NASA Contractor Report 172473

SMP - A SOLID MODELING PROGRAM

Donald P. Randall, Kennie H. Jones,
William H. von Ofenheim, and
Raymond L. Gates

COMPUTER SCIENCES CORPORATION
Hampton, Virginia

Contract NAS1-16078
November 1984

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

Table of Contents

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

The Solid Modeling Program (SMP) provides the capability to model complex solid objects through the composition of "primitive" geometric entities. In addition to the construction of solid models, SMP has extensive facilities for model editing and display. The geometric model produced by the software system can be output in a format compatible with existing analysis programs such as PATRAN-G.

SMP originated as a graphics postprocessor for an advanced spacecraft concepts preliminary design program [1]. Since those early beginnings, SMP has been utilized in a stand alone mode to model proposed space station configurations and large antenna designs [2]. Because the system models objects through the manipulation of basic shapes, it can be employed to support a variety of applications.

A solid model generated by SMP is comprised of a collection of geometric "primitives" (parts). The system provides the designer with a basic set of primitive parts and the capability for defining new primitives. The present version of the SMP software supports five primitives: "boxes", "cones", "spheres", "paraboloids", and "tori". The user defines a primitive part by specifying the dimension and construction attributes required for a given part type. By varying the construction attributes for certain primitives, numerous additional shapes can be represented. For example, a "cylinder" is a special case of a "cone" where both radii are equal. Regularly shaped geometric objects can also be generated through the application of translational or rotational "sweeping". In this case, a two dimensional profile of node points is user supplied, and the modeller "sweeps" out the three dimensional solid object. New primitives can also be generated by operating on existing pairs of primitives with the so-called Boolean ("set") operations of intersection, union, and difference. Finally, primitives can be created external to the SMP software. The only restriction for "external" parts is that the SMP geometry format be rigidly followed. In certain instances, it may be advantageous to group related or frequently used primitives and later reference this group as a single entity. The mechanism for grouping parts in SMP is the "assembly". A representative geometric model illustrating most major part classifications is shown in FIGURE 1. The designer can apply a transformation to any category of part to insure proper orientation. The details for creating each of the major primitive types is presented later in the document.

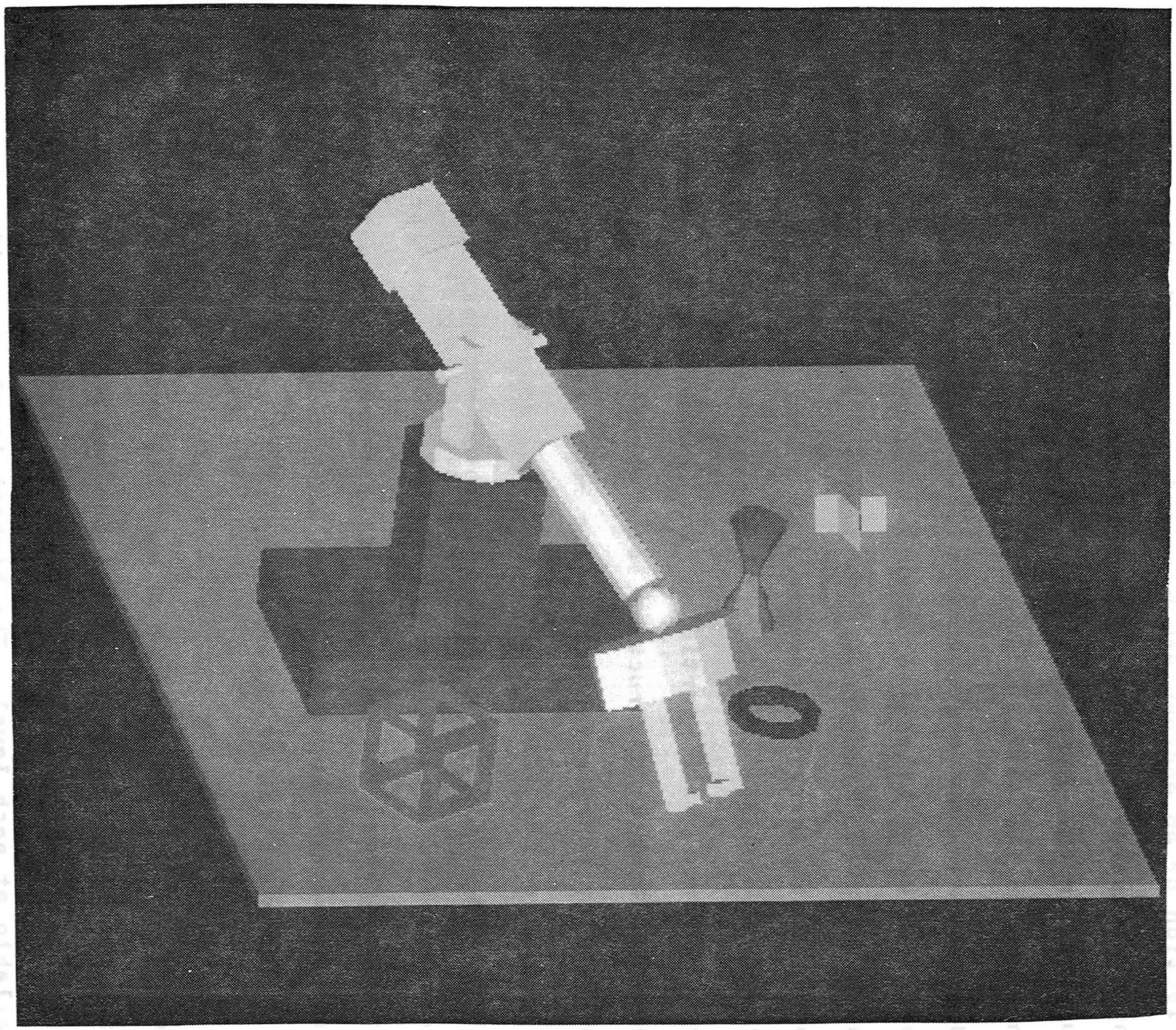


FIGURE 1

ROBOT ARM

One of the major concerns in any solid modeling package is the selection of an appropriate geometric representation scheme. The contemporary schemes for representing solid data are outlined in [3]. SMP employs a dual representation scheme. Internal to the software system, the solid objects are defined by a hybrid boundary representation whereby each part is described by a collection of three dimensional verticies and the associated "connectivity" between the verticies. The boundary is, therefore, comprised of a set of planar facets ("faces"). Unless explicitly requested, the designer need not be concerned with verticies and connectivity. For this reason, a hybrid constructive solid geometry (CSG) scheme is invoked for representing the solid model externally. SMP does not utilize the CSG tree but instead, generates a parts list where each list entry contains the minimal information necessary for reconstructing and orienting the primitive. Both the schemes for geometric representation, together with their respective input and output interfaces, will be detailed in a later section of this report.

The SMP software is structured as a hierarchy with each level being associated with a set of program commands. The system is menu and command driven with an online help facility available at each level. The highest level menu addresses the areas of: reading and writing the solid model geometry, primitive editing, and model display. The menus are never nested more than four deep. If a specific command has options, the user will be prompted as required. Errors will be detected and corrected whenever possible. In some cases, the corrections will require some additional action on the part of the user. In either case, a terminal message will inform the user as to the nature of the error and instruct the user as to an appropriate action. The command levels, sublevels, individual commands, and command options will be described and demonstrated in a subsequent section.

SMP provides a natural approach to creating, manipulating, and displaying solid objects. Because the user unambiguously defines a primitive with only dimension (length, radius,..., etc.) and orientation (rotate, scale, and/or translate) information, user input is kept to a minimum. Through the utilization of externally created primitives, the designer can model objects whose components are irregularly shaped. The intent of this report is to describe the features of the modeller software and to demonstrate their utility.

2. MODELING PRIMITIVES

SMP models solid objects through the construction and orientation of three dimensional (3-D) "primitives". The present version of the modeller software supports five categories of primitives:

(1) basic primitives
(2) swept primitives
(3) Boolean primitives
(4) external primitives
(5) assemblies

The basic primitives include the "box", "cone", "sphere", "parabaloid", and "torus"; swept primitives include 3-D objects generated by either the translational or rotational "sweeping" of a two dimensional (2-D) profile curve; Boolean primitives are the result of the application of one of the "set" operations of union, intersection, or difference to an existing primitive pair; external primitives are 3-D geometric entities created external to SMP; and assemblies provide a mechanism for grouping existing primitives. The intent of this section is to detail the specification and construction of each primitive type from the viewpoint of the user and the internals of the modeling package.

Once a primitive has been created, it must be positioned and oriented in relation to the entire model. SMP supports this requirement by permitting a local transformation to be specified for every primitive. The local transformation takes the form of a rotation, scaling, and translation in each of the x, y, and z directions. SMP employs a standard "right-handed" coordinate system. This nine parameter transformation becomes an attribute of the primitive definition and is carried along with the part throughout its existence. Because the local transformation is retained, the individual transformations are order dependent. The predetermined order is rotation, scaling, and translation on the x, y, and z axes, respectively. This order puts some additional burden on the user, particularly with respect to selecting the proper rotation angles; but a facility for the arbitrary input of rotation angles has been included (see section 4.3.2). If the user chooses not to apply a local transformation, a default identity transformation is supplied. A notable exception to this rule is the initialization of the translation values. For certain primitive types the part may be automatically translated to a position more suitable for local orientation. More information concerning initial positioning will be given in the subsections pertaining to the individual primitives.

Although the specification requirements vary between primitives, two pieces of information are always required. The user can specify an 80 character description for each primitive part (the default is all "blanks"). The user must specify a color value from 1 to 7 for every primitive. The number to color correspondence is as follows:

1 - red
2 - green
3 - yellow
4 - blue
5 - magenta
6 - cyan
7 - white

The color values are ignored on a monochromatic device.

The following subsections describe the specification for every primitive. The reader should note that all primitive attributes can be modified via the "editing" facilities resident within SMP(see section 4).

2.1 BASIC PRIMITIVES

Solid models are very often comprised of basic 3-D geometric shapes. In order to take advantage of this regularity, five basic primitives have been incorporated into the modeling software: "box", "cone", "sphere", "paraboloid", and "torus". These primitives are completely defined through the specification of the appropriate dimension and construction parameters. Although the number of basic primitives is relatively small, altering the construction parameters expands the number of basic shapes. For example, an entire class of conical shapes can be generated by varying the "number of sides" (planar sections). FIGURE 2 exemplifies the utility of the five "basic" primitives.

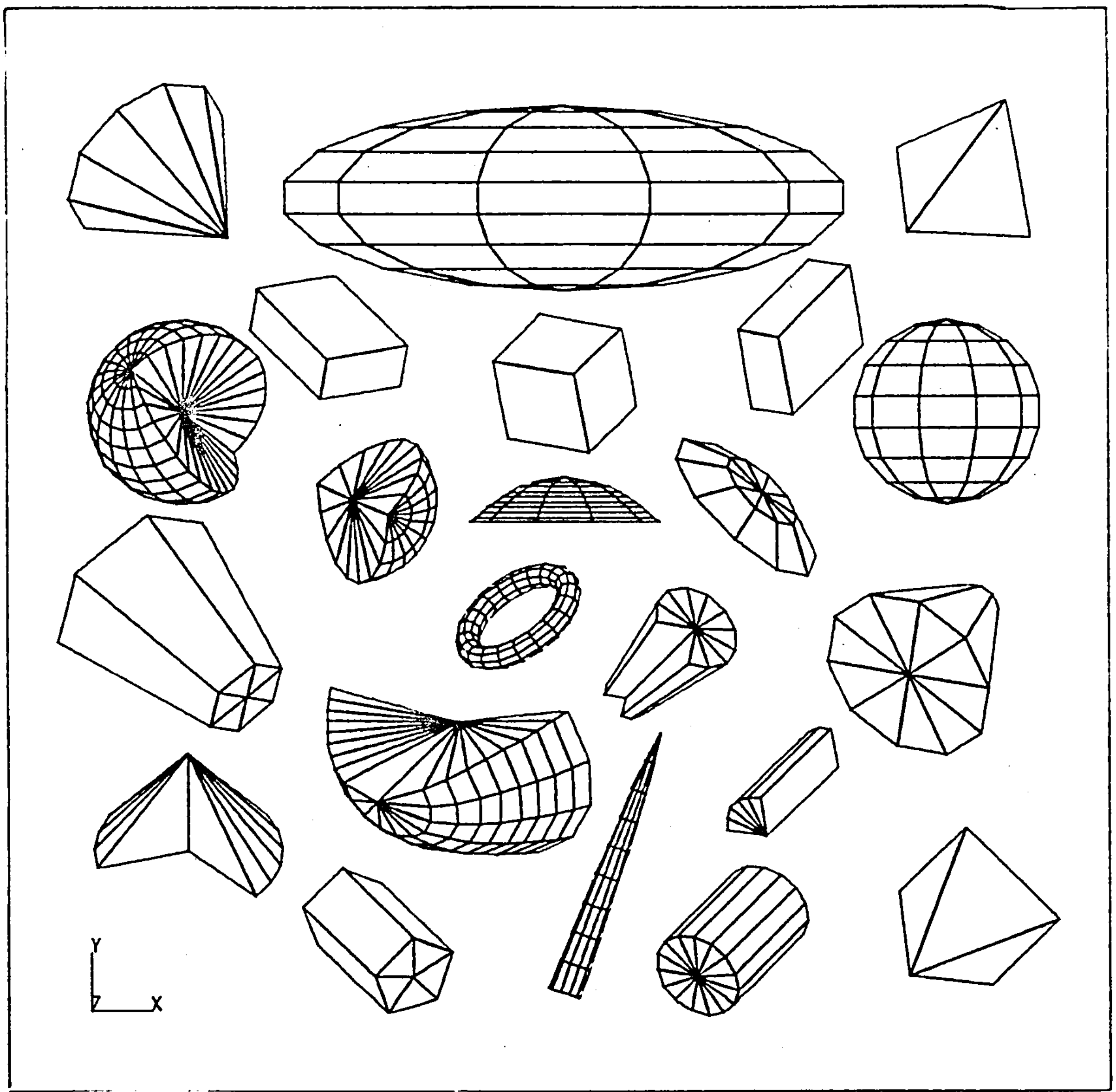


FIGURE 2
GEOMETRIC MODELING PRIMITIVES

Once the designer has selected the desired basic primitive, the SMP software will prompt him/her for the required dimension and construction attributes. Internal error checking is provided to prohibit the user from entering illegal primitive specifications. The following subsections define the specification data required for each basic primitive.

2.1.1 BOX

The "box" is a rectangular hexahedron. The required dimensions are <u>length</u>, <u>width</u>, and <u>height</u> (real values > 0.). Creation of the boundary representation for the "box" is accomplished through the determination of eight verticies such that the specified length is along the x-axis, width is along the z-axis, height is along the y-axis; and the "centroid" of the "box" is at the origin (0,0,0). In addition to the normal uses of a "box", it is useful for simulating other shapes when detail is not significant because of the minimal amount of information required for its boundary representation (see section 3.2).

2.1.2 CONE

The "cone" is a general form used to describe any type of cone, truncated cone, or cylinder. The dimensions of a "cone" are completely defined by supplying two radii of the two circles representing its ends (real values >= 0. where both cannot equal 0.) and the length (real value > 0.). However, the boundary representation for the "cone" is composed of planar subsections.

The "cone" is constructed by revolving (rotationally sweeping) a line segment in a circular path around the x-axis. The endpoints of the line segment are defined by the right most ("positive") and left most ("negative") radii of the circular ends. The slope of the line segment is determined by the relative difference of the two radii. Performing the revolution in discrete increments generates the planar subsections. If an end radius is greater than 0, the end is "capped" by forming triangular subsections emanating from the center of the circle. The resulting "cone" has its "centroid" at the origin with its length parallel to the x-axis.

The user controls the number of increments (and thus the number of planar subsections) by supplying the number of sides for the "cone" (integer value > 2) the boundary representation limitations for a single primitive (see section 3.2) restrict the user's choice to:

(number of sides x 2) + 2 <= 300

Increasing the number of sides tends to "smooth" the "cone's" surfaces but also increases the amount of data in its boundary representation. A reasonable number for the number of sides appears to be 12.

Wedges of a "cone" can be created by varying the revolution angle of the sweep (real value 0 < theta <= 360). An angle of 360 will result in the generation of a complete revolution. Angles less than 360 will result in some relative portion of the "cone" being generated. The interior of the partial "cones" will be constructed by joining the midpoints of the "endcaps".

The "cone" is perhaps the most versatile primitive part. By specifying one radius equal to zero, a true cone is generated. Unequal radii result in a truncated cone and equal radii produce a cylinder. By controlling the radii and the number of sides, special geometric structures such as tetrahedra and pyramids may be constructed. FIGURE 2 illustrates several applications of the "cone" primitive.

2.1.3 SPHERE

The dimensions of a "sphere" are completely defined by supplying the radius (real value >0.). However, the boundary representation for the "sphere" is composed of planar subsections ("faces"). The "sphere" is constructed by revolving (rotationally sweeping) the circle, of the given radius and centered at the origin, around the y-axis. The planar subsections are generated by performing the revolution in discrete increments. The user controls the number of increments (and thus the number of planar subsections) by supplying the number of latitude and longitude lines (integers > 2) sectioning the "sphere". The boundary representation limitations for a single primitive (see section 3.2) restrict the user's choice to:

(number latitudes - 2) x (number longitudes) + 2 <= 300

Increasing the number of latitudes and/or longitudes tends to "smooth" the "sphere's" surfaces but also increases the amount of data in its boundary representation. A reasonable number for both parameters appears to be 10.

Wedges of a "sphere" can be created by varying the revolution angle of the sweep (real value 0. < theta <= 360). An angle of 360 will result in the generation of a complete "sphere". Angles less than 360 will result in some relative portion of the "sphere" being generated. The interior of the partial "spheres" will be constructed from triangular subsections emanating from the "sphere's" center.

The "sphere" is initially positioned at its center which is the origin.

Ellipsoids can be created by supplying nonequivalent scale factors in the local transformation.

2.1.4 PARABOLOID

The dimensions of a "paraboloid" are completely defined by providing the parameters for the equation of a parabola y**2=4px where x is the length of the parabola and p is the distance from the vertex to the focus. However, the boundary representation for the "paraboloid" is composed of planar subsections ("faces"). Analagous to the "cone" and "sphere", the paraboloid is constructed by revolving (rotationally sweeping) the specified parabola around the x-axis. The planar subsections are generated by performing the revolution in discrete increments.

The user controls the number of increments (and thus the number of planar subsections) by supplying the number of latitude and longitude lines (integer values > 2) sectioning the "paraboloid." The boundary limitations for a single primitive (see section 3.2) restricts the users choice to:

(number latitudes - 1) x (number longitudes) + 1 <= 300

Increasing the number of latitudes and/or longitudes tends to "smooth" the "paraboloid's" surfaces but also increases the amount of data in its boundary representation. A reasonable number for both parameters appears to be 10.

Wedges of a "paraboloid" can be created by varying the revolution angle of the sweep (real value 0. < theta <= 360.). An angle of 360 will result in the generation of a complete "paraboloid." Angles less than 360 will result in some relative portion of the "paraboloid" being generated. The interior of the partial "paraboloids" will be constructed from triangular subsections emanating from the "paraboloid's" center.

Elliptical "paraboloids" may be generated by supplying nonequivalent y and z scale factors.

2.1.5 TORUS

The dimensions of a "torus" are completely described by supplying the inner radius and the outer radius. However, the boundary representation for the torus is composed of planar subsection ("faces"). For construction of the "torus", a circle (in the xy-plane) with diameter

$$D = \text{outer radius} - \text{inner radius}$$

and center

$$(-(\text{inner radius} + D/2), 0, 0)$$

is determined. This circle is then revolved (rotationally swept) around the y-axis to form the torus. The planar subsections are generated by the revolution in discrete increments. The resulting torus has its centroid at the origin and its "hole" perpendicular to the y-axis.

The user controls the number of increments (and thus the number of planar subsections) by supplying a number of "sections" (for the revolution) and a number of "sides" (for the circle). The boundary representation limitations for a single primitive (see section 3.2) restrict the user's choice to:

$$(\text{number of sides}) \times (\text{number of sections}) \le 300$$

Increasing the number of sides and/or sections tends to "smooth" the "torus'" surfaces but also increases the amount of data in its boundary representation. A reasonable number for both parameters appears to be 12.

The start angle determines the location of the first increment for the initial circle. This parameter has little effect on "tori" with many sides but does affect the appearance of "tori" with number of sides less than 6. The usual value for the start angle is 0.

2.2 SWEPT PART

A swept part is the 3-D object (solid, shell, or surface) defined by moving ("sweeping") a 2-D profile along a linear or circular path. SMP supports two types of swept parts: translational (linear path) and rotational (circular path); twisted translational and spiral swept parts are not included.

The input for swept parts deviates significantly from the attribute values used in the "basic" primitives (see section 2.1) since the user must explicitly specify the (X,Y) coordinates that define the 2-D profile curve. The user will be notified as to the maximum number of points allowed; this number may be dependent on previously defined part attributes. A minimum of two points is required to generate a swept part. Curves are defined in the plane Z=0 and must be defined in a clockwise order to insure consistency in the boundary representation (see section 3.2). To define a "closed" curve or a loop, the last (X,Y) point must be coincident with the first (X,Y) point entered. Care should be taken when creating an "open" swept curve since the resultant geometric shape is essentially "hollow" (i.e. not solid).

The input for the swept parts allows the user to "cap" the open ends of the part. A flag is used to signal the selection of "end capping": 1 for "end caps" or a 2 for no "end caps". There is no general triangularization algorithm to implicitly construct the "end caps". Consequently, if "end caps" are selected, the user is requested to enter the (X,Y) coordinates of a <u>center point</u>. This feature should not be used if there does not exist a point within the interior of the closed curve such that straight lines can be drawn from each (input) point on the curve to the interior point without any of the lines intersecting the curve itself. The straight lines define the edges of triangular subsections which together form the "end cap".

The "centroid" for a swept part is the center of the smallest bounding rectangular hexahedron (with sides parallel to the coordinate axis) containing the part. The "centroid" value replaces the translation values in the transformation matrix when the part is initially created.

2.2.1 TRANSLATIONALLY SWEPT PARTS

The path of the translationally swept part is defined by entering the (X,Y,Z) coordinates of the new origin. The effect of this new origin (X,Y,Z) is that each coordinate (x,y,0) on the profile curve defined by the user will be "connected" along a linear path to a new point (x+X, y+Y, Z). One restriction is placed on the new origin which is that the Z value may not equal zero since the resulting part would have no "thickness". Currently, the maximum number of points for the user-defined curve is 30.

2.2.2 ROTATIONALLY SWEPT PARTS

Input for the rotationally swept part includes the number of sides (integer value >= 1) and the revolution angle (0<theta<=360). The profile curve is rotated counter-clockwise around the positive X axis in increments corresponding to the number of sides. For best results the curve should have positive Y values and only the end points of an "open" curve or at most one point on a "closed" curve should have Y values equal to zero. "End capping" is only recognized on wedges (i.e. revolution angle less than 360). The maximum number of points for the user-defined curve is inversely proportional to the number of sides, so if the number of side is decreased then more points may be entered. The calculation of the maximum number of points is as follows:

```
Maximum Number of Points = Minimum(30,
                                  300/(Number Sides + 1),
                                  600/(Number Sides + 2))
```

2.3 BOOLEAN PRIMITIVES

Boolean primitives are created by "joining" an existing pair of primitives through one of the "set" (or Euler) operations of: union, intersection, or difference. Several prototype and commercially available solid modeling packages use the Boolean operators as the fundamental mechanism for generating objects [3-4]. The algorithm used for combining primitives with one of the "set" operations is dependent on the internal representation of the geometric data. Because SMP uses a hybrid boundary representation scheme (see section 3.2), it is the boundary representations for each primitive that are computationally "intersected". The algorithm used within the modeller is based on the work of Maruyama [5], Parent[6], and Carlson[7].

The input required for creating a Boolean primitive consists solely of specifying the operator and the operands. The operator may be any one of the three basic "set" operations of:

intersection
union
difference

The operands are specified one at a time and must be selected from among the "active" (see section 4.3.5) primitives which currently comprise the user's model. All primitives with the exception of <u>assemblies</u> (see section 2.5) are legal operands. Existing Boolean primitives can be used as operands for the purpose of creating new primitives.

Reversing the order of the operands usually results in the creation of a different primitive. For the cases of union and intersection, reversing the operands generates primitives of similar shapes but different boundary representations. However, reversing the operand order for the difference operator creates uniquely different shaped objects.

An example of the four basic set operations is shown in <u>FIGURE 3</u> in both hidden line 3(a) and shaded image 3(b) renderings. The "basic" primitives (see section 2.1) of the "box" and "cone" are used to show the penetration of one primitive by another. The following operations are shown:

```
  box (intersect)  cone     (upper left)
  box (difference) cone     (upper right)
  box (union)      cone     (lower left)
 cone (difference) box      (lower right)
```

The extra construction lines on the "faces" of the "box" are due to the restrictions of the data representation scheme (see section 3.2). Specifically, a "face" can be comprised of no more than 4 "verticies".

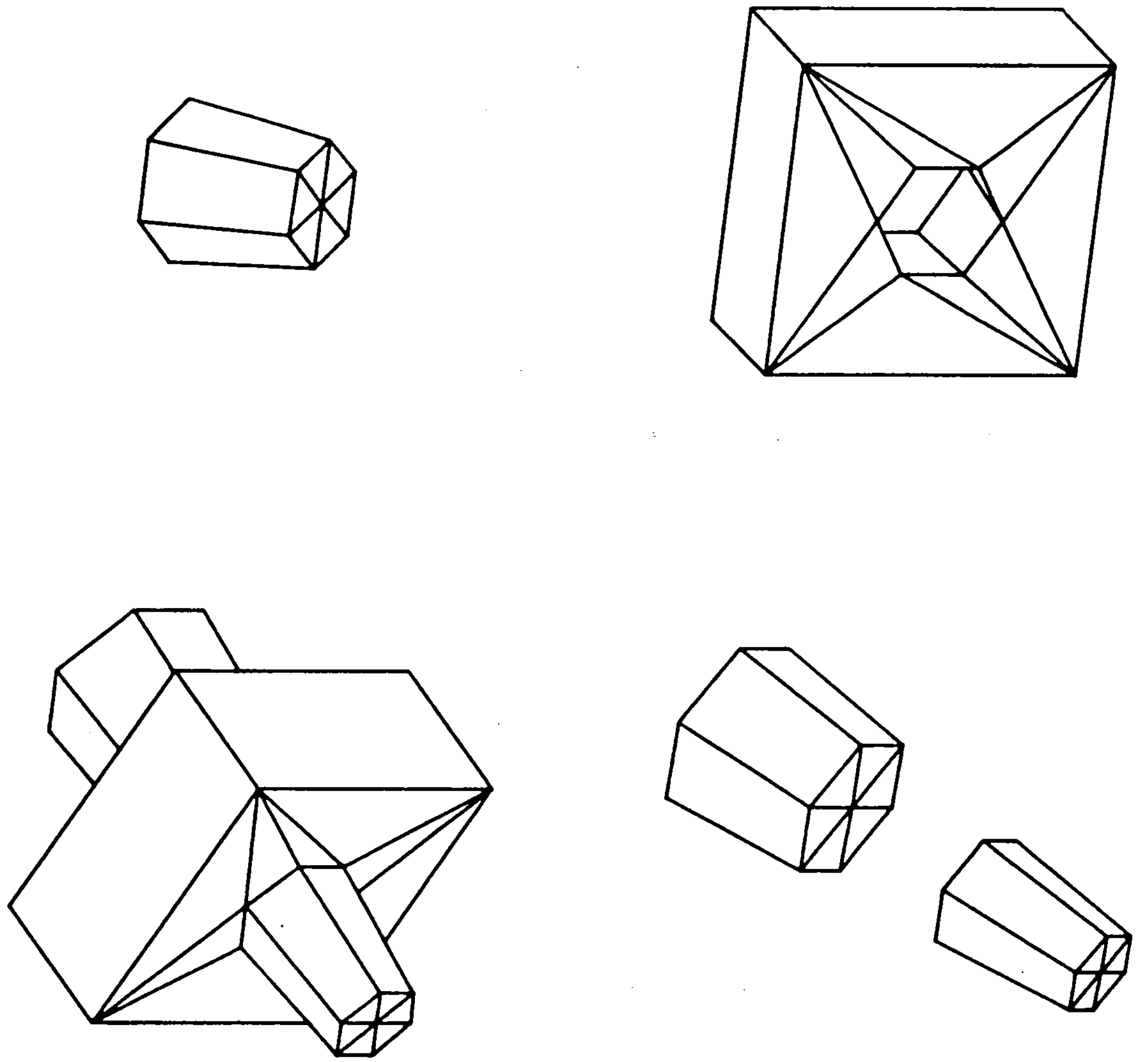


FIGURE 3a

BOOLEAN OPERATION - HIDDEN LINE

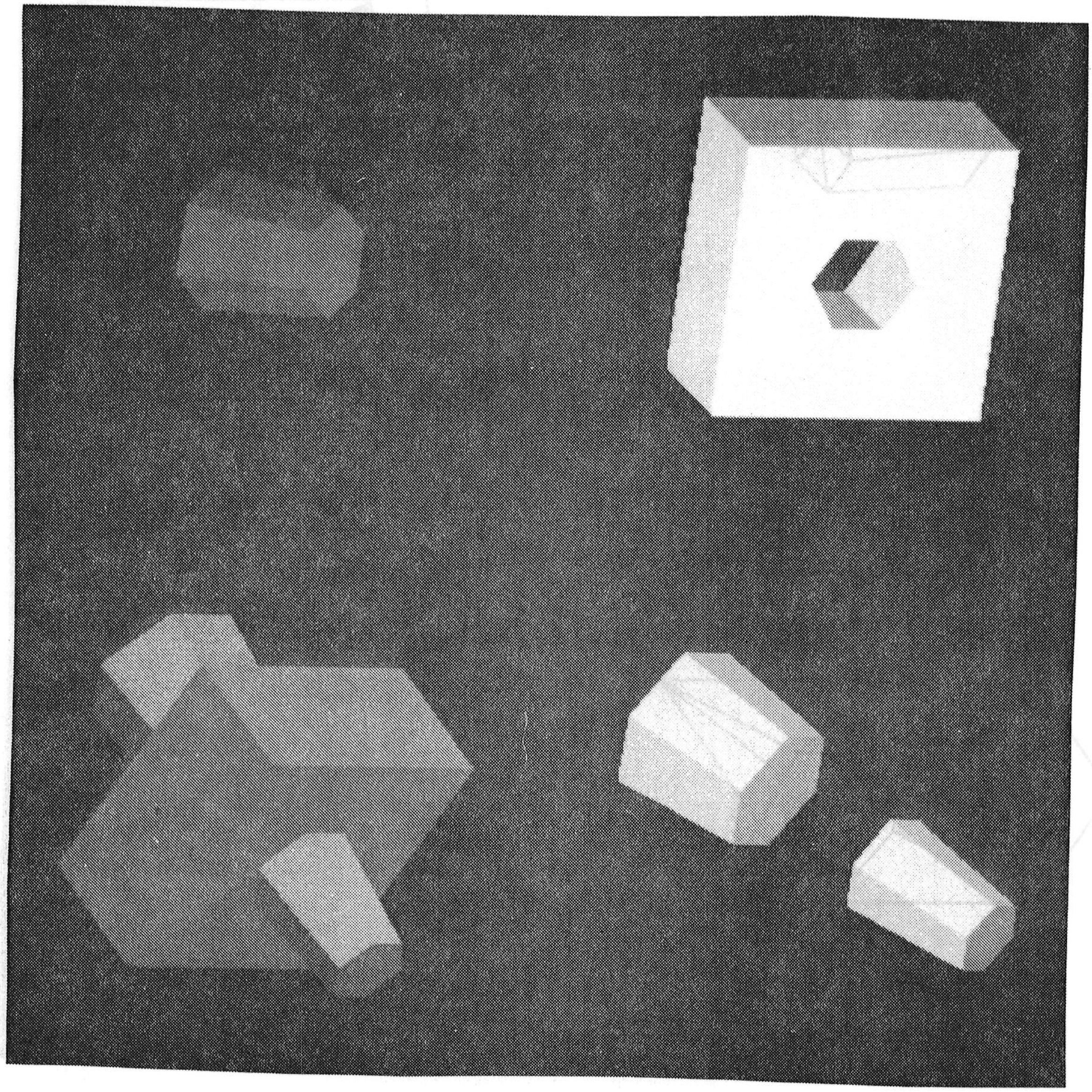


FIGURE 3b
BOOLEAN OPERATION - SHADED IMAGE

If the operands do not geometrically "intersect", a new primitive cannot be constructed. If this occurs, an error message is printed to the user's terminal and all references to this new primitive are implicitly removed from the model.

The algorithm in the current version of the modeller software, may encounter computational difficulties when the operand primitives share common "faces", "edges", or "verticies". If the coincident boundary problems cannot be resolved, the error condition is handled in a manner analogous to the non-intersecting primitives.

The Boolean primitive is initially positioned according to the "centroid" of its <u>bounding box</u>; where the "bounding box" is the minimal rectangular hexahedron that contains the primitive. Consequently, the translation parameters in the local transformation matrix are initially set to the coordinates of this centroid. That is, a translation from the "origin" (0,0,0) to the centroid of the bounding box is implicitly performed.

The "intersection" operator could be employed in "static" <u>interference</u> checking where the resultant object shows the extent of the interference.

2.4 EXTERNALS

Solid objects may be composed of irregularly shaped components that cannot be created through "sweeping" (see section 2.2) or repeated applications of the Boolean operators (see section 2.3). In order to accommodate models of this type, an external interface (in the form of an external primitive) is available. The geometry for the irregular primitive is constructed outside of SMP and later used as modeller input.

External parts may be created by an applications program or by a system editor provided they strictly adhere to the format of an SMP geometry file and do not exceed the limitations for a single primitive (see section 3.2).

An external part is exceptional in that it is the only primitive part that is not initially created using the "parts editor" within the modeller (see section 4.3). One (or more) external parts can only be created by reading their respective boundary representations from a formatted "geometry" file (see section 3.2). The initial boundary representation is then stored for future reference. The "centroid" of the external part is defined as the center of the smallest bounding box that contains the complete part. The translation parameters of the local transformation matrix are initialized to the coordinates of this point. The color is initialized to 7 (white).

After these steps are completed, the external part may be modified using the editors. Modifications, however, are limited to changes in the color, description, and/or local transformation matrix. Should the transformation matrix be changed, the modified matrix will be applied to the initial boundary representation that was stored upon creation. In this way, transformation values are not accumulated.

2.5 ASSEMBLIES

Many solid modeling packages offer a facility for conveniently "grouping" primitives [8]. In SMP the grouping mechanism is called an assembly. Assemblies allow the designer to reference a collection of primitives as a single geometric entity.

The assembly construct is valuable if a collection of primitives is to be replicated one or more times within a model. In conjunction with grouping frequently used primitives, the assembly eases the designer's burden with respect to positioning and orientation. Without the assembly construct, each primitive must be locally transformed in order to achieve proper orientation within the geometric model. With assemblies, the primitive group can be constructed in a convenient location (e.g. the "origin"), and then transformed to the desired orientation with a single local transformation applied to the entire entity.

Assemblies are specified by supplying a list of currently "active" primitives. The present version permits a maximum of 25 primitives (components) to comprise a single assembly. Because assembly components can be selected from the list of all "active" model primitives (including other assemblies), the above limitation is not too severe. That is, the use of "nested" assemblies is permitted. The other restrictions on assembly definitions are as follows: an assembly must contain at least one component, and an assembly definition must not be "circular." A "circular" assembly definition is one in which an assembly is either directly or indirectly a component of itself.

Assemblies are initially positioned according to the "centroid" of the first component. Therefore, although component order is generally not significant, the user should carefully select the first component. The discussions of the other primitives identify the "centroid" for each primitive type. If the first component happens to be an assembly, then the "centroid" of its first component is utilized. The translation parameters of the local transformation matrix are initialized to the coordinates of the "centroid", implying an initial translation from the origin to the "centroid".

The designation of a single assembly component to initially position the entire assembly is founded on the previous experience the model designer has acquired in transforming primitives. The selection of the first component is arbitrary but appears to be convenient. The "robot arm" model in FIGURE 1 makes extensive use of the assemblies construct, including the application of nested assemblies. The reader should note that there are at least three local "pivot points" in the model corresponding to movements of the "arm". In each case, assemblies were utilized to depict these areas, and the first component of each assembly represents the "pivot" for each "arm" motion.

If for reasons of necessity or convenience, the user is not satisfied with the "centroid" of any component within the assembly dictating the original transformation, he/she can create an arbitrarily small primitive (e.g. a "box"), position this primitive in the desired location, and force this primitive to be the first component in the assembly. In this manner, the model designer has total control over the initial orientation of assemblies and/or the creation of local "pivots". Because this primitive is arbitrarily small (in relation to the rest of the model), it should have little or no effect on the geometric model as a whole.

The color specified for an assembly will override the color specified for any of its components. That is, all components of the assembly will take on the color of the assembly. If it is desirable to retain the individual component colors, a value of "0" should be entered for the assembly color.

3. GEOMETRIC REPRESENTATION

Solid modeling packages are often distinguished by the manner in which the geometry defining the 3-D object is represented. Because of the importance in selecting the appropriate representation scheme, numerous pure and hybrid schemes have evolved [3]. SMP employs dual geometric representation schemes: a hybrid boundary representation is utilized to manipulate the geometry within the modeller, and a hybrid CSG (constructive solid geometry) scheme serves as the primary user interface.

The motivation for two distinct representation schemes is illustrated in the following example using the "box" (see section 2.1.1) as a typical primitive. If the geometry of the "box" is required for some further computations (e.g. a hidden surface determination), the actual verticies and "connectivity" defining the "edges" and "faces" of the "box" must be made available. For this type of geometry access a boundary representation scheme is ideal. However, the model designer is not necessarily interested in the 8 verticies, 24 edges, and 6 faces required to represent the boundary of the "box". He/she can unambigously represent a "box" by applying the three parameters: width, height, and length. A form of a CSG representation is more suited for this type of geometric reference. Since both types of applications are justifiable, SMP utilizes both representations.

The geometric representation scheme accessible to the user is implemented via the so-called parts-file. The parts-file is a part data base, and contains the information for all "active" primitives required to reconstruct the associated boundary representations. The parts-file includes entries for: part number, part name, part description, dimensions, construction specifications, color, and local transformation. In the case of the composite primitive parts, Booleans (see section 2.3) and assemblies (see section 2.5), information listing the components replaces dimension and construction specifications.

The boundary representation is contained in the so-called geometry-file. Although a formatted geometry-file is available to the user as an input/output option (see section 4.1.2), the geometry-file is a direct access file used for storing the verticies and connectivity for all primitives and their respective components.

The "external" primitive (see section 2.4) is an exception with respect to its geometric representation. Because the "external" part is defined in terms of a boundary representation, this primitive is specified in terms of its boundary representation on both files.

In the normal mode of operation, there is a one-to-one correspondence between the geometric representation for primitives within the parts-file and the geometry-file. The user has direct control over the contents of the parts-file through the modification of primitive part attributes. Such modifications are reflected in the geometry-file through the reconstruction of the boundary representation for the primitive, thus giving the user indirect contol over this file.

The next two subsections will describe the specifics of the two geometric representation schemes.

3.1 PARTS FILE

The user defines a primitive by supplying the minimum amount of information required to construct a boundary representation of the geometric shape and orient it spatially. For each primitive part, this information is entered into a data base, the so-called parts-file. Because a primitive can be unambiguously represented by relatively few attributes, the parts-file information is memory resident during program execution. The actual parts-file is accessed only during the "read" and "write" operations (see sections 4.1 and 4.2). The compactness of the parts-file makes it the ideal mechanism for saving the model during the various stages of creation.

A parts-file comprised of a representative primitive from each of the major categories is shown in FIGURE 4. Examination of this figure illustrates that the layout of the parts-file is primitive dependent. However, the first three records are common to all part types. Regardless of the primitive the first three records are: part identification number, the part name (BOX, CONE, SPHERE, PARABOLOID, TORUS, ROTA-SWEEP, TRANS-SWEEP, BOOLEAN, EXTERNAL, or ASSEMBLY), and part description. The final record for each part, except swept and external parts, is also common to all primitive types. The nine attributes necessary to build the local transformation matrix reside in this record. The transformation parameters are ordered as follows: x, y, and z <u>rotation angles</u>; x, y, and z <u>scale factors</u>; and the x, y, and z <u>translation values</u>. The differences between the parts-file descriptions for the various primitive types are detailed in the subsequent subsection.

It should be noted that the first record of the parts-file identifies the <u>number of parts</u> to be described.

The user has access to the information in the parts-file via the PRINT command within the EDIT command sublevel (see section 4.3.5).

```
   10
    1
BOX
THIS IS A CUBE
  1.00000E+00  1 50000E+00  1 50000E+00  1.50000E+00  0 00000E-01  0.00000E-01  0.00000E-01  0.00000E-01
  0 00000E-01  0 00000E-01  0 00000E-01  1.00000E+00  1.00000E+00  1.00000E+00 -1.00000E+00  0.00000E-01  0.00000E-01
    2
SPHERE
THIS IS A HEMI-SPHERE
  2 00000E+00  1.10000E+00  1.10000E+01  1.00000E+01  1.80000E+02  0.00000E-01  0.00000E-01  0.00000E-01
  4 50000E+01  4.50000E+01  4.50000E+01  2.00000E+00  3.00000E+00  4.00000E+00  0.00000E-01  0.00000E-01  0.00000E-01
    3
CONE
THIS IS A TRUNCATED CONE
  3 00000E+00  2.00000E+00  5 00000E-01  2.00000E+00  3.60000E+02  4.00000E+00  0.00000E-01  0.00000E-01
  0 00000E-01  0 00000E-01  0.00000E-01  1.00000E+00  1.00000E+00  1.00000E+00  0.00000E-01  0.00000E-01  0.00000E-01
    4
PARABOLOID
THIS IS A PARABOLOID
  4 00000E+00  1.00000E+01  1.50000E+00  1.10000E+01  1.00000E+01  3.60000E+02  0.00000E-01  0.00000E-01
  0.00000E-01  2.70000E+02  0.00000E-01  1.00000E+00  1.00000E+00  1.00000E+00  1.00000E+01  0.00000E-01 -1.00000E+01
    5
TORUS
THIS IS A TORUS
  5 00000E+00  1.50000E+00  1.75000E+00  1.00000E+01  1.00000E+01  0.00000E-01  0.00000E-01  0.00000E-01
  9.00000E+01  0.00000E-01  9.00000E+01  1.00000E+00  1.00000E+00  1.00000E+00  0.00000E-01  1.00000E+02  0.00000E-01
    6
TRAN-SWEEP
THIS IS A SWEPT STAR
  6 00000E+00  5 00000E+00  6.00000E+00  7.00000E+00  1.00000E+00  0.00000E-01  0.00000E-01  0.00000E-01
  0.00000E-01  0 00000E-01  0.00000E-01  1.00000E+00  1.00000E+00  1.00000E+00  2.50000E+00  3.00000E+00  3.50000E+00
    9
 -1 00000E+00  0.00000E-01  0.00000E-01 -2.50000E-01  2.50000E-01  0.00000E-01
  0 00000E-01  1.00000E+00  0.00000E-01  2.50000E-01  2.50000E-01  0.00000E-01
  1 00000E+00  0 00000E-01  0 00000E-01  2.50000E-01 -2.50000E-01  0.00000E-01
  0 00000E-01 -1.00000E+00  0 00000E-01 -2.50000E-01 -2.50000E-01  0.00000E-01
 -1 00000E+00  0 00000E-01  0 00000E-01
    7
ROTA-SWEEP
THIS IS A SWEPT TORUS
  7 00000E+00  1.00000E+01  3 60000E+02  2.00000E+00  0 00000E-01  0.00000E-01  0.00000E-01  0.00000E-01
  0 00000E-01  0 00000E-01  0 00000E-01  1.00000E+00  1.00000E+00  1.00000E+00  1.50000E+00  0.00000E-01 -8.34465E-07
    5
  1 00000E+00  1.00000E+00  0 00000E-01  1.00000E+00  2.00000E+00  0.00000E-01
  2 00000E+00  2.00000E+00  0.00000E-01  2.00000E+00  1.00000E+00  0.00000E-01
  1 00000E+00  1.00000E+00  0.00000E-01
    8
ASSEMBLY
THIS IS AN ASSEMBLY
  1 00000E+00
    3
     3     4     5
  0 00000E-01  1 80000E+02  0.00000E-01  1.00000E+00  1.00000E+00  1.00000E+00  0.00000E-01  0.00000E-01  0.00000E-01
    9
BOOLEAN
THIS A BOOLEAN
  2 00000E+00
UNION
     1     3
  0 00000E-01  0 00000E-01  0 00000E-01  1.00000E+00  1.00000E+00  1.00000E+00 -3.75000E-01 -1.19209E-07 -2.38419E-07
   10
EXTERNAL

  7 00000E+00
  4 50000E+01  9.00000E+01  1.35000E+02  5.00000E+00  1.00000E+00  1.00000E+00 -1.00000E+00  0.00000E-01  0.00000E-01
    8     6
 -1.75000E+00 -7.50000E-01  7.50000E-01 -2.50000E-01 -7.50000E-01  7.50000E-01
 -2.50000E-01  7.50000E-01  7.50000E-01 -1.75000E+00  7.50000E-01  7.50000E-01
 -1 75000E+00 -7.50000E-01 -7.50000E-01 -2.50000E-01 -7.50000E-01 -7.50000E-01
 -2.50000E-01  7.50000E-01 -7.50000E-01 -1.75000E+00  7.50000E-01 -7.50000E-01
    4     1     2     3     4
    4     2     6     7     3
    4     5     8     7     6
    4     1     4     8     5
    4     4     3     7     8
    4     1     5     6     2
```

FIGURE 4
PARTS-FILE

3.1.1 BASIC PRIMITIVES PARTS-FILE DESCRIPTION

The basic primitives (see section 2.1) are determined by the appropriate dimension and construction attributes. These attributes are specified in an eight parameter entry which comprises the fourth record of the parts-file for all basic primitives.

TABLE 1 depicts the correspondence between the primitive attributes and the eight parameters for each basic primitive. Note that none of the basic primitives require more than the first six entries.

TABLE 1

BASIC PRIMITIVE ATTRIBUTES

PARAMETER / PRIMITIVE	1	2	3	4	5	6
BOX	color	length	height	width	–	–
CONE	color	positive radius	negative radius	length	revolution angle	number sides
SPHERE	color	radius	number latitudes	number longitudes	revolution angle	–
PARABOLOID	color	length	width	number latitudes	number longitudes	revolution angle
TORUS	color	inner radius	outer radius	number sections	number sides	start angle

3.1.2 SWEPT PART PARTS-FILE DESCRIPTION

The swept parts (see section 2.2) are determined by the appropriate construction attributes and, following the transformation parameters entry, the data defining the 2-D curve to be swept.

The construction attributes are specified in an eight parameter entry which comprises the fourth record of the parts-file. TABLE 2 depicts the correspondence between the attributes and the eight parameters for each swept part. Note that neither of the swept parts require all eight parameter entries.

The data defining the 2-D curve to be swept consists of an entry for the number of vertices entered by the user and the entries containing the coordinates of the vertices. The coordinate entries contain the X, Y, and Z coordinates for each vertex with two vertices per entry. If there is an odd number of vertices then the last entry will contain the X, Y, and Z coordinates of one vertex. The Z coordinate will always be zero because the user defined a 2-D curve in the XY-plane. The Z coordinate was retained in order to maintain consistency with the format of the external part (see section 2.5) and the geometry-file (see section 3.2).

TABLE 2

BASIC SWEPT PART ATTRIBUTES

PARAMETER PRIMITIVE	1	2	3	4	5	6	7
ROTA-SWEEP	color	number sides	revolution angle	end capping	X end cap	Y end cap	–
TRAN-SWEEP	color	X origin	Y origin	Z origin	end capping	X end cap	Y end cap

3.1.3 BOOLEAN PARTS-FILE DESCRIPTION

The Booleans (see section 2.3) are determined by an entry for <u>color</u>, an entry for the <u>operation</u> to be performed, and an entry containing the part identification numbers of the two <u>operands</u>.

3.1.4 EXTERNAL PARTS-FILE DESCRIPTION

The externals (see section 2.4) are determined by an entry for color and a set of entries, following the transformation parameters entry, containing the parts geometry information (see section 3.2). The geometry information consists of an entry for the number of vertices and the number of faces, a set of entries for the coordinates of the vertices, and a set of entries for the connectivity information.

The coordinate entries contain the X, Y, and Z coordinates for each vertex with two vertices per entry. If there is an odd number of vertices then the last entry will contain the X, Y, and Z coordinates of one vertex.

The connectivity entries contain one entry for each face. The entry consists of the number of vertices which define the face (either 3 or 4) and the four indices which reference the vertex entries. The indices are determined by sequentially numbering the verticies such that the Nth vertex will be referenced by the index number N. This scheme of referencing the vertices by one index number rather than the three coordinates (X,Y,Z), saves space by eliminating the redundancy of explicitly stating the (X,Y,Z) coordinates for every occurrence of the vertex. Note that faces with only three vertices will contain a zero index (i.e. a null reference) in the fourth index position of the connectivity entry.

A comparison of the preceding discussion and TABLE 3 illustrates the close relationship between the external primitive definition and the boundary representation for any primitive.

3.1.5 ASSEMBLY PARTS-FILE DESCRIPTION

The assemblies (see section 2.5) are determined by an entry for color, an entry for the number of components contained in the assembly, and up to three entries for the identification numbers of the components. The component numbers are listed in a ten parameter entry; so for the current maximum of 25 components, three entries would be required.

3.2 GEOMETRY FILE

After user definition of a primitive part, through either the editor "add" command (see section 4.3.1), the "read" command (see section 4.1), or the editor "copy" command (see section 4.3.6), a boundary representation of this part is constructed automatically by SMP. Based on specifications from the part definition (dimension and construction attributes) the boundary representation is generated in a primitive dependent manner (see section 2.0).

The boundary representations for all parts follow certain conventions. Vertices are determined and connected in such a manner as to compose triangular and/or quadrilateral, planar "faces" which represent (as closely as possible) the boundary of the part. In order to avoid ambiguity, the individual faces are formed by connecting the appropriate verticies in a counter-clockwise order (when viewed from "outside" the object). The number of vertices (except for the "box") is controlled through construction attributes supplied by the user. Increasing the number of vertices (and thus the number of faces) will lead to a closer representation of the true boundary for curved surfaces, but will also increase the complexity (and thus the storage requirements) of the representation. For all primitive parts the maximum number of vertices is 300 and the maximum number of faces is 600.

In the case of an external part, these restrictions also apply; but planar faces are not mandatory, and the vertex ordering is not restricted. It should be noted, however, that for many features of SMP (e.g. Booleans, back face cull, or hidden surface removal), the results cannot be guaranteed if all of the above conventions are not obeyed.

Internal to modeller software, the boundary representation is then written in binary format to a direct access version of the geometry-file (where each primitive part corresponds to one logical record). When a part is to be displayed or manipulated in other ways, its boundary representation can be quickly retrieved. By this method, only the geometry for a single part need be memory resident.

When a part is modified (a change in its specifications or local transformation matrix), the boundary representation for the part is reconstructed according to its modified attributes and restored in the geometry file.

If the user requires the geometry of his/her model for reasons of further analysis, the boundary representation is available in a formatted version of the geometry-file. This form of the geometry-file can be generated by invoking the "write" command (see section 4.2). The boundary representation is formatted as shown in TABLE 3.

A sample geometry file entry for a "pyramid" (created with a system editor to be an "external" part) is illustrated in FIGURE 5.

TABLE 3

GEOMETRY FILE DESCRIPTION

RECORD	DESCRIPTION	FORMAT
1	total number of parts on the geometry file	I5
(note: the remaining section is repeated for each part)		
2	total number of vertices and total number of faces	2I5
3 . . . N1 (2+(number of verticies/2))	X, Y, Z coordinates of all vertices with 2 vertices per record. (If the number of vertices is odd, one additional record will be required for the last vertex.)	6(1X, 1P, E12. 5)
N1+1 . . . . . N2((N1+1)+(number of faces))	descriptions of all faces (one face per record). Each face description includes the number of vertices for each face and the vertex indicies forming the face (arranged in a counterclockwise order of connectivity). Faces with only 3 vertices should contain 0 for the fourth vertex.	5I5

```
  1
  5    5
0.00000E-01 -1.00000E+00 -1.00000E+00 -1.00000E+00 -1.00000E+00  0.00000E-01
0.00000E-01 -1.00000E+00  1.00000E+00  1.00000E+00 -1.00000E+00  0.00000E-01
0.00000E-01  1.00000E+00  0.00000E-01
  3    5    1    2    0
  3    5    2    3    0
  3    5    3    4    0
  3    5    4    1    0
  4    1    2    3    4
```

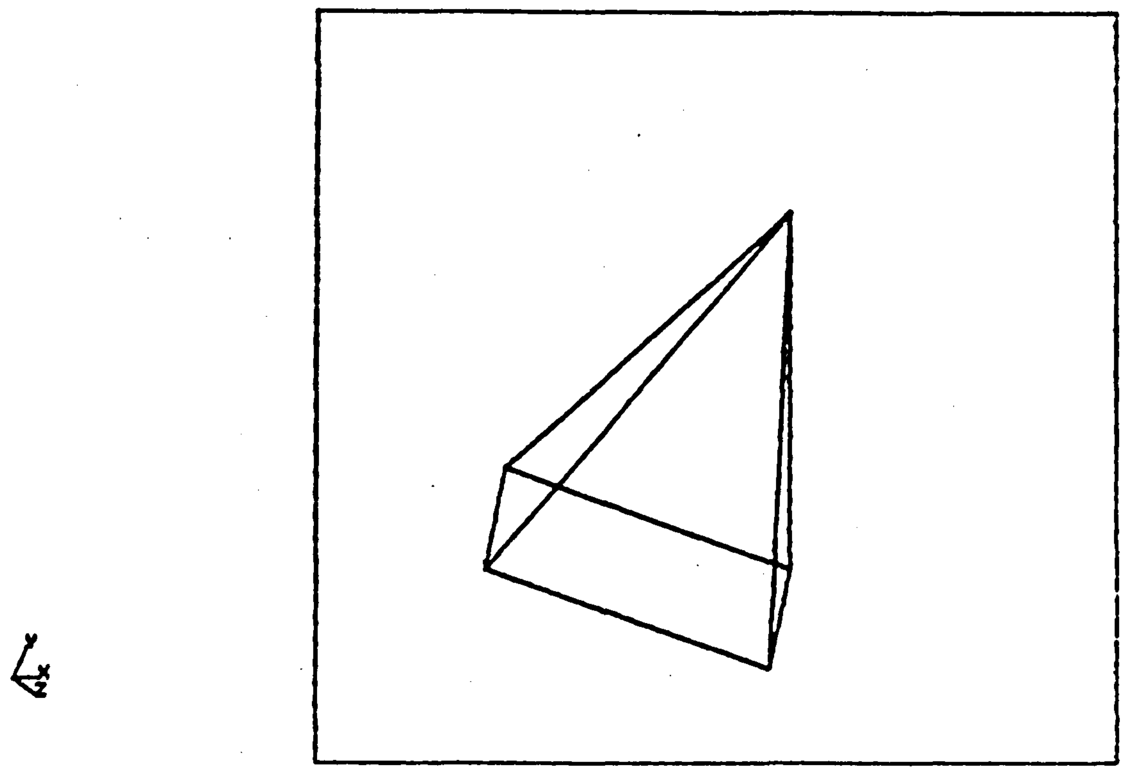


FIGURE 5
GEOMETRY-FILE

4. COMMANDS

SMP provides the designer with a user interface that is both menu and command driven. The user interface can be best understood by examining the hierarchical structure of the modeling system. The individual modeller commands are grouped into levels corresponding to the basic modeling functions.

The "highest" level in SMP is the command level. The available command levels are listed and briefly described below:

READ - input an existing solid model

WRITE - output a solid model

EDIT - modify the solid model by performing basic editing operations on its primitives

DISPLAY- display the solid model

MISCELLANEOUS - perform limited mass properties analysis and dimensioning on the solid model

A command level can be comprised of one or more command sublevels. For example, the EDIT command level has a command sublevel consisting of all available editing operations such as: ADD, MODIFY, COPY, RESTORE, and DELETE. The ADD command sublevel has a second command sublevel menu consisting of the list of available primitives. Once the command sublevels have been exhausted (the level of nesting never goes more than 4 deep), the individual modeling command may require the user to specify <u>subcommands</u> and <u>command options</u>. Continuing with the above example, if the designer selects the Boolean primitive (see section 2.4), he/she must specify the operands (subcommand) and choose from a set of three operators (command option). The relationships between the various levels is illustrated in <u>TABLE</u> 4.

TABLE 4.

SMP COMMAND HIERARCHY

READ (R) EXISTING MODEL

    READ PARTS FILE
        NEW MODEL / ADD TO CURRENT MODEL
        FILE NAME

    READ GEOMETRY FILE
        NEW MODEL / ADD TO CURRENT MODEL
        FILE NAME

    READ MOVIE.BYU GEOMETRY FILE
        NEW MODEL / ADD TO CURRENT MODEL
        FILE NAME

WRITE (W) CURRENT MODEL

    WRITE PARTS FILE
        SELECT PART RANGE
        FILE NAME

    WRITE GEOMETRY FILE
        SELECT PART RANGE
        FILE NAME

    WRITE MOVIE.BYU GEOMETRY FILE
        SELECT PART RANGE
        GEOMETRY FILE NAME
        COLOR COMMAND FILE NAME

    WRITE PATRAN-G NEUTRAL FILE
        SELECT PART RANGE
        FILE NAME

EDIT (E) CURRENT MODEL

    ADD (A) PART

        SUPPLY PART DESCRIPTION

        DESIGNATE PRIMITIVE TYPE
            BOX
            SPHERE

TABLE 4 (cont.)

CONE/CYLINDER
PARABOLOID
TORUS
TRANSLATIONAL SWEEP
ROTATIONAL SWEEP
ASSEMBLY
BOOLEAN

DEFINE ATTRIBUTES (PRIMITIVE DEPENDENT)
DIMENSIONS
CONSTRUCTION PARAMETERS
PROFILE CURVE
COMPONENTS/OPERANDS
BOOLEAN OPERATOR

SPECIFY PART COLOR

MODIFY (M) PART

IDENTIFY PART TO BE MODIFIED

MODIFY OPTIONS
PRINT PART DEFINITION
PRINT PART DEFINITION AND DISPLAY PART
CHANGE PART DESCRIPTION
CHANGE PART SPECIFICATION (PRIMITIVE DEPENDENT)
CHANGE PART TRANSFORMATION
DETERMINE ORDER INDEPENDENT ROTATION ANGLES

DELETE (D) PART

IDENTIFY PART TO BE DELETED
CONFIRM CHOICE

PRINT (P) PART DEFINITIONS

COPY (C) EXISTING PART

IDENTIFY PART TO BE COPIED
CONFIRM CHOICE

RESTORE (R) A DELETED PART

IDENTIFY PART TO BE RESTORED
CONFIRM CHOICE

TABLE 4 (cont.)

DISPLAY (D) THE CURRENT MODEL

RESET (R) VIEWING OPTIONS AND TRANSFORMATION

VIEWING OPTION MENU (M)
- SPECIFY PART RANGE
- VIEWING TRANSFORMATION OPTION
  - DEFINE GLOBAL ROTATIONS
  - DEFINE GLOBAL SCALING
- VIEWING OPTIONS
  - BACK FACE "CULL" OPTION
  - PART LABELING OPTION
  - ELEMENT "SHRINKING" OPTION

DRAW (D) THE CURRENT SELECTED PARTS

DISPLAY FOUR VIEWS (T) OF THE CURRENTLY SELECTED PARTS

ZOOM (Z) ON THE CURRENT VIEW
- REDRAW OPTION
- "PICK" NEW LOWER LEFT CORNER FOR ZOOMED VIEW
- "PICK" NEW UPPER RIGHT CORNER FOR ZOOMED VIEW
- CONFIRM DIMENSIONS FOR ZOOMED VIEW

HIDDEN SURFACE (S)
- HIDDEN SURFACE/LINE OPTION
- IMAGE DISPLAY DEFAULT OVERRIDE OPTION
  - DISPLAY/STORE IMAGE OPTION
  - RESOLUTION CHANGE OPTION
  - SHADING OPTION
    - FLAT
    - SMOOTH
  - SHADING PARAMETER OPTION
    - DIFFUSED LIGHT VALUE
    - REGULAR LIGHT EXPONENT
  - BACK FACE "CULL" OPTION

GRAHICS (G) EDITOR
- PICK A PART (P)
  - SPECIFY XY, YZ, OR ZX-PLANE VIEW
  - PICK PART CENTER (LABEL)
  - (SEE DISPLAY MENU COMMAND)

TABLE 4 (cont.)

PICK AND MODIFY (M) A PART
(SEE GRAPHICS EDITOR PICK A PART COMMAND)
(SEE EDIT MODIFY COMMAND)

PICK AND DELETE (D) A PART
(SEE GRAPHICS EDITOR PICK A PAT COMMAND)
(SEE EDIT DELETE COMMAND)

PICK AND COPY (C) AN EXISTING PART
(SEE GRAPHICS EDITORS PICK A PART COMMAND)
(SEE EDIT COPY COMMAND)

PICK AND PRINT COORDINATES (N) OF A POINT
(SEE GRAPHICS EDITOR PICK A PART COMMAND)
(PICK POINT WHOSE COORDINATES ARE DESIRED

TRANSLATE (T) A PART

TRANSLATION OPTION
TRANSLATE BY PART CENTROID
(SEE GRAPHICS EDITOR PICK A PART COMMAND)
TRANSLATE BY PART VERTEX
(SEE GRAPHICS EDITOR PICK A POINT COMMAND)

PICK NEW/EXISTING TRANSLATION REFERENCE POINT

CONFIRM TRANSLATION SELECTIONS

ROTATE (R) A PART

(SEE GRAPHICS EDITOR PICK A PART COMMAND)
PICK ROTATION REFERENCE POINT
RELOCATE ROTATION REFERENCE POINT
CONFIRM ROTATION SELECTIONS

MISCELLANEOUS (P) COMMANDS

PART DIMENSIONS (D)

DIMENSIONING OPTION
DISTANCE BETWEEN PART CENTROIDS
DISTANCE BETWEEN NODE POINTS
DISTANCES BETWEEN ALL CENTROIDS

(SEE GRAPHICS EDITOR PICK A PART COMMAND)

Although the formal level structure of the SMP commands may appear unnecessarily complex, the procedures for user input remain essentially identical regardless of whether the input refers to a command level, command sublevel, subcommand, or command option. For instance, the appropriate command level and command sublevels are designated by entering the first character of the level name (e.g. "E" for EDIT). If an illegal character is encountered, the user is prompted with a list of legal responses and then permitted to re-enter the selection. In the case of the subcommand and the command option, the user is first prompted for a response. The input for these levels is command dependent; but in all cases, an illegal response will result in the prompt being redisplayed. Input at all SMP levels is "free field".

The program prompts often serve to identify the current command level or subcommand level. For example, the command level is identified by the prompt

ENTER COMMAND>.

The DISPLAY and EDIT command sublevels can be distinguished by the two prompts

ENTER DISPLAY COMMAND>

or

ENTER EDITOR COMMAND>.

There are certain features common to all command levels and command sublevels. These features will be discussed in the following paragraphs.

An on-line "help" facility is available for all command levels. The "help" (HELP) feature permits the user to examine selected sections of this document at his/her interactive terminal during SMP execution. HELP is invoked by entering the character "H" from the command or command sublevel. General information concerning the function of the command level is automatically displayed, and the user can opt for more detailed information regarding the lower levels.

The user can normally traverse back to the next highest level by entering "Q" (for "QUIT"). QUIT works at all command levels, most command sublevels, and at designated lower levels. Entering "Q" from the command level causes an "exit" from the program. If one or more primitives have been modified since the parts-file (see section 3.1) has been last "replaced", the user is prompted regarding the permanence of these modifications.

If the user desires to exit the system without traversing to the upper command levels, the "immediate exit" ("X") option should be used. The "exit" option can be invoked from any command level and most command sublevels. Naturally, the use of the "X" option implies that modifications made to the model since the last "replace" are ignored.

SMP has a built-in interrupt capability which allows the user to stop the execution of a given process and to return to the current menu level. The interrupt is performed by depressing the BREAK key (or <CONTROL> P). After the BREAK key has been struck, the screen will be erased and the current menu level prompt will be issued. This feature is most useful in terminating the drawing of a model that requires corrections, without forcing the user to wait until the display is completed.

Several commands require the user to indicate the primitives to be included in a particular operation. If the model is comprised of a large number of parts, the user may have occasion to view only selected parts as a time saving measure. The selection procedure provides options for the complete model ("*"), part ranges (delimited by a "-"), and individual parts (delimited by a ","). For example, consider the model comprised of 15 primitive parts. The input line:

2,4,6-9,11-13,15

results in parts 2,4,6,7,8,9,11,12,13, and 15 being included in this operation. "Deleted" parts (see section 4.3.3) and "component" parts (see sections 2.3 and 2.5) are exceptions to the above convention. Deleted parts are never included in a specified operation. Component parts may be included or excluded depending on the nature of the operation. If parts 4 and 8 were deleted in the earlier example, the identical input line includes only the parts 2, 6, 7, 9, 11, 12, 13, and 15.

The notion of a "part status" has been alluded to but never explained. A primitive can be either active or inactive. When a part is initially created, its status is "active"; and it remains "active" until one of two conditions exists: the part is explicity or implicitly deleted, or the part becomes a component of a Boolean or assembly. (The rationale behind this second condition is that assembly components and Boolean operands are building tools that can be discarded once the composite part is constructed.) Because the status of a part may effect the result of an SMP command, the user should be alerted as to which commands are impacted by part status. An "inactive" status can be changed to "active" via the RESTORE command within the EDIT command level (see section 4.3.4).

4.1 READ COMMAND LEVEL

The READ (R) command serves as both a means of inputing parts to SMP that have previously been "saved" (see section 4.2) and as an interface to other programs.

Currently, three types of files may be read: a parts-file (see section 3.1), a geometry-file (see section 3.2), or the input geometry from the MOVIE.BYU graphics system [9]. Reading a parts-file is the only means of retaining a part's complete identity; whereas geometry information read from either of the other files is used to create only "external" parts (see section 2.4). The "editing" restrictions applying to "external" parts have been previously discussed (see sections 2.4 and 3.2).

After the file type is specified, a choice is given to either destroy the existing model and make a new model consisting only of the parts contained in the new parts-file, or to append the new parts to the existing model. By employing the append capability new models can be constructed by reading and combining several part-files. This can be convenient for applications where models are composed of similar components.

Finally, a file name must be specified. If the file does not exist or cannot be successfully "opened", the user is informed and asked for another file name. If an error occurs in the attempt to read the file, the current model remains unchanged.

Part numbering ambiguities can result if parts were deleted in an earlier session or if parts-files are being concatenated. One of the functions of the READ operation is to resolve such difficulties by assigning a unique number to each part.

READ also flags each part as "active" or "inactive". All parts are tagged as "active" with the exception of parts which are components of composite parts (i.e. Booleans and assemblies).

4.2 WRITE COMMAND LEVEL

The WRITE (W) command allows a user to save selected parts of the current model at any time during the modeling process. Employing this capability, the user may interrupt a modeling session and continue at a later time, or save designated model parts in separate files for use in other models. The WRITE command also provides a link between SMP and other software systems.

Currently, parts may be saved in four different formats: a parts-file (see section 3.1), a geometry-file (see section 3.2), a MOVIE.BYU [9] geometry input file, or a PATRAN-G [10] "neutral" file.

After specifying the file type, the individual parts to be saved are selected (see section 4.0). For the parts-file, all parts that are explicitly named are saved unless they have been "deleted" (see section 4.3.4). Also, all components of explicitly named composite parts are implicitly saved. Part I.D. numbers remain unchanged on the saved parts-file, but are renumbered consecutively when the file is subsequently "read" (see section 4.1) by SMP. For all three geometry files, explicitly named parts are saved only if they are currently "active" (see section 4.0). (Hence, component parts are not saved unless explicitly "restored" (see section 4.3.4)). In the case of MOVIE.BYU output, the designer can opt to store all components of an "assembly" as a single "MOVIE.BYU part" or to store each assembly component as an individual "part".

Finally, a file name is requested for the saved model. If the file already exists, a choice is given to either overwrite the existing file or input an alternate file name.

For the MOVIE.BYU file option only, a second file may be written containing the current color values for the saved parts. The file is formatted as a MOVIE.BYU "command" file and is used as input for the COLOR command of the MOVIE.BYU DISPLAY program. The use of this file alleviates the need for the user to have to re-enter the part colors specified in SMP within the MOVIE.BYU system. The user is prompted for this second file name. Again, a choice is given to overwrite the file, if it exists, or to specify an alternate name.

4.3 EDIT COMMAND LEVEL

The commands in the EDIT (E) command level permit the designer to either create a new geometric model through the composition of primitives, or to "edit" an existing geometric model through the addition of new primitives or modification of existing primitives. The primitive editing functions include:

ADD
MODIFY
DELETE
RESTORE
PRINT
COPY

The editing commands perform an operation synonymous with the command name. The detailed operation of each of the editing functions is described in a subsequent subsection. The invocation of an EDIT command results in an explicit action being performed on the designated part. However, if the geometric model is comprised of composite parts (either assemblies or Booleans), the editing operation may result in implicit modifications of other composite parts. For example, the explicit deletion of a primitive which is also an assembly component results in the implicit removal of this primitive from the assembly component list. If the composite parts are "nested", an operation on a component can have far reaching effect on the model as a whole. It should be noted that the RESTORE command (see section 4.3.4) prevents operations such as the one described above from being irrevocable.

The preceding discussion presumes the user is working with the parts-file representation (see section 3.1) of the geometric model. In this case, only the geometry-file representation (see section 3.2) of the model is available, some editing functions are illegal. Recall that in a geometry-file representation all parts are treated as externals (see section 2.4). Therefore, editing functions prohibited for external primitives are also prohibited for geometric file entries. More specifically, the user can alter the local transformation parameters, but not the actual part geometry.

4.3.1 ADD

The ADD (A) command sublevel allows the user to create a new part. The added part may be a basic primitive, a swept part, an assembly, or a Boolean part, but not an external part (see section 2.4).

A list of the available primitives is printed; and the user enters the letter which corresponds with the desired primitive, or a "Q" to return to the EDITOR level:

```
ENTER PART NAME  -  BOX(B), SPHERE(S), CONE(C)
 PARABOLOID(P), TORUS(T), TRANSLATIONAL-SWEEP(N)
 ROTATION-SWEEP(R), ASSEMBLY(A), BOOLEAN(E)
 OR  QUIT(Q)
```

After a legal selection, the user is requested to enter the part description (maximum of 80 characters) or a carriage return for no description.

Following the description is a series of requests for the dimension and construction attributes of the selected primitive. These attribute requests are primitive dependent, and the order of these requests is mirrored in the preceding primitive discussions (see sections 2.1-2.3 and 2.5). The final part attribute to be entered is the color code (see section 2.0). If any of the attributes are not within the specified range (again primitive dependent); a warning message is printed specifying the problem, and the corrective action is taken in overcoming the problem.

The local transformation values are defaulted to no rotations, unit scaling, and no translations (except for the non-basic primitive parts which are not centered at the origin (see section 2.0)).

The user is automatically transferred to the MODIFY command sublevel(see section 4.3.2) where attributes can be edited and/or the primitive reoriented.

4.3.2 MODIFY

The MODIFY (M) command sublevel provides a means of changing the attributes of a "non-deleted" (see section 4.3.3) part. The attributes that can be changed include the dimension parameters, construction parameters, orientation parameters, part description, and the color. The MODIFY sublevel can be reached either directly by the MODIFY command, or indirectly by the ADD (see section 4.3.1) or COPY (see section 4.3.6) sublevel commands.

The explicit modification of a component part causes the implicit modification of all composite parts containing this component.

After issuing the MODIFY command, the user is requested to enter the part identification number of the part to be modified or a QUIT (or Q) to return to the EDITOR command level. If an invalid part number is entered (i.e. the part is deleted or non-existent), the request is repeated.

A menu of eight options is displayed next, and the user must enter an integer between 1 and 8 which corresponds with the desired option:

1 PRINT PART DEFINITION

2 PRINT PART DEFINITION AND DISPLAY PART
3 CHANGE PART DESCRIPTION
4 CHANGE PART SPECIFICATION
5 CHANGE PART TRANSFORMATION
6 DETERMINE ORDER INDEPENDENT ROTATION ANGLES
7 CHANGE PART COLOR
8 TO RETURN TO EDITOR (PART OK)

The first two options allow the user to verify that the selected part I.D. number matches the part to be modified and that the most recent modifications are correct by printing the current part definition (see section 4.3.5) or by printing the current part definition and displaying the part (see section 4.4.2). The third option allows the user to enter a new part description (maximum of 80 characters).

The fourth option allows the user to change the dimension and construction parameters of the selected part. The method for changing the parameters varies depending on the part type. Note that the geometry of an external part cannot be modified in this manner (see section 2.4). The part definition is printed to provide a reference for the user.

If the part is a basic primitive, the user is requested to enter an attribute number followed by a new value for each change. The attribute number is the column number which is aligned with the attribute descriptions and values in the printed (see section 4.3.5) part definition. The column number is therefore used as an index for the attribute value.

The attribute number must be between 2 and 9; attribute number 1 is the part identification number which cannot be changed by the user. For instance, if the selected part is a "box", the entry

3,32

changes the height of the box to 32.0 (see TABLE 1). When the user has completed all part attribute changes, he/she terminates by entering 0,0. The changes are then made, and the MODIFY menu is redisplayed.

If the part is a swept primitive then a second submenu is displayed:

1 CHANGE INPUT POINT COORDINATES
2 INSERT A NEW POINT
3 DELETE AN EXISTING POINT
4 CHANGE PART SPECIFICATION
5 RETURN

The first three suboptions allow the user to modify the (X,Y) coordinates that define the 2-D curve (see section 2.2). Each (X,Y) coordinate is assigned an index number; so the user simply enters the index number followed by the new X and Y values or a (0,0,0) to terminate. For the first suboption, the user specified index number must be an existing index number so that the old (X,Y) coordinates can be replaced. For example, if the first suboption is chosen, the entry

3,12.5,15.

changes the coordinates of the third point to

(12.5,15.0).

For the second suboption, the new coordinates will be inserted after the specified index number. For the third suboption, the (X,Y) coordinates are not needed since the coordinates referenced by the index number will be deleted. The fourth suboption is identical to the construction specification modifications used for the basic primitives. The fifth suboption returns directly to the MODIFY menu without making any changes; the first four options return to the MODIFY menu upon normal completion of the operation.

If the part is an assembly, another submenu is displayed:

1 RE-INPUT COMPONENT ARRAY
2 INSERT A NEW COMPONENT
3 DELETE AN EXISTING COMPONENT
4 RETURN

The first suboption allows the user to re-enter the entire component array by entering one component number per line and terminating with a "Q".

For example, if the first suboption is chosen, the entries:

```
9
2
5
Q
```

replace the old component array with the ordered components 9, 2, and 5.

The second suboption either appends the new component to the end of the component array or else "re-activates" the component in its previous location if it had been deleted in a previous session. The third suboption makes an existing component inactive. Care should be taken when modifying the component array since the orientation of the assembly is based on the first component. The second and third suboptions notify the user if the first component is modified and provide a query to allow the user to recover if this is an undesired result. The fourth suboption returns to the MODIFY menu as do the other three options when their actions have been completed.

If the part is a Boolean part, another submenu is displayed:

1 CHANGE OPERATOR
2 CHANGE OPERANDS
3 RETURN

Suboptions one and two provide the same prompts as the ADD sublevel command (see section 4.3.1) and return to the MODIFY menu after completion. The third option returns directly to the MODIFY menu without changing the part.

The fifth option on the MODIFY menu allows the user to modify the orientation parameters. The same indexing technique, as described for the basic primitive attribute modifications, is used to modify the transformation parameters. The nine indices reference the X, Y, and Z rotations, scale factors, and translations. For example, the entries

2, 90
4, 10
9, 5.5
0, 0

change the Y rotation angle to 90.0 degrees, the X scale factor to 10.0, and the Z translation to 5.5.

The sixth option gives the user a means of entering any number of rotation angles and in any order. The result is the equivalent order dependent X, Y, and Z rotations. The user alternates between entering the axis for the rotation and the angle value in degrees. When the user is finished, a "Q" for quit is entered which causes the rotation values to be replaced with the newly calculated order dependent X, Y, and Z rotation angles.

The seventh option allows the user to change the color of the selected part by entering a color code value between 1 and 7.

The eighth option returns to the EDITOR command level when all of the part modifications are complete.

4.3.3 DELETE

The DELETE (D) command allows the user to tag a part as inactive. A "deleted" part cannot be displayed, modified, used as a component, or written to a file unless the part is made active again by issuing the RESTORE command (see section 4.3.4). It should be noted that all "deleted" parts are permanently removed when the user saves (see section 4.2) the geometric model.

After issuing the DELETE command, the user is requested to enter the part identification number of the part to be deleted or a QUIT (or Q) to return to the EDITOR command level. If an invalid part number is entered (i.e. the part is already deleted or non-existent), the request is repeated.

A menu of five options is displayed next, and the user must enter an integer between 1 and 5 which corresponds with the desired option:

1 TO PRINT PART DESCRIPTION
2 TO PRINT PART DESCRIPTION AND DISPLAY PART
3 TO CONFIRM DELETION
4 TO MAKE NEW SELECTON
5 TO RETURN TO THE EDITOR

The first two options allow the user to verify that the selected part I.D. number matches the part to be deleted by printing the part definition (see section 4.3.5) or by _printing_ the part definition (see section 4.3.5) and _displaying_ the part (see section 4.4.2). The third option allows the user to _confirm_ the deletion and return to the EDITOR command level. The fourth option allows the user to _select_ a different part for deletion, and the fifth option _returns_ to the EDITOR command level without deleting the part.

By deleting a part (_explicit deletion_), one or more other parts may also be deleted (_implicit deletion_). The explicit deletion deletes the selected part; and if the part is a component of other composite parts, those components are also implicitly removed. An implicit deletion occurs if the deleted part is the only component of an assembly or one of the operands of a Boolean part. The user is notified when an implicit deletion occurs so that corrective action (see section 4.3.4) could be taken if the implicit deletion was accidental.

If the _first component_ of an assembly is deleted by either an explicit or an implicit part deletion, a warning message and a query will be issued at the first occurrence of this condition. The user can recover by entering "N" to ignore the deletion and return to the editor. By entering "Y", the deletion of the first component of this assembly and all subsequent assemblies in which this component is the first component will be performed. The deletion of the first component of an assembly warrants this level of caution since the first component affects the positioning and orientation of the assembly (see section 2.5).

4.3.4 RESTORE

The RESTORE (R) sublevel command allows the user to re-activate an inactive part. By using the RESTORE command, a part which was deleted in a given session can be re-established as an available part during the same session, and a component part can be returned to active part status.

After issuing the RESTORE command, the user is requested to enter the part identification number of the part to be restored or a QUIT (Q) to return to the EDITOR command level. If an invalid part number is entered (i.e. the part is already active or non-existent), the request is repeated.

A menu of five options is displayed next, and the user must enter an integer between 1 and 5 which corresponds to the desired option:

1 TO PRINT PART DESCRIPTION
2 TO PRINT PART DESCRIPTION AND DISPLAY PART
3 TO CONFIRM RESTORATION
4 TO MAKE NEW SELECTION
5 TO RETURN TO THE EDITOR

The first two options allow the user to verify that the selected part I.D. number matches the part to be restored by printing the part definition (see section 4.3.5) or by printing the part definition and displaying the part (see section 4.4.2). The third option allows the user to confirm the restoration and return to the EDITOR command level. The fourth option allows the user to select a different part for restoration, and the fifth option returns to the EDITOR command level without restoring the part.

The restoration of a non-deleted part which is a component of one or more composite parts causes three changes in the part's status. The three changes are: the part can be displayed, the printing of the part description no longer contains the "component part" message, and the part can be written to a geometry file.

The restoration of a previously deleted part causes the part to be made active again; and if the part is a component of a composite part, the component is also reactivated. It should be noted that unlike the DELETE command, no implicit restorations occur (see section 4.3.3). Consequently, a RESTORE command must be issued for the originally deleted part and each implicitly deleted part when the user wants to undo a previous DELETE command which caused implicit deletions.

If the first component of an assembly is restored by the restoration of a previously deleted part, a warning message and a query will be issued at the first occurrence of this condition. The user can recover by entering "N" to ignore the restoration and return to the editor. By entering "Y", the restoration of the first component and all subsequent first component restorations will be performed. The restoration of the first component of an assembly warrants this level of caution since the first component affects the positioning and orientation of the assembly (see section 2.5).

4.3.5 PRINT

The PRINT (P) command sublevel prints the descriptive information for all parts that have not been "deleted". The descriptive information includes the identification number, the color, the dimension and construction parameters, the orientation parameters, and the part description.

The print format is illustrated in FIGURE 6 where a representative from each part "type" is included. (As a space saving measure the "types" are not separated by pages.) The order used in this figure parallels the order in PRINT.

The part "type" is written at the top of the display area followed by a header. The header consists of a row of column numbers, a row of part attribute descriptions, and a row of transformation descriptions. The descriptive information for one or more parts follows the header. If a part is a component of a composite part, the message:

COMPONENT PART

is appended to the part's descriptive information.

To signal the user that output to the display area is complete, the terminal's "bell" is rung and a message is written informing the user to enter a carriage return to advance to the next "page" of information. The user is returned to the EDITOR command level at the completion of the print output.

It should be noted that the format used in the PRINT command is maintained in all of the EDITOR sublevel commands when printing a part's description.

ASSEMBLIES

PART ID	8		COLOR	1
COMPONENTS	3	4	5	

1	2	3	4	5	6	7	8	9
ROTATEX(DEG)	ROTATEY(DEG)	ROTATEZ(DEG)	SCALE	SCALEY	SCALEZ	TRANSLATEX	TRANSLATEY	TRANSLATEZ
0.00000E-01	1.80000E+02	0.00000E-01	1.00000E+00	1.00000E+00	1.00000E+00	0.00000E-01	0.00000E-01	0.00000E-01

THIS IS AN ASSEMBLY

BOXES

1	2	3	4	5	6	7	8	9
PART ID	COLOR	LENGTH(X)	HEIGHT(Y)	WIDTH(Z)				
ROTATEX(DEG)	ROTATEY(DEG)	ROTATEZ(DEG)	SCALE	SCALEY	SCALEZ	TRANSLATEX	TRANSLATEY	TRANSLATEZ
1	1	1.50000E+00	1.50000E+00	1.50000E+00				
0.00000E-01	0.00000E-01	0.00000E-01	1.00000E+00	1.00000E+00	1.00000E+00	-1.00000E+00	0.00000E-01	0.00000E-01

THIS IS A CUBE
*** COMPONENT PART ***

SPHERES

1	2	3	4	5	6	7	8	9
PART ID	COLOR	RADIUS	#LATITUDES	#LONGITUDES	REV. ANGLE			
ROTATEX(DEG)	ROTATEY(DEG)	ROTATEZ(DEG)	SCALE	SCALEY	SCALEZ	TRANSLATEX	TRANSLATEY	TRANSLATEZ
2	2	1.10000E+00	11	10	1.80000E+02			
4.50000E+01	4.50000E+01	4.50000E+01	2.00000E+00	3.00000E+00	4.00000E+00	0.00000E-01	0.00000E-01	0.00000E-01

THIS IS A HEMI-SPHERE

CONES

1	2	3	4	5	6	7	8	9
PART ID	COLOR	POS. RADIUS	NEG. RADIUS	LENGTH	REV. ANGLE	#SIDES		
ROTATEX(DEG)	ROTATEY(DEG)	ROTATEZ(DEG)	SCALE	SCALEY	SCALEZ	TRANSLATEX	TRANSLATEY	TRANSLATEZ
3	3	2.00000E+00	5.00000E-01	2.00000E+00	3.60000E+02	4		
0.00000F-01	0.00000E-01	0.00000E-01	1.00000E+00	1.00000E+00	1.00000E+00	0.00000E-01	0.00000E-01	0.00000E-01

THIS IS A TRUNCATED CONE
*** COMPONENT PART ***

PARABOLOIDS

1	2	3	4	5	6	7	8	9
PART ID	COLOR	LENGTH(X)	WIDTH(Y)	#LATITUDES	#LONGITUDES	REV. ANGLE		
ROTATEX(DEG)	ROTATEY(DEG)	ROTATEZ(DEG)	SCALE	SCALEY	SCALEZ	TRANSLATEX	TRANSLATEY	TRANSLATEZ
4	4	1.00000E+01	1.50000E+00	11	10	3.60000E+02		
0.00000E-01	2.70000E+02	0.00000E-01	1.00000E+00	1.00000E+00	1.00000E+00	1.00000E+01	0.00000E-01	-1.00000E+01

THIS IS A PARABOLOID
*** COMPONENT PART ***

TORI

1	2	3	4	5	6	7	8	9
PART ID	COLOR	INNER RADIUS	OUTER RADIUS	#SECTIONS	#SIDES	START ANGLE		
ROTATEX(DEG)	ROTATEY(DEG)	ROTATEZ(DEG)	SCALE	SCALEY	SCALEZ	TRANSLATEX	TRANSLATEY	TRANSLATEZ
5	5	1.50000E+00	1.75000E+00	10	10	0.00000E-01		
9.00000E+01	0.00000E-01	9.00000E+01	1.00000E+00	1.00000E+00	1.00000E+00	0.00000E-01	1.00000E+02	0.00000E-01

THIS IS A TORUS
*** COMPONENT PART ***

FIGURE 6
EDITOR PRINT FORMAT

```
                                                                 EXTERNAL

    1            2            3            4            5            6            7            8            9
PART ID      COLOR
ROTATEX(DEG) ROTATEY(DEG) ROTATEZ(DEG) SCALE        SCALEY       SCALEZ       TRANSLATEX   TRANSLATEY   TRANSLATEZ


           10            7
 4 50000E+01  9 00000E+01  1 35000E+02  5 00000E+00  1 00000E+00  1 00000E+00 -1 00000E+00  0 00000E-01  0 00000E-01


                                                                 TRAN-SWEEPS

    1            2            3            4            5            6            7            8            9
PART ID      COLOR        ORIGINX      ORIGINY      ORIGINZ      CAPS (1=YES) CAP (X)      CAP (Y)
ROTATEX(DEG) ROTATEY(DEG) ROTATEZ(DEG) SCALE        SCALEY       SCALEZ       TRANSLATEX   TRANSLATEY   TRANSLATEZ

            6            6  5 00000E+00  6 00000E+00  7 00000E+00            1  0 00000E-01  0 00000E-01
 0 00000E-01  0 00000E-01  0 00000E-01  1 00000E+00  1 00000E+00  1 00000E+00  2 50000E+00  3 00000E+00  3 50000E+00
THIS IS A SWEPT STAR


POINT  1 (X,Y) : -1 00000E+00  0 00000E-01
POINT  2 (X,Y) : -2 50000E-01  2 50000E-01
POINT  3 (X,Y) :  0 00000E-01  1 00000E+00
POINT  4 (X,Y) :  2 50000E-01  2 50000E-01
POINT  5 (X,Y) :  1 00000E+00  0 00000E-01
POINT  6 (X,Y) :  2 50000E-01 -2 50000E-01
POINT  7 (X,Y) :  0 00000E-01 -1 00000E+00
POINT  8 (X,Y) : -2 50000E-01 -2 50000E-01
POINT  9 (X,Y) : -1 00000E+00  0 00000E-01


                                                                 ROTA-SWEEPS

    1            2            3            4            5            6            7            8            9
PART ID      COLOR        #SIDES       REV. ANGLE   CAPS (1=YES) CAP (X)      CAP (Y)
ROTATEX(DEG) ROTATEY(DEG) ROTATEZ(DEG) SCALE        SCALEY       SCALEZ       TRANSLATEX   TRANSLATEY   TRANSLATEZ

            7            7           10  3 60000E+02            2  0 00000E-01  0 00000E-01
 0 00000E-01  0 00000E-01  0 00000E-01  1 00000E+00  1 00000E+00  1 00000E+00  1 50000E+00  0 00000E-01 -8 34465E-07
THIS IS A SWEPT TORUS


POINT  1 (X,Y) :  1 00000E+00  1 00000E+00
POINT  2 (X,Y) :  1 00000E+00  2 00000E+00
POINT  3 (X,Y) :  2 00000E+00  2 00000E+00
POINT  4 (X,Y) :  2 00000E+00  1 00000E+00
POINT  5 (X,Y) :  1 00000E+00  1 00000E+00


                                                                 BOOLEANS

PART ID              9         COLOR          2
OPERATION      UNION                 COMPONENTS          1    3

    1            2            3            4            5            6            7            8            9
ROTATEX(DEG) ROTATEY(DEG) ROTATEZ(DEG) SCALE        SCALEY       SCALEZ       TRANSLATEX   TRANSLATEY   TRANSLATEZ
 0 00000E-01  0 00000E-01  0 00000E-01  1 00000E+00  1 00000E+00  1 00000E+00 -3 75000E-01 -1 19209E-07 -2 38419E-07
THIS A BOOLEAN
```

FIGURE 6 (Continued)
EDITOR PRINT FORMAT

4.3.6 COPY

The COPY (C) sublevel command allows the user to create a new part by making a copy of an existing (non-deleted) part. All attributes and orientation parameters of the part to be copied will be duplicated in the new part.

After issuing the COPY command, the user is requested to enter the part identification number of the part to be copied or a QUIT (or Q) to return to the EDITOR command level. If an invalid part number is entered (i.e. the part is deleted or non-existent) then the request is repeated.

A menu of five options is displayed next, and the user must enter an integer between 1 and 5 which corresponds with the desired option:

1 TO PRINT PART DESCRIPTION
2 TO PRINT PART DESCRIPTION AND DISPLAY PART
3 TO CONFIRM COPY
4 TO MAKE NEW SELECTION
5 TO RETURN TO THE EDITOR

The first two options allow the user to verify that the selected part I.D. number matches the part to be copied by printing the part definition (see section 4.3.5), or by printing the part definition and displaying the part (see section 4.4.2). The third option allows the user to confirm the copy and automatically enter the MODIFY sublevel command (see section 4.3.2). The fourth option allows the user to select a different part for copying. Finally, the fifth option returns to the EDITOR command level without copying the part.

4.4 DISPLAY COMMAND SUBLEVEL

The DISPLAY (D) command sublevel permits the designer to view selected components of the geometric model throughout the creation and modification processes. Subject to the physical limitations of the display device, the user can designate either wire frame or shaded image renderings of the model.

Embedded within the DISPLAY sublevel is a graphics editor. The SMP GRAPHICS EDITOR provides the designer with a mechanism for transforming primitives and querying the model geometry through the manipulation of an interactive graphical input device.

The complete list of viewing operations follows:

R – RESET VIEWING OPTION
M – VIEWING OPTION MENU
D – DRAW MODEL
T – DISPLAY FOUR VIEWS OF MODELS
Z – ZOOM
S – HIDDEN SURFACE
G – GRAPHICS EDITOR

These operations are detailed in subsequent sections.

The general viewing options, set by the "R" and "M" commands, impact the other viewing operations. The viewing options include: setting the global transformation parameters, selecting the form in which "back faces" are to be depicted, toggling the "shrink" switch, and opting for part labels. The terminology describing these viewing options requires further explanation. The global transformation matrix (rotation and scaling only) determines the spatial orientation for the entire model. (Note that the global transformation is not retained in either the parts-file or geometry-file). Recalling the boundary representation scheme (see section 3.2), a "back face" is a planar section of a primitive whose normal points away from the "observer" (the viewing surface). The "shrinking" operation is the standard finite element method (FEM) technique for separating verticies, edges, and faces. Finally, the user can "mark" each part with its identification number via the part labeling option.

4.4.1 RESET COMMAND

The RESET (R) command serves to initialize or reset the general viewing transformation and options.

Employing the terminology of the preceding section, the global transformation matrix is set to an identity matrix. No distinction is made between "front" and "back" faces. Element (edge) "shrinking" is disabled, and parts are not marked with their identification numbers.

RESET does not preselect parts for viewing; this responsibility lies with the module that created the part (either the EDIT commands (see section 4.3) or the READ command (see section 4.1)) or with the MENU command discussed in the next subsection. However, RESET does deselect all component parts.

4.4.2 MENU COMMAND

The MENU (M) command provides facilities for: designating which parts of the model are to be displayed, modifying the global transformation matrix, and specifying the desired viewing options.

The system prompt

DESIGNATE THE "PARTS" TO DISPLAY

signals the user to select parts for viewing. As discussed previously (see section 4.0), an "*" (implying all parts) or a list of parts separated by a "," (individual parts) and/or "-" (part ranges) is input to select parts.

The user is next prompted to

ENTER "Y" TO CHANGE THE VIEWING TRANSFORMATION.

An affirmative response results in the user being prompted to

ENTER ROTATION ANGLES IN X,Y,Z

and then

ENTER SCALE FACTORS IN X,Y,Z.

The transformations are applied in the following order: X rotation, Y rotation, Z rotation, X scale, Y scale, and Z scale. The resulting transformation matrix remains the current global transformation until either the RESET (see section 4.4.1) or MENU commands are reissued. Recall that transformation values are <u>not</u> accumulated on successive calls to MENU. Because there is no "clipping" of the model, with the exception of the ZOOM command (see section 4.4.5), there is no provision for global translation.

The system prompt

ENTER "Y" TO CHANGE VIEWING OPTIONS

signals the user that options controlling the display of "back faces", printing of part labels, and "face shrinking" can now be modified. The removal (<u>cull</u>) of "back faces" can simplify a wire frame display of the model. SMP provides the user with three choices for accommodating the display of "back faces" prompted as follows:

```
ENTER BACK FACE CULL OPTION
-1     NO CULL
 0     HIDE BACK FACES
 1     DASH BACK FACES
```

The prompt

```
ENTER "Y" FOR PART LABELS
OR "N" FOR NO PART LABELS
```

identifies the "part label" option. A "Y" input results in the part identification number (see section 3.2) being output at the part "centroid" (see section 2.0). The "shrink" option is used to separate the individual faces within a part by shortening the edges of the face around its centroid. An affirmative response to the prompt

ENTER "Y" FOR ELEMENT SHRINKING

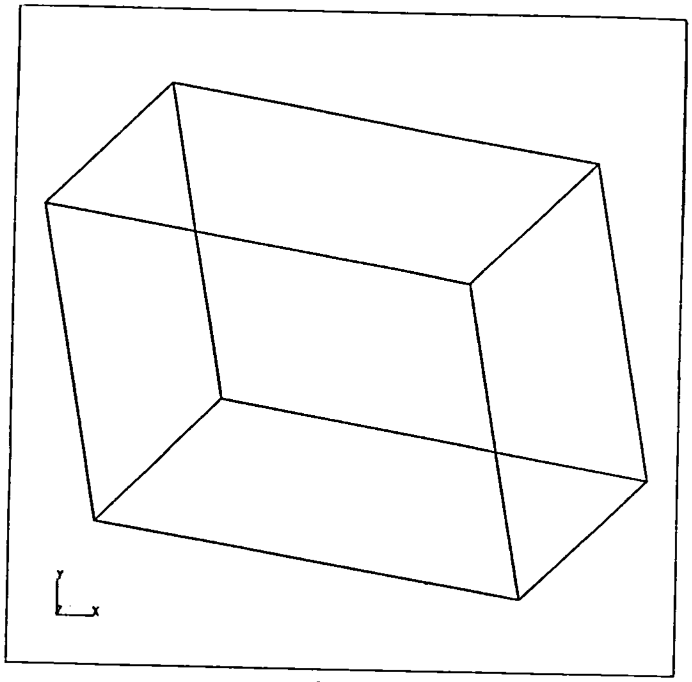
OR "N" FOR NO SHRINKING

requires the user to

ENTER SHRINKAGE FACTOR [0, 1]

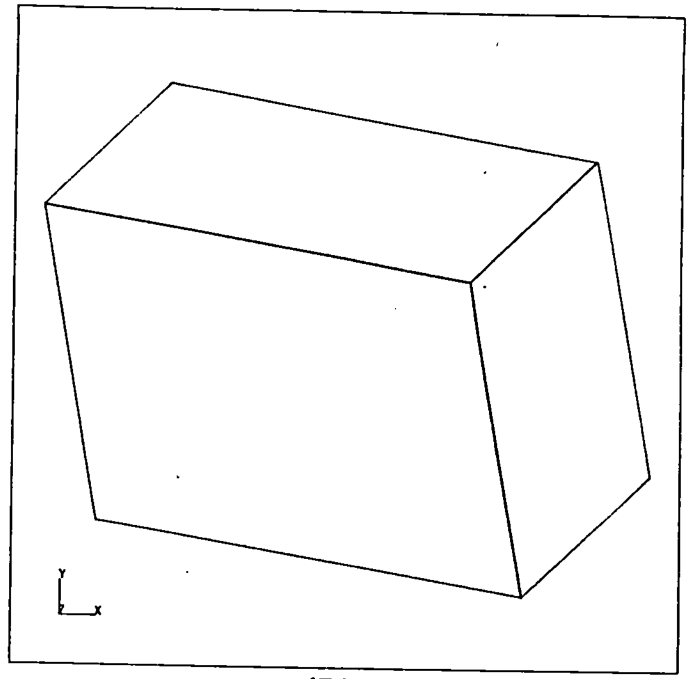
Increasing the "shrinkage factor" increases the extent of the shrinking.

The "back face cull" and "shrink" options are illustrated in FIGURE 7. FIGURE 7(a) shows a primitive "box" with no cull. FIGURE 7(b) shows the same "box" with a cull, and FIGURE 7(c) depicts the "back faces" by dashed lines. The "shrink" option is illustrated in FIGURE 7(d).



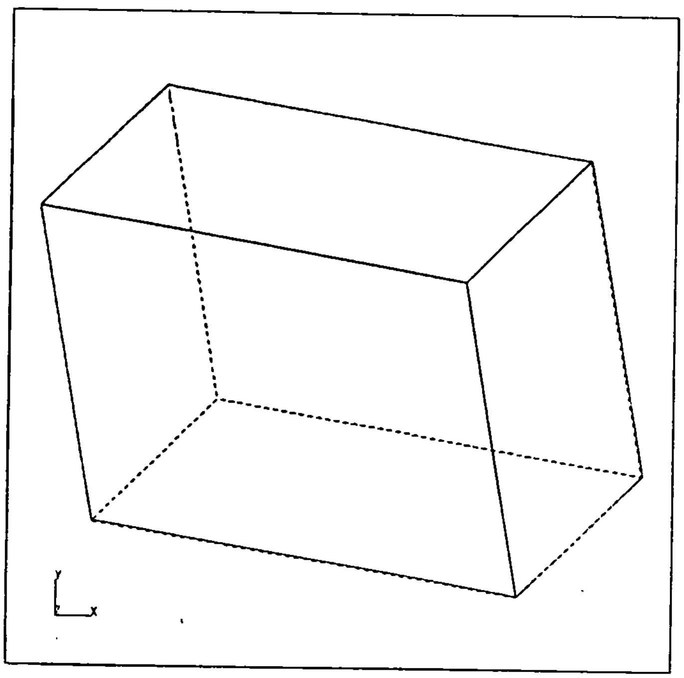
(A)

NO BACK FACE CULL



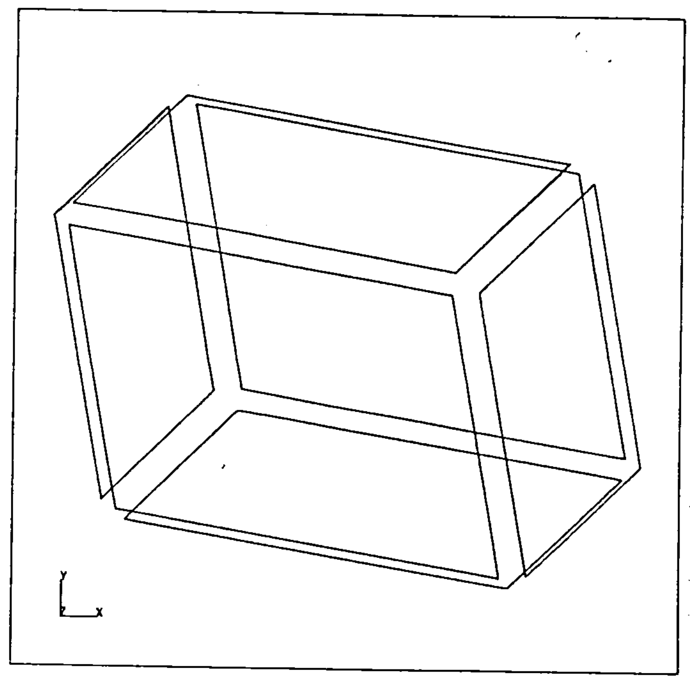
(B)

BACK FACE CULL



(C)

DASH BACK FACES



(D)

SHRINK FACES

Figure 7

4.4.3 DRAW COMMAND

The DRAW (D) command generates a "wire-frame" drawing of the geometric model subject to the part selection, global transformation, and viewing options specified by either the RESET (see section 4.4.1) or MENU (see section 4.4.2) commands.

The DRAW command does not have any options. If the display device supports color, the parts are rendered in their assigned colors.

The 3-D axes positioned in the lower left corner of the display provides the user with the current global orientation. (Recall that SMP utilizes a right-handed coordinate system with the default view being the xy-plane and positive z-axis facing the viewer.)

If the "wire-frame" rendering appears too cluttered, the "back face cull" option (see section 4.4.2) may simplify the image. If specific sections of the model are too small to resolve, selecting specific parts (see section 4.4.0) and viewing only a partial model may be appropriate.

4.4.4 FOUR VIEW COMMAND

The FOUR VIEW (T) command expands the DRAW command (see section 4.4.3) by automatically providing four distinct views of the geometry model subject to same specifications as DRAW.

The display surface is subdivided into four equally sized regions. The default view (xy-plane) is displayed in the upper left region. A yz-plane view is displayed in the upper right region, and a zx-plane view is displayed in the lower left region. The lower right region views the model from the current global orientation as specified in the MENU command. If MENU has not been called, a default rotation of X,Y,Z=45 degrees is supplied.

The FOUR VIEW command is not subject to options.

FIGURE 8 illustrates the FOUR VIEW command.

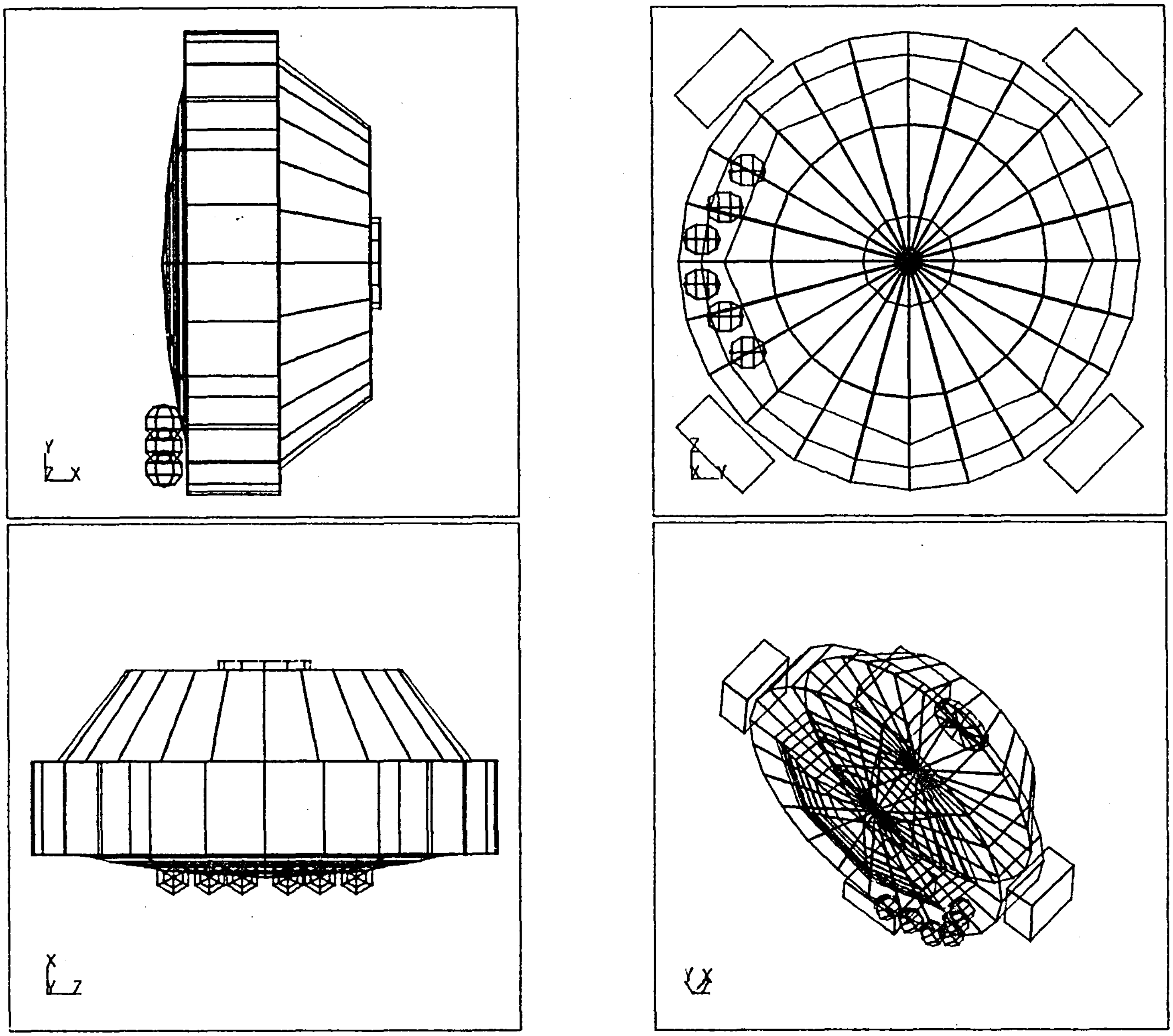


FIGURE 8
SPACE STATION FOUR VIEW

4.4.5 ZOOM COMMAND

The ZOOM (Z) command permits the designer to "zoom-in" on a specified region of the geometric model. The command requires interactive user input implying that the command is applicable only to those graphics terminals which support one or more graphical input devices.

The user is initially prompted as to whether the model should be redrawn. (A negative response is a time-saver if the image is already viewable.) The redrawn view is oriented by the current global transformation matrix and is subject to current part selections and viewing options.

SMP prompts the user to

PICK REGION LL CORNER

and

PICK REGION UR CORNER

which correspond to the lower left ("LL") and upper right ("UR") corners of the rectangular region of the model to be zoomed. The specified zoom region is depicted by a dash-lined rectangle, and the user is required to:

```
ENTER Y (ACCEPT)
   OR N (REJECT)
   OR R (RETURN).
```

A "Y" response results in the display of the zoomed region of the model. If the region is unacceptable, a "N" response forces a return to the zoom region selection process. A response of "R", causes an exit to the DISPLAY command level.

Because the default "window" is not reset upon exit from ZOOM, the zooming process can be repeated to view a still smaller region of the model via repeated calls to ZOOM. References to either RESET (see section 4.4.1) or MENU (see section 4.4.2) cause the default "window" to be restored.

4.4.6 HIDDEN SURFACE COMMAND

The HIDDEN SURFACE (S) command renders the geometric model as a shaded image with all hidden surfaces removed. This command is intended for use on raster graphics devices (either color or monochrome) where some type of "imaging" is possible. However, a hidden line option is available for vector storage tubes or refresh devices.

The hidden surface calculation is an adaptation of the "priority" algorithm of Newell [11]. The user can invoke either of two shading models: "flat" or "smooth". The "flat" (or "constant") shading model is based on the model described by Foley [12], and the "smooth" shading model is a variant of Gouraud's work [13].

The user is initially prompted to enter the display option:

```
ENTER  S - ELIMINATE HIDDEN SURFACE
       L - ELIMINATE HIDDEN LINES
       B - BOTH (SEPARATE VIEWS)
       R - RETURN.
```

A response of "S", "L", or "R" results in an action suggested by the prompt. (However, a selection of "S" on a non-raster device is automatically changed to an "L" without any informative messages.) If both a hidden surface and hidden line rendering are desired, the "B" response should be selected. In this case, two views are produced without user intervention: the hidden line view followed by the hidden surface view.

Limitations of the available graphics display hardware resulted in the inclusion of three special purpose image display options. These options relate to: saving the image on a disk file, changing the computation resolution, and opting for an alternate color table. The user is prompted to

```
ENTER "Y" TO OVERRIDE DEFAULT
IMAGE DISPLAY OPTIONS.
```

Notice that there are no "current" values for these options; any negative response always results in the default values being used.

An affirmative response results in the user being prompted to

```
ENTER DISPLAY/DISK IMAGE OPTION
 0  -  IMAGE OUTPUT TO DISPLAY
 1  -  IMAGE STORED ON FILE "SMGG.IMG"
```

Entering "0" invokes the default action. If "1" is specified, the image is written to the file <u>SMGG.IMG</u>. This file can be later post-processed via a local MOVIE.BYU[9] post-processor program. The option is most applicable to generating a shaded image while working from a non-raster display device.

The second option permits the user to change the computation resolution to be other than the default device resolution.

This option is normally used in conjunction with the first option when the resolution of the "target" display device differs from the present display device.

The prompt

```
ENTER RESOLUTION  CHANGE OPTION
 N  -  DEFAULT TO DEVICE LIMITS
 Y  -  REDUCE FOR FASTER DISPLAY
```

"N" is the default response. As the "Y" prompt implies, reducing the computation resolution will reduce the CPU and "wall-clock" time necessary to generate the image. If a "Y" response is received, the user must

```
ENTER NEW RESOLUTION (SQUARE ONLY)
 RESOLUTION FOLLOWED BY STARTING PIXEL
```

The "square" restriction is intended to prohibit image distortion. The "resolution" (IMAX) and "starting pixel" (IMIN) input cause the image to be displayed on the screen in a square of dimension (IMAX-IMIN) with lower left corner at (IMIN, IMIN). Every effort is made to detect and correct illegal responses.

The third special option relates to overriding the assigned <u>part colors</u> (see section 2.0) and employing a "grey scale" color table. This option is useful (but not necessary) only if the "smooth" shading option is selected. The prompt

```
ENTER COLOR TABLE OPTION
 0  -  USE ASSIGNED PART COLORS
```

1 - OVERRIDE WITH GREY SCALE

signals the option. The default response is "0".

The _shading options_ and _parameters_ must now be specified. The reader should refer to references [12] and [13] for pertinent background information. The user is initially required to

ENTER SHADING OPTION
0 - FLAT ELEMENT SHADING
1 - SMOOTH ELEMENT SHADING.

Because of the reduced computation time, "flat" shading is usually preferred. If the "flat" shaded image does not produce the desired realism, "smooth" shading should be employed.

After the shading model has been selected, the default shading parameters can be modified. The prompt

ENTER "Y" TO CHANGE SHADING PARAMETERS

triggers this action. An affirmative response permits the user to

ENTER DIFFUSED LIGHT VALUE

and

ENTER REGULAR LIGHT EXPONENT.

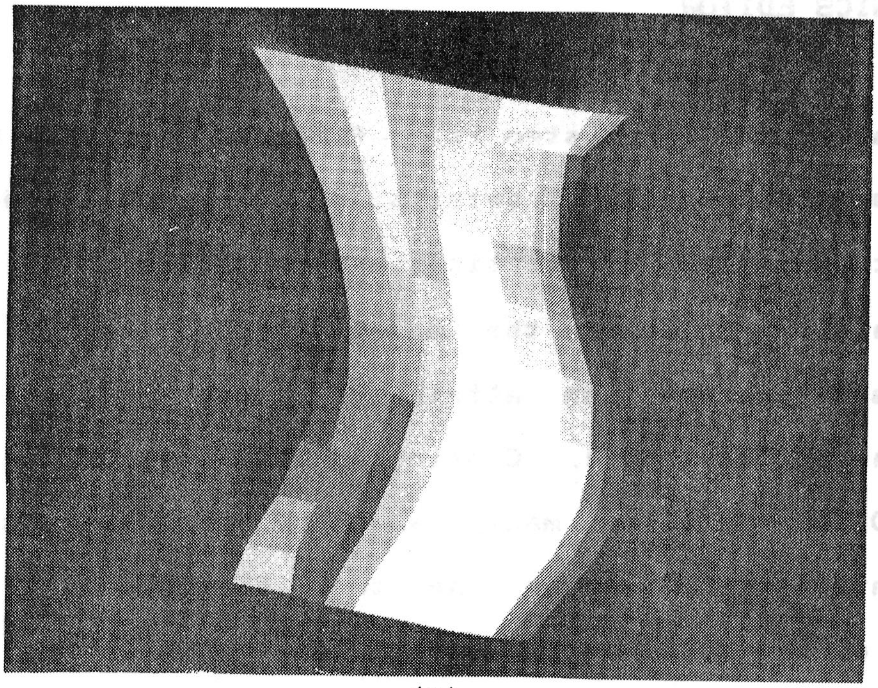
The "diffused" light coefficient defaults to 0.30 and must reside in the interval [0,1]. The regular light exponent defaults to 1. Since shading models are more empirical than analytical, choosing the proper values for these parameters may require experimentation.

The final option is the "back face cull" (see section 4.4.2) option. The user is prompted to

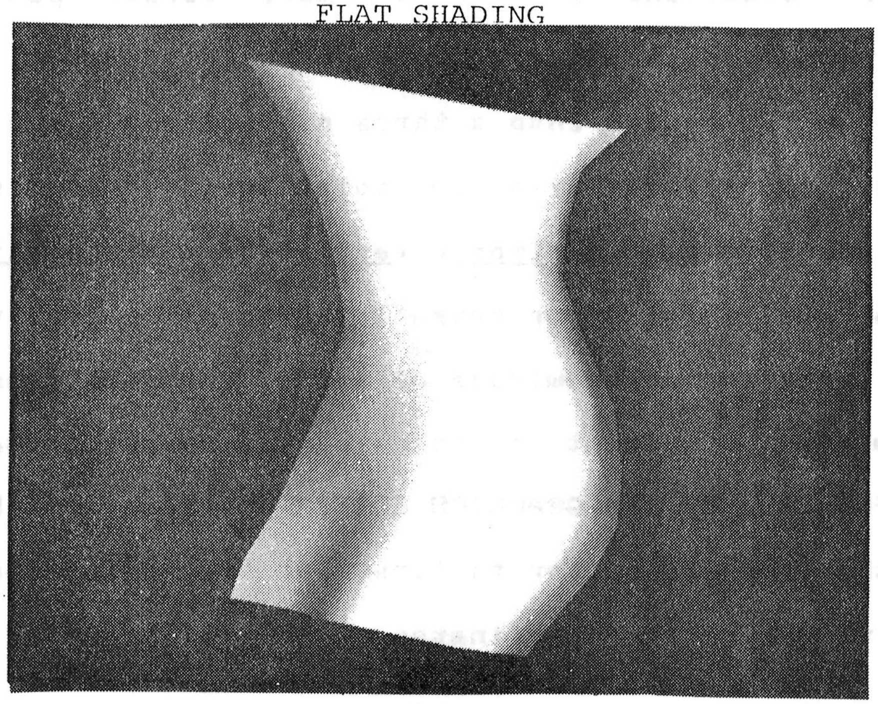
```
ENTER BACK FACE CULL OPTION
 N  -  NO BACK FACE CULL
 Y  -  PERFORM BACK FACE CULL
 R  -  RETURN.
```

A "Y" response will normally reduce the number of faces that must be processed; and thus speed-up the display process. However, this option is not always desirable on objects with "holes". Notice that the "R" response affords the last opportunity to "return" to the DISPLAY command sublevel.

The distinction between "flat" and "smooth" shading is illustrated in FIGURE 9. The surface patch is generated external to SMP and later input as an "external" part (see section 2.5). The effect of "flat" shading is shown in 9(a) and the effect of "smooth" shading is depicted in 9(b).



(A)
FLAT SHADING



(B)
SMOOTH SHADING
FIGURE 9 - SURFACE SHADED IMAGE

4.4.7 GRAPHICS EDITOR

A major goal of the SMP system is to free the user from direct interaction with lower level details of the model geometry (such as coordinates of vertices and connectivity of faces) and allow him to manipulate the model through higher level dimension and construction attributes (such as the length, width and height of a box). Continuing this philosophy, the GRAPHICS EDITOR (G) allows manipulation of model parts through direct interaction between the two dimensional (2-D) projection of the model on the screen and a graphics input device [12]. Locations of one or more screen positions (determined by positioning the graphics cursor) are transformed by the system into a three dimensional (3-D) point that may be identified with some point in the "model space" (such as a specific part centroid, vertex, or a new position). The user may modify (alter or orient) the model using visual cues while requiring no knowledge of actual boundary geometry. Should knowledge of point coordinates become important, a secondary function of the GRAPHICS EDITOR provides a mechanism for using the graphics cursor to first locate a point and then return the values of its coordinates.

All models constructed with SMP are three dimensional but must obviously be projected onto a two dimensional display screen.

A graphics input device may return the 2-D coordinates representing the screen position of the graphics cursor. This approach presents a fundamental problem for graphical editing: how may a 2-D location be used to identify a 3-D point?

The GRAPHICS EDITOR resolves the uniqueness problem for the 3-D point by allowing the user to "edit" multiple views of the model. Upon issuing a GRAPHICS EDITOR command, the user is prompted to enter a choice of view (either the XY, YZ, or ZX plane). A display of the model in the selected view is then generated, and the graphics cursor appears. The user positions the cursor to the point to be "picked" and "triggers" the graphics input device. If the desired point is not found ("pick" not close enough to any point), the screen is cleared, and a view in the next coordinate plane is displayed. The user is prompted to try again. On the third failure to obtain a unique point (a failure on all three coordinate planes) a choice is given to abort or to restart the process. Using this method, if two points eligible for picking overlay one another in a particular view, they may be distinguished in a subsequent view (provided they are not coincident). In order to conserve drawing time, when a view is selected, the system checks this choice against the current view. If the views are identical the model is not re-drawn. Also, should the user desire to abort the process, he/she may do so by entering "Q" when "triggering" the graphics input device.

Currently graphical editing is permitted on only two devices : the TEKTRONIX 401X series terminals, the AED series. In both cases, the graphics cursor is utilized as a "locator" [12] and the "pick" is performed in a manner consistent with the device.

Three basic functions are common to the GRAPHICS EDITOR commands:

1.) "pick" a <u>part</u>

2.) "pick" a <u>vertex</u>

3.) "pick" a <u>new point</u> in space.

All three employ the previously described method for transforming two dimensional screen locations into a three dimensional point.

"Picking" a <u>part</u> requires the user to locate the part "<u>centroid</u>" (which is marked by a "." and the part ID) by selecting a point in close proximity to the part centroid. Should more than one part be returned on the first "pick" (i.e. centroids aligned on an axis perpendicular to the screen), the next view is automatically displayed. However, for the second view, the parts no longer in contention are displayed with dashed lines (thus simplifying the user's choice). This may be repeated for the third view if a unique part has not been returned.

The part selection procedure is illustrated in FIGURE 10(a) and FIGURE 10(b). The centroids of the "box" (part ID 1) and "sphere" (part ID 2) are coincident in the XY-plane view of FIGURE 10(a). A "pick" near this centroid results in the display of FIGURE 10(b). The "sphere" and "box" remain but the "cylinder" (part ID 3) is "dashed" resulting in a unique choice on the second attempt.

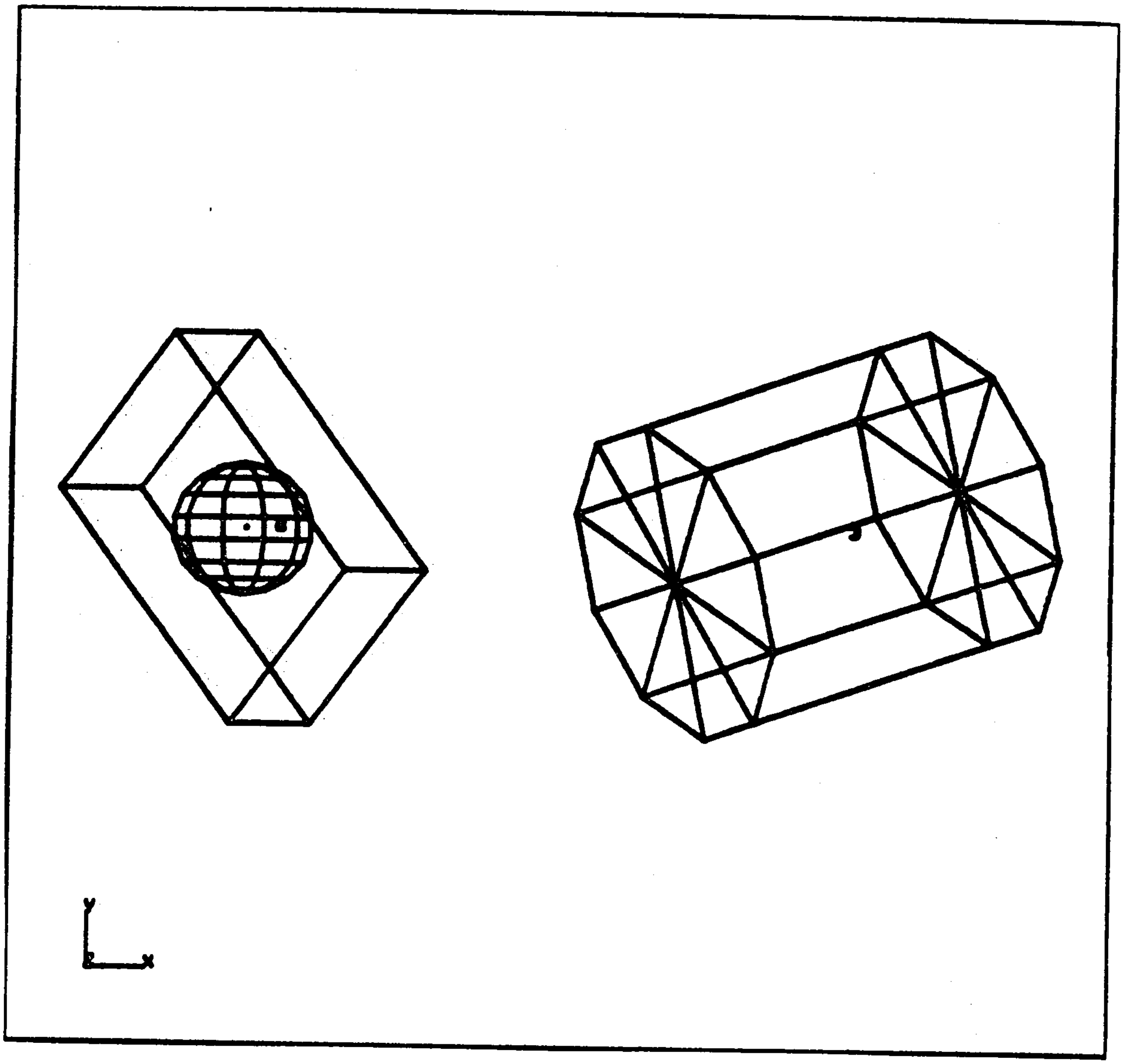


FIGURE 10a
GRAPHICAL PART SELECTION - XY PLANE VIEW

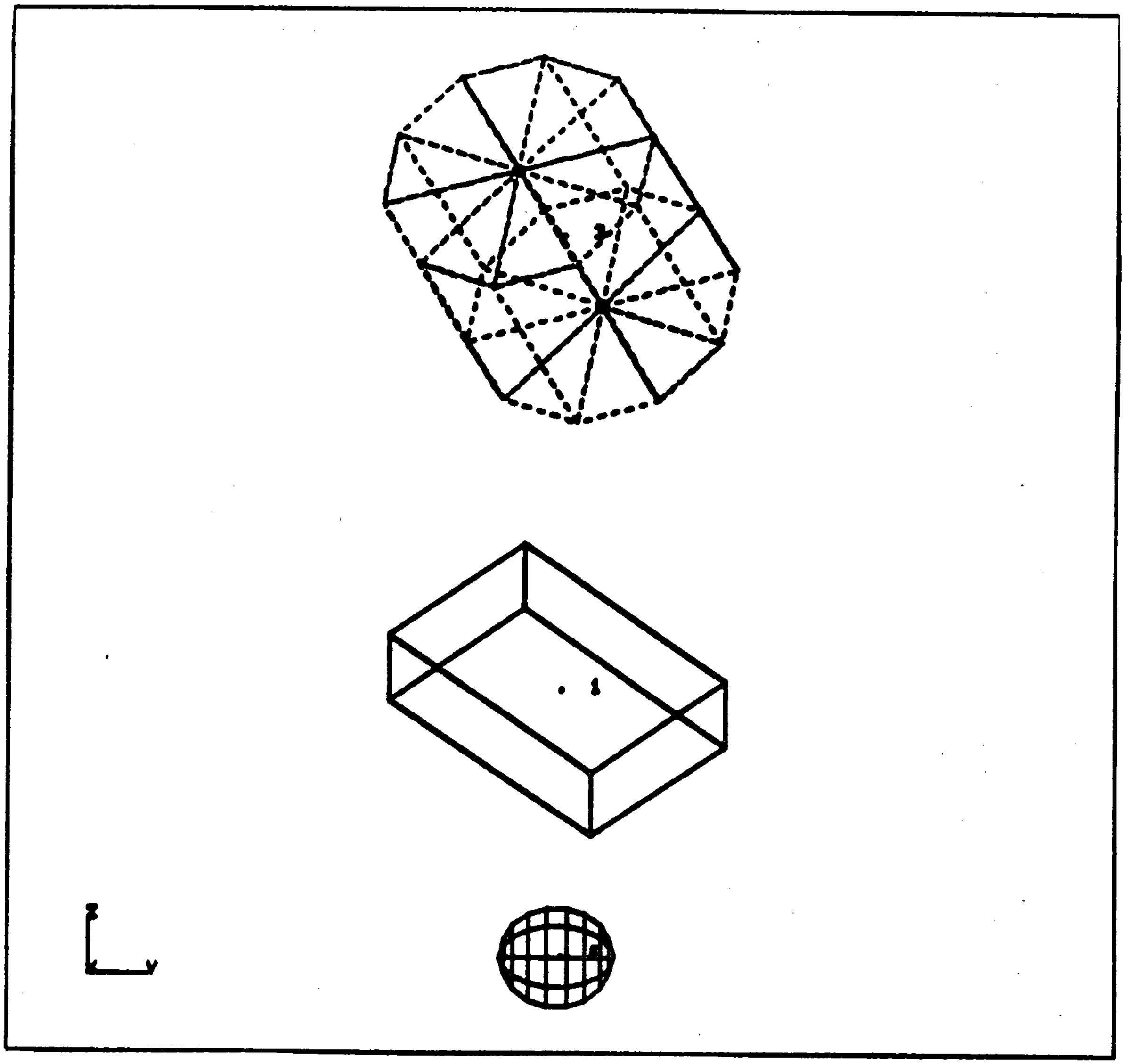


FIGURE 10b
GRAPHICAL PART SELECTION - YZ PLANE VIEW

"Picking" a vertex first requires the user to graphically identify a part. He/she must then locate a three dimensional point "near" the desired part vertex. Similar to identifying a part, "picks" are made in the three coordinate planes until a unique point is identified. Also, after the first pick, the only eligible points are those returned from the previous pick (a "dashed box" is drawn around all eligible points).

Identifying an arbitrary point in model space requires the user to "pick" any desired screen location. The 2-D values returned are assigned to the corresponding coordinates with the third coordinate acquiring the value of zero. The user is then given the opportunity to "pick" from the second view to assign the appropriate value to the third coordinate only.

The following subsections describe the individual GRAPHICS EDITOR commands:

P - Pick a part

M - Pick and modify a part

D - Pick and delete a part

C - Pick and copy on existing part

N - Pick and print the coordinates of a point

T - Translate a part

R - Rotate a part

H - Help (documentation)

Q - Exit graphics editor

X - Exit program

Each command utilizes the procedure outlined in the preceding paragraphs. The use of the word "identify" in the discussion of the individual GRAPHICS EDITOR commands implies one or more applications of the "location" process.

4.4.7.1 PICK A PART

After "identifying" a part, the part description is printed and the part is displayed.

4.4.7.2 PICK A PART AND EDIT

The EDIT (see section 4.3.2, 4.3.3, and 4.3.6) functions for MODIFYing, DELETing or COPYing a part may be accessed through the GRAPHICS EDITOR by graphically "identifying" the part instead of naming the part by its ID number.

4.4.7.3 PRINT VERTEX COORDINATES

After "identifying" a vertex from an "identified" part, its coordinates are displayed.

4.4.7.4 TRANSLATE A PART

A part may be translated in relation to its centroid or one of its verticies. After "identifying" the desired part centroid or vertex, its new position is "identified" as either an existing vertex or a new 3-D point. The display is subsequently redrawn with the part translated to the new location, and the local transformation matrix is updated.

4.4.7.5 ROTATE A PART

Because rotation is not commutative, this function is currently limited to rotation about the X, Y, and Z axes respectively. For a meaningful rotation to occur, all previous rotations are first eliminated by the system. The user may pick the part and reference point for rotation from any view but when the new point for rotation is picked, the system forces the user to choose first from the YZ view (rotation about the X axis) then the ZX view (rotation about the Y axis) and then the XY view (rotation about the Z axis). After a part and its reference point are picked, a new point is picked which, with the centroid, forms a line. The part is then rotated counterclockwise until the reference point intersects with this line. After each rotation, the user is allowed the options to rotate from the next view, rotate again by the same point in the current view, rotate again by a different point in the current view, display the model in 4 views, or return to the editor.

FIGURE 11(a) depicts three boxes as originally positioned. After selecting a part and a reference vertex for rotation (part #1 and top, right vertex), the user is prompted to supply a new point to form the reference line for rotation (FIGURE 11(b)). The chosen point and line are illustrated.

FIGURE 11(c) demonstrates the positions of the boxes after rotation. Note that part #2 is unchanged from FIGURE 11(a).

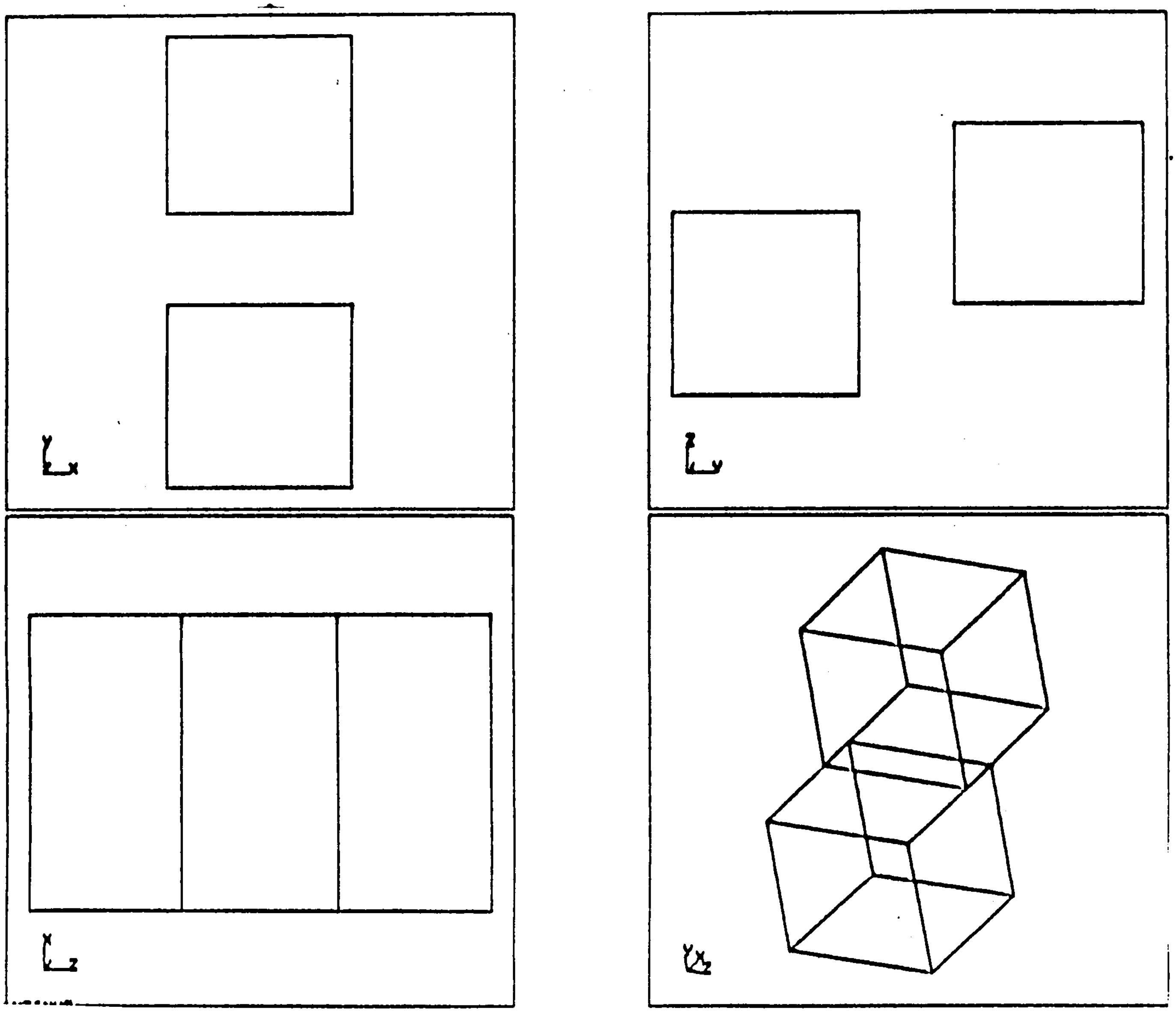


FIGURE 11a

GRAPHICAL PART ROTATION - UNROTATED FOUR VIEW

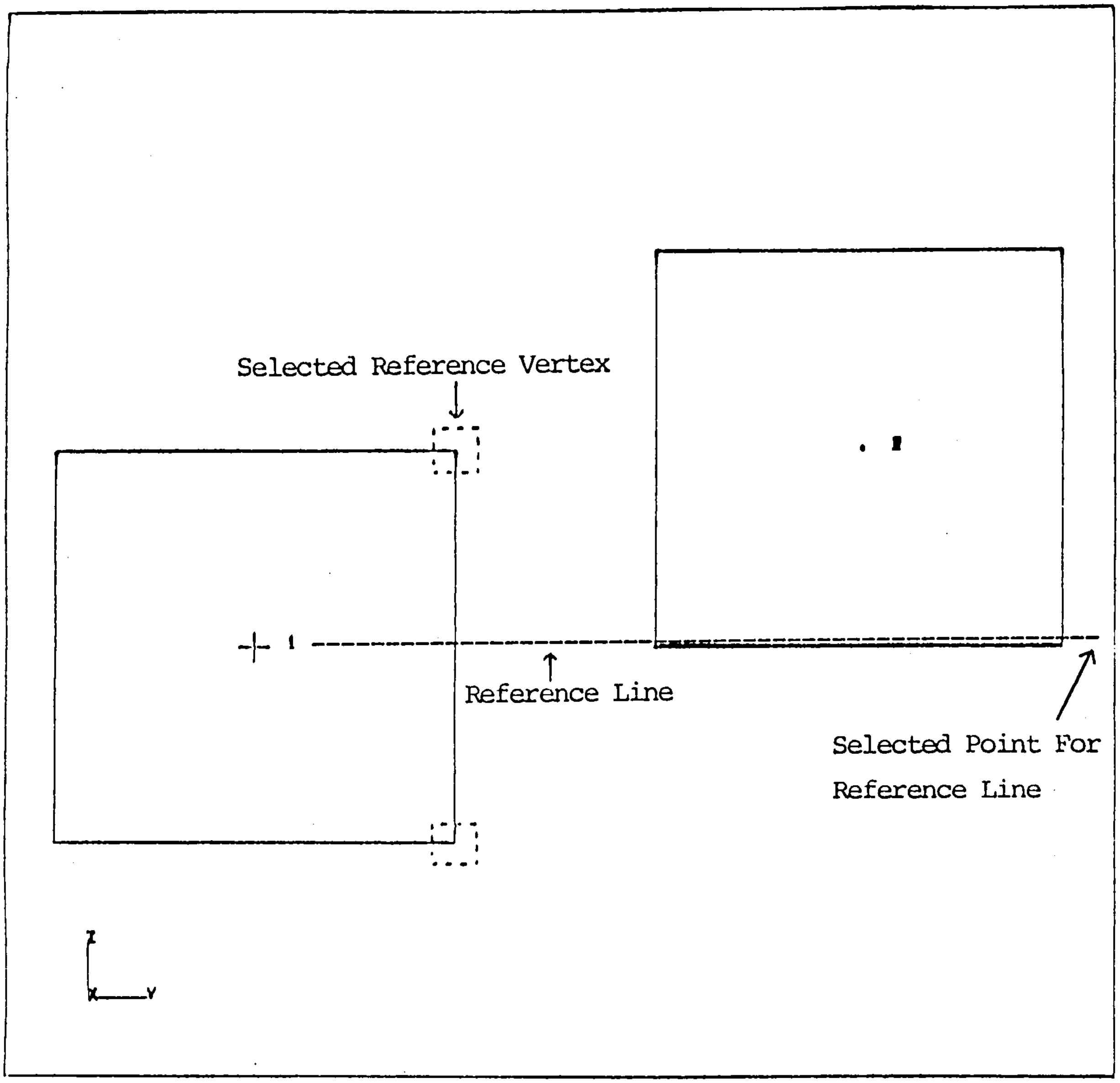


FIGURE 11b

GRAPHICAL PART ROTATION -
REFERENCE VERTEX AND LINE SELECTION

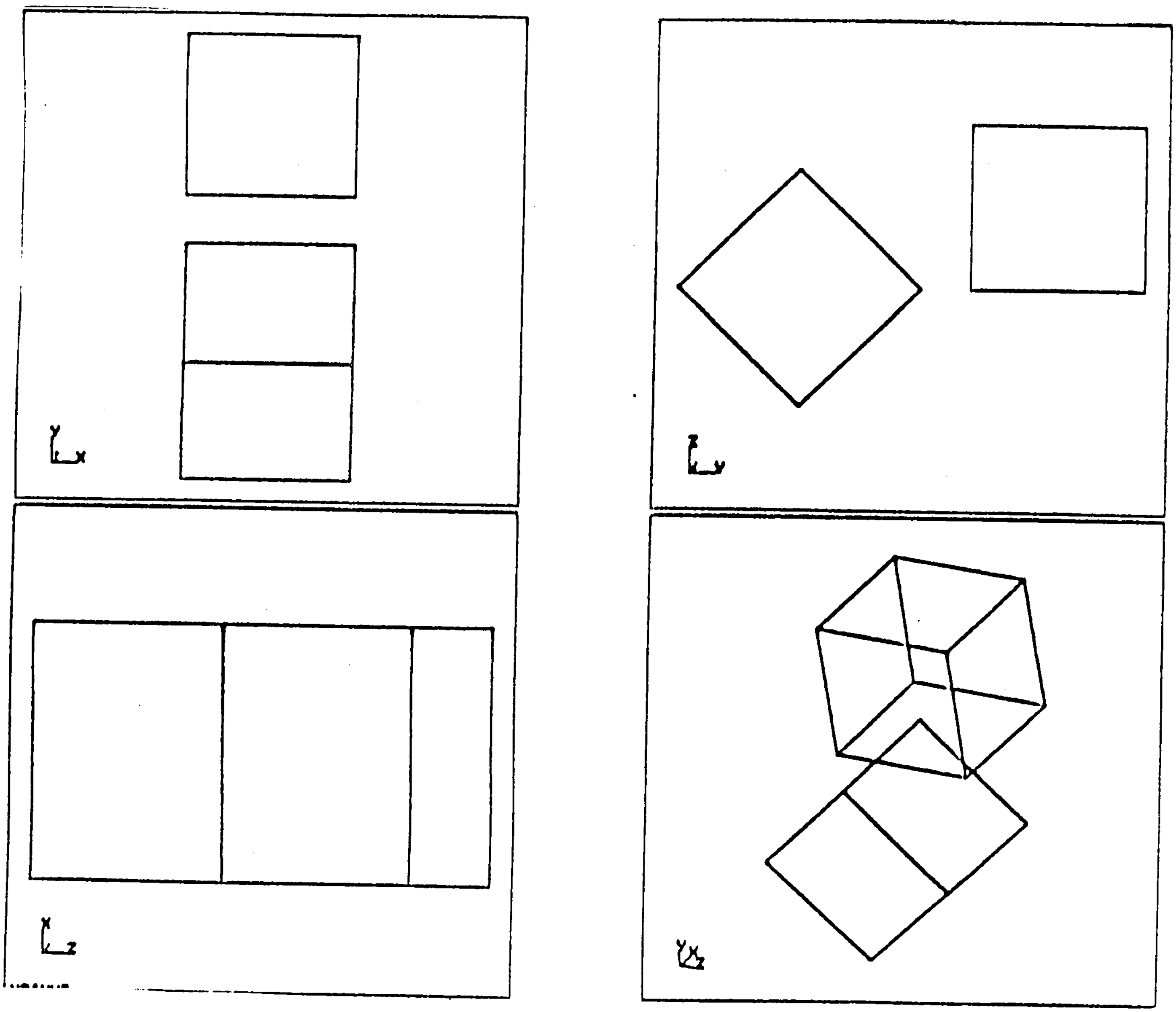


FIGURE 11c
GRAPHICAL PART ROTATION - ROTATED FOUR VIEW

4.5 MISCELLANEOUS COMMANDS

A solid modeling package should provide for more than a capability to create, manipulate, and display geometric data. For example, the user may require relevant mass properties and/or drafting information to supplement the model geometry. Because priorities emphasized model building and model display, SMP is severely limited in areas such as mass properties. Although some effort has been expended in this area, the work is incomplete and thus, will not be discussed in this document.

A "dimensioning" command available under the MISCELLANEOUS command level is useful in determining the relative positions of model primitives. The DIMENSION command will be detailed in the following subsection.

4.5.1 DIMENSIONS BETWEEN PARTS

Using functions of the GRAPHICS EDITOR (see section 4.4.7) the user may request the distance between two part "centroids" or between two selected vertices (on the same or different parts). In addition to the absolute distance between the selected points, the X, Y, and Z displacements are also supplied.

The user may also specify a "range of parts" (see section 4.0). The part labels of the designated parts are displayed and line segments connecting these part centroids are drawn. The length of each line segment is then displayed at its midpoint.

5. SOFTWARE AND USER ENVIRONMENTS

The preceding sections present the SMP software from the standpoint of functionality. The intent of this section is to describe the modeller from a utility perspective by discussing the software and user environments.

No attempt will be made to describe the software algorithms or to teach the "host" operating system. Instead, this section endeavors to identify the computing environment surrounding the modeller and to give the designer a flavor of utilizing SMP in such an environment.

5.1 SOFTWARE ENVIRONMENT

The SMP package resides on a PRIME 850 minicomputer. The software is coded exclusively in FORTRAN 77 with relatively few machine dependencies. (These non-ANSI exceptions evolved for reasons of efficiency and are clearly documented as comments in the source code.) One such dependency is the "interrupt" feature used for premature termination of an SMP process (see section 4.0). SMP is compiled, loaded, and executed under the PRIMOS Rev 19.1 operating system [14], and does rely on the virtual memory facility available within PRIMOS.

The modeller is designed to be exercised in an interactive environment with a strong reliance on graphics (see section 4.4) for user feedback. SMP is interfaced to the following three graphics output devices:

TEKTRONIX 401X
AED 512 (or 767)
ISC 8001

The AED and ISC are color raster devices accommodating the display of "part" color and hidden surface renderings (see section 4.4.6). The TEKTRONIX ignores color information but is capable of producing a simulated hidden line rendering of the model. The graphics device dependent code is localized to facilitate ease of implementation for new devices.

Although the modeller source code is essentially independent of the PRIMOS file structure [14], all files necessary to "build" an executable version of SMP reside within this file structure. In addition, the user's <u>model data files</u> are standard PRIMOS files created through formatted FORTRAN WRITE statements.

The reader should refer to reference [14] and related documentation for more information on PRIMOS and its program development utilities.

5.2 USER ENVIRONMENT

Once "logged" into PRIMOS [14], SMP is executed with the PRIMOS SEG utility as follows:

```
>SEG CSC>SMP
```

For commands that require a file name as input, READ (see section 4.1) or WRITE (see section 4.2), a PRIMOS "pathname" is acceptable input.

The modeller first prompts the designer for the teminal type:

```
ENTER DEVICE CODE
1 = TEKTRONIX
2 = AED
3 = ISC.
```

The prompt is repeated until a legal response is encountered. The next prompt

```
ENTER BAUD RATE
1 = 9600
2 = 1200
```

is included to accommodate "remote logins". Again, this prompt is repeated until a legitimate response is received.

The SMP output is a title page identifying the CSC as the author and NASA LaRC as the sponsor. The user enters <CR> to continue.

At this juncture the designer is positioned at the SMP command level and is prompted accordingly with

ENTER COMMAND>.

The major command classes are detailed in section 4. Recall that information regarding the command levels, command sublevels, subcommands and command options can be obtained through the on-line HELP facility triggered by the character "H". "Help" is available from the command level and the major command sublevels. An illegal response from a command level or command sublevel results in a menu of legal commands being displayed on the user's terminal.

In a typical session the user will enter either the READ or EDIT (see section 4.3) command sublevels. If a model has been created earlier in SMP or from some external source, the READ command should be selected. EDIT should be selected if a model is to be built from "scratch".

The majority of the session will involve transferring between the EDIT and DISPLAY (see section 4.4) commands to properly construct (i.e. "edit") and orient the model geometry.

When the editing process is complete, the WRITE command (see section 4.2) is normally selected to save the edited model geometry.

Once the edited model is saved (as a permanent file), the user can continue the editing process or "exit" the system with the QUIT command.

The above discussion briefly sketches a typical interactive session with the modeller. An experienced user can naturally tailor the command selection to meet his/her requirements.

6. REFERENCES

1. Campbell, R. H.: Spacecraft Design and Cost Model Development Final Report, Aerospace Corp. Report , ATM-76(8191)-3, June 30, 1976. (See also Campbell, R.H.: Systems Cost Performance Analysis. Aerospace Corp. Reports ATM-75(7363).-3 Vol. I-III, March 1975)

2. Jones, K. H.; Randall, D. P.; Gates, R. L.; Von Ofenheim, W.H.: "The Solid Model Geometry Generator (SMGG)", NASA Conference Publication 2272, April 1983, pp. 164-174.

3. Requicha, A. A. G.; Voelcker, H. B.: "Solid Modeling: A Historical Summary and Contemporary Assessment", IEEE Computer Graphics and Applications, Vol. 2, No. 2, March 1982, pp. 9-27.

4. Baer, A.; Eastman, C.; Henrion, M.: "Geometric Modeling: a Survey", Computer Aided Design, Vol. II, No. 5, September 1979, pp. 253-272.

5. Maruyama, K.: "A Procedure to Determine Intersections Between Polyhedral Objects", International Journal of Computer and Information Sciences, Vol. 1, No. 3, 1972, pp. 255-266.

6. Parent, R. E.: "A System for Sculpting 3-D Data", SIGGRAPH '77 Proceedings, Vol. 11, No. 2, 1977, pp. 138-147.

7. Carlson, W. E.: "An Algorithm and Data Structure for 3-D Object Synthesis Using Surface Patch Intersection", Computer Graphics, Vol. 16, No. 3, July 1982, pp. 255-262.

8. Requicha, A.A.G.: "Representations for Rigid Solids: Theory, Methods, and Systems," ACM Computing Surveys, Vol. 12, No. 4, Dec. 1980, pp. 437-464.

9. Christianson, H.; and Stephenson, M.: MOVIE.BYU Training Manual, Brigham Young University, Department of Civil Engineering, 1982.

10. PDA Engineering, PDA/PATRAN-G User's Guide, Vol. 1 and 2, 1980.

11. Newell, M. G.; Newell, R.G.; and Sancha, T. L.: "A Solution to the Hidden-surface Problem", Proceedings

of the 1972 ACM National Conference, 1972, pp. 236-242.

12. Foley, J. D.; Van Dam, A.: Fundamentals of Interactive Computer Graphics, Addison Wesley, 1982, pp. 575-584.

13. Gouraud, H.: "Continuous Shading of Curved Surfaces", IEEE Transactions on Computers, June 1971, pp. 625-628.

14. Patenaude, A. W.: " User's Guide", DOC4/30-190, PRIME Computer, Inc., 1982.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-172473		

4. Title and Subtitle	5. Report Date
SMP - A SOLID MODELING PROGRAM	November 1984
	6. Performing Organization Code
7. Author(s) D.P. Randall, K.H. Jones, W.H. von Ofenheim, R.L. Gates	8. Performing Organization Report No. TAO 34100
9. Performing Organization Name and Address Computer Sciences Corporation 3217 N. Armistead Avenue Hampton, VA 23666	10. Work Unit No.
	11. Contract or Grant No. NAS1-16078
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546	13. Type of Report and Period Covered Contractor Report
	14. Sponsoring Agency Code 505-37-23-01

15. Supplementary Notes

Langley technical monitors: John E. Hogge, Donald L. Lansing and Dariene D. DeRyder

16. Abstract

This paper documents a prototype solid modeling program, SMP, developed by CSC for Langley Research Center (LaRC). The SMP software is employed by the System and Experiments Branch (SEB) of the Space Systems Division (SSD) for preliminary space station design, but is intended as a general purpose tool. The SMP document provides details concerning: the basic geometric modeling primitives and associated operators, the data representation scheme utilized to structure the geometric model, the available commands for both editing and displaying the solid model, the interactive user interface and the input/output interfaces to external software, and the utility of the package in the LaRC computing environment. The document is sufficiently detailed to serve both as a user's guide and reference manual.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
assemblies Boolean operations boundary representation Constructive Solid Geometry (CSG) geometric primitives solid modeling sweeping	Unclassified - Unlimited Subject Category - 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	139	A07

N-305