V2)
   IF(.NOT.NUM) RETURN
   IF(U1.NE.U3.OR.V1.NE.V3) GO TO 90
   N1=NN(I1)
   SCOORD(1,N1)=U1
   SCOORD(2,N1)=V1
   SCOORD(3,N1)=1.
90 IF(U2.NE.U4.OR.V2.NE.V4) RETURN
   N2=NN(I2)
   SCOORD(1,N2)=U2
   SCOORD(2,N2)=V2
   SCOORD(3,N2)=1.
RETURN
END
SUBROUTINE DRWABS(X,Y)
CALL TKPLOT(X,Y,2)
RETURN
END
SUBROUTINE ENDERM
INTEGER*2 US,GS,ESC,SUB,FF,CR
LOGICAL ION,HARCOP,TITLE
LOGICAL ITIT(52)
INTEGER OUTPUT,ERROR
COMMON PLTTER,PLTSIZ,XLAST,YLAST,XDISP
COMMON INTENS,IPH, IPL, IPR, IFX, IMRES, URES
COMMON TITLE TITLE,NUMCHR,ITIT
COMMON ANICR,ICON,HARCOP
COMMON LSTPN1,XLST,YLST
COMMON ILINE,ILINE, NSTZE
COMMON DEVI INPUT,OUTPUT,ERROR
COMMON OFORIO CONTROLS,DEVICE, IRAD, SHOSH R, LBS LPC
COMMON CNTRUS,GS,ESC, SUB, FF, CR

C IF CONTINUOUS-TONE THEN RETURN
C (NOTE: NO CONTINUOUS-TONE DEVICE CALL INCLUDED IN
C THIS VERSION)
C IF (DEVICE .GT. 0) RETURN
C
C TEKTRONIX SCOPE IS PICTURE DEVICE
C
US=31
IF (.NOT. TITLE) GO TO 50
MAXC=URES/15.
Y=URES - (URES/76 +20)
NC=MAXC
X=XDISP + 5.0
IF (NUMCHR .GT. MAXC) GO TO 30
NC=NUMCHP
IM=(MAXC - NUMCHR)/2.0
IM=IM*14.18
IN=(URES/14.18 - MAXC)/2.0 * 14.18
X=IM + IM + XDISP
30 CALL MOVABS(X,Y)
CALL TTOUS(US)
TYPE 40,(ITIT(K),K=1,NC)
40 FORMAT(1H,52A1)
IGS=29
CALL TTOUS(IGS)

C

50 R=IMRES*1.0
CALL MOVABS(0.0,R)
IFSC=27
IETB=23
IBELL=7
IF (.NOT. HARDCOPY) GO TO 100
CALL TOUT(IESC)
CALL TOUT(IETB)
100 CALL TOUT(IBELL)
CALL TOUT(US)
RETURN

C
END
SUBROUTINE EXPL
COMMON/DELSH/ DELTA, SHRK
COMMON/DEVI/ INPUT, OUTPUT, ERROR
COMMON/REST2/ XX
DIMENSION XX(3, 20)
INTEGER OUTPUT

TYPE 40,
10 TYPE 80,
ACCEPT 60, I1, I2, X1, X2, X3
IF(I1.EQ.0) GO TO 30
DO 20 I=I1, I2
   XX(1, I)=X1
   XX(2, I)=X2
20 XX(3, I)=X3
GO TO 10
30 TYPE 70, DELTA
ACCEPT 50, DELTA
RETURN
40 FORMAT(3(AH+<PARTS I1/I2, LOCAL MOTION (X,Y,Z)>/) )

50 FORMAT(F10.0)
60 FORMAT(2T, 3F10.0)

SEE POSSIBLE INCOMPATIBILITIES (2)

70 FORMAT(29H+<LOCAL MOTION SCALE FACTOR (,E12.3, 3H)> $, $)
80 FORMAT(5H+>>> , $)
END
SUBROUTINE FAST
COMMON/DEVI/INPUT,OUTPUT,ERROR
COMMON/LOGI/ISMOOTH,IPPOOR,IMIX,DIRC,ISPEC,ISMA,LINE
$R,
   1 IHLR,IFRING
COMMON/POORPT/ POOR,DIRCA
COMMON/PRO/ NP,NJ,CMO,IC,IREAD,SKALE,NPT
DIMENSION DIRCA(20),POOR(20)
LOGICAL IMIX
INTEGER OUTPUT
REAL LIT
DATA LIT/1HY/

C
DO 10 I=1,NP
   POOR(I)=0.0
10 DIRCA(I)=1.0
   TYPE 140,
   ACCEPT 100, ANS
   IMIX=.FALSE.

C
LIT = "Y"

C
IF(ANS.EQ.LIT) IMIX=.TRUE.
   IF(IMIX) RETURN
   TYPE 150,
   ACCEPT 100, ANS

C
LIT = "Y"

C
IF(ANS.NE.LIT) RETURN

20 DO 30 I=1,NP
30 POOR(I)=1.0
40 TYPE 120,
50 TYPE 160,
   ACCEPT 130, I1,I2
   IF(I1.LE.0) GO TO 70
   IF(I1.GT.NP) GO TO 40
   IF(I2.LT.I1) I2=I1
   IF(I2.GT.NP) I2=NP
   DO 60 I=I1,I2
60 POOR(I)=0.0
   GO TO 50
70 TYPE 110,
80 TYPE 160,
   ACCEPT 130, I1,I2
   IF(I1.LE.0) RETURN
   IF(I1.GT.NP) GO TO 70
   IF(I2.LT.I1) I2=I1
   IF(I2.GT.NP) I2=NP
   DO 90 I=I1,I2
90 DIRCA(I)=0.0
GO TO 80
100 FORMAT(A1)
110 FORMAT(3AH+<PARTS I1/I2 WITH CLOCKWISE ORDERING>/)
120 FORMAT(45H+<PARTS I1/I2 IMMUNE TO POOR MAN'S PROCEDURE>/)

SEE POSSIBLE INCOMPATIBILITIES (1)

130 FORMAT(2I)
SEE POSSIBLE INCOMPATIBILITIES (2)

140 FORMAT(15H+<MIXED DATA?> ,$)
150 FORMAT(25H+<POOR MAN'S PROCEDURE?> ,$)
160 FORMAT(5H+>> ,$)
END
SUBROUTINE FIEL
COMMON/DEV1/INPUT,OUTPUT,ERROR
COMMON/SCAR/DOZ,FIELD,FIELD,TANAL,RES
COMMON/ZRANGE/ZMIN,ZMAX,ZMIOLD,ZMAOLD
INTEGER OUTPUT
C
TYPE 20, FIELD,ZMIOLD,ZMAOLD
ACCEPT 10, FIELD,ZMIN,ZMAX
FIELD=FIELD
FIELD=FIELD*.0087265
TANAL=SIN(FIELD)/COS(FIELD)
ZMIOLD=ZMIN
ZMAOLD=ZMAX
ZMIN=ZMIN*TANAL
ZMAX=ZMAX*TANAL
RETURN
C
SEE POSSIBLE INCOMPATIBILITIES (1)
C
10 FORMAT(3F10.0)
C
SEE POSSIBLE INCOMPATIBILITIES (2)
C
20 FORMAT(21H<ANGLE, ZMIN, ZMAX (*F6.2,2F8.2,3H)> ,$)
END
SUBROUTINE HELP
COMMON/DEV1/ INPUT, OUTPUT, ERROR
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
DIMENSION LIT(2)
INTEGER OUTPUT
REAL LIT
DATA LIT/4, HH HELP, 1, HY/

C
C
LIT (1) = "HELP"

C
C
IF(CMD.NE.LIT(1)) GO TO 20
10 TYPE 40,
RETURN
20 TYPE 50, CMD
ACCEPT 30, ANS

C
C
LIT(2) = "Y"

C
C
IF(ANS.EQ.LIT(2)) GO TO 10
RETURN
30 FORMAT(A1)
40 FORMAT(33H<COMMANDS: ANIMATE, CENTER, DEVICE
1, 10H, DISTANCE>/ 24H <DRAW, EXIT, EXPLODE, FAST
2, 19H, FIELD, HELP, IMMUNE>/
3 43H <NODE, PARTS, PIVOT, POLY, READ, RESTORE, ROTATE
4, 19H, SCALE, SCOPE, SHIFT>/
5 32H <SHRINK, SUMMARY, TRANSLATE, VIEW>/)

C
C
SEE POSSIBLE INCOMPATIBILITIES (2)

C
C
50 FORMAT(2H<, A4, 11H? HELP?>, $)
END
SUBROUTINE IMMUNE
COMMON/DEV/ INPUT, OUTPUT, ERROR
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKELE, NPT
COMMON/ROTIM/ NONROT
DIMENSION NONROT(20)
LOGICAL NONROT
INTEGER OUTPUT

C

DO 10 I=1, NP
10 NONROT(I) = .FALSE.
20 TYPE 60,
30 TYPE 80,
    ACCEPT 70, I1, I2
    IF(I1.LE.0) GO TO 50
    IF(I1.GT.NP) I2=20
    IF(I2.LT.I1) I2=I1
    IF(I2.GT.NP) I2=NP
    DO 40 I=I1, I2
40 NONROT(I) = .TRUE.
    GO TO 30
50 RETURN
60 FORMAT(32H+PART I1/I2 IMMUNE TO ROTATION)/**
C

C

SEE POSSIBLE INCOMPATIBILITIES (1)
C

70 FORMAT(2I)
C

SEE POSSIBLE INCOMPATIBILITIES (2)
C

80 FORMAT(5H+>>> ,$)
END
SUBROUTINE INIT
COMMON/INTENS/IPH, IPL, IPB, IFX, IMRES,URES
COMMON/ANICR/ICON, HARCOP
COMMON/TITLE/TITLE, NUMCHR, ITIT
COMMON/POORPT/ POOR, DIRCA
COMMON/SCMR/ DOZ, FIOLD, FIELD, TANAL, RES
COMMON/CLIP3/ XB, YB, ZB, KB, CB, XE, YE, ZE, KE, CE, LAS, ISH
SARE, NTR
COMMON/DFLUSH/ DELTA, SHRK
COMMON/FEATR/FANG, FEATUR
COMMON/LABEL/NUM, NNUM, PNUM
COMMON/LOGI/ISMOOT, IPOOR, IMIX, D IRC, ISPEC, ISMA, LINEA
$R.
1 IHLR, IFRING
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
COMMON/FOORT/ CONTRS, IDVICE, IBAD, SHOSHR, LBLSPC
DIMENSION DIRCA(20), POOR(20)
DIMENSION DRC(3)
LOGICAL IBAD, NTR, NNUM, PNUM, NUM, IFRING, UFRING, SBOUND
C
LOGICAL ICON, HARCOP, TITLE
LOGICAL*1 ITIT(52)
C
ICON=.FALSE.
TITLE=.FALSE.
HARCOP=.FALSE.
NUM=.FALSE.
NNUM=.FALSE.
PNUM=.FALSE.
IBAD=.FALSE.
NTR=.FALSE.
SHRK=0.0
FANG=1.0
CALL REST
IDVICE=-1
FEATUR=.FALSE.
TPL=0
IPH=255
IC=2
IFX=1024
RES=1023.0
IMRES=780
URES=779.
CALL SUMCEN
10 DO 20 I=1, NP
   POOR(I)=0.0
   DIRCA(I)=1.0
20 CONTINUE
IMIX=.TRUE.
SUBROUTINE LABELS(X,Y,CHR,NCNT)

C
INTEGER OUTPUT,ERROR
COMMON/LSTPNT/XLST,YLST
COMMON/DEVI/ INPUT,OUTPUT,ERROR
COMMON/PLTTER/ PLTSIZ,XLAST,YLAST,XDISP
COMMON/INTENS/IPH, IPL, IPB, IFX, IMRES, URES
COMMON/QFORIO/ CONTRS, IDVICE, IBAD, SHOSHR, LBLSPC
INTEGER CHR(5)

C
IF CONTINUOUS-TONE THEN RETURN
C (NOTE: NO CONTINUOUS-TONE DEVICE CALL INCLUDED IN
C THIS VERSION)
C
IF(IDVICE.GT.0) RETURN
C
SCALE INTERNAL COORDINATES TO EXTERNAL COORDINATES
C
XC=X*PLTSIZ
YC=Y*PLTSIZ
C
TEKTRONIX SCOPE IS PICTURE DEVICE
C
40 XC=XC+XDISP
C
IF((XC.GT.RES).OR.(XC.LT.XDISP))RETURN
IF((YC.GT.URES).OR.(YC.LT.1.0))RETURN
IUS=31
CALL MOVARS(XC,YC)
CALL TTOUT(IUS)
DO 555 K=1,NCNT
CALL TTOUT(CHR(K))
555 CONTINUE
IGS=29
CALL TTOUT(IGS)
C
RETURN
END
SUBROUTINE LASTC
COMMON/CENT/ NCENPT
COMMON/CIP/ IP
COMMON/LASPO/ LASP
COMMON/PAAR/ NPL,NPLS
COMMON/PRO/ NP,NJ,CMD,IC,IREAD,SKALE,NPT
DIMENSION IP(1000)
DIMENSION LASEL(250),NCENPT(250)
DIMENSION LASP(21),NPL(2,20),NPLS(20)
DATA NPTMAX/250/
C
DO 10 I=1,NPTMAX
   LASEL(I)=0
10 NCENPT(I)=0
   NP1=NP+1
   DO 20 I=1,21
20 LASP(I)=0
   DO 50 I=1,NP
   I1=NPL(1,I)
   I2=NPL(2,I)
   LASP(I)=LASEL(I1)
   DO 50 J=I1,I2
   IP0INT=LASEL(J)
   IP0INT=IP0INT+1
   IF(IP(IP0INT).GT.0) GO TO 30
   IF(IP(IP0INT+1).LT.0) GO TO 60
   GO TO 40
30 IP0INT=IP0INT+1
   NCENPT(J)=1
40 CONTINUE
   LASP(NP1)=IP0INT
RETURN
END
SUBROUTINE LINE
COMMON/DEV/ INPUT, OUTPUT, ERROR
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
COMMON/SUX/ SPEC, SPEC1, U, X
COMMON/XNO/ XNORM
DIMENSION U(3, 250), X(3, 250), XNORM(3, 250)
INTEGER OUTPUT
REAL LIT
DATA LIT/1HY/

TYPE 170,
ACCEPT 150, ANS

LIT = "Y"
IF(ANS.NE.LIT) GO TO 20
DO 10 J=1, NJ
   DO 10 I=1, 3
     X(I, J)=X(I, J)+SKALE*U(I, J)
10  TYPE 180,
    ACCEPT 150, ANS

LIT = "Y"
IF(ANS.NE.LIT) GO TO 40
DO 30 J=1, NJ
   DO 30 I=1, 3
     U(I, J)=0.0
30   CALL OPEN(5HDISP1, IUNIT, IREAD, IERROR),
     IF(IERROR) 40, 60, 50
50  READ(IUNIT, 160) ((U(I, J), I=1, 3), J=1, NJ)
60  CALL OPEN(5HDISP2, IUNIT, IREAD, IERROR)
     IF(IERROR) 60, 120, 70
70  READ(IUNIT, 160) ((XNORM(I, J), I=1, 3), J=1, NJ)
120 DO 140 J=1, NJ
   DO 130 I=1, 3
130   U(I, J)=XNORM(I, J)-U(I, J)
140  CONTINUE
RETURN

150 FORMAT(A1)

SEE POSSIBLE INCOMPATIBILITIES (1)

160 FORMAT(6E)

SEE POSSIBLE INCOMPATIBILITIES (2)

170 FORMAT(23H+<INCREMENT GEOMETRY?>, $)
180 FORMAT(46H+<SET DISPLACEMENTS AND SCALAR FUNCTION F $ILES >)
19HTO ZERO> ,$)
END
SUBROUTINE MOVABS(X,Y)
CALL TKPLOT(X,Y,1)
RETURN
END
SUBROUTINE MULTDC(II)
COMMON/COOR/ XP
COMMON/NEVE/ NEDGE,MEDGE,IPART
COMMON/RESTO/ XO,DC
COMMON/SHIFTW/ SR,SU
DIMENSION DC(3,3,21)
DIMENSION XP(3,9)
DIMENSION XO(3)

C
X1=XP(1,II)
X2=XP(2,II)
X3=XP(3,II)
XP(1,II)=DC(1,1,IPART)*X1+DC(2,1,IPART)*X2+DC(3,1,IPART)*X3-SR
XP(2,II)=DC(1,2,IPART)*X1+DC(2,2,IPART)*X2+DC(3,2,IPART)*X3-SU
XP(3,II)=DC(1,3,IPART)*X1+DC(2,3,IPART)*X2+DC(3,3,IPART)*X3
RETURN
END
SUBROUTINE MULTDD(II)
COMMON/COOR/ XP
COMMON/NEME/ NEDGE, MEDGE, IPART
COMMON/RESTO/ XO, DC
COMMON/REST1/ RORG, DD
DIMENSION DC(3, 3, 21), DD(3, 3, 20), RORG(3, 20)
DIMENSION XP(3, 9)
DIMENSION XO(3)

C
C

X1 = XP(1, II) - RORG(1, IPART) + XO(1)
X2 = XP(2, II) - RORG(2, IPART) + XO(2)
X3 = XP(3, II) - RORG(3, IPART) + XO(3)
XP(1, II) = DD(1, 1, IPART) * X1 + DD(2, 1, IPART) * X2 + DD(3, 1, IPART) * X3
1 + RORG(1, IPART) - XO(1)
XP(2, II) = DD(1, 2, IPART) * X1 + DD(2, 2, IPART) * X2 + DD(3, 2, IPART) * X3
1 + RORG(2, IPART) - XO(2)
XP(3, II) = DD(1, 3, IPART) * X1 + DD(2, 3, IPART) * X2 + DD(3, 3, IPART) * X3
1 + RORG(3, IPART) - XO(3)
RETURN
END
SUBROUTINE NODNUM
COMMON/PRO/NP,NJ,CMP,IC,IREAD,SKALE,NPT
COMMON/XNO/XNORM
DIMENSION XNORM(3,250)

DO 10 I=1,NJ
   IF(XNORM(3,I),EQ.0.) GO TO 10
   II=I
   CALL OUTNUM(II,XNORM(1,II),XNORM(2,II))
10 CONTINUE
RETURN
END
SUBROUTINE NORMAL
COMMON/COORD/ XP
COMMON/NEME/ NEDGE,MEDGE,IPART
COMMON/XYN/ XN
DIMENSION XN(3,9),XP(3,9)

C GENERALLY TWO ADJACENT EDGES ARE USED TO FIND NORMAL
C L AT A NODE

DO 20 I1=1,NEDGE
   I2=I1+1
   I3=I1-1
   IF(I1.EQ.1) I3=NEDGE
   IF(I1.EQ.NEDGE) I2=1
   X1=XP(1,I2)-XP(1,I1)
   Y1=XP(2,I2)-XP(2,I1)
   Z1=XP(3,I2)-XP(3,I1)
   X2=XP(1,I3)-XP(1,I1)
   Y2=XP(2,I3)-XP(2,I1)
   Z2=XP(3,I3)-XP(3,I1)
   U1=Y1*Z2-Y2*Z1
   U2=X2*Z1-X1*Z2
   U3=X1*Y2-X2*Y1
   U4=SQRT(U1*U1+U2*U2+U3*U3)
   IF(U4.EQ.0.0) GO TO 30
   GO TO 10

C WHEN NODE LIES ALONG STRAIGHT LINE, NORMAL ROUTINE
C IS ALTERED

30 X1=XP(1,I2)-XP(1,I3)
   Y1=XP(2,I2)-XP(2,I3)
   Z1=XP(3,I2)-XP(3,I3)
   Y2=XP(1,MEDGE)-XP(1,I1)
   Y2=XP(2,MEDGE)-XP(2,I1)
   Y2=XP(3,MEDGE)-XP(3,I1)
   U1=Y1*Z2-Y2*Z1
   U2=X2*Z1-X1*Z2
   U3=X1*Y2-X2*Y1
   U4=SQRT(U1*U1+U2*U2+U3*U3)
10 XN(1,I1)=U1/U4
   XN(2,I1)=U2/U4
   XN(3,I1)=U3/U4
20 CONTINUE
RETURN
END
SUBROUTINE OPEN(FILEID, IUNIT, IOP, IERROR)
COMMON/DEVI/ INPUT, OUTPUT, ERROR
INTEGER OUTPUT, ERROR
LOGICAL*1 XNAME(17), FILEID(5)
DATA DSK/3HDSK/ , MTA/3HMTA/
DATA BLANK/10H /
C
DO 5 K=1,17
XNAME(K)= ' ' 
CONTINUE
IERROR=0
10 TYPE 30, (FILEID(K), K=1, 5) 
ACCEPT 20, NCNT, (XNAME(K), K=1, 17) 
IF(NCNT.EQ.0) RETURN
30 FORMAT(1H, ' < ENTER ',5A1, ' FILENAME > ',$, ) 
20 FORMAT(0,17A1)
   IUNIT=10
   CALL CLOSE(IUNIT)
   IF(IOP.GT.0)OPEN(UNIT=IUNIT, NAME=XNAME, TYPE=' OLD', E $PR=50)
   IF(IOP.LT.0)OPEN(UNIT=IUNIT, NAME=XNAME, TYPE=' NEW', E $PR=50)
   IERROR=1
RETURN
C
C
50 TYPE 60, (XNAME(K), K=1, NCNT) 
60 FORMAT(1H, ' *** OPEN FAILURE ON FILE *** ',17A1) 
   IERROR=-1
RETURN
END
SUBROUTINE OUTNUM(NUM, A, B)
DIMENSION ICHAR(5)

C LOAD ASCII CHARACTERS
C
N=1
IF(NUM.GE.10) N=2
IF(NUM.GE.100) N=3
IF(NUM.GE.1000) N=4
DO 10 I=1, N
10 ICHAR(I)=MOD(NUM/10**(N-I), 10)+48
CALL LABELS(A, B, ICHAR, N)
RETURN
END
SUBROUTINE PART
COMMON/CENT/ NCENTP
COMMON/DELSH/ DELTA,SHRK
COMMON/FYSL/ FUN,YY,SLINR
COMMON/LAIF/ JLAST,IFLAG
COMMON/LASPO/ LASP
COMMON/LINES/LINELM
COMMON/LOGI/ISMOOT,IPoor,IMIX,DIRC,ISPEC,ISMA,LINEA
SR,
IHLR,IFRING
COMMON/XNMEME/ NEDGE,MEDGE,IPART
COMMON/PAAR/ NPL,NPLS
COMMON/QFORIO/ CONTRS,IDVICE,IBAD,SHOSHR,L3LSPC
COMMON/RESTO/ XO,DC
COMMON/REST2/ XX
COMMON/VISI/ NNN,IVIS
DIMENSION DC(3,3,21),LASP(21),NPL(2,20),NPLS(20)
1,XX(3,20)
DIMENSION NCENTP(250)
DIMENSION NITRI(9),NNN(8)
DIMENSION XO(3),FUN(3),YY(3),XNT(3)
LOGICAL IFRING,IFRING,IHLR,IPoor
LOGICAL DIRC,ISMA,LINEAR
C
I1=NPL(1,IPART)
I2=NPL(2,IPART)
YY(1)=XO(1)-DELTA*XX(1,IPART)
YY(2)=XO(2)-DELTA*XX(2,IPART)
YY(3)=XO(3)-DELTA*XX(3,IPART)
JLAST=LASP(IPART)
DO 130 J=I1,I2
C
GET POLYGON CONNECTIVITY
C
JTEST=0
IVIS=1
IQUAD=0
CALL CONEL
IFLAG=1
ICEN=J
C
GET COORDINATES AND NORMALS FOR POLYGON
C
CALL COORD(ICEN)
C
IF LINE ELEMENT, IGNORE EXCEPT ON A DRAW
C
IF(IHLR.AND.LINELM.EQ.1) GO TO 130
IF(IPoor.AND.LINELM.EQ.1) GO TO 130
C CHECK FOR POLYGON ON EDGE
C
IF(IHLR.AND.IPOOR) CALL VISIP
IF(IVIS) 130,20,110
C
SUBDIVIDE POLYGONS ON EDGE INTO TRIANGLES
C
IF NO CENTER POINT SPECIFIED, QUADRILATRALS ARE SUBDIVIDED
C INTO TWO TRIANGLES
C
20 IF(NEDGE.EQ.4.AND.NCENPT(J).EQ.0) GO TO 140
30 NNN(3)=MEDGE
DO 40 I=1,NEDGE
 NNN(1)=I
 NNN(2)=I+1
 IF(I.EQ.NEDGE) NNN(2)=1
 CALL VISIT
40 NITRI(I)=IVIS
 NITRI(MEDGE+1)=NITRI(1)
C
DETERMINE WHICH INTERIOR EDGES TO BE DISPLAYED FOR
C TRIANGULAR ELEMENTS OF SUBDIVIDED POLYGONS
C
DO 90 I=1,NEDGE
 NNN(1)=I
 NNN(2)=I+1
 IF(I.EQ.NEDGE) NNN(2)=1
 JTEST=4
 IF(NITRI(I).NE.NITRI(I+1)) JTEST=2
 IF(I.EQ.1) GO TO 100
 IF(NITRI(I).NE.NITRI(I-1).AND.JTEST.EQ.2) JTE
 $ST=3
 IF(NITRI(I).NE.NITRI(I-1).AND.JTEST.EQ.4) JTE
 $ST=1
50 NNEDEGE=MEDGE
 NEDGE=3
 IF(IPOOR) GO TO 80
 CALL POLY
60 NEDGE=NNEDEGE
 GO TO 90
70 IF(NITRI(I).EQ.-1) GO TO 70
 GO TO 60
80 CONTINUE
 GO TO 130
100 IF(NITRI(I).NE.NITRI(NEDGE).AND.JTEST.EQ.2) JTES
 $ST=3
 IF(NITRI(I).NE.NITRI(NEDGE).AND.JTEST.EQ.4) JTES
 $ST=1
GO TO 50
110   DO 120 I=1,NEDGE
120   NNN(I)=I
       CALL POLY
130   CONTINUE
       RETURN

C \quad \text{QUAD TO TWO TRI ROUTINE} C
140   NEDGE=3
150   IQUAD=1
       DO 150 I=1,3
160   NNN(I)=I
       CALL VISIT
       IF (IVIS.EQ.-1.AND.IPOOR) GO TO 160
       CALL POLY
170   DO 170 I=1,2
180   NNN(I)=I+2
190   NNN(3)=1
       CALL VISIT
       IF (IVIS.EQ.-1.AND.IPOOR) GO TO 130
       CALL POLY
       GO TO 130

C \quad \text{END OF QUAD TO TRI ROUTINE} C
C
END
SUBROUTINE PIVOT
COMMON/DEVI/ INPUT,OUTPUT,ERROR
COMMON/REST1/ RORG,DD
DIMENSION DD(3,3,20),RORG(3,20)
DIMENSION LIT(3)
INTEGER OUTPUT
REAL LIT
DATA LIT/1HX,1HY,1HZ/
C
C
10 TYPE 60,
10 TYPE 100,
ACCEPT 80, I1,I2,X1,X2
IF(I1.EQ.0) GO TO 30

C
C
LIT(1) = "X"
C
IF(X1.EQ.LIT(1)) I3=1
C
LIT(2) = "Y"
C
IF(X1.EQ.LIT(2)) I3=2
C
LIT(3) = "Z"
C
IF(X1.EQ.LIT(3)) I3=3
DO 20 I=I1,I2
    ISAFE=I
    CALL ROTAT(DD,I3,X2,ISAFE)
20 CONTINUE
GO TO 10

30 TYPE 70,
40 TYPE 100,
ACCEPT 90, I1,I2,X1,X2,X3
IF(I1.EQ.0) RETURN
DO 50 I=I1,I2
    RORG(1,I)=X1
    RORG(2,I)=X2
50 RORG(3,I)=X3
GO TO 40

60 FORMAT(27H+<PARTS I1/I2, AXI S, ANG LE>/)
70 FORMAT(31H+<PARTS I1/I2, RELATIVE ORIGIN>/)
C
C
SEE POSSIBLE INCOMPATIBILITIES (1)
C
A0 FORMAT(2I,A1,1X,F10.0)
90 FORMAT(2I,3F10.0)
C
C
SEE POSSIBLE INCOMPATIBILITIES (2)
100 FORMAT(5H>>> ,$)
END
SUBROUTINE PLTLIN(A,B,C,D)

C INTEGER OUTPUT,ERROR
COMMON/DEVI/ INPUT,OUTPUT,ERROR
COMMON/TEK/XBAR,YBAR,SF,RES
COMMON/PLTTER/ PLTSIZ,XLAST,YLAST,XDISP
COMMON/OFORIO/ CONTRS,IDVICE,IBAD,SHOSHR,LBLSPC
COMMON/LSTPNT/XLST,YLST
DIMENSION X(2),Y(2)

C IF CONTINUOUS-TONE THEN RETURN
C (NOTE: NO CONTINUOUS-TONE DEVICE CALL INCLUDED IN THIS VERSION)
C IF(IDVICE.GT.0) RETURN
C
SCALE INTERNAL COORDINATES TO DEVICE COORDINATES

X(1)=A*PLTSIZ
Y(1)=B*PLTSIZ
X(2)=C*PLTSIZ
Y(2)=D*PLTSIZ

C TEKTRONIX SCOPE IS PICTURE DEVICE

60 X(1)=XDISP+X(1)
X(2)=XDISP+X(2)
IF(XLAST.EQ.X(1).AND.YLAST.EQ.Y(1)) GO TO 70
IX=C

C SUBROUTINE MOVARS IS A ROUTINE WHICH LIFTS
C AND MOVES THE PEN TO THE GIVEN LOCATION
C
CALL MOVARS(X(1),Y(1))

C SUBROUTINE DRWABS IS A ROUTINE WHICH DRAGS
C THE PEN TO THE GIVEN LOCATION
C
70 CALL DRWABS(X(2),Y(2))

C XLAST AND YLAST ARE USED TOTALLY WITHIN DEVICE
C
YLAST=X(2)
YLAST=Y(2)

C RETURN

C END
SUBROUTINE POINTS(ICEN)
COMMON/CENT/ NCENPT
COMMON/CONN/ NN
COMMON/CONT/ CONT
COMMON/COOR/ XP
COMMON/FYSL/ FUN,YY,SLINR
COMMON/NME/E NEDGE,MEDGE,IPART
COMMON/PRO/ NP,NJ,CMID,IC,IREAD,SKALE,NPT
COMMON/SPUX/ SPEC,SPEC1,U,X
DIMENSION U(3,250),X(3,250)
DIMENSION NCENPT(250)
DIMENSION CONT(9),NN(9),XP(3,9)
DIMENSION FUN(3),YY(3)

C
DO 30 II=1,MEDGE
   N1=NN(II)
   IF(II.EQ.MEDGE.AND.NCENPT(ICEN).NE.1) RETURN
10  DO 20 I=1,3
20   XP(I,II)=X(I,N1)+SKALE*U(I,N1)-YY(I)
    JSEND=II
    CALL MULTDD(JSEND)
30  CALL MULTDC(JSEND)
RETURN
END
SUBROUTINE POLNOD
COMMON/DEVI/INPUT,OUTPUT,ERROR
COMMON/LABEL/NUM,NNUM,PNUM
COMMON/PRO/NP,NJ,CMD,IC,IREAD,SCALE,NPT
LOGICAL NNUM,PNUM,NUM
INTEGER OUTPUT
REAL LIT
DATA LIT/4HPOLY/

LIT = "POLY"

IF(CMD.EQ.LIT) GO TO 20
NNUM=.NOT.NNUM
NUM=NNUM.OR.PNUM
IF(NNUM) GO TO 10
TYPE 40,
RETURN
10 TYPE 50,
RETURN
20 PNUM=.NOT.PNUM
NUM=NNUM.OR.PNUM
IF(PNUM) GO TO 30
TYPE 60,
RETURN
30 TYPE 70,
RETURN
40 FORMAT(26H+<NODE NUMBERING DISABLED>/)
50 FORMAT(25H+<NODE NUMBERING ENABLED>/)
60 FORMAT(29H+<POLYGON NUMBERING DISABLED>/)
70 FORMAT(28H+<POLYGON NUMBERING ENABLED>/)
END
SUBROUTINE POLNUM
COMMON/PNN/ NN
COMMON/LALPN/ LALPN
COMMON/LALPN/ LALPN
COMMON/LALPN/ LALPN
COMMON/LALPN/ LALPN
COMMON/LALPN/ LALPN
COMMON/LALPN/ LALPN
COMMON/PN/ NP, NJ, CMDIC, IREAD, SKALE, NPT
COMMON/XNO/ XNORM
DIMENSION XNORM(3,250)
DIMENSION LASP(21), NPL(2,20), NPLS(20)
DIMENSION NN(9)
C
DO 30 I=1, NP
   I1=NPL(1, I)
   I2=NPL(2, I)
   IF(NPLS(I), EQ, 0) GO TO 30
   JLAST=LASP(I)
   DO 20 J=I1, I2
      CALL CONEL
      A=0.
      B=0.
      N=0
      DO 10 K=1, NEDGES
         N2=NN(K)
         IF(XNORM(3, N2), EQ, 0) GO TO 10
         N=N+1
         A=A+XNORM(1, N2)
         B=B+XNORM(2, N2)
      10 CONTINUE
      IF(N.LT, NEDGES) GO TO 20
      RN=N
      A=A/RN
      B=B/RN
      J2=J
      CALL OUTNUM(J2, A, B)
30 CONTINUE
RETURN
END
SUBROUTINE POLY
COMMON/NEME/MEDGE,MEDGE,IPART
COMMON/SU80/JTEST,XNT,IOQUAD
COMMON/VTSI/NNN,IVIS
DIMENSION NNN(8)
DIMENSION XNT(3)
C
DO 10 I=1,NEDGE
   I1=NNN(I)
   I2=NNN(I+1)
C
JTEST=0 EXCEPT FOR TRIANGULAR ELEMENTS OF SUBDIVIDED POLYGONS ON EDGE
C
IF(JTEST.GT.0) GO TO 20
IF(I.EQ.NEDGE) I2=NNN(1)
10 CALL DRAW(I1,I2)
   RETURN
20 CALL DRAW(I1,I2)
   IF(JTEST.NE.1) GO TO 30
   I1=NNN(3)
   I2=NNN(1)
   GO TO 50
30 IF(JTEST.NE.2) GO TO 40
   I1=NNN(2)
   I2=NNN(3)
   GO TO 50
40 IF(JTEST.NE.3) RETURN
   I1=NNN(2)
   I2=NNN(3)
   CALL DRAW(I1,I2)
   I1=NNN(3)
   I2=NNN(1)
50 CALL DRAW(I1,I2)
   RETURN
END
SUBROUTINE READ
COMMON/CIP/ IP
COMMON/CLIP3/XB,YB,ZB,KB,CB,XE,YE,ZE,KE,CE,LAS,ISHA
COMMON/DEVF/ INPUT,OUTPUT,ERROR
COMMON/LOGI/ ISMOOT,IPoor,IMIX,DIC3,ISPEC,ISMA,LINE
COMMON/OLPA/NPOL,NSMAX
COMMON/PAAR/NPL,NPLS
COMMON/PRO/NP,NJ,CMD,IC,IREAD,SKALE,NPT
COMMON/QORIO/CONTRO,IDVICE,IBAD,SHOSHR,LBLSPC
COMMON/SPUX/SPEC,SPEC1,U,X
COMMON/TRIS/IKILL,XT
DIMENSION IP(1000)
DIMENSION U(3,250),X(3,250)
DIMENSION NPL(2,20),NPLS(20)
LOGICAL ISPEC,IBAD,NTR
INTEGER OUTPUT,CONMAX
DATA NP/0/,NJ/0/,NPT/0/,Y/1HY/
DATA IREAD/1/
DATA LIT/1HY/

C DATA BELOW WILL NEED TO BE CHANGED IF DIMENSIONS ARE ALTERED
DATA NMAX/20/,NJMAX/250/,NPTMAX/250/,NFRINM/11/,CON
MPMAX/1000/

DO 10 I=1,NJMAX
DO 10 J=1,3
U(J,I)=0.0
X(J,I)=0.0
10 CONTINUE
DO 20 I=1,CONMAX
20 IP(I)=0
DO 30 I=1,NMAX
NPLS(I)=1
DO 30 J=1,2
30 NPL(J,I)=0

C READ GEOMETRY FILE
C
DO 40 CALL OPEN(5HGFOM,,IUNIT,IREAD,IERROR)
TF(IERROR) 40,70,50
DO 50 READ(IUNIT,140) NP,NJ,NPT,NCON
IF(NP.GT.NPOL) TYPE 200,
NPOL=NP
TF(NCON,NE,0) GO TO 60
TYPE 220,
STOP
50 IF(NP.GT.NP MAX) TYPE 160, NP, NP MAX
IF(NJ.GT.NJ MAX) TYPE 170, NJ, NJ MAX
IF(NPT.GT.NPT MAX) TYPE 180, NPT, NPT MAX
IF(NCON.GT.NCON MAX) TYPE 210, NCON, NCON MAX
IF(NP.GT.NP MAX ORNJ.GT.NJ MAX OR NPT.GT.NPT MAX) STOP
$P
IF(NCON.GT.NCON MAX) STOP
READ(IUNIT,140) ((NPL(I,J),I=1,2),J=1,NP)
READ(IUNIT,150) ((X(I,J),I=1,3),J=1,NJ)
READ(IUNIT,140) (IP(I),I=1,NCON)
TYPE 190, NP,NJ,NPT
READ DISPLACEMENT FILE
70 IF(NP.EQ.0) GO TO 40
CALL OPEN(5HDISP.,IUNIT,IREAD,IERROR)
IF(IERROR) 70,90,80
90 READ(IUNIT,150) ((U(I,J),I=1,3),J=1,NJ)
CALL LASTC
IBAD=.FALSE.,
NTR=.FALSE.,
ICODE=1
CALL SEPART(ICODE)
RETURN
140 FORMAT(16I5)
150 FORMAT(6E12.5)
160 FORMAT(16H+<NP .GT.NP MAX .GT. 2I5,1H>/)
170 FORMAT(16H+<NJ .GT.NJ MAX .GT. 2I5,1H>/).
180 FORMAT(16H+<NPT .GT.NPT MAX .GT. 2I5,1H>/)
190 FORMAT(7H+<READ:.I5,7H PARTS:.I5,13H COORDINATES:,
115,11H ELEMENTS.>/)
200 FORMAT(4H+<WARNING: NEW FILE HAS MORE PARTS THAN 0
$LD FILE>/,
127H <ISSUE "RESTORE" COMMAND> /)
210 FORMAT(15H+<CONN. LENGTH(.I7,18H) EXCEEDS MAX. OF(. 
$17,2H)>/
220 FORMAT(53H+<ERROR: OLD DATA FORMAT] RUN FILE THROUG
$H "UPDATE" >/
END
SUBROUTINE REST
COMMON/PRO/NP,NJ,CMD,IC,ITRED,SKALE,NPT
COMMON/REST0/ XO,DC
COMMON/REST1/ RORG,DD
COMMON/SHlFTW/ SR,SU
DIMENSION DC(3,3,21),DD(3,3,20),RORG(3,20)
DIMENSION XO(3)
C
SR=0.0
SU=0.0
NP1=NP+1
DO 20 K=1,NP1
   L=K
   IF(K.EQ.NP1) L=K-1
   DO 20 J=1,3
      DO 10 I=1,3
         DC(I,J,K)=0.0
   10     DD(I,J,L)=0.0
         RORG(J,L)=0.0
         DC(J,J,K)=1.0
   20 DD(J,J,L)=1.0
RETURN
END
SUBROUTINE ROTA
COMMON/DEV1/INPUT, OUTPUT, ERROR
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
COMMON/RESTO/ XO, DC
COMMON/ROTIM/ NONROT
DIMENSION DC(3,3,21), NONROT(20)
DIMENSION LIT(3), X0(3)
INTEGER OUTPUT
LOGICAL NONROT
REAL LIT
DATA LIT/1HX, 1HY, 1HZ/

C TYPE 60,
ACCEPT 50, X1, X2
I1=0
C
C LIT(1) = "X"
C
C IF(X1.EQ.LIT(1)) I1=1
C
C LIT(2) = "Y"
C
C IF(X1.EQ.LIT(2)) I1=2
C
C LIT(3) = "Z"
C
C IF(X1.EQ.LIT(3)) I1=3
C IF(I1.GT.0) GO TO 10
C TYPE 40, X1
C RETURN
10 NP1=NP+1
DO 30 I=1, NP1
   ISAFE=I
   IF(I.F0 .NP1) GO TO 20
   IF(NONROT(ISAFE)) GO TO 30
20 CALL ROTAT(DC, I1, X2, ISAFE)
30 CONTINUE
C RETURN
40 FORMAT(2H+<A1,7H AXIS?>>)
C
C SEE POSSIBLE INCOMPATIBILITIES (1)
C
C 50 FORMAT(A1,1X,F10.0)
C
C SEE POSSIBLE INCOMPATIBILITIES (2)
C
C 60 FORMAT(15H+<AXIS, ANGLE> ,$)
END
SUBROUTINE ROTAT(X, IDIR, THETA, K)
DIMENSION X(3,3,20)

COMPUTE SINE AND COSINE

CS = COS(THETA*0.017453)
SS = SIN(THETA*0.017453)
GO TO (10, 30, 50), IDIR

10 DO 20 I = 1, 3
   X2 = X(I, 2, K)
   X3 = X(I, 3, K)
   X(I, 2, K) = CS*X2 - SS*X3
   20 X(I, 3, K) = CS*X3 + SS*X2
RETURN

X2 DIRECTION

30 DO 40 I = 1, 3
   X1 = X(I, 1, K)
   X3 = X(I, 3, K)
   X(I, 1, K) = CS*X1 + SS*X3
   40 X(I, 3, K) = CS*X3 - SS*X1
RETURN

X3 DIRECTION

50 DO 60 I = 1, 3
   X1 = X(I, 1, K)
   X2 = X(I, 2, K)
   X(I, 1, K) = CS*X1 - SS*X2
   60 X(I, 2, K) = CS*X2 + SS*X1
RETURN
END
SUBROUTINE SCAL
COMMON/DEV1/INPUT, OUTPUT, ERROR
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
INTEGER OUTPUT

C
TYPE 20, SKALE
ACCEPT 10, SKALE
RETURN

C
SEE POSSIBLE INCOMPATIBILITIES (1)

C
10 FORMAT(F10.0)

C
SEE POSSIBLE INCOMPATIBILITIES (2)

C
20 FORMAT(29H<DISPLACEMENT SCALE FACTOR (.F7.1,3H>), $$
END
SUBROUTINE SCOP
COMMON/INTENS/IPH, IPL, IPR, IFX, IMRES, URES
COMMON/SCMR/DOZ, FIELD, FIELD, TANAL, RES
COMMON/TRIS/IKILL, XT
DATA YES/1HY/

C

C

C

TYPE 160
ACCEPT 100,ANS
IKILL=0
IF(ANS.EQ.YES)IKILL=1
IF(IKILL.EQ.0)GO TO 50
TYPE 170
ACCEPT 100,ANS
IF(ANS.EQ.YES)IKILL=2
50 TYPE 180,IMRES
180 FORMAT(1H,'< ENTER DISPLAY RESOLUTION (1 - ',
$14,'>') > ',$:)
ACCEPT 190,IM
190 FORMAT(I)
IF(IM.LT.100)IM=100
IF(IM.GT.IMRES)IM=IMRES
URES=IM*1.0-1.0
RETURN
100 FORMAT(A1)
160 FORMAT(1H,'<SUPPRESS COORDINATE TRAID?>',$:)
170 FORMAT(1H,'<DISPLAY NON-ROTATED TRAID?>',$:)
END
SUBROUTINE SEPAR(ICODE)
COMMON/DEV1/ INPUT, OUTPUT, ERROR
COMMON/PARR/ NPL, NPLS
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
DIMENSION NPL(2, 20), NPLS(20)
INTEGER OUTPUT

DO 10 I=1, NP
   NPLS(I) = 0
   IF(NPL(1, I) .LE. NPL(2, I)) NPLS(I) = 1
10 CONTINUE
   IF(ICODE.EQ.1) RETURN
20 TYPE 50,
30 TYPE 70,
   ACCEPT 60, II1, II2
   IF(II1 .LE. 0) RETURN
   IF(II1 .GT. NP) GO TO 20
   IF(II2 .LT. II1) II2 = II1
   IF(II2 .GT. NP) II2 = NP
   DO 40 J=II1, II2
40 NPLS(J) = 0
   GO TO 30
50 FORMAT(1H 'ENTER PARTS I1/I2 TO BE WITHHELD FROM DISPLAY'>/')

SEE POSSIBLE INCOMPATIBILITIES (1)

60 FORMAT(16I)

SEE POSSIBLE INCOMPATIBILITIES (2)

70 FORMAT(5H >>> $)
END
SUBROUTINE SHRINK
COMMON/COORD/ XP
COMMON/DELSH/ DELTA, SHRK
COMMON/NEME/ NEDGE, MEDGE, IPART
DIMENSION XP(3,9)

C
F1=1.0-SHRK
DO 10 I=1,NEDGE
   DO 10 J=1,3
10 XP(J,I)=XP(J,I)*F1+SHRK*XP(J,MEDGE)
RETURN
END
SUBROUTINE SHRKR
COMMON/DELSH/Delta,SHRK
COMMON/DEVI/INPUT,OUTPUT,ERROR
INTEGER OUTPUT

C
10 TYPE 30, SHRK
ACCEPT 20, SHRK
IF(SHRK.LT.0.0.0.0.R.0.R.0.SHRK.GE.1.0) GO TO 10
RETURN

C SEE POSSIBLE INCOMPATIBILITIES (1)

C
20 FORMAT(F10.0)

C SEE POSSIBLE INCOMPATIBILITIES (2)

C
30 FORMAT(14,'<CHANGE SHRINK FACTOR (0.0 - 0.99) FROM
$ ('$,$,F5.2,'') TO>',$)
END
SUBROUTINE SUMCEN
COMMON/CIP/ IP
COMMON/DEF/ INPUT, OUTPUT, ERROR
COMMON/LASPO/ LASP
COMMON/LOGI/ ISMOTH, IP00R, IMIX, DIRC, ISPEC, ISMA, LINEA
$R,
1   IHLR, IFRING
COMMON/PAAR/ NPL, NPLS
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
COMMON/RESTO/ XO, DC
COMMON/SCMR/ DOZ, FIOLD, FIELD, TANAL, RES
COMMON/SHIFTW/ SR, SU
COMMON/SPLX/ SPEC, SPEC1, U, X
COMMON/ZRANGE/ ZMIN, ZMAX, ZMIOLD, ZMAOLD
DIMENSION IP(1000)
DIMENSION U(3, 250), X(3, 250)
DIMENSION DC(3, 3, 21), LASP(21), NPL(2, 20), NPLS(20)
DIMENSION X0(3)
LOGICAL ISPEC
INTEGER OUTPUT
REAL LIT
DATA LIT/4HSUMM/
C
SR=0.0
SU=0.0
DO 10 II=1, NP
   IF(NPLS(II).NE.0) GO TO 20
10   CONTINUE
   RETURN
20   I1=LASP(II)+1
   I2=IIABS(IP(I1))
   XL=X(1, I2)-XO(1)
   XS=XL
   YL=X(2, I2)-XO(2)
   YS=YL
   ZL=X(3, I2)-XO(3)
   ZS=ZL
   UL=U(1, I2)
   US=UL
   DO 50 J=II, NP
      IF(NPLS(J).EQ.0) GO TO 50
      I1=LASP(J)+1
      I2=LASP(I1+1)
      DO 40 J=I1, I2
         K1=IIABS(IP(J))
         IF(K1.EQ.0) GO TO 40
         X1=X(1, K1)-XO(1)
         Y1=Y(2, K1)-XO(2)
         Z1=Y(3, K1)-XO(3)
         IF(XL.LT.X1) XL=X1
      40   CONTINUE
50   CONTINUE
   RETURN
   END
IF(YS.GT.X1) XS=X1
IF(YL.LT.Y1) YL=Y1
IF(YS.GT.Y1) YS=Y1
IF(ZL.LT.Z1) ZL=Z1
IF(ZS.GT.Z1) ZS=Z1
DO 30 K2=1,3
    X1=U(K2,K1)
    IF(UL.LT.X1) UL=X1
    IF(US.GT.X1) US=X1
30 CONTINUE
IF(SL.LT.X1) SL=X1
IF(SS.GT.X1) SS=X1
CONTINUE
50 CONTINUE
TYPE 70, XS,XL,YS,YL,ZS,ZL
IF(US.NE.0.0.) OR(UL.NE.0.0.) TYPE 100, US,UL
LIT = "SUMM"
IF(CMD.EQ.LIT) GO TO 60
XO(1)=XO(1)+(XS+XL)/2.
XO(2)=XO(2)+(YS+YL)/2.
XO(3)=XO(3)+(ZS+ZL)/2.
XL=YL-YS
IF(YL-YS.GT.XL) XL=YL-YS
IF(ZL-ZS.GT.XL) XL=ZL-ZS
DO2=3.5*XL
ZMIN=0.1
ZMAX=7.0*XL
ZMIOLD=ZMIN
ZMAOLD=ZMAX
FIOLD=28.
TANAL=0.24932365
60 TYPE 80, (XO(I),I=1,3)
TYPE 90, DO2,FIOLD,ZMIOLD,ZMAOLD
ZMIN=ZMIN*TANAL
ZMAX=ZMAX*TANAL
RETURN
$3,5H <Z< ,
1 F9.3,1H>/)
80 FORMAT(19H+<ORIGIN MOVED TO:,3F9.3,1H>/)
90 FORMAT(40H+<DISTANCE TO ORIGIN, ANGLE, ZMIN, ZMAX:,
$F6.3,
1 F6.2,2F8.3,1H>/)
SEE POSSIBLE INCOMPATIBILITIES (3)
100 FORMAT(2H+<,1PE12.5,19H <VECTOR FUNCTION< ,1PE12.5,
$1H>/)
110 FORMAT(2H<,1PE12.5,19H<SCALAR FUNCTION<,1PE12.5,
$1H>/)
END
SUBROUTINE TITLE
COMMON/INTENS/IPM, IPL, IPR, IFX, IMRES, URES
COMMON/TITLE/TITLE, NUMCHR, ITIT
LOGICAL TITLE
LOGICAL*1 ITIT(52)

C

TITLE=.FALSE.
TYPE 100
100 FORMAT(1H,'<DO YOU WANT A TITLE? >',$)
ACCEPT 200,ANS
200 FORMAT(A1)
IF(ANS.EQ.'Y')TITLE=.TRUE.
C
C**** CLEAR THE LINE
C
250 DO 300 K=1,52
   ITIT(K)= ' ' 
300 CONTINUE
IF(TITLE)GO TO 400
RETURN
C
C*** GET THE TITLE LINE
C
400 MAXC=URES/15
   IF(MAXC.GT.52)MAXC=52
   TYPE 500,MAXC
500 FORMAT(/1H,'*** MAXIMUM NUMBER OF CHARACTERS WITH 
   $',/1H,'$*** CURRENT RESOLUTION IS =',/1H,'$ 
   $ENTER YOUR 
   $LINE $')
   ACCEPT 600,NUMCHR,(ITIT(K),K=1,MAXC)
600 FORMAT(Q,52A1)
   IF(NUMCHR.LE.MAXC)RETURN
C
C
TYPE 700
700 FORMAT(1H,' MAXIMUM NUMBER OF CHARACTERS EXCEEDED'
   $',/1H,'$ '<DO YOU WANT TO TRY AGAIN? >',$)
   ACCEPT 200,ANS
   IF(ANS.EQ.'Y')GO TO 250
RETURN
END
SUBROUTINE TKPLOT(X, Y, N)
INTEGER*2 HIXY(32), LOWY(32), LOWX(32)
INTEGER*2 BUF(4), US, GS, ESC, SUB, FF, CR
DATA US, GS, ESC, SUB, FF, CR / 31, 29, 27, 26, 12, 13 /
DATA HIXY / 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45 
  46, 47, 48, 149, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63 /
DATA LOWY / 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 1  
  122, 123, 2124, 125, 126, 127 /
DATA LOWX / 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77 
  78, 79, 80, 181, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95 /
COMMON / CONTR/US, GS, ESC, SUB, FF, CR
PX = MAX0 (0, MIN1 (1023, X))
PY = MAX0 (0, MIN1 (780, Y))
J = 1 + PY / 32
BUF(1) = HIXY(J)
J = 1 + AMOD (PY, 32,)
BUF(2) = LOWY(J)
J = 1 + PX / 32
BUF(3) = HIXY(J)
J = 1 + AMOD (PX, 32,)
BUF(4) = LOWX(J)
IF (N .LT. 1) RETURN
IF (N .GT. 2) RETURN
IF (N .EQ. 1) CALL TTOUT(GS)
CALL TTOUT(BUF(1))
CALL TTOUT(BUF(2))
CALL TTOUT(BUF(3))
CALL TTOUT(BUF(4))
RETURN
END
SUBROUTINE PLERAS
INTEGER*2 FF, ESC, GS
ESC = 27
FF = 12
GS = 29
CALL TTOUT(ESC)
CALL TTOUT(FF)
RETURN
END
SUBROUTINE TRAN
COMMON/DEVI/INPUT, OUTPUT, ERROR
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
COMMON/RESTO/ XO, DC
COMMON/SHIFTW/ SR, SU
DIMENSION DC(3, 3, 21)
DIMENSION XO(3)
INTEGER OUTPUT
REAL LIT
DATA LIT/4HSHIF/

LIT = "SHIF"

IF(CMD.EQ.LIT) GO TO 10
TYPE 40, (XO(I), I=1, 3)
ACCEPT 20, (XO(I), I=1, 3)
RETURN
10 TYPE 50, SR, SU
ACCEPT 30, SR, SU
RETURN

SEE POSSIBLE INCOMPATIBILITIES (1)

20 FORMAT(3F10.0)
30 FORMAT(2F10.0)

SEE POSSIBLE INCOMPATIBILITIES (2)

40 FORMAT(15H+<ORIGIN FROM (.3F8.2,6H).TO> ,$)
50 FORMAT(20H+<SHIFT WINDOW X,Y (.2F8.2,3H)>, $)
END
*TITLE   TTOUT
.*CALL   TTYOUT

******************************************************************************

*                     TTOUT  FORT CHAR OUT TO TTY (CRT)                   *
*                     CALLING SEQUENCE                                   *
*                     CALL TTOUT(N)                                      *
*                     OR                                              *
*                     CALL TTOUT(M,J)                                   *
*                     WHERE N IS A ONE WORD INTEGER WITH                 *
*                     CHARACTER RIGHT JUSTIFIED                        *
*                     M IS A ONE WORD (INTEGER*2)                        *
*                     ARRAY ONE CHARACTER PER WORD RIGHT                *
*                     JUSTIFIED AND J IS THE LAST POSITION             *
*                     IN THE ARRAY TO BE OUTPUT                        *
*                     I.E.  M(L)*L=1,J                                   *
*                     
******************************************************************************

TTOUT:          MOV   (R5)+,R4                    ;GET NUMBER OF PARMS
               INC   #177000,R4                   ;CLEAR HIGH ORDER BITS
               CMP   R4,#1                      ;MORE THAN ONE PARM MEANS
               BNE   ARRAY                     ;ADDRESS OF CHAR
               MOV   (R5),R3                    ;CHAR
               MOV   (R3),R0                    ;CHAR
               .TTYOUT RS                       ;RETURN TO CALLER
               RTS                            ;RETURN TO CALLER

ARRAY:         MOV   (R5)+,R3                    ;POINT TO BEGGING OF
               MOV   (R5),R4                    ;POINT TO NUMBER OF WORDS IN
               MOV   (R4),R4                    ;VALUE

LOOP:          MOV   (R3)+,R0                    ;FIRE OFF A BYTE
               DEC   R4                        ;GET NEXT ONE
               BNE   LOOP                      ;RETURN TO CALLER
               RTS                            ;RETURN TO CALLER

.END
SUBROUTINE VIEDRA
COMMON/AN1/ DA, DT, DR, CPF, IFR1, IFR2, DDOZ, DDELTAA, IPM,
$NFRAME,
1 SFDEL, AMISHR
COMMON/CLIPS/ XB, YB, ZB, KB, CB, XE, YE, ZE, KE, CE, LAS, ISH
$ARE, NTR
COMMON/ANTCR/ICON, HARCOP
COMMON/DELSH/ DELTA, SHRK
COMMON/DEVI/ INPUT, OUTPUT, ERROR
COMMON/FYSL/ FUN, YY, SLINR
COMMON/LABEL/NUM, NNUM, PNUM
COMMON/LAST/ JLAST, IFLAG
COMMON/LOGI/ ISMOTH, IPOOR, IMIX, DIRC, ISPEC, ISMA, LINEA
$R,
1IHLR, ITFRING
COMMON/NOME/ NEDGE, MEGDE, IPART
COMMON/PAAR/ NPL, NPLS
COMMON/POORPT/ POOR, DIRCA
COMMON/PRO/ NP, NJ, CMD, IC, IREAD, SKALE, NPT
COMMON/QFROI0/ CONTROS, IDVICE, IBAD, SHOSHR, LBLSPC
COMMON/RFSTO/ X0, DC
COMMON/REST1/ RORG, DD
COMMON/SCMR/ DOZ, FIELO, FIELD, TANAL, RES
COMMON/SMOTH/ JSMOTH
COMMON/XMO/ XNORM
DIMENSION XNORM(3, 250)
DIMENSION DA(3, 20), DC(3, 3, 21), DD(3, 3, 20), DIRCA(20)
1, JSMOTH(20), NPL(2, 20), NPLS(20), POOR(20), RORG(3, 20)
DIMENSION X0(3), OT(3), DR(3), FUN(3), YY(3)
LOGICAL LINEAR, ISMA, IHLR, IBAD, ITFRING, DIRC, IPOOR
LOGICAL IPM, ISHARE, CONTROS, SBOUND
LOGICAL SHOSHR, NTR, NNUM, PNUM, NUM, ICON, HARCOP
INTEGER OUTPUT
REAL LIT
DATA LIT/4HVIEW/
C
   IHLR= .FALSE.
C
C
   LIT = "VIEW"
C
   IF(CMD, EQ, LIT) IHLR= .TRUE.
   IF(IDVICE, LE, 0) GO TO 10
   IHLR= .TRUE.
   CONTROS= .FALSE.
   NNUM= .FALSE.
   PNUM= .FALSE.
   10 ISHARE= IDVICE, LT, 0
   SLINR= 0.0
   XMAGN= SKALE
   AMPZ= 1.0
XFRAME=NFRAME
DO 120 IIMOVE=1,NFRAME
   IF(NFRAME.EQ.0) GO TO 40
   INCREMENT DISPLACEMENTS, ROTATIONS, TRANSLATIONS, ETC. FOR ANIMATE
   XIMOVE=IIMOVE
   XMAGN=XMAGN+SFDEL
   SKALE=XMAGN
   IF(LINEAR) SLINR=XIMOVE/XFRAME
   IF(LINEAR) SKALE=XMAGN*SLINR
   IF(CPF.EQ.0.0) GO TO 20
   ANG=360.0*CPF*XIMOVE
   SKALE=XMAGN*SIN(ANG*.017453)
20   AMP=180.0*XIMOVE/XFRAME
   AMP=COS(AMP*.017453)
   DAMP=0.5*(AMPZ-AMP)
   IF(.NOT.ISMA) DAMP=1.0/XFRAME
   AMPZ=AMP
   DOZ=DOZ+DDOZ*DAMP
   SHRK=SHRK+ANISHR*DAMP
   DELTA=DELTA+DDELTA*DAMP
   DO 30 T=1,3
      ISAFE=YES
      XO(I)=XO(I)+DT(I)*DAMP
   DO 30 J=1,NP
      JSAFE=N
      DDD=DR(I,J)*DAMP
      IF(DDD.NE.0.0) CALL ROTAT(DC,ISAFE,DDD,JSA)
$FE)
      DDD=DA(I,J)*DAMP
      IF(DDD.NE.0.0) CALL ROTAT(DD,ISAFE,DDD,JSA)
$FE)
30 CONTINUE
   IF(.NOT.IPM) GO TO 120
   IF(IIMOVE.LT.IFR1.OR.IIMOVE.GT.IFR2) GO TO 120
   PROCESS PARTS INDIVIDUALLY
   CALL RGNNFRM
   CALL COARRO
   IF(NFRAME.EQ.0) HARCOP=.FALSE.
   IBAD=.FALSE.
   DO 50 T=1,NJ
50   XNORM(3,I)=0.
   DO 60 T=1,NP
      IPART=T
      IPOOR=.FALSE.
      DIRC=.FALSE.
XI=POOR(IPART)
X2=DIRCA(IPART)

IF(X1.EQ.1.0) IPOOR=.TRUE.
IF(X2.EQ.0.0) DIRC=.TRUE.
ISMOTH=JSMOTH(IPART)
SHOSHR=.TRUE.
IF(ISMOTH.EQ.-1) SHOSHR=.FALSE.
IF(NPLS(I),EQ.0) GO TO 60
CALL PART

60 CONTINUE
IF(.NOT.NUM) GO TO 70
IF(NNUM) CALL NODNUM
IF(PNUM) CALL POLNUM
70 CALL ENDFRM
IF(IIMOVE.GT.1) GO TO 100
IF(IDVICE.GT.0) GO TO 80
GO TO 90
80 IF(IC.EQ.1) TYPE 160.
IF(IC.EQ.2) TYPE 170.
90 IF(NFRAME.LT.1) GO TO 120
100 IF(IDVICE.NE.-1) GO TO 110
IF(IIMOVE.EQ.NFRAME) GO TO 110
TYPE 190, IIMOVE,NFRAME
IF(.NOT.INCON) GO TO 105
DO 101 IVIC=1,10000
   ICOTR=5*5*6
   VIC=5*5*7
101 CONTINUE
GO TO 120
105 ACCEPT 150, ANS
GO TO 120
110 TYPE 180, IIMOVE,NFRAME
120 CONTINUE
NFRAME=0
SKALE=XMAGN
LINEAR=.FALSE.
RETURN
130 CALL ENDFRM
TYPE 140,
NFRAME=0
SKALE=XMAGN
LINEAR=.FALSE.
IRAD=.FALSE.
NTR=.FALSE.
RETURN
140 FORMAT(18H+<HIDDEN FAILURE>/)
150 FORMAT(A1)
160 FORMAT(13H+<COLOR PASS>/)
170 FORMAT(23H+<BLACK AND WHITE PASS>/)
180 FORMAT(2H<,I3,1H/,I3,1H>/)
C SEE POSSIBLE INCOMPATIBILITIES (2)
C 190 FORMAT(2H<,I3,1H/,I3,1H>,$)
END
SUBROUTINE VISIP
COMMON/COOR/ XP
COMMON/LOGI/SMOTH, IPOOR, IMIX, DIRC, ISPEC, ISMA, LINEA
$P.
1 IHLR, IFRING
COMMON/NEME/ NEDGES, MEDGES, IPART
COMMON/SCMR/ DOZ, FIELDS, FIELD, TANAL, RES
COMMON/VISI/ NNN, IVIS
COMMON/XXN/ XN
DIMENSION NNN(8), XN(3,9), XP(3,9)
LOGICAL DIRC, IPOOR
C
C DIRECTION OF NORMAL AT EACH NODE FOR A POLYGON IS CALCULATED
C
DO 20 I=1, NEDGE
    TEST=-XP(1, I)*XN(1, I)-XP(2, I)*XN(2, I)+(DOZ-XP(3, 6T))*XN(3, I)
    IF(I.LT.1) GO TO 10
    IVIS=-1
    IF(TEST.GT.0.0) IVIS=1
    GO TO 20
10   IF(TEST.GT.0.0.AND.IVIS.EQ.1) GO TO 20
    IF(TEST.LE.0.0.AND.IVIS.EQ. -1) GO TO 20
    IVIS=0
    RETURN
20 CONTINUE
    IF(IVIS.EQ.0) RETURN
    IF(IVIS.LT.0) GO TO 40
    IF(.NOT.DIRC) RETURN
    IVIS=-1
30   IF(.NOT.IPOOR) IVIS=1
    RETURN
40   IF(.NOT.DIRC) GO TO 30
    IVIS=1
    RETURN
END
SUBROUTINE VISIT
COMMON/COOR/ XP
COMMON/FEATR/ FANG,FEATUR
COMMON/LOGI/ISMOOTH,IPoor,IMIX,DIRC,ISPEC,ISMA,LINEA
$R,
1 IHLR,IFRING
COMMON/SM/R/ DOZ,FIELD,FIELD,TANAL,RES
COMMON/SUBD/ UTES,TNT,ITQUAD
COMMON/VISI/ NNN,IVIS
DIMENSION NNN(N),XP(3,9)
DIMENSION TNT(3)
LOGICAL DIRC,FEATUR
C
T1=NNN(1)
T2=NNN(2)
T3=NNN(3)
C
NORMAL FOR TRIANGULAR ELEMENT CALCULATED AND TESTED
FOR
DIRECTION IT IS FACING
C
X1=XP(1,T2)-XP(1,T1)
Y1=XP(2,T2)-XP(2,T1)
Z1=XP(3,T2)-XP(3,T1)
X2=XP(1,T3)-XP(1,T2)
Y2=XP(2,T3)-XP(2,T2)
Z2=XP(3,T3)-XP(3,T2)
U1=Y1*Z2-Z1*Y2
U2=Z1*X2-X1*Z2
U3=X1*Y2-Y2*Y1
IF(.NOT.FEATUR) GO TO 10
U4=SQR(U1*U1+U2*U2+U3*U3)
TNT(1)=U1/U4
TNT(2)=U2/U4
TNT(3)=U3/U4
10 X1=-XP(1,T2)*U1-XP(2,T2)*U2+(DOZ-XP(3,T2))*U3
IVIS=-1
IF(X1.LT.0.0.AND.DIRC) IVIS=1
IF(X1.GT.0.0) GO TO 20
RETURN
20 IF(.NOT.DIRC) IVIS=1
RETURN
END
APPENDIX C

PROGRAMS OVERLAY STRUCTURES
R L INK
DK1:UTILITY, DK1:UTILITY=DK1:UTILITY, OPEN, HELP, OVER, CMD/
DK1:CLEAN, MAKE, MESC OR/O:1
DK1:DISP/O:1
DK1:GE OM/O:1
DK1:MERGE/O:1
DK1:ORDER/O:1
DK1:SYMM/O:1
DK1:R DDISP/O:2
DK1:WRDISP/O:2
DK1:MOVE/O:2
DK1:REDGE OM/O:2
DK1:WRGE OM/O:2
DK1:ELLIP/O:2
DK1:HE XGEN/O:2
DK1:QUADGN/O:2
DK1:REVOLV/O:2
DK1:CONNAR/O:3
DK1:CONNSH/O:3
DK1:INTERS/O:3
DK1: MESH/0V/O:3
DK1:COORDP/O:3
DK1: COR NPT/O:3
DK1:REVC OR/O:3
DK1:TRAROT/O:3
DK1: NO DF, NODEN/O:4
DK1:LOA D/O:4
DK1:CROSSX/O:4
DK1: DIST/O:4
//
R LINK
DK1:MOVIE,MOVIE=DK1:COMM/
DK1:ANIMAT/O:1
DK1:TITLE/O:1
DK1:CLEAR/O:1
DK1:DISTT/O:1
DK1:EXPLT/O:1
DK1:FAST/O:1
DK1:FIEL/O:1
DK1:HELPP/O:1
DK1:IMMUNE/O:1
DK1:INTT,IIABS,SCOP,SUMCEN,REST/O:1
DK1:LINE/O:1
DK1:PIVOT/O:1
DK1:POLMOD/O:1
DK1:READ,SEPAR/O:1
DK1:ROTA/O:1
DK1:SCAL/O:1
DK1:SHKRN/O:1
DK1:TRAN/O:1
DK1:VIDRA,DEVICE,TKPLOT,TTOUT/O:1
DK1:OPENN/O:2
DK1:ROTAT/O:2
DK1:LASTC/O:2
DK1:COARRO/O:2
DK1:NONNUM/O:2
DK1:PART/O:2
DK1:POLNUM/O:2
DK1:OUTNUM/O:3
DK1:CONEL/O:3
DK1:COCORD/O:3
DK1:POLY/O:3
DK1:VISIP/O:3
DK1:VISIT/O:3
DK1:POINT/U/O:4
DK1:NORMAL/O:4
DK1:SHRINK/O:4
DK1:DRAW/O:4
DK1:MUTDC/O:5
DK1:MULTDD/O:5
//
APPENDIX D

COMPUTER GRAPHICS PACKAGE USER'S MANUAL
MOVIEW USER'S MANUAL

MOVIEW IS AN INTERACTIVE PROGRAM FOR THE DISPLAY AND ANIMATION OF ANY MODEL COMPOSED OF POLYGONS. THE PROGRAM ALLOWS THE USER TO MANIPULATE THE MODEL (ROTATE, TRANSLATE, ETC.), AND DISPLAY THE MANIPULATED MODEL ON THE TEKTRONIX 4051 WITH OR WITHOUT HIDDEN LINES/SURFACES REMOVED. THE PROGRAM PROCEEDS IN THE FOLLOWING MANNER.

<GEOM FILE>

ENTER THE GEOMETRY DEV:FILENAME:MAT.

THE GEOMETRY FILE IS NOW READ USING THE FOLLOWING FORTRAN STATEMENTS.

READ(IUNIT,140) NP,NJ,NPT,NCON
READ(IUNIT,140) ((NPL(T,J),I=1,7),J=1,NP)
READ(IUNIT,150) ((X(I,J),I=1,3),J=1,NJ)
READ(IUNIT,140) (IP(I),I=1,NCON)

140 FORMAT(16I5)
150 FORMAT(6E12.5)

THE VARIABLES ARE DEFINED AS FOLLOWS:

NP = THE NUMBER OF PARTS
NJ = THE NUMBER OF NODES OR JOINTS
NPT = THE NUMBER OF ELEMENTS OR POLYGONS
NPL = THE PARTS LIST
NCON = NUMBER OF ELEMENTS IN CONNECTIVITY ARRAY

X = THE COORDINATES OF THE NODES
IP = THE CONNECTIVITY OF THE ELEMENTS OR POLYGONS

NOTE: IF CONCAVE ELEMENTS ARE USED, ONE MUST START THE NODE
NUMBERING AT A CONVEX NODE TO INSURE CONSISTENT ORIENTATION
OF THE NORMALS.

<DISP FILE>

ENTER THE NAME OF THE DISPLACEMENT FILE IN
THE SAME FORMAT USED FOR THE GEOMETRY FILE. A ZERO
FILE DESIGNATION WILL SKIP THE DISPLACEMENT FILE.

AT THIS POINT THE DISPLACEMENT FILE IS READ USING
THE FOLLOWING FORTRAN STATEMENTS.

READ(IUNIT1,150) ((U(I,J),I=1,3),J=1,NJ)

150 FORMAT(6F12.5)

U = NODE DISPLACEMENTS IN THE X, Y, AND Z DIRECTIONS

INITIALIZATION PROCESS

BEFORE ISSUING THE COMMAND PROMPT, SEVERAL PROGRAM
VARIABLES ARE INITIALIZED SO THAT THE MODEL CAN BE DISPLAYED
IMMEDIATELY.

COMMAND PROMPT
FOLLOWING THE INITIALIZATION PROCESS, THE USER WILL RECEIVE THE BLIND PROMPT >>. TO LIST THE COMMANDS AVAILABLE AT THIS POINT SEE THE COMMAND HELP. SINCE MANY OF THE COMMANDS ARE IMPLEMENTED AT THE PARTS LEVEL, QUITE OFTEN THE USER WILL BE GIVEN REQUESTS OF THE FORM:

< . . . PARTS 11/12 . . . >

>>>


ALLOWABLE COMMANDS

THE PROGRAM IS NOW READY TO ACCEPT ONE OF THE ALLOWABLE COMMANDS. THE COMMANDS ARE LISTED IN ALPHABETICAL ORDER, AND THE INFORMATION THEY REQUEST IS DISCUSSED IN THE FOLLOWING PARAGRAPHS.

ANIMATE

THE ANIMATE COMMAND ALLOWS THE USER TO SPECIFY A SEQUENCE OF FRAMES.
<NUMBER OF FRAMES>

Enter the number of frames. This option is not only useful when generating long sequences of frames for movies but also for as few as two or three frames to view the model from different positions.

<SEND FRAMES 11/72>

Enter the numbers of the first and last frames to actually be displayed. This option is useful when the system crashes while in the middle of a long sequence. Instead of regenerating all frames of the sequence, it is only necessary to give the numbers of the frames wanted. If a zero number of frames is given, the program sends all the frames to the display device.

<SMOOTH ANIMATION?>

Enter Y or YFS if smooth animation (instead of uniform steps) is desired. Smooth animation introduces a cosine variation in the animation steps to avoid the sensation of instantaneous velocity changes.

<TRANSIENT DATA?>

Enter Y or YFS to achieve a linear variation between two scalar function files and a uniform increase in the displacement scale factor from zero to full value over the specified number of frames. Use the command linear to set up the desired changes for the displacements and the scalar functions.

<VIBRATION CYCLES/FRAMF>

If a non-zero displacement scale factor is in effect, this prompt will be typed on the user's terminal. Enter the number of vibration cycles per
FRAME FOR SIMULATION OF HARMONIC VIBRATION. THIS
MAY SEEM LIKE AN AKWARD DEFINITION OF THE VIBRATION
FREQUENCY, BUT IT HAS THE ADVANTAGE THAT STILL
MOTION IS SPECIFIED BY THE ENTRY OF A ZERO.

<TOTAL TRANSLATION CHANGE (DX, DY, DZ)>

ENTER THE TOTAL CHANGE IN THE TRANSLATION OF
THE ORIGIN IN THE X, Y, AND Z DIRECTIONS, FOR THE
ANIMATED SEQUENCE.

<TOTAL ROTATION CHANGE (DX, DY, DZ)>

ENTER THE TOTAL CHANGE IN ROTATION (IN
DEGREES) ABOUT THE TRANSLATED ORIGIN IN THE GLOBAL
X, Y, AND Z DIRECTIONS. THE INCREMENTAL ROTATIONS
WILL BE MADE IN THE Y, Z, X ORDER. REMEMBER, FINITE
ROTATIONS DO NOT ADD AS VECTORS!

<PIVOT PARTS 11/12, DX, DY, DZ>

ENTER THE PART NUMBERS AND THE PIVOTS (I.E.
LOCAL ROTATIONS) IN THE X, Y, AND Z DIRECTIONS ABOUT
THE RELATIVE ORIGIN SPECIFIED IN THE PIVOT COMMAND.

<DISTANCE TO ORIGIN (DELTA)>

ENTER THE CHANGE IN THE DISTANCE TO THE
ORIGIN. A NEGATIVE VALUE WILL BRING THE MODEL
TOWARD THE OBSERVER.

<SHRINK FACTOR (DELTA)>

ENTER THE CHANGE IN THE SHRINK FACTOR.
REMEMBER THE SHRINK FACTOR PLUS THE CHANGE MUST
REMAIN IN THE RANGE OF 0.0 TO 1.0.

<DISPLACEMENT SCALE FACTOR (DELTA)>
ENTER THE CHANGE IN THE DISPLACEMENT SCALE FACTOR.

<POSITION SCALE FACTOR (DELTA)>

ENTER THE CHANGE IN THE LOCAL MOTION SCALE FACTOR. THIS COMMAND WILL PRODUCE ANIMATION OF THE EXPLOSION PATTERNS DEFINED IN THE EXPLODE COMMAND.

<SUPPRESS PICTURES?>

ENTER A CARRIAGE RETURN TO SEND PICTURES TO THE DISPLAY DEVICE, ENTER A Y OR YES TO CAUSE THE NEXT DISPLAY COMMAND TO PERFORM THE ANIMATION BUT WILL NOT SEND THE PICTURES TO THE DISPLAY DEVICE. THIS IS HELPFUL IN CHECKING KEY FRAMES IN A MOVIE SINCE THE FINAL SCENE CAN BE DISPLAYED BY ISSUING A SECOND DISPLAY COMMAND.

<DO YOU WANT CONTINUOUS OPERATION?>

ENTER YES TO GET CONTINUOUS OPERATION FROM FRAME TO FRAME. OTHERWISE A <CR> WILL BE REQUIRED TO ADVANCE THE ANIMATION FROM ONE FRAME TO THE OTHER.

<DO YOU WANT A HARD-COPY OF EVERY FRAME?>

ENTER YES TO GET A HARD-COPY OF EVERY FRAME OF THE ANIMATION SEQUENCE.

NOTE: THE ANIMATION COMMAND SPECIFIES THE PARAMETERS OF THE ANIMATION. ACTUAL EXECUTION OF THE DISPLAYS REQUIRES THAT A SINGLE DRAW OR VIEW COMMAND BE GIVEN FOLLOWING THE ANIMATION COMMAND.
THE CENTER COMMAND INVOKES THE SUMMARY COMMAND AND
TRANSLATES THE ORIGIN TO THE CENTER OF THE DISPLAYED PARTS
OF THE MODEL. IN THIS CENTERING ROUTINE, ROTATIONS AND
PIVOTS HAVE BEEN NEGLECTED, AND THUS THE CENTERING MAY NOT
ALWAYS BE EXACT. IF ROTATIONS HAVE BEEN MADE THE ERROR IS
GENERALLY VERY SMALL AND ANY NECESSARY FINE TUNING CAN
EASILY BE ACCOMPLISHED WITH THE SHFf COMMAND. HOWEVER, IF
ANY SIGNIFICANT PIVOTS HAVE BEEN MADE, THE ROUTINE MAY NOT
BE VERY SATISFACTORY.

DISTANCE TO THE ORIGIN, ANGLE OF VIEW, ZMIN, AND
ZMAX. CLIPPING PLANE LOCATIONS (SET COMMANDS DISTANCE AND
FIELD) ARE ALSO RECALCULATED AND THEIR DEFAULT OR CALCULATED
VALUES WILL BE TYPED ON THE USERS TERMINAL.

DISTANCE

THE DISTANCE COMMAND ALLOWS THE USER TO SPECIFY THE
DISTANCE BETWEEN THE ORIGIN OF THE GLOBAL COORDINATE SYSTEM
(WHICH MAY HAVE BEEN TRANSLATED) AND THE OBSERVER.

<TO ORIGIN (CURRENT VALUE)>

ENTER THE DISTANCE FROM THE OBSERVER TO THE
MODEL ORIGIN. DECREASING THE DISTANCE PRODUCES A
"ZOOM" EFFECT.

DRAW

THE DRAW COMMAND SENDS THE PICTURE DEFINED BY ALL
PREVIOUS COMMANDS TO THE DISPLAY DEVESE. WHEN THIS
COMMAND IS USED THE HIDDEN LINES/SURFACES ALGORITHM IS NOT
APPLIED TO THE MODEL. THE BLIND PROMPT >> IS GIVEN TO
INDICATE THAT THE DISPLAY HAS BEEN COMPLETED AND THE SYSTEM
IS READY TO ACCEPT A NEW COMMAND.
EXII

THE EXII COMMAND PROVIDES A CONTROLLED TERMINATION OF THE PROGRAM (INCLUDING WRITING THE OUTPUT BUFFER).

EXPLODE

THE EXPLODE COMMAND ALLOWS THE USER TO SPECIFY LOCAL MOTION (EXPLOSION) PATTERNS FOR ANY GROUP OF ELEMENTS.

<PARTS I1/I2, LOCAL MOTION (X,Y,Z)>

>>> ENTER THE PAPI NUMBERS AND THE LOCAL MOTION PATTERN COMPONENTS IN THE X, Y, AND Z DIRECTIONS.

<LOCAL MOTION SCALE FACTOR (CURRENT VALUE)>

ENTER THE SCALE FACTOR TO BE USED IN SCALING THE ABOVE MOTION PATTERN PRIOR TO THEIR BEING ADDED TO THE NODAL COORDINATES.

FAST

THE FAST COMMAND ALLOWS THE USER TO DEFINE THE DATA ORGANIZATION AND INVOKE THE POOR MAN'S HIDDEN SURFACE PROCEDURE.
<MIXED DATA?>

ENTER Y OR YES IF THE POLYGONAL VERTEXES IN
THE GEOMETRY FILE ARE NOT ORDERED IN A CONSISTENT
CLOCKWISE OR COUNTER-CLOCKWISE DIRECTION AND CAUSE
THE NEXT THREE REQUESTS TO BE SKIPPED.

NOTE: THE POOR MAN'S HIDDEN SURFACE PROCEDURE CANNOT BE
SUCCESSFULLY USED UNLESS THE DATA IS CONSISTENT. UTILITY MAY
BE USED TO ORDER MIXED DATA GEOMETRY FILES.

<POOR MAN'S PROCEDURE?>

ENTER Y OR YES TO INVOKE THE POOR MAN'S
VISIBLE SURFACE PROCEDURE. OTHERWISE THE NEXT TWO
REQUESTS WILL BE SKIPPED. THE POOR MAN'S PROCEDURE
WILL NOT SEND TO THE DISPLAY TERMINAL ANY POLYGON
WHICH IS FACING AWAY FROM THE OBSERVER.

<PARTS I1/I2 IMMUNE TO POOR MAN'S PROCEDURE>

>>>

ENTER THE PART NUMBERS FOR THOSE PARTS WHICH
ARE TO BE IMMUNE TO THE POOR MAN'S VISIBLE SURFACE
PROCEDURE. A COMPLETE DEFINITION IS REQUIRED AS THE
PREVIOUS DEFINITION IS DESTROYED WHEN THIS COMMAND
IS ENTERED (I.E. ALL PARTS ARE DEFINED TO HAVE POOR
MAN'S PROCEDURE IN EFFECT).

<PARTS I1/I2 WITH CLOCKWISE ORDERING>

>>>

ENTER THE PART NUMBERS FOR THOSE PARTS WHOSE
ELEMENTS HAVE A CLOCKWISE ORDERING OF THE POLYGON
VERTEXES AS VIEWED FROM THE OUTSIDE. A COMPLETE
DEFINITION IS REQUIRED AS THE PREVIOUS DEFINITION IS
DESTROYED WHEN THIS COMMAND IS ENTERED (I.E. ALL
PARTS ARE DEFINED AS HAVING A COUNTER-CLOCKWISE
ORDERING).
NOTE: IF YOU SPECIFY, FOR EXAMPLE, THAT THE DATA HAS BEEN ORDERED IN A CLOCKWISE ORIENTATION AND THE REVERSE IS TRUE, THE ROUTINE WILL DISCARD THE FORWARD FACING ELEMENTS FOR THAT GROUP OF PARTS. THIS CAN BE AN INTERESTING AND USEFUL FEATURE. MOSTLY, HOWEVER, YOU WILL FIND IT TO BE CONFUSING.

FIELD

THE FIELD COMMAND ALLOWS THE USER TO DEFINE THE FRUSTUM OF VISION.

<ANGLE, ZMIN, ZMAX (CURRENT VALUES)>

ENTER THE ANGLE OF VIEW (IN DEGREES), THE DISTANCE FROM THE OBSERVER TO THE NEAR CLIPPING PLANE, AND THE DISTANCE FROM THE OBSERVER TO THE FAR CLIPPING PLANE. THE NEAR AND FAR CLIPPING PLANES SHOULD BE PLACED FAR ENOUGH AWAY FROM THE MODEL TO ALLOW FOR ALL ROTATIONS AND TRANSLATIONS TO WHICH THE MODEL WILL BE SUBJECTED. IF THE NEAR AND FAR CLIPPING PLANES ARE DEFINED CLOSE TOGETHER, THE EFFECT IS TO DISPLAY A CROSS-SECTION OF THE MODEL. A SMALL ANGLE OF VIEW WILL REDUCE THE PERSPECTIVE AND INCREASE THE MODEL SIZE ON THE DEVICE, WHILE A LARGE ANGLE OF VIEW WILL INCREASE THE PERSPECTIVE AND REDUCE THE MODEL SIZE ON THE DEVICE.

HELP

THE HELP COMMAND TYPES THE AVAILABLE COMMANDS ON THE USER’S TERMINAL IN ALPHABETICAL ORDER AS FOLLOWS:

<COMMANDS: ANIMATE, CENTER, DISTANCE, DRAW, EXIT, EXPLODE, FAST, FIELD, HELP, IMMUNE, LINEAR, NODE, PARTS, PILOT, POLY, HEAD, RESOLVE, ROTATE, SCALE, SCOPE, SHIFT, SHRINK, SUMMARY, TITLE, TRANSLATE, VIEW>
THE IMMUNE COMMAND ALLOWS THE USER TO MAKE PARTS TEMPORARILY IMMUNE TO GLOBAL ROTATIONS. THE IMMUNITY ONLY APPLIES TO ROTATION COMMANDS GIVEN AFTER IMMUNITY IS ESTABLISHED AND BEFORE IT IS REVOKED.

NOTE: IF USED ON PARTS THAT WILL LATER BE PIVOTED, IT MUST BE RECOGNIZED THAT THE GLOBAL COORDINATE TRIAD IN THE LOWER LEFT OF THE PICTURE MAY NO LONGER COINCIDE WITH THE ORIGINAL AXIS ORIENTATION OF THAT PART. (SEE PIVOT COMMAND)

<PART 11/12 IMMUNE TO ROTATION>

ENTER THE PART NUMBERS TO BE MADE IMMUNE TO GLOBAL ROTATIONS. A COMPLETE DEFINITION IS REQUIRED AS THE PREVIOUS DEFINITION IS DESTROYED WHEN THIS COMMAND IS ENTERED (I.E. ALL IMMUNITY IS REVOKED).

LINEAR

THE LINEAR COMMAND ALONG WITH THE TRANSIENT DATA OPTION IN THE ANIMATE COMMAND ALLOWS THE USER TO LINEARLY INTERPOLATE BETWEEN TWO DISPLACEMENT FILES.

<INCREMENT GEOMETRY?>

ENTER Y OR YES TO MODIFY THE CURRENT GEOMETRY (I.E. THE NODE COORDINATES) BY ADDING THE CURRENT DISPLACEMENTS MULTIPLIED BY THE CURRENT SCALE FACTOR.

<SET DISPLACEMENTS FILE TO ZERO?>
ENTER Y OR YES TO CLEAR THE DISPLACEMENT FILE AND BOTH SCALAR FUNCTION FILES.

<DIS1 FILE>

ENTER THE FILE NAME .EX1 OF THE DISPLACEMENT FILE AT TIME=1. A NULL FILE DESIGNATION WILL SKIP TO THE NEXT REQUEST WITHOUT READING A FILE. THE DISPLACEMENTS ARE READ IN THE SAME FORMAT AS DESCRIBED IN COMMAND READ.

<DIS2 FILE>

ENTER THE NAME OF THE DISPLACEMENT FILE AT TIME=1+1 IN THE SAME FORMAT AS ABOVE.

AT THIS POINT THE FOLLOWING OPERATIONS ARE PERFORMED:

1) THE DISPLACEMENTS ARE SET EQUAL TO THE DIFFERENCE BETWEEN THE SECOND AND FIRST DISPLACEMENT FILES. THIS AND SUBSEQUENT CALCULATIONS CAUSE BOTH THE FIRST AND SECOND DISPLACEMENT FILES TO BE LOST WITH ONLY THE DIFFERENCE BETWEEN THE FILES BEING RETAINED.

NODE

THE NODE COMMAND ALLOWS THE USER TO DISPLAY NODE NUMBERS. THIS COMMAND IS A SWITCH. WHEN GIVEN THE FIRST TIME, NODE NUMBERING IS ENABLED (<NODE NUMBERING ENABLED>). WHEN GIVEN AGAIN, NODE NUMBERING IS DISABLED (<NODE NUMBERING DISABLED>), ETC.

NOTE: IF A GIVEN NODE APPEARS AT MORE THAN ONE LOCATION (DUE TO THE USE OF SUCH COMMANDS AS EXPLODE), THE NODE WILL

PARTS

THE PARTS COMMAND ALLOWS THE USER TO SELECT ALL OF THE MODEL OR A SUBSET OF THE MODEL FOR DISPLAY.

<PARTS I1/I2 WITHHELD FROM DISPLAY>

>>> ENTER THE PART NUMBERS FOR THE PARTS THAT ARE TO BE WITHHELD FROM THE SCENE. THIS MUST BE A COMPLETE DEFINITION OF THE PARTS TO BE WITHHELD. ALL PREVIOUS DEFINITIONS OF THE SCENE COMPONENTS ARE DISCERNED (I.E., ALL PARTS ARE DEFINED TO BE SHOWN).

PIVOT

THE PIVOT COMMAND ALLOWS THE USER TO ROTATE INDIVIDUAL PARTS OF THE MODEL ABOUT AN ORIGIN DEFINED FOR THAT PART IN THE ORIGINAL AXIS DIRECTIONS OF THE MODEL. THE COORDINATE TRIAD IN THE LOWER LEFT OF THE PICTURE IS DESIGNED TO HELP THE USER REMEMBER THE ORIGINAL AXIS ORIENTATION. AFTER THE IMMUNE COMMAND HAS BEEN USED ON A PART AND A SUBSEQUENT ROTATION COMMAND HAS BEEN GIVEN, HOWEVER, THE GLOBAL COORDINATE TRIAD NO LONGER COINCIDES WITH THE DIRECTIONS FOR THAT PART, THUS THE ORIGINAL MODEL AXIS ORIENTATION MUST BE REMEMBERED. PREVIOUSLY DEFINED PIVOT COMMANDS REMAIN IN EFFECT. A RESTORE COMMAND IS USED TO REVOKE ALL PIVOT COMMANDS.

<PARTS I1/I2, AXIS, ANGLE>

>>> ENTER THE PART NUMBERS, THE AXIS (X, Y, OR

<PARTS 11/12, RELATIVE ORIGIN>


POLY

THE POLY COMMAND ALLOWS THE USER TO DISPLAY POLYGON NUMBERS.

THIS COMMAND IS A SWITCH. WHEN GIVEN THE FIRST TIME, POLYGON NUMBERING IS ENABLED (<POLYGON NUMBERING ENABLED>), IF GIVEN AGAIN, POLYGON NUMBERING IS DISABLED (<POLYGON NUMBERING DISABLED>), ETC.

NOTE: IF THIS OPTION IS USED IN CONJUNCTION WITH THE EXPLODE CAPABILITY (OR RELATED COMMANDS WHICH RESULT IN A GIVEN NODE APPEARING AT MORE THAN ONE LOCATION ON THE DISPLAY DEVICE), THE LOCATION OF THE POLYGON NUMBERS MAY NOT BE AT THE ELEMENT CENTERS. THIS IS BECAUSE THE ROUTINE LOOKS AT THE LATEST DEFINITION OF THE NODE LOCATIONS (FOR THE PURPOSE OF CALCULATING THE AVERAGE VALUE) FOR THE NODES WHICH (BY NUMBER) ARE ASSOCIATED WITH THE POLYGON.

READ

THE READ COMMAND RETURNS CONTROL TO THE BEGINNING OF THE PROGRAM SO THE USER MAY READ IN NEW GEOMETRY, AND DISPLACEMENT FILES. IT IS COMMON TO ISSUE THE RESTORE
AND CENTER AFTER READING IN ANOTHER FILE.

RESTORE

THE RESTORE COMMAND ZEROS ROTATIONS, PIVOTS, AND TRANSLATIONS AND Initializes THE ROTATION AND PIVOT TRANSFORMATION MATRICES. RESTORE DOES NOT MODIFY ANY PARAMETERS SET BY THE EXPLODE COMMAND.

ROTATE

THE COMMAND ROTATE ALLOWS THE USER TO ROTATE THE MODEL ABOUT THE TRANSLATED ORIGIN. THE ROTATIONS OF THE MODEL ARE SPECIFIED RELATIVE TO AXES WHICH ARE PARALLEL (X TO THE RIGHT, Y UP) AND NORMAL (Z TOWARD THE OBSERVER) TO THE PLANE OF THE DISPLAY DEVICE. A RIGHT-HANDED COORDINATE SYSTEM AND NOTATION FOR ROTATIONS IS USED. PREVIOUSLY DEFINED ROTATE COMMANDS REMAIN IN EFFECT. A RESTORE COMMAND IS USED TO REVOKE ALL ROTATE COMMANDS.

<AXIS, ANGLE>


SCALF

THE SCALF COMMAND ALLOWS THE USER TO SELECT A SCALE FACTOR FOR THE DISPLACEMENTS. THE DISPLACEMENTS WILL BE MULTIPLIED BY THE SCALE FACTOR BEFORE THEY ARE ADDED TO THE
NODAL COORDINATES.

<DISPLACEMENT SCALE FACTOR (CURRENT VALUE)>

ENTER THE DISPLACEMENT SCALE FACTOR. BOTH POSITIVE AND NEGATIVE VALUES ARE ACCEPTABLE.

SCOPE

THE SCOPE COMMAND REQUESTS INFORMATION NECESSARY TO DEFINE CERTAIN PICTURE VARIABLES.

<SUPPRESS COORDINATE TRIAD?>

THIS REQUEST IS ISSUED FOR LINE DRAWING DEVICES ONLY. ENTER Y OR YES TO HAVE THE GLOBAL COORDINATE TRIAD IN THE LOWER LEFT OF THE PICTURE SUPPRESSED. ANY OTHER RESPONSE WILL CAUSE IT TO BE DRAWN. THE COORDINATE TRIAD WILL ROTATE WITH THE MODEL, DEFINING THE "ORIGINAL" COORDINATE AXES DIRECTIONS.

NOTE: THE COORDINATE TRIAD WILL RESPOND TO ALL ROTATION COMMANDS WHILE SOME (OR ALL) OF THE PARTS MAY BE IMMUNE TO THE ROTATION COMMAND. THE COORDINATE TRIAD DOES NOT RESPOND TO PIVOT COMMANDS.

<DISPLAY NOT ROTATED COORDINATE TRIAD?>

ENTER Y OR YES TO HAVE DISPLAY AT THE LOWER COORDINATE CORNER A TRIAD WITHOUT APPLYING THE ROTATION TRANSFORMATION.

<DISPLAY RESOLUTION (CURRENT VALUE)>

ENTER THE DISPLAY RESOLUTION WHICH DETERMINES THE SIZE OF THE DISPLAY SQUARE. THE RESOLUTION SHOULD BE A VALUE FROM 100 TO 760.
SHIFT

The shift command allows the user to move the viewing window in the plane of the display device.

<SHIFT WINDOW X, Y (CURRENT VALUES)>

Enter the X (to the right) and Y (up) distances by which the viewing window is to be moved. These distances are based on the same scale as the model. The entered values are absolute in nature, that is, they are not changes to the current values.

SHRINK

The shrink command allows the user to move the nodes of the elements towards the element centroid, thus shrinking elements away from each other. This option is useful in the verification of elements.

<CHANGE SHRINK FACTOR FROM (CURRENT VALUE) TO>

Enter a value greater than or equal to 0.0 and less than 1.0. The picture will vary from no shrink effects at value=0 to nothing but a point for each element at a value=1. Generally good shrink factor values range from 0.2 to 0.4.

SUMMARY

TITLE

THE TITLE COMMAND ALLOWS THE USER TO HAVE A STRING OF CHARACTERS TYPED AT THE TOP OF THE DISPLAY FRAME.

TRANSLATE

THE TRANSLATE COMMAND ALLOWS THE USER TO SHIFT THE ORIGIN OF THE MODEL TO A NEW LOCATION.

<ORIGIN FROM (CURRENT VALUE) to>


VIEW

THE VIEW COMMAND SENDS THE PICTURE DEFINED BY ALL PREVIOUS COMMANDS TO THE DISPLAY DEVICE. THIS DISPLAY
COMMAND WILL INVOCF POOR'S VAN ALGORITHM TO REMOVE HIDDEN LINES OR SURFACES. THE BLANK PROMPT >> IS GIVEN TO INDICATE THAT THE DISPLAY HAS BEEN COMPLETED AND THE SYSTEM IS READY TO RECEIVE A NEW COMMAND.

ERROR MESSAGES

IF A COMMAND IS NOT RECOGNIZED DURING THE EXECUTION OF THE PROGRAM, THE ILLEGAL COMMAND, "AAAA", IS TYPED ON THE USER'S TERMINAL AS FOLLOWS.

<AAAA? HELP?>

IF THE USER WISHES TO SEE A LIST OF THE AVAILABLE COMMANDS, HE SHOULD ANSWER Y OR YES TO THIS ERROR MESSAGE.

THERE ARE TEN WARNING AND TWO ERROR MESSAGES THAT MAY BE ISSUED DURING A SESSION. THE ERROR MESSAGES ARE CONSIDERED FATAL AND WILL CAUSE PROGRAM EXECUTION TO TERMINATE. THE MESSAGES ARE:

<WARNING: NEW FILE HAS MORE PARTS THAN OLD FILE>
<ISSUE "RESTORE" COMMAND>

THIS MESSAGE RESULTS WHEN READING IN A FILE THAT HAS MORE PARTS THAN THE PREVIOUS FILE. A RESTORE COMMAND IS NECESSARY TO INITIALIZE IMPORTANT PARAMETERS, OTHERWISE ALL THE PARTS WILL NOT BE DISPLAYED.

<ERROR: MAXIMUM NUMBER OF EDGES EXCEEDED>

THIS ERROR MESSAGE RESULTS WHEN AN ELEMENT OR POLYGON IS PRESENT IN THE DATA THAT HAS MORE SIDES THAN THE DIMENSIONED NSMAX. EITHER THE DATA FILE HAS ERRORS OR THE NSMAX DIMENSIONING IN THE FORTRAN CODE NEEDS TO BE INCREASED.
ERROR: DATA PROBLEMS

This message is given when the lower limit element number of a part in the parts array (NPL) exceeds the dimensioned value for Elements(NPTMAX). See subroutine Lastc.
UTILITY USER'S MANUAL

MOVIE SYSTEM UTILITY AND MODEL GENERATION

INTRODUCTION

UTILITY IS A ROUTINE DESIGNED TO CREATE OR EDIT FORTRAN DATA FILES IN A FORMAT WHICH IS COMPATIBLE WITH THE OTHER PROGRAMS IN THE MOVIE SYSTEM. THE PROGRAM IS BASED UPON A FOUR LETTER KEY WORD SYSTEM WHICH ALLOWS THE USER TO SPECIFY COMMANDS TO MAKE DATA FILES USING THE MODEL GENERATION AND TRANSFORM CAPABILITIES, TO READ, WRITE, OR CHANGE DATA FILES, TO PERFORM SYMMETRY OPERATIONS (E.G., TO CREATE A MODEL OF A COMPLETE SPHERE BASED UPON A MODEL OF 1/8 OF THE SPHERE LOCATED IN THE FIRST QUADRANT), TO ORDER PANEL DATA CONSISTENTLY, TO MERGE OR REORGANIZE DATA FILES, OR TO EXIT FROM THE PROGRAM IN A CONTROLLED MANNER.

WHILE THE FOLLOWING INSTRUCTIONS ARE RATHER LENGTHY, USERS WILL NORMALLY NOT FIND MUCH REASON TO REFER TO THEM. THE SYSTEM IS EASILY LEARNED SINCE THE PROGRAM ASKS SPECIFIC QUESTIONS OR GIVES ONE OF THE FOLLOWING PROMPTS (>, FOR LEVEL 1, >> FOR LEVEL 2, OR >>> FOR LEVEL 3). WHEN THESE PROMPTS ARE ENCOUNTERED, A LISTING OF THE AVAILABLE OPTIONS MAY BE OBTAINED BY ENTERING ? OR HELP. ESCAPE FROM REPEATED REQUESTS (IF ESCAPE IS APPROPRIATE) IS ACCOMPLISHED WITH A CARRIAGE RETURN. A CARRIAGE RETURN FOLLOWING THE PROMPT (>>>) FOR LEVEL 3 WILL TRANSFER CONTROL TO LEVEL 2 AND GIVE THE PROMPT (>>). ESCAPE TO LEVEL 1 IS ALSO OBTAINED BY A CARRIAGE RETURN.

SOME LEVEL THREE COMMANDS WILL GENERATE REQUESTS TO THE USER IN THE FORM:
< REQUEST FOR MULTIPLE INPUTS >

THIS LEVEL 4 PROMPT IS USED TO SAVE WRITING THE LEVEL THREE REQUEST ON A REPEATED BASIS, THE LEVEL 4 PROMPTS CANNOT BE INTERROGATED, BUT ESCAPE TO THE NEXT LEVEL THREE REQUEST IS ACCOMPLISHED WITH A CARRIAGE RETURN.

COMMAND LEVEL 1


COMMAND LEVEL 2

THE LEVEL 2 ALGORITHM ENTERED DEPENDS UPON THE COMMAND GIVEN IN LEVEL 1. GEOM FUNCTIONAL COMMANDS ARE READ, WRITE, CHANGE, PRINT AND EXIT.
DISP HAS THESE SAME COMMANDS PLUS TRANSFORM. THE
LEVEL TWO COMMANDS ASSOCIATED WITH THE LEVEL ONE COMMAND
MAKE, ARE ENTIRELY DIFFERENT. IF THE COMMAND GIVEN IS
UNACCEPTABLE, THE LEVEL 2 OPTIONS ARE LISTED AND THE PROMPT
IS REPEATED. THE RESPONSE TO THE ACCEPTABLE COMMANDS
DEPENDS UPON THE COMMANDS GIVEN AT BOTH LEVEL 1 AND LEVEL 2.
The responses for GEOM and DISP will now be discussed
ACCORDING TO THE VARIOUS COMBINATIONS OF LEVEL 1 - LEVEL 2
COMMANDS. FOLLOWING THIS, THE INSTRUCTIONS FOR USING THE
MODEL GENERATOR ASSOCIATED WITH THE MAKE COMMAND WILL BE
GIVEN.

GEOM-READ

<READ GEOM FILE>

ENTER THE INPUT GEOMETRY FILENAME, EXT. THE
FILENAME SHOULD NOT EXCEED SIX CHARACTERS, AND THE
EXTENSION SHOULD NOT EXCEED THREE.

AT THIS POINT THE GEOMETRY FILE WILL BE READ USING
THE FOLLOWING FORTRAN STATEMENTS.

READ(UNIT,180) NP,NJ,NPT,NEDGE
READ(UNIT,180) ((NPL(I,J),I=1,2),J=1,NP)
READ(UNIT,170) ((X(I,J),I=1,3),J=1,NJ)
READ(UNIT,180) (JP(I),I=1,NEDGE)

170 FORMAT(6F12.5)
180 FORMAT(16I5)

THE VARIABLES ARE DEFINED AS FOLLOWS:

NEDGE = TOTAL NUMBER OF EDGES
NP = THE NUMBER OF PARTS
NJ = THE NUMBER OF NODES OR JOINTS
NPT = THE NUMBER OF ELEMENTS
NPL = THE PARTS ARRAY
X = THE COORDINATES OF THE NODES
JP = THE CONNECTIVITY OF THE ELEMENTS

AFTER READING THE GEOMETRY FILE THE PROGRAM REQUESTS VERIFICATION THAT THE PARTS ARRAY IS DEFINED PROPERLY.

<PART LIMITS>
(LIST OF TOTAL NPL ARRAY)
<number of elements in part groups>
(LIST OF DIFFERENCED NPL ARRAY)
<changes?>

ENTER Y OR YES IF THE PARTS ARRAY IS NOT DEFINED PROPERLY. ANY OTHER RESPONSE WILL REQUEST THE USER TO REDEFINE THE PARTS ARRAY.

<MULTIPLY DEFINED PART GROUPS>

THIS MESSAGE IS PRINTED IF THE PROGRAM DETECTS MULTIPLY DEFINED PARTS WHILE ATTEMPTING TO CONSTRUCT THE PARTS ARRAY. THE PROGRAM WILL REQUEST THE USER TO SUPPLY THE CORRECT PARTS ARRAY ACCORDING TO THE FIRST DEFINITION OF THE PARTS ARRAY GIVEN IN THE NOTE JUST ABOVE.

<number of part groups>

ENTER THE NUMBER OF ELEMENT GROUPINGS

<number of elements in groups>
ENTER THE NUMBER OF ELEMENTS IN EACH GROUP.

GEOM-WRITE

<WRITE GEOM FILE>

ENTER THE OUTPUT GEOMETRY FILENAME, EXT. THE FILENAME SHOULD NOT EXCEED SIX CHARACTERS, AND THE EXTENSION SHOULD NOT EXCEED THREE.

BEFORE WRITING THE GEOMETRY FILE, THE PROGRAM ALLOWS THE USER TO VERIFY THE PARTS ARRAY AND TO CHANGE IT IF DESIRED.

<ELEMENT LIMITS OF PARTS>
(List of NPL array)
<CHANGES?>

ENTER Y OR YES IF THE LOWER AND UPPER LIMITS OF THE PARTS ARRAY ARE NOT CORRECT. ANY OTHER RESPONSE WILL REQUEST NEW ELEMENT LIMIT INFORMATION.

<NUMBER OF PARTS>

ENTER THE NUMBER OF PARTS.

<ELEMENT LIMITS OF PARTS>

ENTER THE LOWER AND UPPER LIMIT NUMBERS OF THE BOUNDING ELEMENTS FOR EACH PART. (REMEMBER, THIS MAY BE AN OVERLAPPING DEFINITION.) THESE INTEGERS ARE ACCEPTED ACCORDING TO A (20I) FORMAT.

AT THIS POINT THE GEOMETRY FILE WILL BE WRITTEN USING THE FOLLOWING FORIPAN STATEMENTS.
WRITE(TUNIT,120) NP,NJ,NPI,NEDE
WRITE(TUNIT,120) ((KP(I,J),I=1,NP),J=1,NJ)
WRITE(TUNIT,160) ((X(I,J),I=1,3),J=1,NJ)
WRITE(TUNIT,120) (JP(I),J=1,NEDE)

120 FORMAT(16I5)
160 FORMAT(1PE12.5)

FOR VARIABLE DESCRIPTIONS, SEE GEOM-READ.

GEOM-CHANGE

THIS COMMAND RESULTS IN THE PROGRAM ENTERING LEVEL 3 WITH THE USER RECEIVING THE PROMPT(>>). THE ALLOWABLE COMMANDS IN THIS INSTANCE ARE GROUP, COORDINATES, ELEMENTS, MOVE, SCALE, SHIFT, AND EXIT.

GEOM-CHAN-GROUP

<NUMBER OF ELEMENT GROUPS>

ENTER THE NUMBER OF ELEMENT GROUPS (NP).

<REORDER OLD GROUPS?>

ENTER Y OR YES TO INVOKE THE OPTION TO REORDER THE ELEMENT LIST BY PART GROUPS. ANY OTHER RESPONSE WILLskip TO A REQUEST FOR THE <NUMBER OF ELEMENTS IN GROUPS>. THIS OPTION IS ESPECIALLY USEFUL TO REORDER THE ELEMENT CONNECTIVITY ARRAY FOLLOWING A SYMMETRY OPERATION. USE THE LEVEL ONE COMMAND "ORDER" TO PRODUCE CONSISTENT CLOCKWISE OR COUNTER-CLOCKWISE CONNECTIVITY.
<NEW GROUP NUMBER = OLD GROUP NUMBERS>
<GROUP 1 = >

ENTER OLD GROUP NUMBERS IN THE ORDER THAT THEY SHOULD NOW COMBINE TO BECOME THE NEW GROUP 1.

<GROUP 2 = >

REPEATS FOR NUMBER OF ELEMENT GROUPS PREVIOUSLY INDICATED. WHEN FINISHED, THE REQUEST FOR THE NUMBER OF ELEMENTS IN EACH GROUP IS SKIPPED AND THE LEVEL THREE PROMPT IS GIVEN.

<NUMBER OF ELEMENTS IN GROUPS>

ENTER THE COMPLETE ELEMENT GROUP LIST 
(NPL (1, I), I=1, NF).

GEOM-CHAN-COORDINATES

<CHANGE NUMBER OF NODES?>

ENTER Y OR YES TO CHANGE THE TOTAL NUMBER OF NODES (NJ) IN THE MODEL. ANY OTHER RESPONSE WILL SKIP THE NEXT REQUEST.

<NUMBER OF NODES>

ENTER THE NEW NUMBER OF NODES.

<NODE NUMBER AND COORDINATES> >>>>

ENTER THE APPROPRIATE NODE NUMBER AND
CURVING X, Y, AND Z COORDINATE VALUES (RIGHT
HANDED SYSTEM) ACCORDING TO THE FORMAT (I,3E). IF
THE NODE NUMBER IS LARGER THAN THE CURRENT TOTAL
NUMBER OF NODES, THE TOTAL NUMBER OF NODES IS
INCREASED TO THE VALUE OF THE ENTERED NODE NUMBER.

COORD-ELEMENT

<CHANGES FOR N-NODE ELEMENTS. ENTER VALUE FOR N>

ENTER NUMBER OF NODES (NOT COUNTING A
CENTRAL NODE) FOR THE ELEMENTS TO BE MODIFIED. IF,
FOR EXAMPLE, THE NUMBER 3 IS ENTERED, ONLY TRIANGLES
CAN BE MODIFIED. IF, FOR EXAMPLE, BOTH TRIANGLES
AND PENTAGONS NEED TO BE MODIFIED, ENTER THIS
COMMAND TWICE; ONCE FOR N=3 AND ONCE FOR N=5.

<ADD, DELETE, OR REPLACE>

ENTER A OR ADD TO ADD ELEMENTS, D OR DELETE
TO DELETE ELEMENTS, OR K OR REPLACE TO REPLACE
ELEMENTS. IF THE ADD COMMAND IS GIVEN, YOU WILL
RECEIVE THE PROMPT <GROUP, N NODES, CENTRAL NODE>
IF THE DELETE COMMAND IS GIVEN, YOU WILL RECEIVE THE
PROMPT <GROUP, N NODES>; IF THE REPLACE COMMAND IS
GIVEN, YOU WILL RECEIVE THE PROMPT <GROUP, N OLD
NODES, N NEW NODES, CENTRAL NODE>.

NOTE: IF CONCAVE ELEMENTS ARE USED, ONE MUST START THE NODE
NUMBERING AT A CONVEX NODE TO INSURE CONSISTENT ORIENTATION
OF THE NormALS.

NOTE: PLANAR OR NEARLY PLANAR POLYGONS WILL GENERALLY NOT
REQUIRE USER SPECIFIED CENTER NODES (HAPPILY THIS IS
GENERALLY THE CASE). IF, HOWEVER, QUITE HIGHLY WARPED
POLYGONS ARE TO BE DISPLAYED, THE DEFINING OF REASONABLE
CENTER NODES MAY BE NECESSARY TO CORRECTLY REPRESENT THE
SURFACE. ESPECIALLY IF THESE WARPED POLYGONS ARE ALSO
CONCAVE AND/OR HAVE THREE OR MORE NODES ALL IN A STRAIGHT
LINE. THE PROGRAM WILL GENERATE "AVERAGE" CENTER NODES IF
NEEDED, BUT IF THE DISPLAY IS NOT SATISFACTORY, ONE SHOULD SUPPLY "BETTER" CENTER NODE DATA. IN THE VERY WORST CASES, SUBDIVISION OF THE WARPED POLYGON INTO LESS WARPED POLYGONS MAY BE NECESSARY.

GROUP, N NODES, CENTRAL NODE

TO ADD ELEMENTS, ENTER THE GROUP NUMBER, THE NODE NUMBERS FOR THE ELEMENT, AND THE CENTRAL NODE NUMBER IF THERE IS ONE.

GROUP, N NODES

TO DELETE ELEMENTS, ENTER THE GROUP NUMBER AND THE NODE NUMBERS FOR THE ELEMENT. THE CENTRAL NODE NUMBER, IF IT EXISTS, IS NOT REQUIRED.

GROUP, N OLD NODES, N NEW NODES, CENTRAL NODE


GROUP-CHAN-MOVE

MOVE I1 ELEMENTS STARTING AT I2 TO FOLLOW I3

ENTER THE APPROPRIATE VALUES FOR I1, I2, AND I3. TO ILLUSTRATE THE USE OF THIS COMMAND, CONSIDER THE FOLLOWING EXAMPLE. SUPPOSE THAT WE WISH TO REORDER A LIST OF SIX ELEMENTS SUCH THAT THE FIRST TWO ELEMENTS REMAIN WHERE THEY ARE, THE FIFTH ELEMENT BECOMES THE THIRD ELEMENT, THE THIRD AND
FOURTH ELEMENTS BECOME THE FOURTH AND FIFTH ELEMENTS AND THE SIXTH ELEMENT REMAINS ITS POSITION. THIS REORDERING MAY BE ACHIEVED BY THE MOVE COMMAND (2,3,5) OR (1,5,2). MOVE COMMANDS DO NOT AUTOMATICALLY RESULT IN CHANGES TO THE PARTS ARRAY. THE USER MAY MODIFY THE PARTS ARRAY AS APPROPRIATE, EITHER BEFORE OR AFTER USING THE MOVE COMMAND. CAUTION SHOULD BE EXERCISED IN THE USE OF REPEATED MOVE COMMANDS TO AVOID GETTING MIXED UP. USE OF PRINT COMMANDS BETWEEN MOVE COMMANDS TO ESTABLISH ELEMENT GROUP LIMIT LOCATIONS IS RECOMMENDED.

GLCM-CHAN-SCALE

<COORDINATE SCALE FACTOR>

ENTER THE FACTOR BY WHICH THE GLOBAL COORDINATES ARE TO BE MULTIPLIED.

GLCM-CHAN-SHIFT

<NEW ORIGIN (X, Y, Z)>

ENTER THE LOCATION (X,Y,Z) OF THE NEW COORDINATE SYSTEM ORIGIN (E.G., ENTER 3,4,5 TO REDUCE ALL X,Y,Z COORDINATES BY 3,4,5 RESPECTIVELY). THE SHIFT COMMAND IS USEFUL IN EXTENDING THE CAPABILITIES OF SYMMETRY AND MIRROR.

GFDM-PRINT
THIS COMMAND PAIR IS STRUCTURED SO AS TO FACILITATE QUICK CHECKS OF SUBSETS OF THE DATA. THESE COMMANDS RESULT IN THE PROGRAM ENTERING A LEVEL 3 OPERATION WITH THE PROMPT (>>>) WHICH IS SIMILAR TO THAT DISCUSSED IN GEOM-CHAN. THE ACCEPTABLE COMMANDS AT THIS LEVEL ARE GROUP, COORDINATES, ELEMENTS, AND EXIT. ISSUE OF ONE OF THESE COMMANDS RESULTS IN THE FOLLOWING ACTION.

**GEOM-PRIN-GROUP**

THE COMPLETE PART LIST \( (NPL(1,i), i=1, NP) \) IS PRINTED.

**GEOM-PRIN-COORDINATE**

\(<\text{NO}DF\ \text{I1} \text{ THRU I2} \rangle\)

ENTER I1 AND I2 WITH I1 BEING THE LOWER LIMIT AND I2 BEING THE UPPER LIMIT NODE NUMBERS. TO OBTAIN THE COORDINATES OF A SINGLE NODE SIMPLY ENTER JUST THE DESIRED NODE AS I1 AND DO NOT ENTER I2. I2 WILL BE ADJUSTED TO BE NOT GREATER THAN THE TOTAL NUMBER OF NODES (NJ).

**GEOM-PRIN-ELEMENT**

\(<\text{ELEMENTS I1 THRU I2} \rangle\)

ENTER I1 AND I2 WITH I1 BEING THE LOWER
LIMIT AND 12 BEING THE UPPER LIMIT ELEMENT NUMBERS.
TO OBTAIN THE NODE NUMBERS OF A SINGLE ELEMENT
SIMPLY ENTER JUST THE DESIRED ELEMENT AS I1 AND DO
NOT ENTER I2. I2 WILL BE ADJUSTED TO BE NOT GREATER
THAN THE TOTAL NUMBER OF ELEMENTS (KPT).

DISP-READ

<READ DISP FILE>

ENTER THE INPUT DISPLACEMENT FILENAME.EXT.
THE FILENAME SHOULD NOT EXCEED SIX CHARACTERS, AND
THE EXTENSION SHOULD NOT EXCEED THREE.

AT THIS POINT THE DISPLACEMENT FILE WILL BE READ
USING THE FOLLOWING FURTRAN STATEMENTS.

```
READ(IUNI,40) ((U(I,J),I=1,3),J=1,NJ)
```

40 FORMAT(6E12.5)

U = NODE DISPLACEMENTS IN THE X, Y, AND Z DIRECTIONS

DISP-WRITE

<WRITE DISP FILE>

ENTER THE OUTPUT DISPLACEMENT FILENAME.EXT.
THE FILENAME SHOULD NOT EXCEED SIX CHARACTERS, AND
THE EXTENSION SHOULD NOT EXCEED THREE.

AT THIS POINT THE DISPLACEMENT FILE WILL BE WRITTEN
USING THE FOLLOWING FURTRAN STATEMENTS.
WRITE(TUNIT,40) ((U(I,J),I=1,3),J=1,NTYPE)

40 FORMAT(1PE12.5)

U = NODE DISPLACEMENTS IN THE X, Y, AND Z DIRECTIONS

DISP-CHANGE

<NODE NUMBER AND DISPLACEMENTS>

>>>>

ENTER THE NODE NUMBER AND THE X, Y, AND Z COORDINATE DIRECTION DISPLACEMENT COMPONENTS.

DISP-PRINT

<NODE I1 THRU I2>

ENTER I1 AND I2 WITH I1 BEING THE LOWER LIMIT AND I2 BEING THE UPPER LIMIT NODE NUMBERS. TO OBTAIN THE DISPLACEMENTS OF A SINGLE NODE SIMPLY ENTER JUST THE DESIRED NODE AS I1 AND DO NOT ENTER I2. I2 WILL BE ADJUSTED TO BE NOT GREATER THAN (NJ).

DISP-TRANSFORM

<READ GEO1 FILE>

THIS IS A REQUEST FOR THE FIRST GEOMETRY FILE. SEE GEOM-READ FOR FORMAT.

<READ GEO2 FILE>

THIS IS A REQUEST FOR THE SECOND GEOMETRY FILE. SEE GEOM-READ FOR FORMAT.


MAKE

THIS LEVEL ONE COMMAND FACILITATES THE GENERATION OF
MODELs BASED UPON A SET OF PRIMITIVES WHOSE PARAMETERS AND DIMENSIONS ARE USER DEFINED. FOR EXAMPLE, ONE OF THE PRIMITIVES IS A HEXAHEdRON WHICH THE USER IS ABLE TO OBTAIN IN ANY POSITION IN SPACE AND WHOSE FACES MAY HAVE ANY DIMENSIONS OR DEGREE OF WARP. ALSO, THE USER MAY GENERATE ANY PORTION OF THE PRIMITIVE.

A TABLE SHOWING THE BREAKDOWN OF THE COMMANDS WITHIN THE MODEL GENERATOR APPEARS BELOW.

```
> MAKE

>>> SHEETS VOLUMES

>>> QUADRILATERAL HEXAHEdRON
PARALLELGRAM ELLIPSOID
SHELL REVOLUTION
REVOLUTION
```

BEFORE ENTRY INTO LEVEL two, THE USER RECEIVES THE PROMPT:

THE LEVEL TWO COMMANDS ASSOCIATED WITH THE MAKE COMMAND ARE SHEETS AND VOLUMES. SHEETS REFER TO THOSE OBJECTS WHICH ARE QUASI TWO-DIMENSIONAL. THEY MAY TAKE ON CURVATURE AND WARP OUT OF A PLANE, MUCH AS A PIECE OF PAPER, BUT THEY HAVE NO THICKNESS WHEN VIEWED ON EDGE. VOLUMES, ON THE OTHER HAND, MAY BE THOUGHT OF AS SOLID OBJECTS WHICH INDEED OCCUPY SOME VOLUME OF SPACE. THEREFORE, ONLY PANEL ELEMENT DATA MAY BE GENERATED FOR MAKE-SHEETS OBJECTS AND EITHER PANEL OR SOLID DATA MAY BE GENERATED FOR MAKE-VOLUMES OBJECTS.

A DISCUSSION OF THE VARIOUS LEVEL THREE OPTIONS ALONG WITH DEFINITIONS OF THE TERMS USED WILL NOW BE GIVEN.
A QUADRILATERAL IS A SURFACE (NOT JUST ONE ELEMENT) WHICH HAS STRAIGHT EDGES AND IS DEFINED BY 4 CORNER POINTS. THE SURFACE MAY BE EITHER PLANAR OR WARPED. AFTER GIVING THE QUADRILATERAL COMMAND, THE PROGRAM RESPONDS WITH

<COORDINATES (X,Y,Z) OF THE SEQUENTIAL CORNER POINTS>
1 2 3 4


<CORRECTIONS?>

ENTER Y OR YFS FOR CORRECTIONS, OTHERWISE THE PROGRAM JUMPS TO THE PROMPT <NUMBER OF ELEMENTS BETWEEN CORNER POINTS: 1 AND 2 (II), AND 1 AND 4 (I2)>.

<CORNER POINTS AND COORDINATES>

ENTER THE CORNER POINT NUMBER AND THE COORDINATES OF EACH CORNER POINT FOR WHICH CHANGES ARE REQUESTED. A CARRIAGE RETURN OR ZERO CORNER POINT NUMBER CAUSES THE NEW COORDINATES OF THE CORNER POINTS TO BE LISTED AND AGAIN CORRECTIONS ARE ASKED FOR. THIS PROCESS IS REPEATED UNTIL NO CORRECTIONS ARE REQUESTED. THE PROGRAM THEN RESPONDS WITH

<NUMBER OF ELEMENTS BETWEEN CORNER POINTS: 1 AND 2 (II), AND 1 AND 4 (I2)>

INPUT THE NUMBER OF ELEMENTS ALONG THE EDGE
DEFINED BY CORNER POINTS 1 AND 2, AND THE NUMBER OF
ELEMENTS ALONG THE EDGE DEFINED BY CORNER POINTS 1
AND 4

AT THIS POINT THE PROGRAM WILL PRINT A SUMMARY OF
THE TOTAL NUMBER OF PARTS, NODES, AND ELEMENTS AND RETURN TO
A LEVEL 3 PROMPT.

MAKE-SHEETS-PARALLELGRAM

A PARALLELGRAM IS ANY PLANAR SURFACE (AGAIN, NOT
JUST A SINGLE ELEMENT) Whose Opposite SIDES ARE PARALLEL.
THE PROCEDURE FOLLOWED IS THE SAME AS FOR A
MAKE-SHEETS-QUADRILATERAL COMMAND EXCEPT THAT ONLY THREE
CORNER POINTS ARE REQUIRED TO BE INPUT.

MAKE-SHEETS-SHELL

A SHELL IS THE SURFACE, OR ANY PORTION OF IT, OF AN
ELLIPSOID. AFTER ENTERING THE SHELL COMMAND, THE PROGRAM
RESPONDS WITH

<COORDINATES (X,Y,Z) OF THE ELLIPSOID’S CENTER>

ENTER THE COORDINATES OF THE COMMON ORIGIN
OF THE ELLIPSOID. ALL RADII ARE MEASURED FROM THIS
POINT.

<OUTER RADII IN THE X, Y AND Z DIRECTIONS>

ENTER THE RADII ALONG EACH COORDINATE
DIRECTION. IF ONLY ONE RADIUS IS ENTERED, A SPHERE IS GENERATED. IF TWO RADIi ARE ENTERED, A SPHEROID WITH THE RADIUS IN THE Y AND Z DIRECTIONS BEING THE SAME IS GENERATED.

<STARTING AND ENDING LATITUDE
(MEASURED IN DEGREES FROM THE XZ PLANE)>

ENTER THE STARTING AND ENDING LATITUDES.

<STARTING AND ENDING LONGITUDE
(MEASURED IN DEGREES FROM THE XY PLANE)>

ENTER THE STARTING AND ENDING LONGITUDE.

(NUMBER OF ELEMENTS:
IN LATITUDE(I1), AND IN LONGITUDE (I2)>

ENTER THE NUMBER OF ELEMENTS IN LATITUDE AND LONGITUDE.

AT THIS POINT THE PROGRAM WILL PRINT A SUMMARY OF THE TOTAL NUMBER OF PARTS, NODES, AND ELEMENTS AND RETURN TO A LEVEL 3 PROMPT.

MAKE-SHEETS-REVOLUTION

THIS SET OF COMMANDS CAUSES A SURFACE OF REVOLUTION AND/OR TRANSLATION TO BE GENERATED. IT IS FORMED BY REVOLVING A CURVE AROUND AND TRANSLATING ALONG THE Y AXIS. EXAMPLES OF THIS INCLUDE CONES, CYLINDERS, SPIRALS, AND AXISYMMETRIC SHELLS. AFTER THE USER GIVES THIS STRING OF COMMANDS, THE PROGRAM RESPONDS WITH

(NUMBER OF NODES ON THE CURVE, ANGLE OF REVOLUTION>
THE USER IS REQUIRED TO ENTER ALL THE NODES THAT ARE NECESSARY TO DESCRIBE THE CURVE THAT IS TO BE REVOLVED. ENTER THE NUMBER OF NODES THAT WILL BE REQUIRED IN ORDER TO ACCOMPLISH THIS. ALSO ENTER THE ANGLE OF REVOLUTION. KEEP IN MIND THAT THIS ANGLE MAY BE EITHER VERY SMALL (EVEN ZERO WHICH IS USEFUL TO DEVELOP A SURFACE OF TRANSLATION) OR VERY LARGE DEPENDING ON THE DESIRED FINAL RESULTS.

<COORDINATES (X,Y,Z) OF EACH NODE>

ENTER THE COORDINATES OF ALL OF THE NODES ON THE CURVE THAT IS TO BE REVOLVED. REMEMBER THAT THE NODES DO NOT HAVE TO LIE WITHIN A PLANE.

<TOTAL: TRANSLATION ALONG THE Y AXIS, CHANGE IN RADIUS>

IF IT IS DESIRED THAT THE CURVE BE INCREMENTALLY TRANSLATED ALONG THE Y AXIS AS IT IS BEING REVOLVED, ENTER THE TOTAL AMOUNT OF TRANSLATION THAT IT IS DESIRED TO TAKE PLACE. THIS ALLOWS FOR THE CREATION OF SPIRALS OF TRANSLATION. ALSO ENTER THE TOTAL CHANGE IN RADIUS THAT IS TO TAKE PLACE WHILE THE CURVE IS BEING REVOLVED. THIS TOTAL CHANGE IN RADIUS WILL BE INCREMENTALLY DIVIDED AMONG THE CIRCUMFERENTIAL ELEMENTS. THIS PROVIDES FOR THE CREATION OF RADIAL SPIRALS.

<NUMBER OF CIRCUMFERENTIAL ELEMENTS>

ENTER THE NUMBER OF ELEMENTS THAT ARE DESIRED ALONG THE CIRCUMFERENCE OF THE SURFACE OF REVOLUTION.

<WOULD YOU LIKE TO TRANSLATE AND(OR) ROTATE THIS PART?>

ENTER Y OR YES IF THIS PART IS TO BE MOVED TO A NEW LOCATION IN SPACE. IF NOT, THE PROGRAM WILL PRINT THE TOTAL NUMBER OF PARTS, NODES, AND ELEMENTS AND RETURN A LEVEL 3 PROMPT. THE METHOD
FOR ACCOMPLISHING A TRANSLATION AND/OR ROTATION IS TO MOVE THE COORDINATE AXES OF THE PART. THE NEW AXES ARE DEFINED BY:

<NEW ORIGIN>

ENTER THE COORDINATES OF THE NEW ORIGIN OF THE PART

<POINT ALONG NEW X AXIS>

ENTER THE COORDINATES OF ANY POINT ON THE NEW X AXIS

<POINT IN NEW XY PLANE>

ENTER THE COORDINATES OF ANY POINT LYING WITHIN THE NEW XY PLANE, BUT NOT ON THE NEW X AXIS.

AT THIS POINT THE PROGRAM WILL PRINT A SUMMARY OF THE TOTAL NUMBER OF PARTS, NODES, AND ELEMENTS AND RETURN TO A LEVEL 3 PROMPT.

MAKEVOLUMES-HEXAHEXDRON

A HEXAHEDRON IS DEFINED AS A VOLUME WITH STRAIGHT EDGES AND 9 CORNER POINTS. THE NUMBER OF CORNER POINTS WHICH THE USER MUST SPECIFY DEPENDS UPON THE COMPLEXITY OF THE OBJECT AND VARIES FROM FOUR IN THE SIMPLEST CASE TO EIGHT IN THE MOST COMPLEX. THE SURFACE DEFINED BY THE FIRST 4 CORNER POINTS OF THE HEXAHEDRON IS REFERRED TO AS THE FRONT FACE AND THE SURFACE DEFINED BY THE OTHER 4 CORNER POINTS IS REFERRED TO AS THE BACK FACE. CORNER POINTS 1 AND 5, 2 AND 6, 3 AND 7, AND 4 AND 8 ALWAYS DEFINE EDGES. IT IS IMPORTANT TO REMEMBER THIS WHEN ENTERING THE CORNER POINTS ON THE BACK FACE. AFTER THE HEXAHEDRON COMMAND IS GIVEN, THE PROGRAM RESPONDS WITH
<FRONT FACE INFORMATION>
<IS THIS FACE A PARALLELOGRAM?>

ENTER Y OR YES IF THE FRONT FACE IS A PARALLELOGRAM.

<COORDINATES (X,Y,Z) OF THE SEQUENTIAL CORNER POINTS>
1 2 3 4

>>>>

IF THE FRONT FACE IS A PARALLELOGRAM, ONLY 3 CORNER POINTS WILL BE REQUESTED. IF IT IS NOT, ALL 4 CORNER POINTS WILL BE REQUESTED. ENTER THE CORNER POINTS IN A COUNTERCLOCKWISE MANNER AROUND THE PERIMETER OF THE FACE. IF COUNTERCLOCKWISE ELEMENT NUMBERING IS DESIRABLE, WHEN THE ENTRY HAS BEEN COMPLETED, THE CORNER POINTS AND THEIR CORRESPONDING COORDINATES ARE LISTED.

<CORRECTIONS?>

ENTER Y OR YES FOR CORRECTIONS, OTHERWISE THE PROGRAM JUMPS TO THE PROMPT <IS THE HEXAHEDRON PRISMATIC?>.

<CORNER POINTS AND COORDINATES>

>>>>

ENTER THE CORNER POINT NUMBER AND THE COORDINATES OF EACH CORNER POINT FOR WHICH CHANGES ARE REQUESTED. A CARRIAGE RETURN OR ZERO CORNER POINT NUMBER CAUSES THE NEW COORDINATES OF THE CORNER POINTS TO BE LISTED AND AGAIN CORRECTIONS ARE ASKED FOR. THIS PROCESS IS REPEATED UNTIL NO CORRECTIONS ARE REQUESTED. THE PROGRAM THEN Responds WITH

<IS THE HEXAHEDRON PRISMATIC?>

ENTER Y OR YES IF THE HEXAHEDRON HAS A
CONSTANT CROSS-SECTION. ALSO ANSWER YES IF THE BACK FACE IS PARALLEL TO THE FRONT FACE AND HAS THE SAME SHAPE AND DIMENSIONS (JUST ROTATED AND TRANSLATED RELATIVE TO THE FRONT FACE). IF A NEGATIVE RESPONSE IS GIVEN, THE PROGRAM WILL JUMP TO THE PROMPT <PACK FACE INFORMATION>. AN AFFIRMATIVE ANSWER IS ACKNOWLEDGED WITH

<COORDINATES (X,Y,Z) OF CORNER POINT 5>

ENTER THE COORDINATES OF THE CORNER POINT ON THE BACK FACE THAT IS ON THE SAME EDGE AS CORNER POINT 1. THE PROGRAM WILL THEN JUMP TO THE PROMPT <NUMBER OF ELEMENTS BETWEEN CORNER POINTS 1 AND 7>

<BACK FACE INFORMATION>

<IS THIS FACE A PARALLELOGRAM?>

THE SAME PROCEDURE USED TO DEFINE THE FRONT FACE IS AGAIN FOLLOWED IN ORDER TO DEFINE THE BACK FACE. WHEN THE INPUT AND CORRECTION PROCESS HAS BEEN COMPLETED, THE PROGRAM PROMPTS WITH

<NUMBER OF ELEMENTS BETWEEN CORNER POINTS: 1 AND 2 (I1), 1 AND 4 (I2), AND 1 AND 5 (I3)>

INPUT THE NUMBER OF ELEMENTS ALONG THE EDGE DEFINED BY CORNER POINTS 1 AND 7, AND THE NUMBER OF ELEMENTS ALONG THE EDGE DEFINED BY CORNER POINTS 1 AND 4, AND THE NUMBER OF ELEMENTS ALONG THE EDGE DEFINED BY CORNER POINTS 1 AND 5.

AT THIS POINT THE PROGRAM WILL PRINT A SUMMARY OF THE TOTAL NUMBER OF PARTS, NODES, AND ELEMENTS AND RETURN TO A LEVEL 3 PROMPT.

MAKE-VOLUMES-ELLIPSOID
<number of elements: radially(11),
in latitude(12), and in longitude(13)>

enter the number of elements that are
desired radially on the thick-shelled volume, the
number of elements in latitude, and the number of
elemental divisions in longitude.

at this point the program will print a summary of
the total number of parts, nodes and elements, and return to
a level 3 prompt.

make-volumes-revolution

this set of commands causes a closed surface of
revolution and/or translation to be generated. two special
cases of this are the torus, which is formed when the loop
(i.e. closed surface) is a circle, and the cylindrical
shell, created by a rectangular loop. the input information
and dialogue required for their creation is the same as that
of make-sheets-revolution, the only difference being that
the word loop replaces curv. geometrically, their only
difference is that elements are generated between the last
and first user specified nodes, thus closing the loop, for a
volume of revolution.

symmetry

the symmetry command provides modification to the
geometry and displacement. the symmetry operation will
double the number of parts and elements, but the
number of nodes may not double due to the presence of nodes
on the plane of symmetry (which are not repeated). if
before the symmetry operation there were (np) parts, the
symmetrical counterpart to the nth part is found to be the
np+nth part.
AN ELLIPSOID IS A THICK-SHELLED VOLUME WHOSE
BOUNDARIES MAY BE DESCRIBED BY ELLIPSOIDS. ANY PORTION OF
THESE OBJECTS MAY BE GENERATED. FURTHERMORE, THE OUTER AND
INNER ELLIPSOIDS OF THE VOLUME NEED NOT BE SIMILAR. THE
PROGRAM'S REPLY TO THE ELLIPSOID COMMAND IS

<COORDINATES (X,Y,Z) OF THE ELLIPSOID'S CENTER>

ENTER THE COORDINATES OF THE COMMON ORIGIN
OF THE OUTER AND INNER ELLIPSOIDS. ALL RADII ARE
MEASURED FROM THIS POINT.

<OUTER RADII IN THE X, Y AND Z DIRECTIONS>

ENTER THE RADII ALONG EACH COORDINATE
DIRECTION OF THE ELLIPSOID WHICH DESCRIBES THE OUTER
BOUNDARY OF A THICK-SHELLED VOLUME. IF ONLY ONE
RADIUS IS ENTERED, A SPHERE IS GENERATED. IF TWO
RADII ARE ENTERED, A SPHEROID WITH THE RADII IN THE
Y AND Z DIRECTIONS BEING THE SAME IS GENERATED.

<INNER RADII IN THE X, Y AND Z DIRECTIONS>

ENTER THE RADII ALONG EACH COORDINATE
DIRECTION OF THE ELLIPSOID WHICH DESCRIBES THE INNER
BOUNDARY OF A THICK-SHELLED VOLUME. THE SAME RULES
FOR SPHERE AND SPHEROID GENERATION THAT WERE
EXPLAINED FOR THE OUTER ELLIPSOID APPLY HERE.

<STARTING AND ENDING LATITUDE
(MEASURED IN DEGREES FROM THE XZ PLANE)>

ENTER THE STARTING AND ENDING LATITUDES.

<STARTING AND ENDING LONGITUDE
(MEASURED IN DEGREES FROM THE XY PLANE)>

ENTER THE STARTING AND ENDING LONGITUDES.
<WHICH SYMMETRY PLANE>

ENTER ONE OF THE ACCEPTABLE SYMMETRY PLANES XY, XZ, OR YZ, OR A CARRIAGE RETURN TO PROVIDE ESCAPE.

<MULTIPLE FILES?>

ENTER Y OR YES TO PERFORM THE SYMMETRY OPERATION ON MULTIPLE DISPLACEMENT FILES. ANY OTHER RESPONSE WILL SKIP THE REMAINING REQUESTS.

(NUMBER OF DISPLACEMENT FILES>

ENTER THE NUMBER OF DISPLACEMENT FILES FOR WHICH SYMMETRY OPERATIONS ARE DESIRED. A ZERO OR CARRIAGE RETURN WILL SKIP FURTHER REQUESTS FOR DISPLACEMENT FILE INFORMATION.

<READ DISP FILE>

SEE DISP-READ FOR FORMAL INFORMATION.

<WRITE DISP FILE>

SEE DISP-WRIT FOR FORMAL INFORMATION

THE TWO REQUESTS ABOVE ARE REPEATED FOR THE NUMBER OF DISPLACEMENT FILES SPECIFIED.

ORDER

THE ORDER COMMAND ATTEMPTS TO CONSISTENTLY ORDER THE
PANEL DATA OF THE IP ARRAY IN A CLOCKWISE OR COUNTER-CLOCKWISE MANNER. THE FIRST ELEMENT IN EACH GROUP IS ASSUMED TO BE ORDERED CORRECTLY. THIS ASSUMPTION MAY BE CHECKED USING A GEOM-PRINT-ELEMENT SEQUENCE OF COMMANDS. ALL OTHER ELEMENTS IN THE GROUP ARE MATCHED AGAINST PREVIOUSLY ORDERED ELEMENTS UNTIL THE PROCESS IS COMPLETE. IF AN ELEMENT CANNOT BE MATCHED, THE MESSAGE <WARNING: ELEMENT HAS NO NEIGHBORS! IS Typed ON THE USERS TERMINAL.

MERGE

THE MERGE COMMAND ALLOWS THE USER TO APPEND GEOMETRY AND DISPLACEMENT FILES TO THOSE PREVIOUSLY READ IN. ANY NUMBER OF FILES MAY BE ADDED. IF NO DISPLACEMENT FILE IS READ, THE CORRESPONDING LOCATIONS OF THE DISPLACEMENT ARRAY IS SET TO ZERO. SEE GEOM-READ, DISP-READ, FOR VARIABLE LISTS AND SPECIFIED FORMAT. A BLANK GEOMETRY FILE NAME TERMINATES THIS COMMAND.

CLEAN

THE CLEAN COMMAND ALLOWS THE USER TO REDUCE DATA FILES WHICH CONTAIN MORE THAN ONE NODE AT A GIVEN LOCATION AND/OR HAVE DEGENERATE POLYGONS (I.E. A DEGENERATE QUADRILATERAL IS REDUCED TO A TRIANGLE). THE CHECKING FOR MULTIPLE NODES AT NEARBY LOCATIONS IS DONE ON AN ELEMENT GROUP BASIS. TWO NODES AT THE SAME LOCATION WILL SURVIVE THIS PROCESS AS LONG AS NO INDIVIDUAL GROUP REFERS TO BOTH NODE NUMBERS.

<NODE CONSOLIDATION DISTANCE>

ENTER THE MAXIMUM DISTANCE BY WHICH TWO NODES MAY BE SEPARATED AND YET BE REPLACED BY THE COORDINATES OF THE LOWER NUMBERED NODE AND GIVEN A COMMON NODE NUMBER. PROVISION IS MADE FOR CORRESPONDING REDUCTIONS IN DISPLACEMENT FILE. SEE THE SYMFIRI COMMAND STARTING AT <MULTIPLE FILES?>.
BEFORE RETURNING TO LEVEL ONE, THE PROGRAM REPORTS 
THE RESULTS OF THIS COMPUTATION IN TERMS OF THE NUMBER OF 
NODES AND POLYGON EDGES WHICH HAVE BEEN ELIMINATED. 

USE OF A MAKE COMMAND AUTOMATICALLY EXECUTES THE 
CLEAN COMMAND WITH THE NODE CONSOLIDATION DISTANCE SET EQUAL 
TO ZERO AND THE REPORT ON THE REDUCTION PROCESS SKIPPED. 

EXIT 

THIS COMMAND ALLOWS A GRACEFUL EXIT FROM THE 
PROGRAM. THIS COMMAND CAN BE USED AT LEVEL ONE, TWO, OR 
THREE. 

ERROR MESSAGES 

THERE ARE THREE WARNING AND THREE ERROR MESSAGES THAT 
MAY BE ISSUED DURING A SESSION. THE ERROR MESSAGES ARE 
CONSIDERED FATAL AND WILL CAUSE PROGRAM EXECUTION TO 
TERMINATE. THE MESSAGES ARE: 

<WARNING: ELEMENT NOT THERE!> 

ATTEMPT TO DELETE AN ELEMENT THAT WAS NOT 
FOUND. SEE GEO-M-CHAN-FPLM. 

<WARNING: ELEMENT HAS NO NEIGHBORS!> 

WHILE ATTEMPTING TO ORDER PANEL DATA, AN 
ELEMENT WAS FOUND IN AN ELEMENT GROUPING THAT DID 
NOT HAVE A COMMON EDGE WITH ANY OTHER ELEMENT IN 
THAT GROUP. See ORDER.
<WARNING: TRANSFORM COMMAND ABORT!>

GEOMETRICAL FILES ARE NOT COMPATIBLE IN THE NUMBER OF NODES. THE NUMBER OF NODES AND THE VALUES IN THE DISPLACEMENT ARRAY HAVE BEEN LEFT UNTOUCHED BY THE TRANSFORM COMMAND.

<ERROR: ATTEMPT TO EXCEED NPMAX OF NNNNN!>

THE USER ATTEMPTED TO EXCEED THE MAXIMUM DIMENSION OF NPMAX. INCREASE THE VALUE OF NPMAX IN THE MAIN PROGRAM AND TRY AGAIN.

<ERROR: ATTEMPT TO EXCEED NJMAX OF NNNNN!>

THE USER ATTEMPTED TO EXCEED THE MAXIMUM DIMENSION OF NJMAX. INCREASE THE VALUE OF NJMAX IN THE MAIN PROGRAM AND TRY AGAIN.

<ERROR: ATTEMPT TO EXCEED NPIMAX OF NNNNN!>

THE USER ATTEMPTED TO EXCEED THE MAXIMUM DIMENSION OF NPIMAX. INCREASE THE VALUE OF NPIMAX IN THE MAIN PROGRAM AND TRY AGAIN.
APPENDIX E

RESULTS
UTILITY Program Results

Figure 15 thru Figure 30 are presented to demonstrate program UTILITY capabilities. Preceding every figure the program run used to generate the figure is presented.
RUN DF1:UTILITY
<MOYIE SYSTEM UTILITY AND MODEL GENERATION>

+> MAKE
< WILL THIS BE A "SHEET OR VOLUME " TYPE >
+>> SHEET
< PARA , QUAD , SHELL OR REVOL >
+>>> QUAD
<ENTER 4 CORR(X,Y,Z) OF THE QUADRILATERAL SEQUENTIAL CORNER POINTS> ( 1, 2, 3, 4)
+>>> 25,25,0
+>>> -30,30,0
+>>> -10,-10,0
+>>> 25,-15,0
<THESE ARE THE ENTERED CORNER POINTS AND COORDINATES>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>25.00</td>
<td>25.00</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>-30.00</td>
<td>30.00</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>-10.00</td>
<td>-10.00</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>25.00</td>
<td>-15.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

<CORRECTIONS? > N

<NUMBER OF ELEMENTS BETWEEN CORNER POINTS: 1 AND 2 (I1), AND 1 AND 4 (I2)) 5,5
<CURRENT TOTALS: 7 PARTS, 36 NODES, 25 ELEMENTS>

< PARA , QUAD , SHELL OR REVOL >
+>>> < WILL THIS BE A "SHEET OR VOLUME " TYPE >
```plaintext
+>> GEOM
+>> WRITE
< ENTER WRIT GEOM FILE >DK1:QUAD.TST

<ELEMENT LIMITS OF PARTS>
   1   25
<OK?> Y
+>> MAKE
+WILL THIS BE A "SHEET OR VOLUME " TYPE >
+>> SHEET
< PARA, QUAD, SHELL OR REVOL >
+>> QUAD
<ENTER 4 COOR.(X,Y,Z) OF THE QUADRILATERAL SEQUENTIAL CORNER POINTS: ( 1, 2, 3, 4)
+>>> 25,25,10
+>>> -20,30,0
+>>> -10,-10,-10
+>>> 25,-15,0
<THESE ARE THE ENTERED CORNER POINTS AND COORDINATES>

  1  25.00  25.00  10.00
  2 -20.00  30.00   0.00
  3 -10.00 -10.00 -10.00
  4  25.00 -15.00   0.00

<CORRECTIONS?> N
<NUMBER OF ELEMENTS BETWEEN CORNER POINTS>
```
1 AND 2 (I1), AND 1 AND 4 (I2) 5,5
(CURRENT TOTALS: 2 PARTS, 72 NODES, 50 ELEMENTS)

< PARA, QUAD, SHELL OR REVOL >
++++
< WILL THIS BE A "SHEET OR VOLUME " TYPE >
++++
> GEOM
++++ WRITE
< ENTER WRIT GEOM FILE >DK1:QUADW.TST
<ELEMENT LIMITS OF PARTS>
   1  25  26  50
Figure 15. This figure shows two quadrilateral 90 degrees rotation around the X-axis. The line is the planar quadrilateral and the other is the warped quadrilateral.
.RUN DK1:UTILITY
<MOVIE SYSTEM UTILITY AND MODEL GENERATION>

+> MAKE
< WILL THIS BE A "SHEET OR VOLUME " TYPE >
+>> SHEET
< PARA , QUAD , SHELL OR REVOL >
+>> PARA
< ENTER 3 COOR.(X,Y,Z) OF THE PARALLELOGRAM SEQUENTI
CORNER POINTS >( 1, 2, 3)
+>>> 25,25,0
+>>> -25,25,0
+>>> -40,-25,0
<THESE ARE THE ENTERED CORNER POINTS AND COORDINATES>

1  25.00  25.00  0.00
2  -25.00  25.00  0.00
3  -40.00 -25.00  0.00
4   10.00 -25.00  0.00

<CORRECTIONS?> N

<NUMBER OF ELEMENTS BETWEEN CORNER POINTS:
  1 AND 2 (11), AND 1 AND 4 (12)> 5,5,
<CURRENT TOTALS:  1 PARTS, 36 NODES, 25 ELEMENTS>
< PARA, QUAD, SHELL OR REVOL >

< WILL THIS BE A "SHEET OR VOLUME " TYPE >

< WRITE
< GEOM
< WRITE

< ENTER WRIT GEOM FILE >DK1:PARA.TOT
< ELEMENT LIMITS OF PARTS>
Figure 16. This figure shows a parallelogram created by the MAKE-PARALLELogram command.
RUN DK1:UTILITY
<MOVIE.SYSTEM UTILITY AND MODEL GENERATION>

> MAKE
< WILL THIS BE A "SHEET OR VOLUME " TYPE >
> SHEET
< PARA, QUAD, SHELL OR REVOL >
>>> SHELL
<COORDINATES (X,Y,Z) OF THE ELLIPSOID'S CENTER> 0,0,0
<OUTER RADIUS IN THE X, Y AND Z DIRECTIONS> 10
<STARTING AND ENDING LATITUDE
<MEASURED IN DEGREES FROM THE XZ PLANE> -90,90
<STARTING AND ENDING LONGITUDE
<MEASURED IN DEGREES FROM THE XY PLANE> 0,360
(NUMBER OF ELEMENTS:
IN LATITUDE (I1), AND IN LONGITUDE (I2) > 10,12
<CURRENT TOTALS: 1 PARTS, 119 NODES, 120 ELEMENTS>

< PARA, QUAD, SHELL OR REVOL >
>>>>
< WILL THIS BE A "SHEET OR VOLUME " TYPE >
>>>>
>>> GEOM
>>> WRITE
< ENTER WRIT GEOM FILE > DK1:CSHELL.TST
<ELEMENT LIMITS OF PARTS>
1 120
Figure 17. This figure shows a shell sphere created by the MAKE-SHELL-ELLIPsoid command.
Figure 18. This figure shows figure 17 sphere with the hidden lines removed.
.RUN DK1:UTILITY
(MOVIE SYSTEM UTILITY AND MODEL GENERATION)

+ > MAKE
  < WILL THIS BE A "SHEET OR VOLUME " TYPE >
+ >> SHELL
  < WILL THIS BE A "SHEET OR VOLUME " TYPE >
+ >> SHEET
  < PARA , QUAD , SHELL OR REVOL >
+ >> SHELL
  < COORDINATES (X,Y,Z) OF THE ELLIPSOID'S CENTER> 0,0,0
  < OUTER RADII IN THE X, Y AND Z DIRECTIONS> 20,5
  < STARTING AND ENDING LATITUDE
  (MEASURED IN DEGREES FROM THE XZ PLANE)> -90,90
  < STARTING AND ENDING LONGITUDE
  (MEASURED IN DEGREES FROM THE XY PLANE)> 30,360
  < NUMBER OF ELEMENTS:
    IN LATITUDE (I1), AND IN LONGITUDE (I2)> 10,12
  < CURRENT TOTALS:  1 PARTS, 119 NODES, 120 ELEMENTS>
  < PARA , QUAD , SHELL OR REVOL >
+ >>>
  < WILL THIS BE A "SHEET OR VOLUME " TYPE >
+ >>
+ > GEOM
+ >> WRITE
  < ENTER WRIT GEOM FILE >DK1:SH ELL.TST
  < ELEMENT LIMITS OF PARTS>
    1 120
Figure 19. This figure shows a shell spheroid created by the MAKE-SHELL-ELLIPsoid command.
RUN DK1:UTILITY <MOVIE SYSTEM UTILITY AND MODEL GENERATION>

MAKE SHEET, OR SHELL OR REVOL?

> SHEET OR VOLUME "TYPE >

COORDINATES (x, y, z) OF THE ELLIPSOID'S CENTER: 0, 0, 0

COORDINATES OF THE ELLIPSOID DIRECTIONS: 20, 10, 5

OUTER RADIUS IN THE x, y AND z DIRECTIONS: 60, -60

STARTING AND ENDING LATITUDE FROM THE xz PLANE: 0, 180

STARTING AND ENDING LONGITUDE FROM THE xy PLANE: 16, 12

NUMBER OF ELEMENTS:

IN LATITUDE (I2): 16, 12

IN CURRENT TOTALS: 1 PARTS, 221 HOLES, 192 ELEMENTS

CURRENT TOTALS:

PARA, QUAD, SHELL OR REVOL

WILL THIS BE A "SHEET OR VOLUME " TYPE >

WILL THIS BE A "SHEET OR VOLUME " TYPE >

WRITE <ELEMENT LIMITS OF PARTS>

WRT GEOM FILE >DK1:ESHELL.TST

1 192
Figure 20. This figure shows a shell ellipsoid created by the MAKE-SHELL-ELLipsoid command.
Figure 21. This figure shows figure 20 rotated 90 degrees around the Y-axis.
\texttt{MAKE}

\texttt{< WILL THIS BE A "SHEET OR VOLUME " TYPE >}

\texttt{>>> SHEET}

\texttt{< PARA, QUAD, SHELL OR REVOL >}

\texttt{>>> REVOL}

\texttt{< NUMBER OF NODES ON THE CURV. ANGLE OF REVOLUTION > 7, 360}

\texttt{< COORD. S (X, Y, Z) OF EACH NODE >}

\texttt{>>> 10, 0, 0}

\texttt{>>> 15, 2.5, 0}

\texttt{>>> 15, 6, 0}

\texttt{>>> 13, 10, 0}

\texttt{>>> 13, 13, 0}

\texttt{>>> 20, 20, 0}

\texttt{>>> 20, 23, 0}

\texttt{< TOTAL: TRANSLATION ALONG Y AXIS, CHANGE IN RADIUS > 0, 0}

\texttt{< NUMBER OF CIRCUMFERENTIAL ELEMENTS > 15}

\texttt{< WOULD YOU LIKE TO TRANSLATE AND(OR) ROTATE THIS PART >} \texttt{N}

\texttt{< CURRENT TOTALS: 1 PARTS, 105 NODES, 90 ELEMENTS >}

\texttt{< PARA, QUAD, SHELL OR REVOL >}

\texttt{>>>}

\texttt{< WILL THIS BE A "SHEET OR VOLUME " TYPE >}

\texttt{>>>}

\texttt{GEOM}

\texttt{>>> WRITE}

\texttt{< ENTER WRIT GEOM FILE >DK1:SREV1.TST}

\texttt{< ELEMENT LIMITS OF PARTS >}

\texttt{1 90}
Figure 22. This figure shows an object created using the MAKE-SHELL-REVOLUTION command.
Figure 23. This figure shows figure 22 with the hidden lines removed.
RUN DK1:UTILITY
(MOVIE SYSTEM UTILITY AND MODEL GENERATION)

+> MAKE
(WILL THIS BE A "SHEET OR VOLUME " TYPE )
+>> SHEET
(PARA, QUAD, SHELL OR REVOL )
+>> REVOL
(NUMBER OF NODES ON THE CURV, ANGLE OF REVOLUTION ) 2,180
(COORD.S (X,Y,Z) OF EACH NODE)
+>>> 5,0,0
+>>> 10,20,0
(TOTAL : TRANSLATION ALONG Y AXIS, CHANGE IN RADIUS) 20,10
(NUMBER OF CIRCUMFERENTIAL ELEMENTS) 20
(WOULD YOU LIKE TO TRANSLATE AND (OR) ROTATE THIS PART ?) N
(CURRENT TOTALS: 1 PARTS, 42 NODES, 20 ELEMENTS)

(PARA, QUAD, SHELL OR REVOL )
+>>>)
(WILL THIS BE A "SHEET OR VOLUME " TYPE )
+>>)
+> GEOM
+>> WRITE
(ENTER WRIT GEOM FILE >DK1:SREV2.TST
(ELEMENT LIMITS OF PARTS)
  1 20
(OK?) Y
+>> EXIT
Figure 24 - This figure shows a model created by the MAKE-SHELL-REVOLUTION command using the translation on the Y-axis and change in radius.
MAKE

< WILL THIS BE A "SHEET OR VOLUME " TYPE >

VOLUME

ENTER ( HEXA, ELLI OR REVO )

HEXA

FRONT FACE INFORMATION

< IS THIS FACE A PARALLELOGRAM > Y

< ENTER 3 COOR. ( X,Y,Z ) OF THE PARALLELOGRAM SEQUENTI
CORNERS POINTS >( 1, 2, 3)

25,25,10

-25,25,10

-25,-25,10

< THESE ARE THE ENTERED CORNER POINTS AND COORDINATES >

1 25.00 25.00 10.00

2 -25.00 25.00 10.00

3 -25.00 -25.00 10.00

4 25.00 -25.00 10.00

CORRECTIONS ? N

< IS THE HEXAHEDRON PRISOMATIC > N

BACK FACE INFORMATION

< IS THIS FACE A PARALLELOGRAM > Y

< ENTER 3 COOR. ( X,Y,Z ) OF THE PARALLELOGRAM SEQUENTI
CORNERS POINTS >( 5, 6, 7)

40,40,10

-40,40,10

-40,-40,10

< THESE ARE THE ENTERED CORNER POINTS AND COORDINATES >
<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>40.00</td>
<td>40.00</td>
<td>10.00</td>
</tr>
<tr>
<td>6</td>
<td>-40.00</td>
<td>40.00</td>
<td>10.00</td>
</tr>
<tr>
<td>7</td>
<td>-40.00</td>
<td>-40.00</td>
<td>10.00</td>
</tr>
<tr>
<td>8</td>
<td>40.00</td>
<td>-40.00</td>
<td>10.00</td>
</tr>
</tbody>
</table>

CORRECTIONS? Y

ENTER CORRECTION (NODE, X, Y, Z) ENTER NODE=0 TO FINISH

>>> 5, 40, 40, -10

>>> 6, -40, 40, -10

>>> 7, -40, -40, -10

THESE ARE THE ENTERED CORNER POINTS AND COORDINATES

< CORRECTIONS? > N

NUMBER OF ELEMENTS BETWEEN CORNER POINTS:
1 AND 2 (I1), 1 AND 4 (I2), AND 1 AND 5 (I3): 5, 5, 4
CURRENT TOTALS: 1 PARTS, 132 NODES, 130 ELEMENTS

ENTER < HEXA, ELLI OR REVO>

WILL THIS BE A "SHEET OR VOLUME" TYPE?
GEOM
WRITE
FILE
>OK1:HEXA1.TST

<ELEMENT LIMITS OF PARTS>

1
130

<OK?> Y

EXIT
Figure 25. This figure shows an hexahedron created using the MAKE-VOLUME HEXahedron command.
Figure 26. This figure shows figure 25 rotated - 90 degrees around the Y-axis.
Figure 27. This figure shows figure 25 with hidden lines removed.
MAKE
<br>
WILL THIS BE A "SHEET OR VOLUME " TYPE
<br>
VOLU
<br>
ENTER ( HEXA , ELLI OR REVO )
<br>
ELL
<br>
COORDINATES (X, Y, Z) OF THE ELLIPSOID'S CENTER> 0,0,0
<br>
OUTER RADI IN THE X, Y AND Z DIRECTIONS> 10,5,10
<br>
INNER RADI IN THE X, Y AND Z DIRECTIONS> 5,2,5,5
<br>
STARTING AND ENDING LATITUDE
<br>
(MEASURED IN DEGREES FROM THE XZ PLANE)> 0,30
<br>
STARTING AND ENDING LONGITUDE
<br>
(MEASURED IN DEGREES FROM THE XY PLANE)> 0,180
<br>
NUMBER OF ELEMENTS: RADIALLY (I1)
<br>
IN LATITUDE (I2) , AND IN LONGITUDE (I3) > 5,5,8
<br>
CURRENT TOTALS: 1 PARTS, 212 NODES, 210 ELEMENTS
<br>
ENTER ( HEXA , ELLI OR REVO )
<br>
WILL THIS BE A "SHEET OR VOLUME " TYPE
<br>
ORDER
<br>
GEOM
<br>
WRITE
<br>
ENTER WRIT GEOM FILE >DK1:VELL11.TST
<br>
ELEMENT LIMITS OF PARTS>
<br>
1 210
<br>
OK? Y
Figure 28. This figure shows a solid ellipsoid created by the MAKE-VOLUME-ELLIPsoid command.
MAKE
< WILL THIS BE A "SHEET OR VOLUME " TYPE >

<<<< VOLUME
< ENTER ( HEXA, ELLI OR REVO) >
<<<< REVO

(NUMBER OF NODES ON THE LOOP, ANGLE OF REVOLUTION) 6, 180
(COORDS (X,Y,Z) OF EACH NODE)
<<<< 5, 0, 0
<<<< 30, 10, 0
<<<< 5, 20, 0
<<<< 5, 18, 0
<<<< 25, 10, 0
<<<< 5, 3, 0

(TOTAL: TRANSLATION ALONG Y AXIS, CHANGE IN RADIUS) 5, 0
(NUMBER OF CIRCUMFERENTIAL ELEMENTS) 15
(WOULD YOU LIKE TO TRANSLATE AND/OR ROTATE THIS PART?) N
(CURRENT TOTALS: 1 PARTS, 96 NODES, 90 ELEMENTS)

<<<< EN T E R ( HEXA, ELLI OR REVO) >
<<<<
<<<< WILL THIS BE A "SHEET OR VOLUME " TYPE >
<<<<
<<<< DK1: VREV.TST
<<<< GEOM
<<<< WRITE
<<<< ENTER WRIT GEOM FILE >DK1: VREV.TST
<<<< ELEMENT LIMITS OF PARTS
<<<< 1 20
Figure 29. This figure shows a solid object created using the MAKE-VOLUME-REVOLUTION command.
MOVIE Program Results

The first part of this section presents the UTILITY program run used to generate a "Bullet" model. This "Bullet" model is used in Figure 31 thru Figure 45 to demonstrate program MOVIE capabilities. Preceding every figure program MOVIE run used to display the figure is presented.
```
.RUN DI1:UTILITY
<NOVIE SYSTEM UTILITY AND MODEL GENERATION>

+> GEOM
+> CHANGE
+> GROUP
(NUMBER OF ELEMENT GROUPS) 3
(REORDER OLD GROUPS?) NO
(NUMBER OF ELEMENTS IN GROUPS)

+>>> COOR
(CHANGE NUMBER OF NODES?) NO
(NODE NUMBER AND COORDINATES)
>>> 1,0,0,12
>>> 2,0,1.5,11.6
>>> 3,0,0.75,1.3,11.6
>>> 4,1.3,0.75,11.6
>>> 5,1.5,0.11.6
>>> 6,0,2.6,10.5
>>> 7,1.3,2.25,10.5
>>> 8,2.25,1.3,10.5
>>> 9,2.6,0,10.5
>>> 10,0,3,9
>>> 11,1.5,2.6,9
>>> 12,2.6,1.5,9
>>> 13,3,0,9
>>> 14,0,3,6
```
\begin{verbatim}
>>> 15, 1, 5, 2, 6, 6
>>> 16, 2, 6, 1, 5, 6
>>> 17, 3, 0, 6
>>> 18, 0, 3, 3
>>> 19, 1, 5, 2, 6, 3
>>> 20, 2, 6, 1, 5, 3
>>> 21, 3, 0, 3
>>> 22, 0, 3, 0
>>> 23, 1, 5, 2, 6, 0
>>> 24, 1, 6, 1, 5, 0
>>> 25, 7, 0, 0
>>> 26, 0, 0, 0

>>> ELEM
<CHANGES FOR N-NODE ELEMENTS, ENTER VALUE FOR N> 4
<ADD, DELETE, OR REPLACE> ADD
<GROUP, 4 NODE>, CENTRAL NODE

>>> 1, 1, 2, 3
>>> 1, 1, 3, 4
>>> 1, 1, 4, 5
>>> 1, 2, 6, 7, 3
>>> 1, 3, 7, 8, 4
>>> 1, 4, 8, 9, 5
>>> 1, 6, 10, 11, 7
>>> 1, 7, 11, 12, 8
>>> 1, 8, 12, 13, 9
>>> 2, 10, 14, 15, 11
>>> 2, 11, 15, 16, 12
\end{verbatim}
<table>
<thead>
<tr>
<th>NODE</th>
<th>X-COOR</th>
<th>Y-COOR</th>
<th>Z-COOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0000E+01</td>
<td>0.0000E+01</td>
<td>1.2000E+01</td>
</tr>
<tr>
<td>2</td>
<td>0.0000E+01</td>
<td>1.5000E+00</td>
<td>1.1600E+01</td>
</tr>
<tr>
<td>3</td>
<td>7.5000E+00</td>
<td>1.3000E+00</td>
<td>1.1600E+01</td>
</tr>
<tr>
<td>4</td>
<td>1.3000E+00</td>
<td>7.5000E+00</td>
<td>1.1600E+01</td>
</tr>
<tr>
<td>5</td>
<td>1.5000E+00</td>
<td>0.0000E+00</td>
<td>1.1600E+01</td>
</tr>
<tr>
<td>6</td>
<td>0.0000E+01</td>
<td>2.6000E+00</td>
<td>1.0500E+01</td>
</tr>
<tr>
<td>7</td>
<td>1.3000E+00</td>
<td>2.2500E+00</td>
<td>1.0500E+01</td>
</tr>
<tr>
<td>8</td>
<td>2.2500E+00</td>
<td>1.3000E+00</td>
<td>1.0500E+01</td>
</tr>
<tr>
<td>9</td>
<td>2.6000E+00</td>
<td>0.0000E+00</td>
<td>1.0500E+01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ELEMENT I1/I2</th>
</tr>
</thead>
<tbody>
<tr>
<td>+&gt;&gt;COOR</td>
</tr>
<tr>
<td>NODE I1/I2 1,30</td>
</tr>
</tbody>
</table>

Note: The table represents the coordinates of nodes for a finite element analysis.
<table>
<thead>
<tr>
<th>No</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.0000E-01</td>
<td>3.0000E+00</td>
<td>9.0000E+00</td>
<td>1.5000E+00</td>
<td>2.6000E+00</td>
<td>1.5000E+00</td>
<td>3.0000E+00</td>
<td>0.0000E+00</td>
<td>3.0000E+00</td>
<td>0.0000E+00</td>
<td>3.0000E+00</td>
<td>0.0000E+00</td>
<td>3.0000E+00</td>
<td>0.0000E+00</td>
<td>3.0000E+00</td>
<td>0.0000E+00</td>
<td>3.0000E+00</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NODE 11/12**

```
<<< DISP
>>> CHANGE

<NODE NUMBER AND DISPLACEMENTS>

>>> 14, 0, -0.4, 1.5
>>> 15, -0.2, -0.35, 1.5
>>> 16, -0.35, -0.2, 1.5
>>> 17, -0.4, 0, 1.5
>>> 18, 0, -1.5, 3.4
```

```
20 -1.3000E+00 -7.5000E-01 0.0000E+00
21 -1.5000E+00 0.0000E+00 3.4000E+00
22 0.0000E+00 -1.5000E+00 0.0000E+00
23 -7.5000E-01 -1.3000E+00 0.0000E+00
24 -1.3000E+00 0.0000E+00 -1.5000E+00
25 -1.5000E+00 0.0000E+00 0.0000E+00
26 0.0000E+00 0.0000E+00 0.0000E+00

(NODE I1.I2)

+> SYMM
(WHICH SYMMETRY FLANE) YZ
(MULTIPLE FILES?) NO

+> GROUP

(ELEMENT GROUP LIST)

9 9 3 9 9 3

+> GROUP

(CHAN)

(NUMBER OF ELEMENT GROUPS) 3
(REORDER OLD GROUPS?) NO
(NUMBER OF ELEMENTS IN GROUPS)
15, 15, 6

(MOVE)

(MOVE I1 ELEMENTS STARTING AT I2 TO FOLLOW ELEMENT I3)
9, 22, 9
9, 31, 27

SYMM
(WHICH SYMMETRY PLANE) XZ
(MULTIPLE FILES?) NO

GEOM
PRINT
GROUP
(ELEMENT GROUP LIST)
18 18 6 18 18 6

CHANGE
GROUP
(NUMBER OF ELEMENT GROUPS) 3
(REORDER OLD GROUPS?) YES
(NEW GROUP NUMBER = OLD GROUP NUMBERS)
(GROUP) 1 = 1, 4
(GROUP) 2 = 2, 5
(GROUP) 3 = 3, 6

PRINT
GROUP
<ELEMENT GROUP LIST>
   36 36 12

+>>>
+>> WRITE
< ENTER WRIT GEOM FILE >DK1:GBULET.TST
<ELEMENT LIMITS OF PARTS>
   1   36  37  72  73  94
<OK?> YES
+>>
+> DISP
+>> WRITE
< ENTER WRIT DISP FILE >DK1:DBULET.TST
+>>
+> EXIT
Figure 30. Bullet model displayed as defined in program UTILITY. No rotation or translation has been applied.
HELP> YES
(COMMANDS: ANIMATE, CENTER, DEVICE, DISTANCE)
(DRAW, EXIT, EXPLODE, FAST, FIELD, HELP, IMMUNE)
(NODE, PARTS, PILOT, POLY, READ, RESTORE, ROTATE, SCALE, SCOPE, SHIFT)
(SHRINK, SUMMARY, TRANSLATE, VIEW)
>> FAST
(MIXED DATA?) N
(POOR MAN'S PROCEDURE?) Y
(PARTS II/12 IMMUNE TO POOR MAN'S PROCEDURE)
>>
(PARTS II/12 WITH CLOCKWISE ORDERING)
>> 1, 3
>>
>> ROTATE
(AXIS, ANGLE) X, 90
>> VIEW
Figure 31. Bullet model shown with 90 degree rotation around the X-axis and hidden lines removed.
>> TITLE
<DO YOU WANT A TITLE ? > Y

*** MAXIMUM NUMBER OF CHARACTERS WITH
*** CURRENT RESOLUTION IS = 37
  NOW ENTER YOUR LINE
***** BULLET 90 X-AXIS -45 Z-AXIS *****
  MAXIMUM NUMBER OF CHARACTERS EXCEEDED
< DO YOU WANT TO TRY AGAIN ? > Y

*** MAXIMUM NUMBER OF CHARACTERS WITH
*** CURRENT RESOLUTION IS = 37
  NOW ENTER YOUR LINE
BULLET 90 X-AXIS -45 Z-AXIS
>> ROTA
<AXIS, ANGLE> Z,-45
>> DRAW
Figure 32. Bullet model shown with a title and the tip of the picture frame.
SCOP
<SUPPRESS COORDINATE TRAID?> N
< ENTER DISPLAY RESOLUTION (1 - 780 ) > 570
EXPLODE
(PARTS II-12, LOCAL MOTION (X,Y,Z))
>>> 1,1,0,0,2
>>> 3,3,0,0,-2

<LOCAL MOTION SCALE FACTOR ( 0.000E+00) > 1
DIST
< TO ORIGIN ( 42.00 ) > 30
DRAW
Figure 33. bullet model shown after part 1 and 3 has been translated using the EXPLode command. In addition the distance from the observer has been reduced.
RESTORE
CENT
< -3.000 <X< 3.000 -3.000 <Y< 3.000 -6.000 <Z<
<-1.50000E+00 <VECTOR FUNCTION< 3.40000E+00>
<ORIGIN MOVED TO: 0.000 0.000 6.000>
<DISTANCE TO ORIGIN, ANGLE, ZMIN, ZMAX: 42.000 28.00 0.100>

IMMUNE

PART 11/12 IMMUNE TO ROTATION>

2,2

ROTA
<AXIS, ANGLE> X,90

PART

<ENTER PARTS 11/12 TO BE WITHHELD FROM DISPLAY>

3,3

DRAW
Figure 34. Bullet model shown after part 2 has been set IMMUNE to rotations and part 3 has been withheld from display.
EXPLODE

<PARTS 11/12, LOCAL MOTION (X,Y,Z)>

LOCAL MOTION SCALE FACTOR ( 0.100E+01) 0.0

RESTORE

CENT

-3.000 <X< 3.000 -3.000 <Y< 3.000 -6.000 <Z<

-1.50000E+00 (VECTOR FUNCTION 3.40000E+00)

<ORIGIN MOVED TO: 0.000 0.000 6.000>

<DISTANCE TO ORIGIN, ANGLE, ZMIN, ZMAX: 42.000 28.00 0.100>

ROTA

<AXIS, ANGLE> X,-90

FIELD

<ANGLE, ZMIN, ZMAX (28.00 0.10 84.00)> 10, 1.84

DRAW
Figure 35. Bullet model shown after the rotation - 90 degrees X-axis and the angle of view has been reduced.
RESTORE
CENTER
<-3.000 <X< 3.000  -3.000 <Y< 3.000  -6.000 <Z<
<-1.50000E+00 (VECTOR FUNCTION: 3.40000E+00)
<ORIGIN MOVED TO:  0.000  0.000  6.000>
<DISTANCE TO ORIGIN, ANGLE, ZMIN, ZMAX:  42.000  28.00  0.10
DIST
<TO ORIGIN ( 42.00)> 35
DRAW
Figure 36. Bullet model at a distance of 35 units from the observer.
FIELD
<ANGLE, ZMIN, ZMAX ( 23.00 0.10 84.00) > 15
DIST
<TO ORIGIN ( 35.00) > 60
DRAW
Figure 37. Bullet model shown as the angle of view has been reduced and distance to the observer has been increased. This demonstrates that by reducing the angle of view and increasing the distance the object has the same size and the perspective effect is less. (Compare with figure 35).
> CENT
< -3.000 <X> 3.000 -3.000 <Y> 3.000 -6.000 <Z>
< -1.50000E+00 <VECTOR FUNCTION> 3.40000E+00>
< ORIGIN MOVED TO: 0.000 0.000 6.000>
< DISTANCE TO ORIGIN, ANGLE, ZMIN, ZMAX: 42.000 28.00 0.100
> DIST
< TO ORIGIN ( 42.00)> 20
> SHIFT
< SHIFT WINDOW X,Y ( 0.00 0.00)> 3,3
> VIEW
Figure 38. Bullet model shown after the view window has been shifted.
TRANSLATE FROM (0.00, 0.00) TO (-5.00, -5.00)
Figure 39. Bullet model shown after a translation of the model center.
>> SHRINK
<CHANGE SHRINK FACTOR (0.0 - 0.99) FROM (0.00) TO >0.5
>> DRAW
Figure 40. Bullet model shown with an 0.5 shrinking factor.
\( \text{ROTATE (AXIS ANGLE)} \): \( x, -90 \)

\( \text{DIST TO ORIGIN} \): \( 42.00 \)

\( \text{SCALE (DISPLACEMENT SCALE FACTOR)} \): \( (0.0) > 0.4 \)

\( \text{DRAW} \)
Figure 41. Bullet model shown with an 0.4 displacement scale factor.
>> EXPLDOE
<PARTS I1/I2, LOCAL MOTION (X,Y,Z)>
>>> 1,1,0,0,2
>>> 3,3,0,0,-2

<LOCAL MOTION SCALE FACTOR ( 0.000E+00)> 1
>> PIVOT
<PARTS I1/I2, AXIS, ANGLE>
>>> 1,1,X,180
>>> 2,2,Y,30

<PARTS I1/I2, RELATIVE ORIGIN>
>>> 1,1,0,0,12
>>> 2,2,0,0,4.5

>> ROTA
<AXIS, ANGLE> X,-90
>> 2,45
<Z,45? HELP?>
>> ROTA
<AXIS, ANGLE> Z,45
>> DRAW
Figure 42. Bullet model shown as local rotation has been performed by using the PIVOT command.
Figure 43. Bullet model shown as the first frame of a three frame animation sequence.
Figure 44. Bullet model shown as the second frame of a three frame animation sequence.
Figure 45. Bullet model shown as the third frame of a three frame animation sequence.
LITERATURE CITED


BIBLIOGRAPHY


