Feature-Based Geometric Modeling using B-Spline Surfaces

and a Natural Language Approach

by

Ashit R. Gandhi

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Mechanical Engineering

APPROVED:

Dr. A. Myklebust, Chairman

Dr. M. P. Deisenroth

Dr. R. H. Fries

Dr. J. R. Mahan

Dr. C. F. Reinholtz

April 14, 1989

Blacksburg, Virginia

Feature-Based Geometric Modeling using B-Spline Surfaces and a Natural Language Approach

by

Ashit R. Gandhi

Dr. A. Myklebust, Chairman

Mechanical Engineering

(ABSTRACT)

Traditionally, design geometries are represented using orthographic views which require a human being to interpret them and recognize geometric features to evaluate the design. Solid modeling systems have made the task somewhat easier, but they often require tedious and complex operations using simple geometric primitives. This has led to investigation of modeling systems which allow the creation of assemblies the way engineers conceive them - as features.

To be able to efficiently describe models in terms of features, a rich set of feature descriptors is necessary. An exhaustive study of English words describing form, shape, feature, shape altering transformations and surface conditions was done to establish a wide vocabulary for feature based description. Classification based on topology and form was done and prototype constraint relations were implemented to differentiate between some features. A feature is created from a topological group by computing points and interpolating them with uniform rational B-spline patches. Control points needed to compute the patches are computed from the interpolation points by an inverse relation. A designer-oriented modeling language, based on simple English syntax, was devised to specify procedures to be executed by the modeler in order to create features using minimal user input.

Acknowledgements

I would like to extend my deepest gratitude to the chairman of my supervisory committee, Dr. Arvid Myklebust, for his guidance, encouragement and support throughout my graduate studies.

I wish to express my sincere appreciation to Drs. Deisenroth, Fries, Mahan and Reinholtz, who served as members of my committee and offered valuable advise and support.

For her infinite support, I wish to thank my wife, , without whom this work would have taken longer. Her devotion and sacrifices have helped me overcome many hard and trying moments during the last two years.

Last and foremost, I am indebted to my parents and family members for their unfailing love and support which allowed me to make my dreams come true. I know, that no one will be happier than them to see this work completed.

Table of Contents

List of Illustrations

List of Tables

Chapter 1: Introduction

1.1 Overview

1.1.1 Need to Step-up Automation

Increasing competition throughout the world has put tremendous pressure on manufacturers to step up automation in order to make better products - and to make them more quickly and less expensively than before. This means reduced lead times for both design and manufacturing cycles and an increase in required levels of productivity and flexibility. This has led to an attempt to develop computer tools that will help automate, and hence speed up, the cycles. One way of achieving this is by integration of CAD and CAM. Yet a great many computer tools purporting to serve these ends automate only a small subset of production. What are touted as "computer-aided design" (CAD) systems, for example, usually are systems that offer dazzling graphics but few tools to improve the product that is on the drawing board.

"Computer-aided manufacturing" (CAM) often amounts to using a computer to create NC instructions and to run machine tools that are not integrated with other factory operations. Most importantly, CAD/CAM has so far done little to bridge the long-standing chasm between those who design the products and those who make them. The new generation of CAD/CAM will have to bring computer power to bear on far more of the design and production cycles.

1.1.2 Conventional CAD/CAM systems

One of the primary aims of geometric modeling is to provide a computer shape model that is as useful in design and manufacture as any physical prototype. Hence, one vital component of this stored description is a geometric model which contains information about the shape of the object under consideration. The richness and the completeness of the geometric model have a profound effect on the capabilities of the system.

In the earliest systems, data stored were only two-dimensional graphical entities which made up engineering drawings. Dimensions and notes on such drawings describe the geometric model, but their meaning is usually subjective and difficult for a computer to interpret. Such systems still predominate the field and lack the necessary 'intelligence' to review the engineering drawing, interpret it, and extract a set of useful information.

More recent systems have attempted to store descriptions of physical objects themselves. This description can be stored by using wire-frames, surface models or solid models and can be used for many different purposes such as generation of

engineering drawings and analyses and generation of NC instructions. An object is best modeled using solid models, since they contain maximum information that would help supporting engineering functions like mass property calculations, interference checks, etc. Of the several methods available for solid model construction, three methods are commonly used. These are sweep representation, B-Rep (boundary representation) and CSG (constructive solid geometry). B-Rep requires the creation of surface models and their conversion to solids. However, the methods by which surface models are created require a very high level of human interaction. A major disadvantage of using CSG is the limited number of primitives that act as the building blocks for the model. It is difficult for a designer to visualize complex objects in terms of these primitives and even harder to model. Hence, CSG often requires complex, tedious and time-consuming operations.

1.1.3 Trends in CAD/CAM system development

The ease with which geometric models can be created governs the speed of design. In trying to develop more sophisticated applications algorithms delivering higher levels of automation, it becomes clear that better schemes for geometric modeling and geometry representation must be devised. These schemes must overcome the deficiencies of many current modeling systems. Because design is iterative in nature, the topology, geometry or dimensioning of a geometric model must be modified many times during the design cycle. However, the effectiveness of future CAD/CAM systems will depend not only upon the ease with which geometric models can be created and modified, but also on the availability of convenient user

interfaces for automatic analyses, design evaluations, re-design suggestions and generation of manufacturing information.

This has led to research activity in feature-based geometric modeling systems which will contain more effective procedures for definitions and modification of geometric models. It is generally agreed that commonly used geometric primitives (like spheres, cylinders, cubes) are not a sufficiently rich vocabulary with which to describe engineering components. Designers use a richer vocabulary of higher level descriptors (like slots, keyways, chamfers, fillets) to convey information about geometry and topology of the component. It is the aim of feature-based modeling to allow engineers to design parts and assemblies by specifying readily understandable features or to extract feature information from an existing model. Thus common entities like holes, grooves, notches and bosses can be called for and manipulated without referring to individual geometric elements of which they are composed. A set of commonly used engineering shapes and features could be used as building blocks in creating parts or assemblies. The use of such higher level descriptors to convey information about the geometry and topology of a part allows both designers and production engineers to perform their separate tasks using a common language (of feature-based vocabulary) and to use the same solid model from conceptual design to the finished part. Ultimately, a feature model should provide sufficient information to drive NC tools such as mi g machines, lathes and drill presses that make the part.

In addition to easing and speeding the modeling process, feature-based modeling systems are expected to have the highest level of integration of CAD and

CAM as yet. It will only be reasonable that such systems, in coming years, may replace solid modeling as a tool for geometric modeling.

1.1.4 Advantages of Feature Based Modeling

A major disadvantage of conventional CSG solid modelers is the availability of only a limited number of primitives that can be used in creating a part model. It is not easy to visualize complex shapes and even harder to model using these primitives. The availability of a large number of pre-defined generic features in a feature modeler will not only make the task of modeling easier but also faster, since they will eliminate several steps that are generally required to create a feature from basic primitives. Solid modelers are not capable of interactively changing a design. The use of different techniques such as parameterization of feature-based models may allow generation of variations of a basic design by simply changing these parameters. This will help users to understand the engineering relationship within their design and those that exist between different parts of an assembly. Perhaps the most important advantage of a feature-based design system is that all essential geometric and manufacturing information can reside within each feature. These features could then be used to perform design analysis and to recognize the kinds of machines and processes that are needed to produce the part.

1.2 Literature Review

Since one of the objects of this research is to develop a new technique to create and store geometric models, it is necessary to review work done by different people in addressing various issues of geometric modeling and improvement techniques in CAD/CAM. The papers discussed in this review are grouped according to the issues they address.

1.2.1 Survey Papers in Modeling

The following papers review the growing interest in feature-based modeling. Brody (1987) provides a good introduction to feature-based modeling, plus a preliminary study of research by individuals and corporations in that area. Woo (1977) discusses the problems and techniques of shape description for mechanical components. Allen (1984) describes some schemes used for representing geometry on solid modeling systems. He also describes some construction and editing techniques, which may be used to create geometry. Pratt (1988) has done a survey of different approaches to feature-based modeling techniques and has tried to find an optimal approach to design with features and interfacing to automated applications.

1.2.2 Improvement Techniques in CAD

Work on improving techniques in CAD has been done by several researchers. Hatvany (1984) identified knowledge-based techniques as the most important tool for CAD systems of the future. Davidson (1985) has drawn up a list of requirements for a CAD decision-making tool. Serrano and Gossard (1986) have developed interactive software which integrates mathematical models with solid models to aid in the preliminary design of mechanical components.

In order to combine the use of expert systems with geometric modeling, Sankar and Myklebust (1989) have developed a prototype expert system for automatic generation of interfaces between design application programs and CAD/CAM systems. New computer languages are being developed by researchers to improve the current techniques in CAD. Lai (1987) and Wilson (1987) have developed a "Function-Description" language which describes the function of relationships between elements in a mechanical design. Takase and Nakajima (1984) developed a "Feature-Description" language for describing and modeling assembled machines.

1.2.3 Feature Based Geometric Modeling

The area of feature-based geometric modeling is relatively new and the literature is somewhat sparse, compared to information available regarding improvement of techniques used in CAD/CAM. Significant work in feature-based modeling has been done by Luby, Dixon and Simmons (1986). They have created and used a features database for evaluation of manufacturability of castings. This

database, however, appears to consist of a relatively small number of features. They also do not mention whether their program performs addition and subtraction of the features being added to the component. Dixon and Simmons (1986) have worked in the area of creating a features database and talk about the use of such databases. Dixon and Dym (1986) have done some work on geometric representation and reasoning in design. Libardi *et al.* (1986) have tried to apply feature-based modeling techniques to extrusions. Nnaji and Vishnu (1987) have investigated methods for representing fundamental geometric data and for developing an algorithm for automatic conversion of wire-frame data to solid geometry. Light and Gossard (1982) have developed flexible procedures for definition and modification of geometric models. Gossard, Zuffante and Sakurai (1988) have treated dimensions, tolerances, and features in solid models. Dimensions are represented by a relative position operator and changes in dimensional values automatically translate into changes in geometry and topology. More recently, Cunningham and Dixon (1988) have investigated the origin of features and their role in design and manufacturing activities. However, rules and guidelines for such process-activities generate a very limited set of features. Tikerpuu and Ullman (1988) have established representations for a general description of a design object's form and functions in terms of the state of design. Shah and Rogers (1988) have grouped product information into sets based on the meaning of the information. This helps in the automation of engineering tasks. Shah, Bhatnagar and Hsiao (1988) have developed a feature mapping and application shell by implementing specific applications such as Group Technology coding and Manufacturing Process selections.

It is apparent that although there is a lot of interest in developing feature-based modeling techniques, there is little consensus on how to approach the problem. It is

also apparent that researchers have selected their own way of formulating such techniques and that no comprehensive solutions have surfaced.

1.2.4 Surface Modeling Techniques

Geometric modeling and its techniques have evolved over the last 15 years. This documentation covers the formulation of different kinds of curves and surfaces, their approximations, subdivisions, placement of knots, smoothing and parameterization. A variety of information can be found on this subject. Of particular interest is the work on B-spline curves and surfaces by Riesenfeld (1973) and Versprille (1975). Both have used B-spline surfaces for computer-aided design applications. Tiller (1983) and Piegl (1987) have used rational B-splines for representing quadric primitives and free-form surfaces. Formulations for different modeling schemes and use of different surfaces are well documented by Mortenson (1985), Faux and Pratt (1979), Barsky (1988), Bartels *et al.* (1988), Barnhill and Boehm (1983), Greville (1969), Schoenberg (1969), Brodlie (1980), Gardan (1985) and Bezier(1986).

1.3 Problem Definition

Most research into the issues of feature representations has been in the area of feature extraction from existing CSG or boundary representation models, geometric tolerances and manufacturing evaluation of designs. Little thought has

been given to designing with features from the outset, and the manner in which they are defined or created. Current methods for feature definitions are application specific and do not provide designers with generalized procedures for feature creation. In addition, these procedures can be applied to only a limited number of features and hence restrict the ability of a designer to produce detailed models. Hence, development of generalized schemes to create shapes and features is necessary.

A designer's understanding of a design feature originates from its English and technical definitions. It forms the basis by which he describes a mechanical assembly. He uses simple English words to convey information about the part's size, shape and functions. Since designers find it convenient to communicate in this manner, the use of English words for feature creation and representation is a logical move towards easing the task of feature creation.

Every mechanical part consists of a unique set of features. Such sets may be similar for some parts, but very rarely the same. In many cases this set must be modified for appropriate use. Such modifications or shape alterations are further described by additional English words. Thus the vocabulary involved in the complete description of features is very large. The primary intention of this work is to lay a foundation for definition and creation of a geometric model using features and to identify an approach that makes best use of the vast vocabulary among designers. Some objectives of this research were:

1. To create an exhaustive library of design features based on English, technical and non-technical terms for shape and form together with the vocabulary for their modification and use.

2. To classify the features by form and topology.
3. To define the features parametrically.
4. To produce suitable feature surface models based on incomplete specification of parameters.

This results in several areas which must be addressed individually and in detail.

1.3.1 A Feature Dictionary and Library

Although many people talk about designing with shapes and features, there has been no exhaustive research on a feature vocabulary which relate the English definition of a feature to its mathematical characterization. The first aim of this research is to create an exhaustive dictionary of geometrical terms that relate to features. This dictionary will contain English definitions of the geometrical terms and will contain information on their synonyms. For ease in feature selection and modeling, features will be classified into groups according to topology and form. This vocabulary will provide a base for further research.

1.3.2 Feature Parameterization

A feature model that can be used with different objects and perform different functions must have versatility and must be able to adapt to changes in design. Any change should be accomplished quickly and with ease. More importantly, this change must be reflected in all related dimensions of the feature and other

components that are related to it. Parameterization of models meets these requirements and allows the feature to have different dimensions, orientations and variable relative dimensions. Each topology group will be parameterized and will be assigned a set of parameters that can be used to create and modify a feature model.

1.3.3 Feature Definition through Incomplete Specification

Depending on its function or application, a feature may have different dimensions. However, these dimensions may be governed by different mathematical rules resulting from existing geometric constraints or those specified by a designer. Shape altering words can be used to change mathematical rules that apply to a feature. A feature must satisfy all these rules but may have any dimensions within the constraints of its definition and mathematical rules. One can take advantage of such rules and use them effectively in creation of feature models based on incomplete specifications of parameters. A methodology to solve for unspecified parameters based on mathematical inequalities will be developed and presented.

1.3.4 Geometric Modeling of Features

Another objective of this research is to mathematically map the verbal and mathematical description of each feature or a class of features to a surface description. Geometric models would be based on parametric definition of the feature class. A method will be developed that will use the points computed from the

parametric definition of a feature and interpolate them to produce uniform rational B-spline surface models that pass through these points.

1.3.5 Designer-Oriented Modeling Language

Conventional geometric modelers are menu driven, require large amounts of input data and do not allow data transfer between two different programming environments. A designer-oriented modeling language, based on simple English syntax, will be devised to specify procedures to be executed by the modeler in order to create features using minimal user input. One major advantage of this will be to allow the modeler to operate in other programming environments.

1.3.6 Other Support Work

A high-level environment will be created to impart some degree of intelligence to the modeler. Rules will be used to create an inheritance network (to determine parameter values inherited from a parent feature) and cognition rules (to see if a feature-parameter is used in a valid manner). A modeling environment for creating, modifying and deleting features, will be created. Object-oriented programming techniques will be used to support user definition of form features that can be positioned and manipulated in a logical manner. The IBM version of PHIGS, graPHIGS, will be used to provide device independent graphics support.

1.4 Definition of a Feature

Feature-based modeling has applications in both design and manufacturing cycles of product development. In design it can be used for object modeling, mass property analysis, interference checks and failure analysis. In manufacturing it has applications in process planning, manufacturability evaluation, cost analysis, etc. The most general definition of a "feature" is "any entity, geometric or otherwise, that influences any step of the product development cycle". However, for practical purposes, a "feature" definition varies with the application. Most applications do not consider both aspects of feature design. Thus most definitions of a "feature" are limited to the particular application for which it is being used. Cunningham (1988) defines a feature as "a higher order abstract geometric form or entity that is used in reasoning about topology and geometry". Shah (1988) defines features as "recurring patterns of information related to a part's description". The scope of this research is limited to use of features in geometric modeling. It does not take into consideration any manufacturing features. For this work, a feature will be defined as **a geometric entity that defines the attributes of a part's nominal size, geometry, shape and topology.** The emphasis will be on geometric subsets of a part that can be recognized and readily specified by a designer.

Chapter 2: A Feature Dictionary and Library

2.1 Feature Vocabulary

To ease modeling, a designer should be allowed to create parts the same way he perceives them - as features. The geometric reasoning process that a designer uses in visualizing a conceptual design should also be reflected in the modeling process. For most engineering applications, the conceptual design consists of complex shapes and features. The vocabulary used to describe such complex features is large and varied. If a designer is to efficiently describe a geometric part in terms of features, a rich set of feature descriptors is necessary. This means that the set of features available to a designer must be a complete and natural set. If the list is complete, a designer must be able to completely describe a part by a set of features that form the list. In addition, any word that a designer uses to describe shape or form should be a feature or a synonym of a feature. Hence, an exhaustive study of English words describing shape and form of engineering features was done to establish a wide vocabulary for feature-based description. This 373 word set

should account for almost all feature shapes and form required to model an engineering part. To help a designer decide which feature is best suited for a specific application, each feature has been described by means of its English, technical or non technical definitions. These will help him better understand the feature's functions and characteristics. For example, a hole is defined as a cavity or aperture in a solid mass or body and performs the function of creating a hollow space in the part. A wedge is a V-shaped piece of solid material and is used in holding different components of a part in place. Many features possess similar characteristics and can be used to perform similar functions. Such features are synonyms of each other and can be used interchangeably without any change in the model geometry. Such closely related features have been placed in synonym lists and give a designer more choice of features for specific applications. For example, the features packing, pad, padding and wadding are synonyms of each other. A designer may choose to use one to perform the function of cushioning. However, if he is unaware of the function of a packing, there is a possibility that he knows the meaning of one of its synonyms. A designer should now be able to describe any part with a set of features or shapes from this vocabulary. Tables 1 and 2 contain the complete list of all the features that have been considered for this work. This list and the definitions were compiled using various sources. Some notable references were Oberg *et al.* (1988), Parker (1989), French and Svensen (1966), Yaslow (1969), Horner and Abbey (1952), Del Vecchio (1962), Nayler and Nayler (1967) and Giachino and Beukema (1964).

Table 1. List of Features - I.

1. abutment
2. aiguille
3. angle
4. annular-hole
5. aperture
6. apex
7. apron
8. arbor
9. arbor-hole
10. arc
11. arch
12. arm
13. arris
14. astragal
15. axle
16. baffle-plate
17. ball
18. band
19. bar
20. barrel
21. base
22. basket
23. bead
24. beam
25. bed
26. bedding
27. belt
28. bench
29. bend
30. bevel
31. billet
32. block
33. bolt
34. bore
35. boss
36. bow
37. box
38. brace
39. bracket
40. break
41. brick
42. broach
43. bubble
44. bucket
45. bulb
46. bulge
47. bullet
48. bump
49. bushing
50. button
51. cable
52. cage
53. cam
54. can
55. canal
56. cantilever
57. cap
58. capsule
59. cartridge
60. case
61. casing
62. cave
63. cavity
64. cell
65. chamber
66. chamfer
67. channel
68. chase
69. chased-groove
70. chassis
71. chimney
72. chord
73. chute
74. cleat
75. cleavage
76. cleft
77. clevis
78. clip
79. clough
80. coil
81. coin
82. collar
83. collet
84. column
85. compartment
86. cone
87. conic
88. conical
89. constriction
90. convex
91. cord
92. core
93. corner
94. cotter
95. counter-bore
96. counter-sink
97. cover
98. crack
99. crater
100. crescent
101. crevice
102. crown
103. cube
104. cup
105. curl
106. curve
107. cusp
108. cut
109. cylinder
110. deep
111. deflation
112. dent
113. depression
114. depth
115. dial
116. diaphragm
117. dilation
118. dip
119. dish
120. disk
121. ditch
122. dome
123. donut
124. doughnut
125. dove-tail
126. dowel
127. drift-pin
128. drill
129. drop
130. droplet
131. drum
132. duct
133. dune
134. edge
135. elbow
136. ellipse
137. ellipsoid
138. emboss
139. entasis
140. face
141. fence
142. fillers
143. fillet
144. filling
145. film
146. filter
147. fin
148. fissure
149. flake
150. flange
151. flue
152. flume
153. flute
154. foil
155. fold
156. fracture
157. frame
158. furrow
159. gap
160. gasket
161. gauze
162. gear
163. gib
164. gland
165. globe
166. globule
167. gorge
168. grain
169. grid
170. grill
171. groove
172. guide
173. gusset
174. gutter
175. hand-wheel
176. hatch
177. helix
178. hem
179. hemisphere
180. hex-bolt
181. hex-nut
182. hole
183. hollow
184. hood
185. hub
186. hump
187. impression
188. inlet
189. journal
190. keel
191. kerf

Table 2. List of Features - II.

192. key
193. keyhole
194. keyseat
195. keyway
196. knob
197. knurl
198. ladle
199. land
200. leaf
201. ledge
202. leg
203. line
204. lip
205. lobe
206. loop
207. lug
208. mandrel
209. manifold
210. mask
211. mass
212. mat
213. membrane
214. mesh
215. messenger
216. mosaic
217. nail
218. neck
219. needle
220. nest
221. net
222. nile
223. nipple
224. node
225. notch
226. nozzle
227. nut
228. o-ring
229. opening
230. orifice
231. oval
232. packing
233. pad
234. padding
235. pail
236. pallet
237. pan
238. parallelopiped
239. parting
240. partition
241. patch
242. pattern
243. peak
244. peen
245. peg
246. pellet
247. perforation
248. perimeter
249. pillar
250. pin
251. pipe
252. pit
253. plane
254. plate
255. plug
256. plunger
257. pocket
258. pod
259. point
260. pot
261. pouch
262. prism
263. probe
264. projection
265. prominence
266. protrusion
267. protuberance
268. pulley
269. puncture
270. pyramid
271. rabbet
272. race
273. rail
274. raise
275. ravine
276. ream
277. receptacle
278. recess
279. reel
280. rib
281. ribbon
282. ridge
283. rifle
284. rim
285. ring
286. rivet
287. rod
288. roller
289. rope
290. round
291. rounding
292. rung
293. rupture
294. rut
295. scar
296. scoop
297. screen
298. screw
299. seal
300. seam
301. seat
302. separation
303. shaft
304. sharp
305. sheave
306. sheet
307. shell
308. shield
309. shim
310. shoulder
311. sieve
312. slab
313. sleeve
314. sleeving
315. slit
316. slot
317. socket
318. sphere
319. spike
320. spindle
321. spiral
322. spline
323. split
324. split-ring
325. spoke
326. spool
327. spot-face
328. spring
329. stick
330. stiffener
331. stopper
332. string
333. strip
334. stud
335. stuffing
336. swell
337. swelling
338. t-bolt
339. t-slot
340. table
341. tack
342. tang
343. tank
344. taper
345. tapped-hole
346. tear
347. telescope
348. tetrahedron
349. thickening
350. thread
351. tile
352. torus
353. track
354. trepan
355. tube
356. tunnel
357. twine
358. twist
359. undercut
360. valley
361. vane
362. vein
363. vent
364. vertex
365. void
366. wadding
367. washer
368. web
369. wedge
370. weld
371. well
372. wheel
373. wire

2.2 *Feature Parameterization*

There is a need to develop a methodology that will allow easy definition and creation of features. Since designers find it easier to visualize a feature in terms of its parameters or dimensions, a method that allows definition of features in terms of their parameters would be desirable. Such parametric definitions of feature models have distinct advantages since many features have regular and symmetrical shapes which are dependent on these sets of parameters. Often these parameters represent direct geometrical or manufacturing information with only a small change in interpretation. For example, a 10-mm hole in design is translated into a manufacturing process (drilling) that requires a 10-mm drill bit. Defining a feature using such parameters will ultimately help in process planning procedures. Parametrics relate feature parameters directly to parametric equations defining the feature and indirectly to its surface characterization. This link not only allows quick creation of models but also easy and efficient modification of previously created features. Parameterization helps capture the intent of a designer and allows creation of geometric relations between different features in a part. In many cases changes in even one parameter affect values of related parameters, and each change is quickly transmitted through the entire model. For example, when a designer creates a 'through hole' in a plate, he creates a geometrical relation that requires that the length of the hole be equal to the thickness of the plate. If the designer later modifies the plate thickness, the hole dimensions will have to automatically adjust to reflect that change. Many designs vary only in a few parameters. They are basically the same part, with some size variations. With parametric modeling in effect, instead of producing separate designs for each part, only one design is necessary to design the

entire group. This significantly increases productivity by making the design cycle faster and less expensive. Use of parametrics also increases the ability of translating files between one CAD system and another. This is hence, a logical development to promote standards for such movement interfaces.

Parameters are prime factors in deciding the final shape or topology of a feature or class of features. Any group of features that possess a similar topology can be defined by the same or similar set of parameters. Once groups have been formed, it is straightforward to deduce and assign a set of parameters that can be used to represent each feature group. Such grouping and classification techniques are discussed in the next section. However, since one classification scheme uses topology as the basis of grouping, it makes the job of assigning feature parameters a little easier. Each topology group has been assigned parameters that can be used to define and create any feature that has been classified under that group. Figures 1 to 5 show each topology group, parameters that define the group and some example features from each group. Positive Boolean features are represented by solid lines whereas negative Boolean features are represented by dotted lines. Some features have specialized shapes and cannot be grouped into a general category. Such features can still be parameterized. Figures 6 and 7 show parameters defining some such features. Although most features can be grouped into one parameter or topology group, some can have various profiles and cross-sections that are described by different parameters. For example, a bar can have a circular cross-section (defined by radius) or a rectangular cross-section (defined by length and breadth). It is therefore necessary to parameterize different profiles. Some common profiles and their defining parameters are shown in Figure 8.

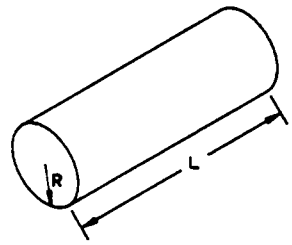
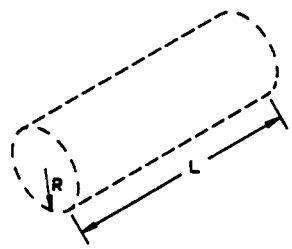
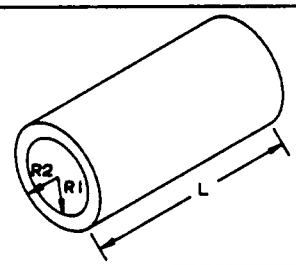
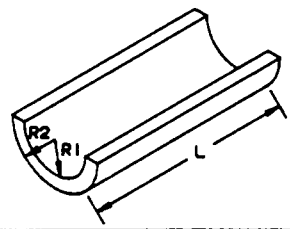
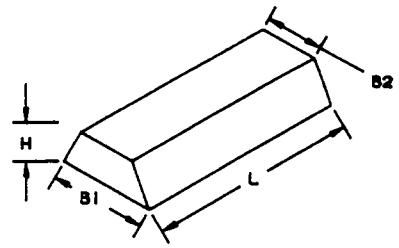
TOPOLOGY GROUP	EXAMPLE FEATURES	PARAMETERS TO DEFINE GROUP
TOPOLOGY - A	AXLE, BAR, CYLINDER, DISK, PLATE	
TOPOLOGY - B	APERTURE, BORE, HOLE, ORIFICE, REAM	
TOPOLOGY - C	BUSHING, CLIP, GLAND, RING, WASHER	
TOPOLOGY - D	ARC, ARCH, CAVE, RACE, TUNNEL	
TOPOLOGY - E	BAR, BEAM, BILLET, BOSS, CUBE	

Figure 1. Parameters defining Major Topology Groups - I.

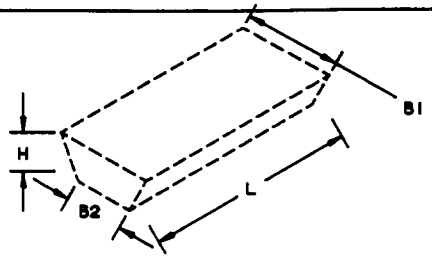
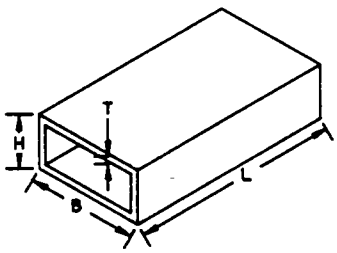
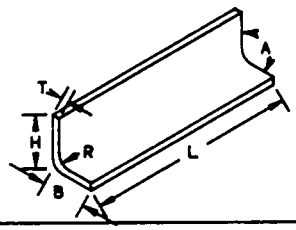
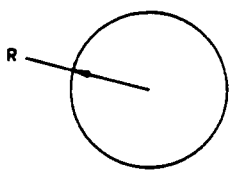
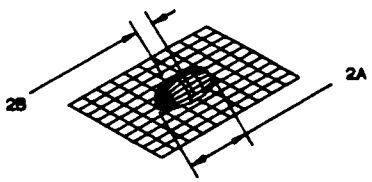
TOPOLOGY GROUP	EXAMPLE FEATURES	PARAMETERS TO DEFINE GROUP
TOPOLOGY - F	CHANNEL, GROOVE, KEYWAY, POCKET, SLOT	
TOPOLOGY - G	CASE, CELL, CHAMBER, FRAME, SHELL	
TOPOLOGY - H	ANGLE, BEND, BRACKET, CORNER, ELBOW	
TOPOLOGY - I	BALL, BULB, CROWN, GLOBULE, SPHERE	
TOPOLOGY - J	BUMP, CUSP, HUMP, PROTRUSION, SWELLING	

Figure 2. Parameters defining Major Topology Groups- II.

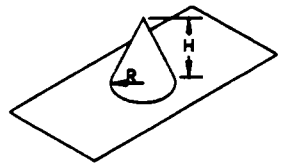
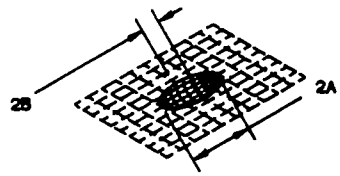
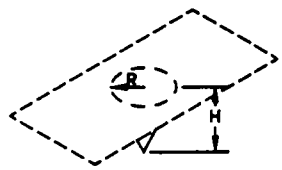
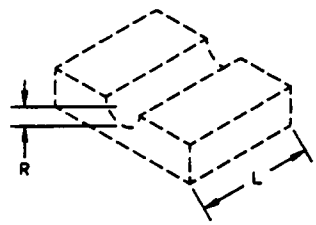
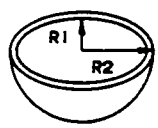
TOPOLOGY GROUP	EXAMPLE FEATURES	PARAMETERS TO DEFINE GROUP
TOPOLOGY - K	APEX, PEAK, SHARP, SPIKE, VERTEX	
TOPOLOGY - L	CRATER, DEFLATION, DENT, DEPRESSION	
TOPOLOGY - M	DIP, NOTCH	
TOPOLOGY - N	FLUTE, RACE	
TOPOLOGY - O	CAP, CUP, DOME, HEMISPHERE	

Figure 3. Parameters defining Major Topology Groups - III.

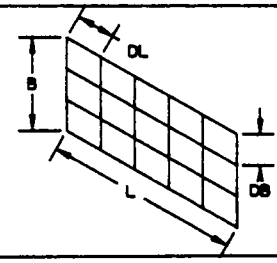
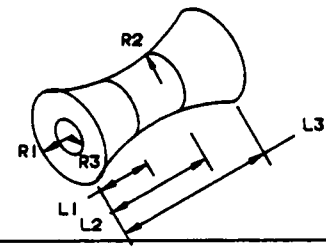
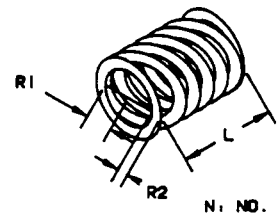
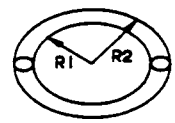
TOPOLOGY GROUP	EXAMPLE FEATURES	PARAMETERS TO DEFINE GROUP
TOPOLOGY - P	GRID, MESH, NEST, NET	
TOPOLOGY - Q	PULLEY, REEL, SPOOL	
TOPOLOGY - R	CURL, HELIX, SPIRAL, SPRING	
TOPOLOGY - S	DONUT, RING, TORUS	

Figure 4. Parameters defining Major Topology Groups - IV.

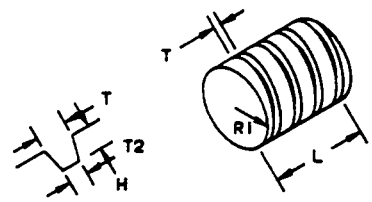
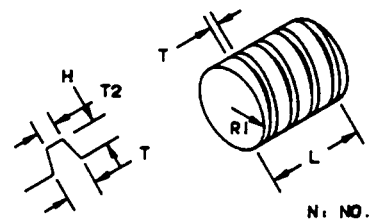
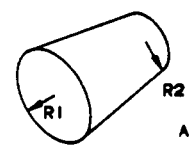
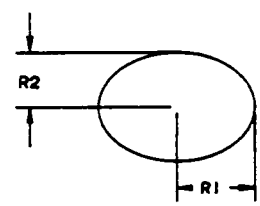
TOPOLOGY GROUP	EXAMPLE FEATURES	PARAMETERS TO DEFINE GROUP
TOPOLOGY - T	RIFLE, TAPPED-HOLE	
TOPOLOGY - U	SCREW	
TOPOLOGY - V	BEVEL, CONE, COTTER, TAPER	
TOPOLOGY - W	ELLIPSE, ELLIPSOID, OVAL	

Figure 5. Parameters defining Major Topology Groups - V.

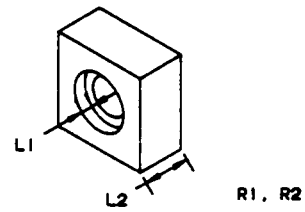
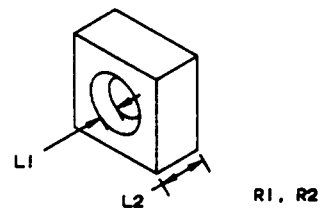
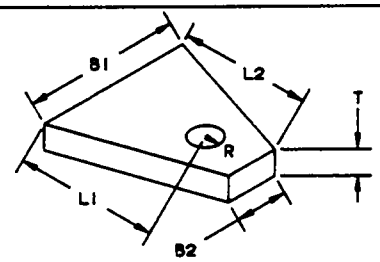
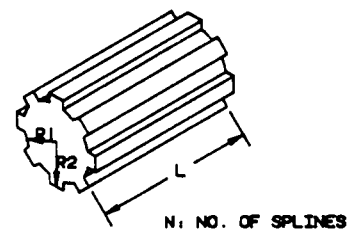
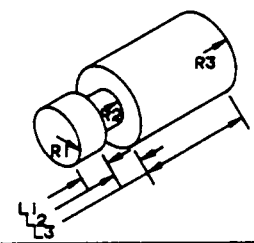
OTHER FEATURES	PARAMETERS TO DEFINE FEATURE
COUNTER-BORE	
COUNTER-SINK	
FLANGE	
SPLINE	
NECK	

Figure 6. Parameters defining Other Features - I.

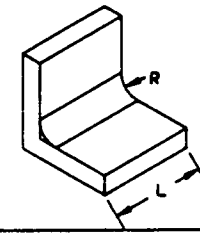
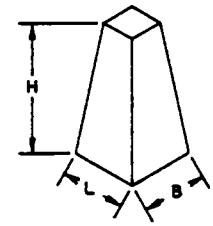
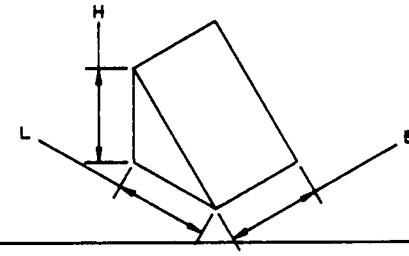
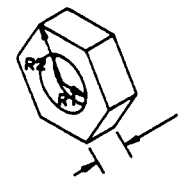
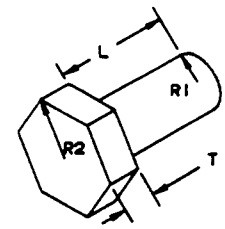
OTHER FEATURES	PARAMETERS TO DEFINE FEATURE
FILLET	
PRISM	
WEDGE	
HEX-NUT	
HEX-BOLT	

Figure 7. Parameters defining Other Features - II.

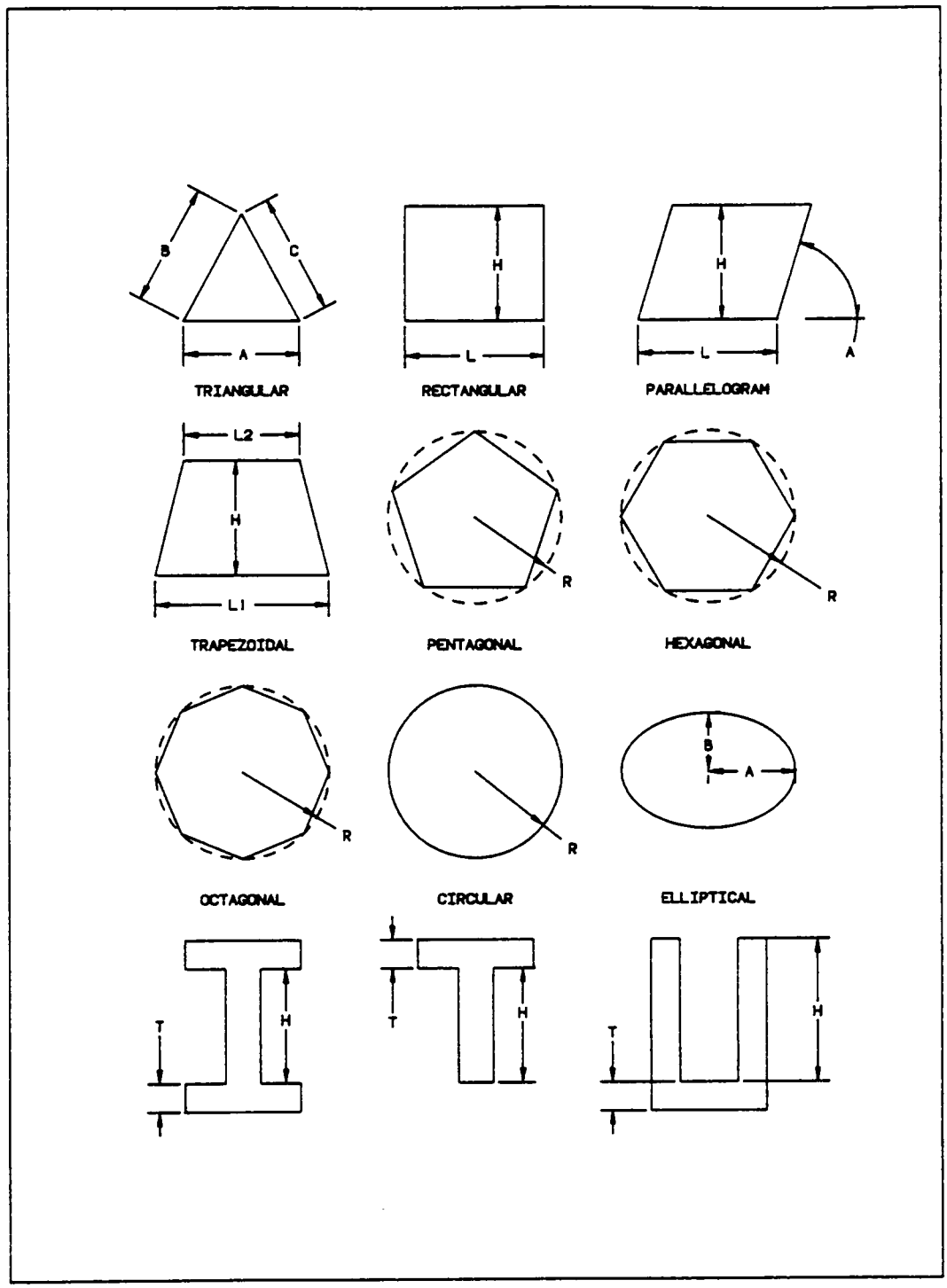


Figure 8. Common Profiles and Parameters Defining Them.

2.3 *Feature Classification*

Although each feature has its own function, many can be described or modeled using the same or very similar set of parameters. For example, a circular bar and a circular plate have different functions, but can be described using the same parameters, for example, radius and length/thickness. On the other hand many features can be used to model a part with specific functional requirements but are described by different parameters. For example, rotundity form can be created by a cylinder or a sphere both of which are described by different parameters. Since the number of features available is large, it is necessary to classify them into groups that will allow the designer to access them intelligently and efficiently. Hence, classification schemes must be devised which will make feature selection and accessing a simple task. The features were classified according to two criteria: topology and shape form. Features under the same topology group are governed by the same set of parameters that provide an easy way of defining them. Classification by shape form will help a designer select a feature that will impart the correct shape to the part. These two schemes will be combined together to allow feature selection based on form and creation based on topology.

2.3.1 Classification by Topology

For the purpose of this work **all physical or natural features (Independent of size) which can be expressed as a set of points possessing similar properties of geometric configurations and have similar structural relationships** will be termed as

topologically similar. Features that can be described by a similar set of parameters are classified under different groups which in turn are defined or described by parametric equations. Due to the nature of these groups, all features within a particular group would possess a similar topology. This makes implementation of modeling schemes straightforward, since a large number of features can be described using a single set of parametric equations. A detailed discussion about parameterizing each group has been presented earlier. The 23 major topological or parametric groups represent both positive and negative Boolean features. For example, groups A and B are both defined by parameters for radius (R) and length/thickness (L). However, both topologies represent different characteristics that will be imparted to the object that is being modeled. 'Boss' which is a part of group A is a positive feature and will act as a projection or a protrusion whereas a blind 'hole' classified under group B is a negative feature and will create a depression in the object. As discussed earlier, some features have different profiles and must be classified under more than one topology group. In such cases, correct interpretation of its topology must be arrived at by means of the shape profiles or shape descriptors. Such descriptors are discussed later in the chapter. Tables 3 through 7 contain the complete classification of the feature list by topology.

2.3.2 Classification by Form

Frequently a part that is being modeled has a specific geometric function. This function may put constraints on the shape-form which features that form the part may have. Although, many features have the capability of imparting the required form to the part, a designer should be given a wide choice to choose the most suitable

Table 3. Feature Classification by Topology - I.

Topology A

1. aiguille
2. arbor
3. axle
4. bar
5. beam
6. boss
7. broach
8. bullet
9. button
10. cable
11. cap
12. chord
13. cleat
14. coin
15. column
16. cord
17. core
18. cover
19. cylinder
20. dial
21. diaphragm
22. dish
23. disk
24. dowel
25. drift-pin
26. drill
27. filling
28. filter
29. fin
30. hatch
31. key
32. leg
33. mandrel
34. messenger
35. nail
36. needle
37. packing
38. pad
39. padding
40. peg
41. pellet
42. pillar
43. pin
44. plate
45. plug
46. plunger
47. rail
48. rod
49. roller
50. rope
51. seal
52. shaft
53. spindle
54. spoke
55. stick
56. string
57. stud
58. tack
59. twine
60. wadding
61. wheel
62. wire

Topology B

1. aperture
2. arbor-hole
3. bore
4. broach
5. constriction
6. dilation
7. hole
8. inlet
9. keyhole
10. opening
11. orifice
12. pan
13. perforation
14. puncture
15. ream
16. scar
17. well

Topology C

1. annular-hole
2. baffle-plate
3. barrel
4. basket
5. bucket
6. bushing
7. can
8. capsule
9. cartridge
10. chimney
11. clip
12. collar
13. drum
14. duct
15. filling
16. flue
17. gasket
18. gland
19. hub
20. journal
21. lip
22. loop
23. manifold
24. nipple
25. nozzle
26. o-ring
27. packing
28. pad
29. padding
30. pail
31. perimeter
32. pipe
33. probe
34. rim
35. ring
36. sleeving
37. split-ring
38. stopper
39. tube
40. vein
41. wadding
42. washer

Table 4. Feature Classification by Topology - II.

Topology D

1. arc
2. arch
3. bow
4. cave
5. crescent
6. curve
7. race
8. tunnel

Topology E

1. abutment
2. apron
3. baffle-plate
4. band
5. bar
6. base
7. beam
8. bed
9. bedding
10. belt
11. bench
12. billet
13. block
14. boss
15. cantilever
16. cap
17. cleat
18. column
19. core
20. cover
21. cube
22. diaphragm
23. face
24. filling
25. film
26. filter
27. fin
28. flake
29. foil
30. fold
31. gib
32. guide
33. gusset
34. hatch
35. hood
36. keel
37. key
38. leaf
39. leg
40. mask
41. mat
42. membrane
43. packing
44. pad
45. padding
46. pallet
47. partition
48. patch
49. plane
50. plate
51. rail
52. ribbon
53. rod
54. seal
55. seat
56. sheet
57. shield
58. slab
59. spoke
60. strip
61. stuffing
62. table
63. thickening
64. tile
65. track
66. vane
67. wadding

Topology F

1. canal
2. cavity
3. channel
4. chased-groove
5. chute
6. cleavage
7. cleft
8. clough
9. collet
10. crack
11. crevice
12. cut
13. deep
14. depth
15. ditch
16. fissure
17. flume
18. fracture
19. furrow
20. gap
21. gorge
22. groove
23. guide
24. gutter
25. hole
26. hollow
27. kerf
28. keyhole
29. keyseat
30. keyway
31. notch
32. parting
33. pit
34. pocket
35. pouch
36. rabbet
37. ravine
38. receptacle
39. recess
40. ridge
41. rupture
42. rut
43. scar
44. separation
45. slit
46. slot
47. socket
48. split
49. valley
50. vein

Table 5. Feature Classification by Topology - III.

Topology G

1. box
2. cage
3. case
4. casing
5. cell
6. chamber
7. chassis
8. compartment
9. frame
10. shell
11. tank

Topology H

1. angle
2. arm
3. arris
4. bend
5. brace
6. bracket
7. chamfer
8. corner
9. edge
10. elbow
11. line
12. rib
13. shoulder

Topology I

1. ball
2. bead
3. bubble
4. bulb
5. crown
6. drop
7. droplet
8. globe
9. globule
10. mass
11. pellet
12. pot
13. round
14. rounding
15. sphere
16. tear

Topology J

1. astragal
2. bulge
3. bump
4. chase
5. convex
6. cusp
7. dune
8. emboss
9. entasis
10. grain
11. hump
12. knob
13. ledge
14. lobe
15. nile
16. node
17. pod
18. projection
19. prominence
20. protrusion
21. protuberance
22. raise
23. swell
24. swelling

Topology K

1. apex
2. peak
3. point
4. sharp
5. spike
6. taper
7. vertex

Table 6. Feature Classification by Topology - IV.

Topology L

1. crater
2. deflation
3. dent
4. depression
5. impression

Topology M

1. dip
2. notch

Topology N

1. flute
2. race

Topology O

1. cap
2. cup
3. dome
4. hemisphere
5. ladle
6. scoop

Topology P

1. fence
2. gauze
3. grid
4. grill
5. mesh
6. mosaic
7. nest
8. net
9. pattern
10. screen
11. sieve
12. web

Topology Q

1. pulley
2. reel
3. spool

Topology R

1. coil
2. curl
3. helix
4. spiral
5. spring
6. thread
7. twist

Table 7. Feature Classification by Topology - V.

Topology S

1.	donut	2.	doughnut	3.	ring	4.	torus

Topology T

1.	rifle	2.	tapped-hole

Topology U

1.	screw

Topology V

1.	bevel	3.	conic	5.	cotter	7.	taper
2.	cone	4.	conical	6.	hub		

Topology W

1.	ellipse	2.	ellipsoid	3.	oval

feature for that application. Hence, it is necessary that the available features be classified according to the form that they have. Also, a designer may wish to browse a group of features classified by form, based on the recollection of one feature in that group. Figure 9 shows the general classification scheme adopted for grouping features according to their form. Shape forms can generally be classified into three major groups. The primary or general form group gives a designer a general idea of the presence of form or regularity. In this case all features are classified under presence of form and regularity. The secondary or special form group represents the nature of the whole feature. The tertiary or superficial form group represents the superficial form that will be imparted to the part. Each of these groups is further classified. The special form group is divided into five sub-groups that represent the shape of the feature. One sub-group, circularity, is further subdivided into simple and complex types. The superficial form group is divided into 14 categories, based on the superficial characteristics that a feature may have. Each feature in the list was examined and tested for all groups and subgroups. Some were classified under one group while others were found to be eligible for numerous groups. For example, the feature 'sphere' can be used not only to give rotundity to the part but also simple circularity and convexity. Another example is a hemisphere which can additionally perform the function of imparting concavity to the part. Figures 10 to 14 show some examples in each category. A user may find it easier to understand a shape form based on use of some adjectives or verbs. Thus adjectives or verbs that may represent a shape form are also classified under that shape form. Tables 8 through 13 contain the complete classification of the feature list by shape form.

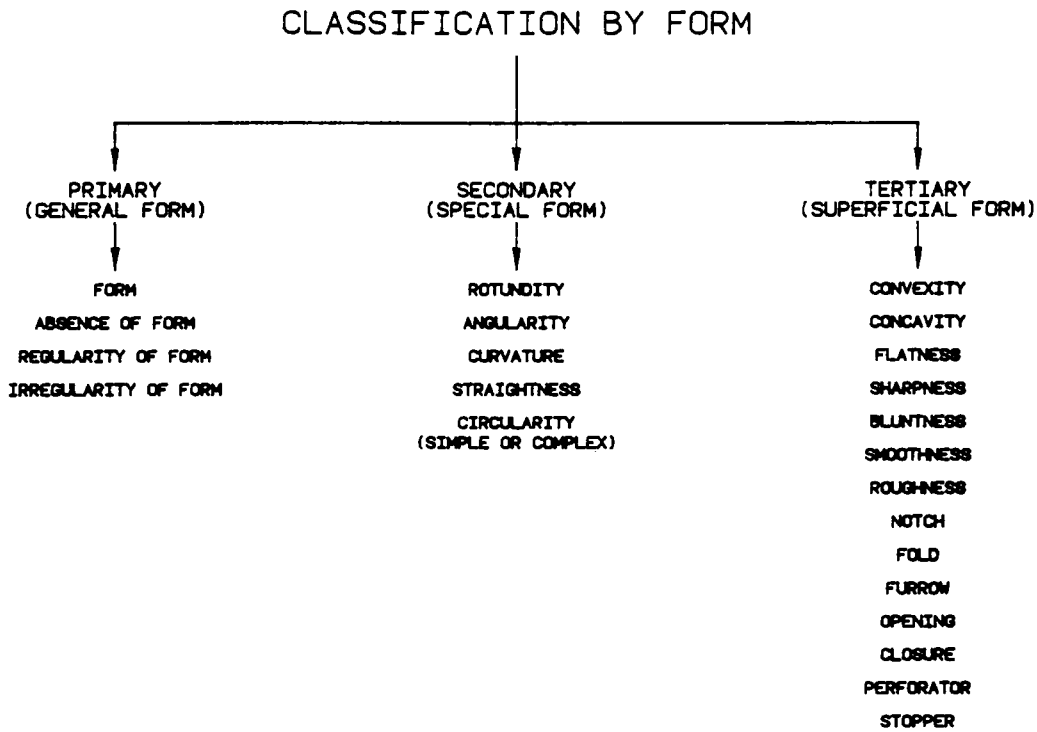


Figure 9. Feature Classification by Shape Form.

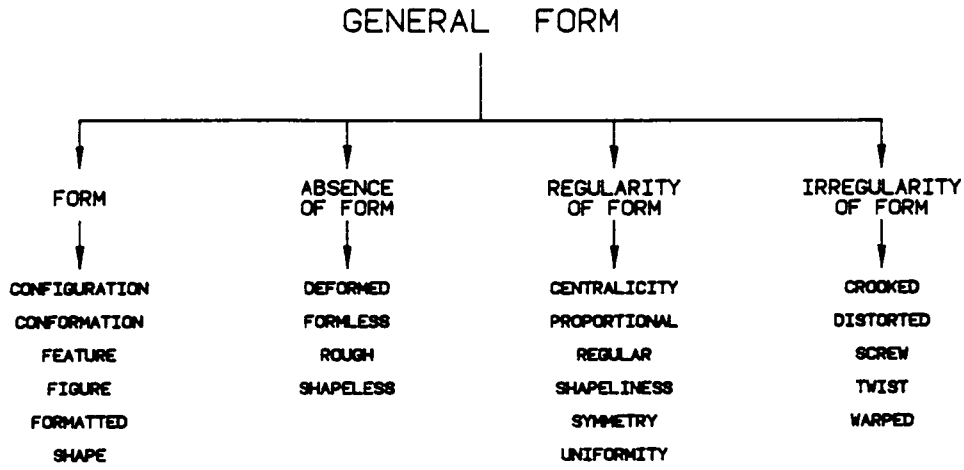


Figure 10. Examples for General Form Features.

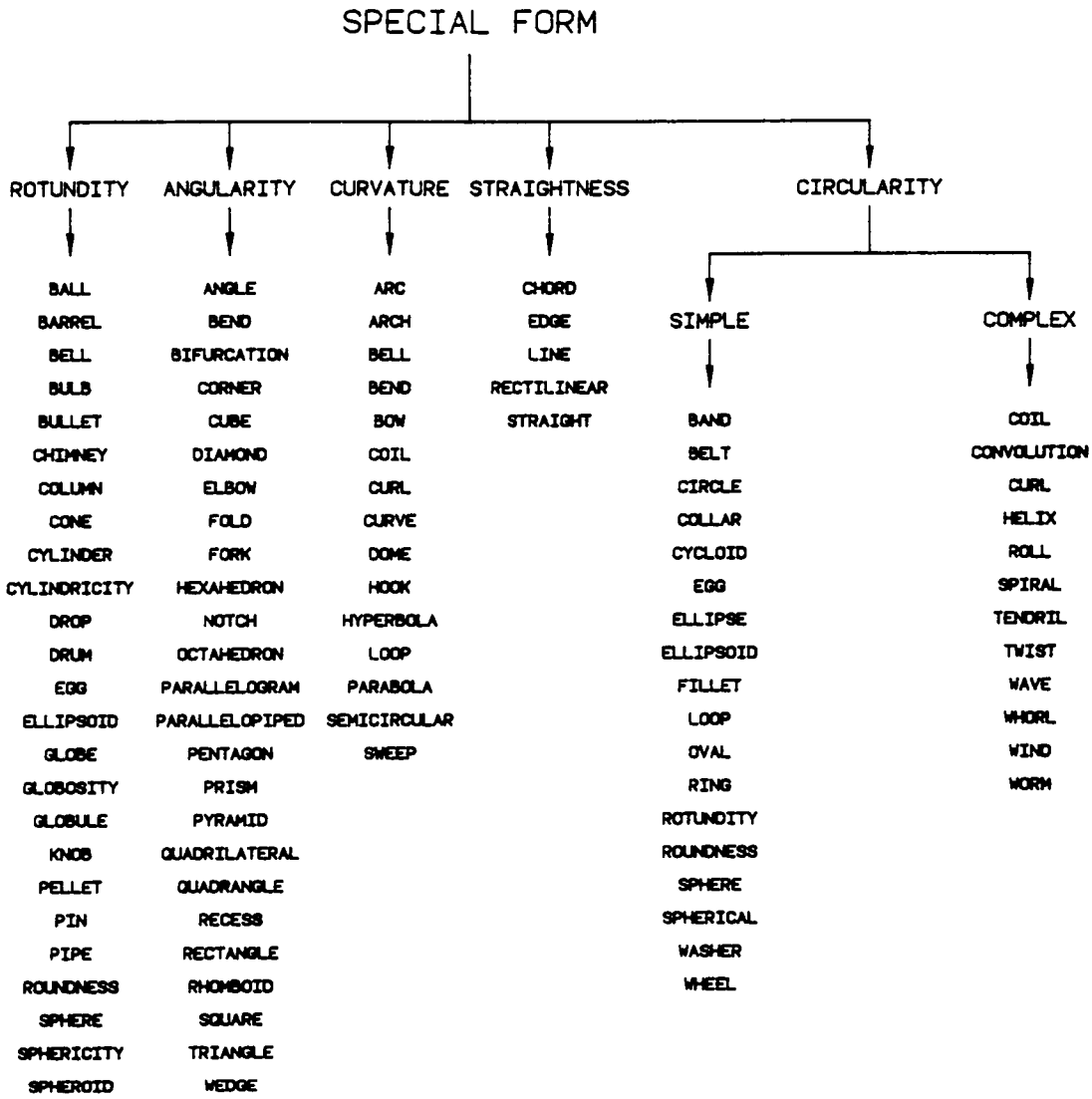


Figure 11. Examples for Special Form Features.

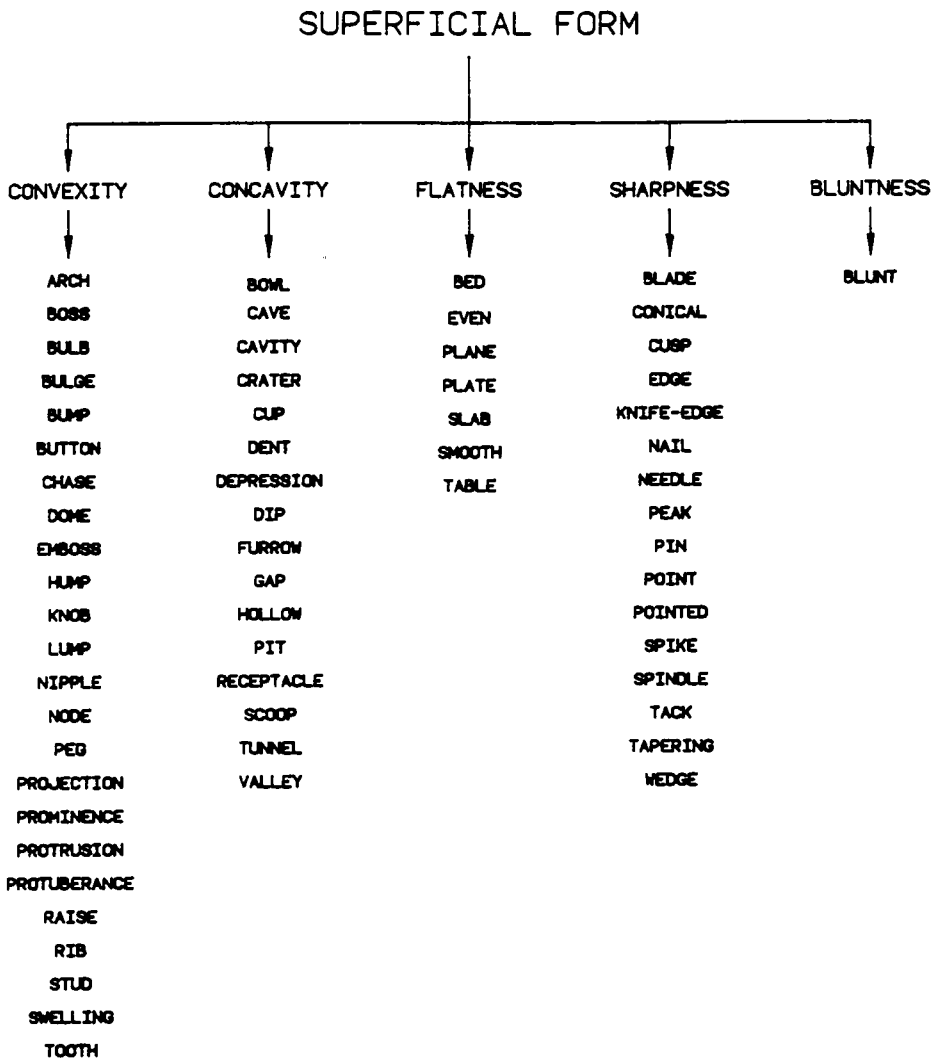


Figure 12. Examples for Superficial Form Features - I.

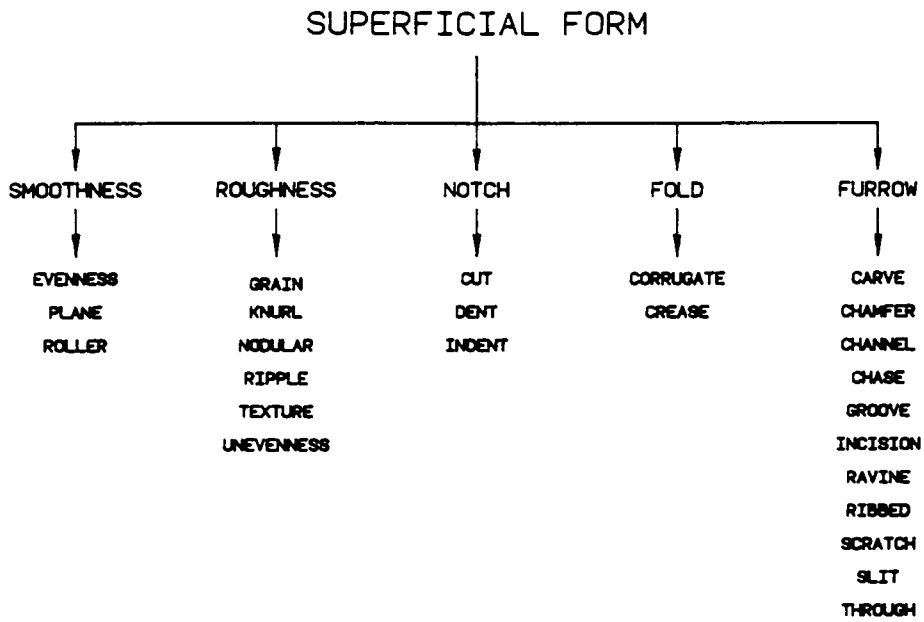


Figure 13. Examples for Superficial Form Features - II.

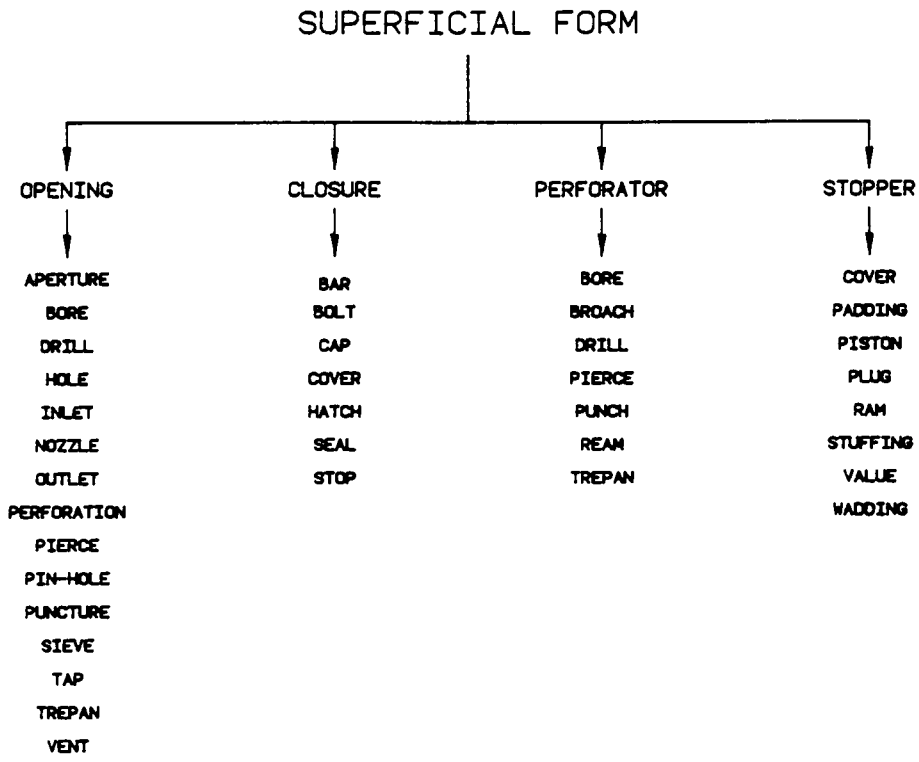


Figure 14. Examples for Superficial Form Features - III.

Table 8. Feature Classification by Form - I.

Form Group Rotundity

1. aiguille
2. annular-hole
3. aperture
4. apex
5. arbor
6. arbor-hole
7. astragal
8. axle
9. baffle-plate
10. ball
11. bar
12. barrel
13. base
14. basket
15. bead
16. beam
17. bevel
18. bore
19. boss
20. broach
21. bubble
22. bucket
23. bulb
24. bulge
25. bullet
26. bump
27. bushing
28. button
29. cable
30. can
31. cap
32. capsule
33. cartridge
34. cave
35. chimney
36. cleat
37. clip
38. coil
39. coin
40. collar
41. column
42. cone
43. conic
44. conical
45. cord
46. core
47. cotter
48. counter-bore
49. cover
50. crown
51. cusp
52. cylinder
53. deflation
54. dent
55. depression
56. dial
57. diaphragm
58. dish
59. disk
60. donut
61. doughnut
62. dowel
63. drift-pin
64. drill
65. drop
66. droplet
67. drum
68. dune
69. ellipse
70. ellipsoid
71. filling
72. filter
73. flake
74. flange
75. flue
76. gasket
77. gland
78. globe
79. globule
80. grain
81. hatch
82. helix
83. hole
84. hump
85. journal
86. knob
87. ledge
88. leg
89. lip
90. lobe
91. loop
92. mandrel
93. manifold
94. mass
95. membrane
96. messenger
97. nail
98. needle
99. nile
100. nipple
101. node
102. o-ring
103. orifice
104. oval
105. packing
106. pad
107. padding
108. pail
109. pallet
110. pan
111. peak
112. peen
113. peg
114. pellet
115. perimeter
116. pillar
117. pin
118. pipe
119. plate
120. plug
121. plunger
122. pocket
123. pod
124. point
125. pot
126. probe
127. projection
128. prominence
129. protrusion
130. protuberance
131. race
132. rail
133. ream
134. receptacle
135. rifle
136. rim
137. ring
138. rod
139. roller
140. rope
141. round
142. rounding
143. scoop
144. screen
145. screw
146. seal
147. shaft
148. shell
149. shield
150. shim
151. sleeving
152. sphere
153. spindle
154. stick
155. stopper
156. string
157. stuffing
158. swell
159. swelling
160. tack
161. taper
162. tear
163. torus
164. tube
165. tunnel
166. twine
167. vein
168. vertex
169. wadding
170. washer
171. well
172. wheel
173. wire

Table 9. Feature Classification by Form Group - II.

Form Group Angularity (Solid)

1. abutment
2. angle
3. apron
4. baffle-plate
5. bar
6. base
7. beam
8. bed
9. bedding
10. bench
11. bend
12. billet
13. block
14. boss
15. canal
16. cantilever
17. cap
18. cavity
19. chamfer
20. channel
21. cleat
22. column
23. core
24. corner
25. cover
26. cube
27. diaphragm
28. elbow
29. filling
30. film
31. filter
32. fin
33. flue
34. foil
35. gib
36. gusset
37. gutter
38. keel
39. key
40. keyhole
41. keyseat
42. keyway
43. leaf
44. leg
45. mask
46. membrane
47. packing
48. pad
49. padding
50. pallet
51. pan
52. partition
53. patch
54. pellet
55. plate
56. pocket
57. rabbet
58. rail
59. receptacle
60. rib
61. rod
62. screen
63. seal
64. seat
65. sheet
66. shield
67. shim
68. shoulder
69. slab
70. slot
71. socket
72. strip
73. stud
74. stuffing
75. table
76. thickening
77. tile
78. track
79. vane
80. vein
81. wadding
82. washer

Form Group Angularity (Hollow)

1. cage
2. case
3. casing
4. cell
5. chamber
6. chassis
7. compartment
8. duct
9. frame
10. shell
11. tank

Form Group Curvature

1. arc
2. arch
3. arm
4. arris
5. bend
6. bow
7. brace
8. bracket
9. corner
10. crater
11. crescent
12. cup
13. curve
14. dome
15. hemisphere
16. ladle
17. notch
18. scoop

Form Group Straightness

1. chord
2. edge
3. guide
4. line
5. rectilinear
6. seam
7. straight

Table 10. Feature Classification by Form Group - III.

Form Group Circularity (Simple)

1. arbor
2. axle
3. ball
4. band
5. bar
6. barrel
7. basket
8. bead
9. beam
10. belt
11. bevel
12. bore
13. boss
14. bubble
15. bulb
16. bullet
17. bushing
18. button
19. cable
20. can
21. cap
22. capsule
23. cartridge
24. cleat
25. clip
26. coin
27. collar
28. column
29. cone
30. conic
31. conical
32. cord
33. core
34. cover
35. crown
36. cylinder
37. dial
38. diaphragm
39. dish
40. disk
41. donut
42. doughnut
43. dowel
44. drill
45. drop
46. droplet
47. drum
48. dune
49. ellipse
50. ellipsoid
51. fillet
52. filling
53. filter
54. flange
55. flue
56. gasket
57. gland
58. globe
59. globule
60. grain
61. hole
62. hub
63. journal
64. leg
65. lip
66. loop
67. mandrel
68. manifold
69. mass
70. membrane
71. messenger
72. nail
73. needle
74. o-ring
75. oval
76. packing
77. pad
78. padding
79. pail
80. pallet
81. pan
82. peg
83. pellet
84. perimeter
85. pillar
86. pin
87. pipe
88. plug
89. plunger
90. pocket
91. pod
92. pot
93. probe
94. rail
95. receptacle
96. ribbon
97. rim
98. ring
99. rod
100. roller
101. rope
102. round
103. rounding
104. scoop
105. screen
106. seal
107. shaft
108. shell
109. shield
110. sleeving
111. sphere
112. spindle
113. stick
114. stopper
115. string
116. swell
117. swelling
118. tack
119. taper
120. tear
121. torus
122. tube
123. tunnel
124. twine
125. vein
126. wadding
127. washer
128. well
129. wheel
130. wire

Form Group Circularity (Complex)

1. aiguille
2. chase
3. chased-groove
4. chute
5. coil
6. curl
7. helix
8. rifle
9. screw
10. spiral
11. spring
12. tapped-hole
13. thread
14. twist

Table 11. Feature Classification by Form Group - IV.

Form Group Convexity

1. arbor
2. arch
3. astragal
4. bead
5. boss
6. bulb
7. bulge
8. bump
9. button
10. cap
11. chase
12. chimney
13. convex
14. crown
15. cup
16. cusp
17. cylinder
18. dilation
19. dome
20. drop
21. droplet
22. dune
23. emboss
24. entasis
25. flake
26. globe
27. globule
28. grain
29. hemisphere
30. hump
31. knob
32. lobe
33. mass
34. nile
35. nipple
36. node
37. oval
38. peen
39. pod
40. projection
41. prominence
42. protrusion
43. protuberance
44. raise
45. round
46. rounding
47. sphere
48. stick
49. swell
50. swelling
51. tear

Form Group Concavity

1. cap
2. cave
3. cavity
4. channel
5. constriction
6. crater
7. cup
8. deflation
9. dent
10. depression
11. dip
12. dome
13. gutter
14. hemisphere
15. hollow
16. impression
17. ladle
18. pit
19. pulley
20. race
21. reel
22. scoop
23. spool
24. tunnel

Form Group Flatness

1. abutment
2. apron
3. baffle-plate
4. band
5. bar
6. base
7. beam
8. bed
9. bedding
10. belt
11. bench
12. billet
13. block
14. box
15. cantilever
16. case
17. casing
18. coin
19. column
20. core
21. cover
22. cube
23. face
24. fence
25. filling
26. film
27. filter
28. fin
29. flume
30. foil
31. gauze
32. gib
33. grid
34. grill
35. gusset
36. keel
37. key
38. leaf
39. leg
40. mask
41. mat
42. membrane
43. mesh
44. mosaic
45. nest
46. net
47. packing
48. pad
49. padding
50. pallet
51. pan
52. partition
53. patch
54. pattern
55. pellet
56. plane
57. plate
58. rail
59. ribbon
60. screen
61. seat
62. sheet
63. shield
64. shim
65. sieve
66. slab
67. strip
68. stud
69. stuffing
70. table
71. tank
72. thickening
73. tile
74. track
75. wadding
76. washer
77. web

Table 12. Feature Classification by Form Group - V.

Form Group Sharpness

1. apex
2. cone
3. conic
4. conical
5. corner
6. cotter
7. dowel
8. drift-pin
9. nail
10. needle
11. nile
12. peak
13. peg
14. pin
15. point
16. screw
17. sharp
18. spike
19. tack
20. taper
21. vertex

Form Group Bluntness

1. blunt
2. edge
3. round

Form Group Smoothness

1. evenness
2. plane
3. polished
4. roller

Form Group Roughness

1. bump
2. dilation
3. grain
4. knurl
5. pod

Form Group Notch

1. deflation
2. dent
3. depression
4. dip
5. notch
6. valley

Form Group Fold

1. fold

Table 13. Feature Classification by Form Group - VI.

Form Group Furrow

1. break
2. canal
3. cavity
4. channel
5. cleavage
6. cleft
7. clough
8. collet
9. crack
10. cut
11. deep
12. depth
13. ditch
14. fissure
15. flute
16. fracture
17. furrow
18. gap
19. gorge
20. groove
21. gutter
22. hole
23. kerf
24. keyhole
25. keyseat
26. keyway
27. parting
28. pocket
29. pouch
30. rabbet
31. race
32. ravine
33. receptacle
34. recess
35. ridge
36. rupture
37. rut
38. scar
39. separation
40. slit
41. slot
42. socket
43. split
44. valley

Form Group Opening

1. annular-hole
2. aperture
3. arbor-hole
4. bore
5. counter-bore
6. crevice
7. flange
8. hole
9. inlet
10. nozzle
11. opening
12. orifice
13. perforation
14. puncture
15. tapped-hole

Form Group Closure

1. baffle-plate
2. cap
3. cover
4. hatch
5. hem
6. plug
7. seal

Form Group Perforator

1. bore
2. broach
3. counter-bore
4. ream

Form Group Stopper

1. bushing
2. clip
3. gasket
4. stopper

2.4 Rules and Guidelines for Feature Definition

Although features under the same topology group are modeled using the same set of parameters, their physical nature and purpose govern the values that the parameters may take. In general, different cognition rules govern the size, placement and use of each feature. This puts constraints in the way that features can be defined. Many times, design standards place rules or guidelines on a feature. The guidelines state what conditions should or must be imposed on the feature's topology and geometry in order for it to perform its required function. The rules often place quantitative or qualitative restrictions on the parameters that describe the feature. Any program that has to impart intelligence to the design process must have a representation that includes knowledge of feature defining parameters.

2.4.1 Mathematical Relations Between Parameters

Often, similar features impart different characteristics to the object being modeled and perform different functions. Each topology class will effectively represent only one feature unless a scheme to differentiate feature members is evolved. A conceptual study of features demonstrates that such differentiation can be established by creation of mathematical rules which govern the range of parameter values, relationships between various parameters and consist of equality and inequality relations between the parameters. These relations should be formulated taking design and other practical considerations into account. For example load conditions will constrain a 'column' to have a minimum

diameter-to-length ratio, or a 'washer' is expected to have a certain thickness-to-diameter ratio. Since dimensions influence many aspects of design, it is important that these sets of rules reflect the designer's perceptions of the feature's proportions. Once this differentiation is in place, a topology group can represent multiple features and can advise a designer about the numerical values of input parameters required to create a particular feature. Application of these rules will constrain feature definition within the specified ranges and ensure that it meets the design rules, guidelines and standards. A feature can be created only if it meets the inequality constraints. Thus a designer who is unaware of a feature's proportion is guided by these rules in creating a reasonable feature.

Prototype mathematical inequality relations for example features from different groups were created. These relations differentiate features within the same topology form group. Obviously, synonym features will be governed by the same set of rules. The relations put constraints on the relative values of parameters and may also control the values of individual parameters. For example, Topology group A is defined by two parameters: radius (R) and length (L). The relations put constraints on the values of the ratio R/L. In the case of Topology E the defining parameters are length (L), breadths (B_1, B_2), and height (H). The relations govern the ratios B_2/B_1, B_1/L and H/L . Tables 14 through 16 show some suggested mathematical relations for example features. Relations for other topology and form groups can be formed in a similar manner. Figures 15 and 16 show some extreme cases of features that were created using these relations.

Table 14. Suggested Mathematical Relations For Example Features - I.

Feature	Shape Form	Topology	Suggested Mathematical Relations
Bar	Rotundity	A	$0.1 \leq R/L \leq 0.5$
Boss	Rotundity	A	$0.5 \leq R/L \leq 5.0$
Rim	Rotundity	C	$0.25 \leq R_1/R_2 < 1.0$ $1.0 \leq R_2/L \leq 2.0$
Hub	Rotundity	V	$0.5 \leq R_2/R_1 \leq 1.0$ $1.0 \leq R_1/L \leq 2.0$
Bevel	Rotundity	V	$0.5 \leq R_2/R_1 \leq 0.75$ $2.0 \leq R_1/L \leq 5.0$
Bar	Angularity	E	$B_2/B_1 = 1$ $0.2 \leq B_1/L < 1.0$ $0.2 \leq H/L < 1.0$
Boss	Angularity	E	$B_2/B_1 = 1$ $1.0 < B_1/L \leq 10.0$ $1.0 < H/L \leq 10.0$
Case	Angularity	G	$0.5 \leq B/L \leq 1.0$ $0.5 \leq H/L \leq 1.0$ $0.1 \leq T/L \leq 0.25$
Keyway	Angularity	F	$B_2/B_1 = 1$ $0.1 \leq B_1/L \leq 0.5$ $0.1 \leq H/L \leq 0.5$
Packing	Angularity	E	$B_2/B_1 = 1$ $20.0 \leq B_1/L \leq 40.0$ $20.0 \leq H/L \leq 10.0$
Plate	Angularity	E	$B_2/B_1 = 1$ $10.0 \leq B_1/L \leq 20.0$ $10.0 < H/L \leq 20.0$
Shell	Angularity	G	$0.5 \leq B/L \leq 1.0$ $0.5 \leq H/L \leq 1.0$ $0.01 \leq T/L \leq 0.1$

Table 15. Suggested Mathematical Relations For Example Features - II.

Feature	Shape Form	Topology	Suggested Mathematical Relations
Angle	Curvature	H	$10° \leq A \leq 170°$ $1.0 \leq R/T \leq 2.0$
Arch	Curvature	D	$0.75 \leq R_1/R_2 < 1.0$ $1.0 \leq R_2/L \leq 2.0$
Dome	Curvature	O	$0.75 \leq R_1/R_2 < 1.0$
Tunnel	Curvature	D	$0.75 \leq R_1/R_2 < 1.0$ $0.01 \leq R_2/L < 1.0$
Coin	Circularity	A	$10.0 \leq R/L \leq 20.0$
Dowel	Circularity	A	$0.01 \leq R/L \leq 0.1$
Sleeving	Circularity	C	$0.75 \leq R_1/R_2 < 1.0$ $0.1 \leq R_2/L \leq 1.0$
Wire	Circularity	A	$0.001 \leq R/L \leq 0.01$
Hemisphere	Convexity	O	$0.0 \leq R_1/R_2 < 1.0$
Membrane	Flatness	E	$B_2/B_1 = 1.0$ $40.0 \leq B_1/L \leq 100.0$ $40.0 \leq H/L \leq 100.0$
Pad	Flatness	A	$10.0 \leq R/L \leq 20.0$

Table 16. Suggested Mathematical Relations For Example Features - III.

Feature	Shape Form	Topology	Suggested Mathematical Relations
Spike	Sharpness	K	$A.R. = 0.0$ $0.1 \leq R/H \leq 0.2$
Taper	Sharpness	K	$0.0 < A.R. < 1.0$ $0.5 \leq R/H \leq 1.0$
Crack	Furrow	F	$0.0 \leq B_2/B_1 \leq 0.01$ $0.0 \leq B_1/L \leq 0.01$ $0.0 \leq H/L \leq 0.01$
Dip	Furrow	M	$0.0 < A.R. < 1.0$ $0.5 \leq R/H \leq 1.0$
Flute	Furrow	N	$0.05 \leq R/L \leq 0.25$
Notch	Furrow	M	$A.R. = 0.0$ $0.1 \leq R/H \leq 0.2$
Pocket	Furrow	F	$0.5 \leq B_2/B_1 \leq 1.0$ $0.5 \leq B_1/L \leq 2.0$ $0.1 \leq H/L \leq 0.5$
Slot	Furrow	F	$B_2/B_1 = 1$ $0.05 \leq B_1/L \leq 0.25$ $0.025 \leq H/L \leq 0.25$
Collar	Stopper	C	$0.25 \leq R_1/R_2 \leq 0.75$ $1.0 \leq R_2/L \leq 2.0$
Washer	Stopper	C	$0.25 \leq R_1/R_2 \leq 0.75$ $10.0 \leq R_2/L \leq 20.0$

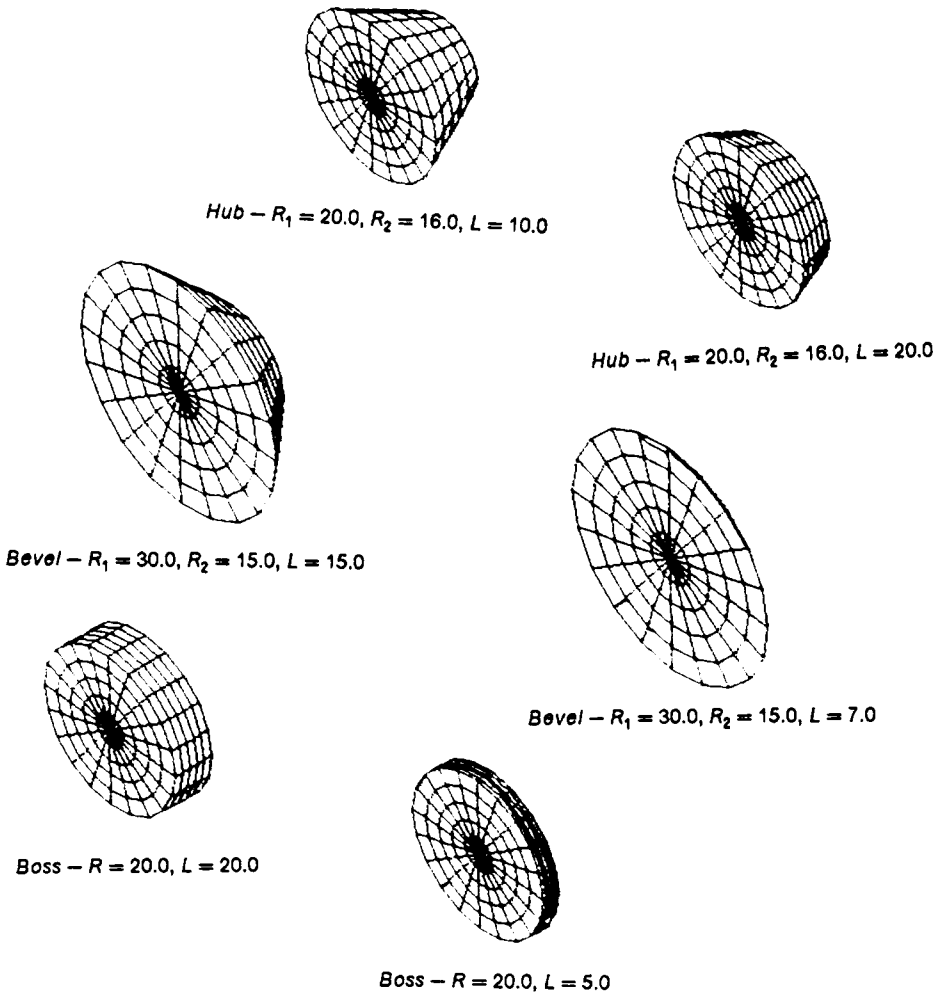


Figure 15. Example Features Based On Suggested Guidelines - I.

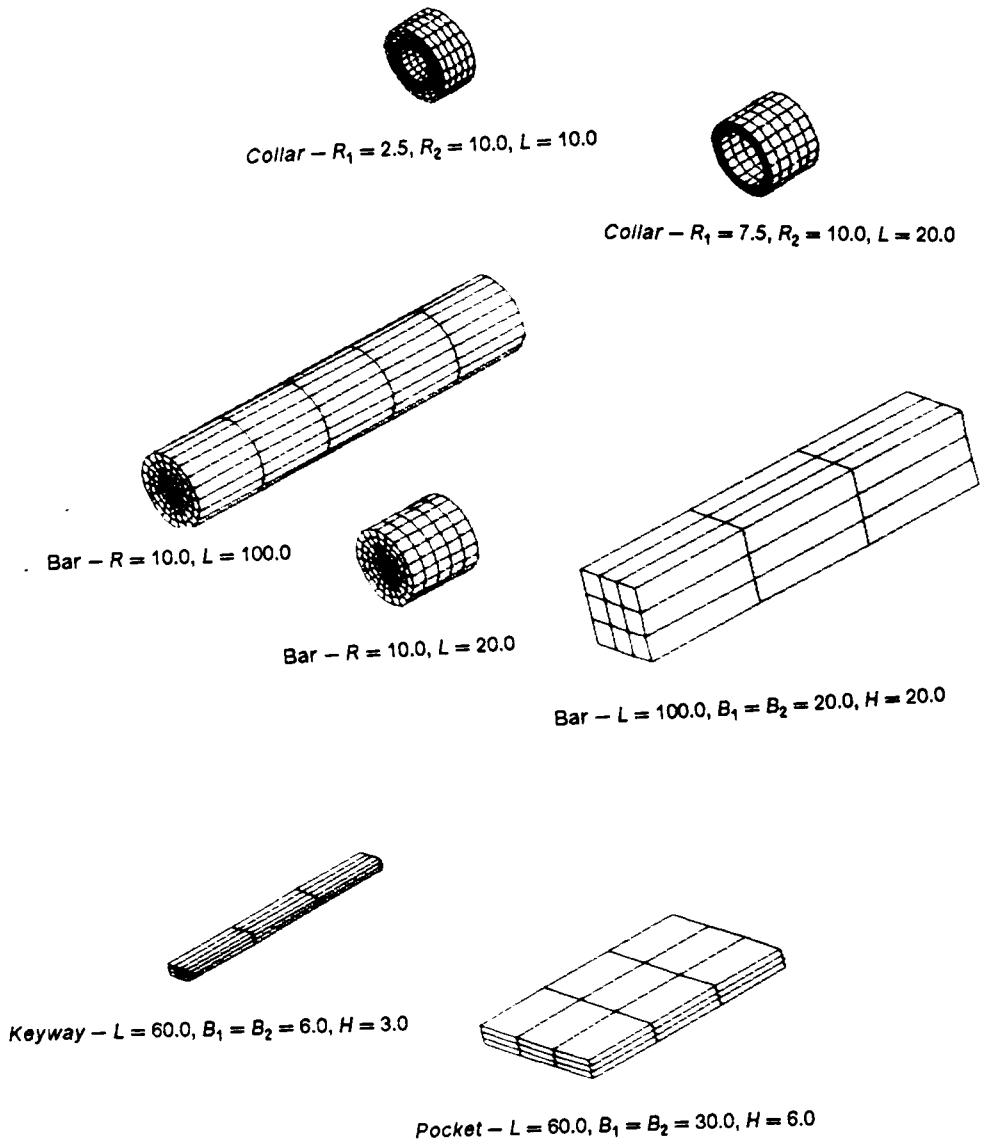


Figure 16. Example Features Based On Suggested Guidelines - II.

2.4.2 Shape-Altering Words or Adjectives

Use of shape-altering words or descriptions puts additional constraints on the defining relations. The above relations are used as guidelines to define a default feature. Many times designers need variations of these features and describe them by using additional words like 'long', 'thick' or 'large'. The use of such words put additional constraints on a feature's definition. Hence, the original relations are no longer valid and must be modified appropriately to match the new description of the feature being defined. Most words have a numerical effect on the constraints. A few words (e.g., smooth, rough) have a qualitative effect on the feature and do not affect the mathematical relations in any way. Still such words are part of a designer's vocabulary and must be accounted for. A set of shape-altering words and parameters has been defined to make such changes in the mathematical relations if required. These parameters, called 'adjectives', act as multiplicative factors to all or specified groups of parameters that define a feature. Because a descriptive word may induce different changes in the dimensions of different features, one must be careful when interpreting the meaning of such factors. This is best explained by means of an example. Consider the descriptive word 'thin' being applied to two different features, a circular 'rod' and a rectangular 'bar'. The radius-to-length ratio (R/L) will be affected in case of the rod but in case of the bar two ratios, B_1/L and H / L will be changed. Table 17 provides a listing of common shape-altering words and their suggested effect on the mathematical relations. Their effect on mathematical relations that govern feature-parameter values can be better understood by means of an example. In case of a rectangular 'bar', the word 'long' will affect all relations, but 'wide' will affect only that which concerns its breadth. The use of such

shape-altering words gives a designer additional control over the overall shape of the feature and also allows him to apply his normal vocabulary to feature definitions. It also allows him to create gradual changes in a feature's dimensions by repeated use of different or the same shape-altering words. The designer can hence change the description of the feature till it meets his requirements. For example, if a designer is not satisfied with the dimension range for a 'long rod', he can use the word 'long' again to further modify the mathematical relations to obtain a 'long long rod'. Normal vocabulary uses adverbs to further describe nouns. Thus provision should be made for the use of adverbs such as 'very' for additional descriptions. Thus a 'long long rod' can be described as a 'very long rod'.

2.5 Feature Definition By Incomplete Specification of Parameters

In the early stages of conceptual design, a designer may only have a rough idea about the dimensions of a part or about the features that make up that part. To require him to enter all parameter values for defining a feature defeats the purpose of an intelligent system. However, the presence of mathematical rules and generic parameter values helps a designer to stay within reason and allow him to create a feature using the default values or by partial specification of parameters. The minimum number of parameters to be specified is governed by the number of independent rules. A designer may be required to specify some parameters and may be given an option of choosing the other parameters. Once has specified the

Table 17. Shape-Altering Words And Their Effect on Mathematical Rules.

Shape Altering Word	Suggested Scale Factors	Relations Affected
blunt	N.A.	N.A.
broad	1.25	B/L
bumpy	N.A.	N.A.
compressed	1.25	B/L, H/L, R/L
deep	1.50	H/L
	0.67	R/L
default	1.00	N.A.
extended	0.80	B/L, H/L, R/L
fat	1.50	B/L, H/L, R/L
fine	0.67	B/L, R/L
glossy	N.A.	N.A.
high	1.50	H/L
	0.67	R/L
jagged	N.A.	N.A.
long	0.67	B/L, H/L, R/L
low	0.67	H/L
	1.50	R/L
narrow	0.80	B/L
normal	1.00	N.A.
plump	1.25	B/L, H/L, R/L
pointed	0.00	$R_2/R_1, B_2/B_1$
polished	N.A.	N.A.
rough	N.A.	N.A.
shallow	0.67	H/L
	1.50	R/L
sharp	0.00	$R_2/R_1, B_2/B_1$
short	0.67	H/L
	1.50	R/L
sleek	N.A.	N.A.
slender	0.67	B/L, H/L, R/L
slim	0.80	B/L, H/L, R/L
spiked	0.00	$R_2/R_1, B_2/B_1$
tall	1.50	H/L
	0.67	R/L
tapered	0.80	$R_2/R_1, B_2/B_1$
thick	1.50	B/L, H/L, R/L, R_2/R_1
thin	0.67	B/L, H/L, R/L, R_2/R_1
wide	1.50	B/L

parameters of his choice, the remaining parameters in the set can be computed automatically by applying the rules. This procedure will help speed up the decision process for a designer.

In order to compute the complete set of parameters for a general feature, it is necessary to formulate a general method for parameter computations. A mathematical model that can be used for this purpose is presented below. This model can be used in conjunction with linear inequalities.

Consider a feature (F) that can be represented by the following function

$$F \ : \ f \ (\ P_1, \ P_2, \ldots\ldots, \ P_n \ ), \tag{2.1}$$

where f is a function that depends on n parameters. If the feature is governed by m linear constraints (l) involving k parameters, they can be represented as

$$l_i \ : \ \sum_{j=1}^{n} C_{ij} \ P_j \qquad i \ = \ 1,2,3,\ldots,m \tag{2.2}$$

where, C_{ij} are coefficients generated from the mathematical rules. $C_{ij} \neq 0$ for k parameters and $C_{ij} = 0$ for the other $n - k$. This results in a matrix equation that can be solved using a variety of methods. However, since the constraints use only k parameters, the other n - k parameters must be specified by the user. Let us first address the equality constraints only. Different cases are possible and must be addressed separately.

If $m = 0$, then there are no constraints and the user must specify all the parameters.

If $m \neq 0$, then two cases exist.

1. If $k \leq m$, then $m - k$ are dependent constraints and are redundant. The k independent relations can be used to simultaneously compute unique solutions for the k parameters.
2. If $k > m$, then an additional $k - m$ parameters must be specified. After they are specified, the m independent relations can be used to simultaneously compute unique solutions for the m parameters.

In general, the constraint set consists of equality and inequality relations. In such cases the rules must be classified into two groups. If a constraint has a $\geq$ or a $\leq$ relation, then it must be classified into both groups. The first group must consist of equality relations and must be solved first using the procedure mentioned above. The solutions obtained from this group must be replaced in the second group consisting of inequality relations. Now, the second group may be considered. Change all inequalities to $>$ relations, by multiplying by *(-1)* if necessary. Now this set of relations can be solved by using the process of elimination to get unique values or a range for the parameters. In the latter case, the designer may be given the option of selecting a value, or a conservative estimate can be made automatically using the mean value of the range.

In either case, once all parameters have been computed, they must be used to check that all mathematical constraints are satisfied. If not, numerical methods may be used to solve for the constraints using these values as an initial guess as is illustrated by the two examples given below.

Consider a feature defined by three parameters *x, y and z* with three constraints

$x - 2y + z = 3$

$3x + 4y - z = 26$

$8x + 2y - 3z > 30$

In this case the first two equations are put in one group. The equality group contains two equations in three unknowns; that is, m = 2 and k = 3. Hence, one parameter must be specified. Let x = 4 (chosen arbitrarily). Now the two constraints can be solved simultaneously to get y = 6.5 and z = 12. The computed values must now be used in the third constraint to check the validity of the solution. It turns out that 8x + 2y - 3z = 33 > 30 and hence meets the entire constraint set. Hence $x = 4.0$, $y = 6.5$, $z = 12.0$ is one of the many acceptable solutions for the feature.

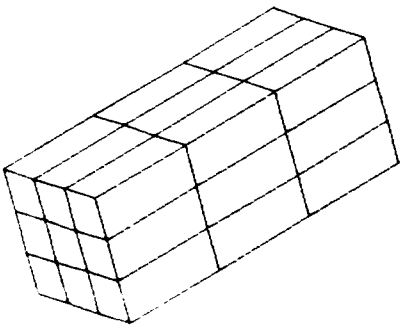
As another example, consider the mathematical relations that govern the definition of a rectangular bar. The relations are:

$B_2/B_1 = 1.0$

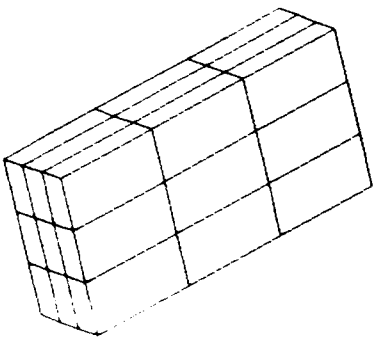
$0.2 \leq B_1/L < 1.0$

$0.2 \leq H/L < 1.0$

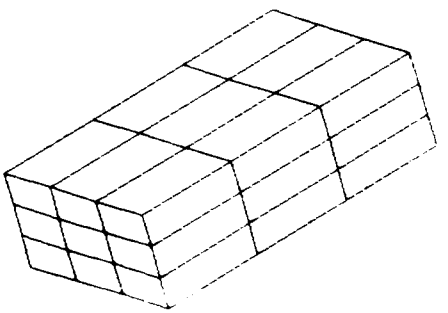
There are effectively three parameters that describe the feature but there are only two relations. Hence one dimension must be specified by the user. If the designer chooses the parameter L to be 100.0, then using the above method we get a range for B_1 and H. Using the mean value of the ranges we get one acceptable solution to be $B_1 = B_2 = 40.0$ and $H = 40.0$. Some other acceptable solutions would be $B_1 = B_2 = 25.0$ and $H = 50.0$ or $B_1 = B_2 = 60.0$ and $H = 30.0$. The features that would be created using these dimensions are shown in Figure 17.



Bar – $L = 100.0$, $B_1 = B_2 = 40.0$, $H = 40.0$



Bar – $L = 100.0$, $B_1 = B_2 = 25.0$, $H = 50.0$



Bar – $L = 100.0$, $B_1 = B_2 = 60.0$, $H = 30.0$

Figure 17. Some Acceptable Dimensions for a Rectangular Bar.

2.6 *A Feature Library*

The establishment of a feature library based on definition, classification, parameterization and mathematical constraints is now feasible. The library will contain the following information for each feature:

1. Feature name.
2. Feature definition.
3. Feature classification (Topology and Shape Form).
4. Feature synonyms.
5. Mathematical Inequalities or rules governing the feature.
6. A generic model listing default parameters defining the feature.

This library can be used as a tool by the designer to define and create feature-based models. Since this library is not dependent on a particular design environment, it can be used as an interface for any kind of design, modeling or programming environment. It can be used not only to check whether a feature definition meets the constraints, design rules and procedures but also to help a user understand the meaning of the feature, to give the user the choice of using other features which are its synonyms, to create generic models of features and to allow feature creation by incomplete specifications of parameters. Appendix A contains the complete feature dictionary that can be used in conjunction with an application program. Only those features that have been considered in the previous sections contain a set of mathematical relations as examples of feature modeling. As mentioned earlier, the

remaining mathematical relations can be established at a later stage, in the process of establishing a production feature modeler.

Chapter 3: Designer-Oriented Modeling Language

Although CAD/CAM systems automate and speed up the design process, they require a high level of human interaction to get input data. The ease with which such data can be input will control the speed of such interaction. Most systems use graphics input/output as means of interacting with the designer. Such menu-driven systems require the user to have an in-depth knowledge of the menu structure and hierarchy of the system. For efficient use of such systems, a user must learn how to use them. This has tremendous cost disadvantages since considerable time must be spent in training users. In addition, menu interfaces are less flexible and work only in their own database domains. They are more time consuming to customize and require applications programmers to anticipate the kind of questions that will be asked. A flexible designing system must be able to interact freely with other programming environments and any data transfer should be simple. Future systems will integrate novel methods of input like voice recognition and natural language so that input will become simpler and faster. In most cases however, such complex and costly interfaces are not required. A simple language that would allow a user to have linguistic coverage, deductive abilities or intelligent prompting would suffice. One

of the aims of this research is to develop a designer-oriented modeling language called Universal Modeling Language (UML), that will be fast and robust enough to serve users with no knowledge of the underlying domain. This language should be completely portable (i.e. can run on different hardware under different operating systems). The language is a preliminary step to implementing a voice input interface.

Using a natural language approach towards feature selection and modeling will make the task of the designer simpler and faster. In order to be able to devise a method that allows such an approach, one must take a close look at the manner in which a designer thinks and formulates his modeling procedure. In addition, one must consider the grammar of the language structure involved in this process. The elements that make up this structure depend on the kind of processing to be done and the action to be taken. One planned application of UML is its use in a modeling environment coupled to a graphics display. Hence the structure elements will have to handle modeling, system and graphics commands issued by the designer.

3.1 Language Structure Elements

For modeling an object using feature instances, a general language structure will have the form:

[verb] [adjective] [component] [name] [parameters] [location] [orientation]

where the elements of the structure are as described below:

[verb]: A command can be of different types. It could indicate a modeling, transformation, display, inquiry, archive or control verb. A modeling verb allows creation, modification, deletion of different components and the relative placement of components. These words will operate on a component specified by the user. A transformation verb executes viewing transformations (zoom, pan, rotations, etc.). An inquiry verb helps a user make numerous inquires about a model, component or a feature. Archive verbs allow execution of archiving commands like 'recall', 'start' or 'exit'. Thus, as in all languages a verb is the principal element in the entire structure. It helps decide the level of interpretation required and the flow of the program. The verb list and the classification scheme employed is shown in Table 18.

[adjective]: Use of shape altering would be restricted to features. It indicates what coefficients will be applicable in modifying mathematical constraints for defining that feature.

[component]: It could be 'feature', 'object' or 'assembly' and indicates the position of the component in the object hierarchy. In case of a modeling verb, it plays a major role in interpretation of the other elements of the structure.

[name]: Any component must be named in order to identify it. This allows the program to keep track of all components existing in the model. This simplifies commands for any further operations on the named component.

[parameters]: These are numerical values (to be specified in a fixed order) that allow definition of features. This list must be completely specified by the user. In case it is not, then mathematical relations defined for that feature are applied to compute the missing parameters. In some cases they may represent graphics parameters.

[location]: This represents the location of the component in question. It uses object hierarchy to compute the actual location. Assemblies are located in global coordinates, objects in assembly coordinates and features in object coordinates.

Table 18. Command Verbs - List and Classification

Modeling Verbs

- add
- addto
- alter
- append
- assemble
- attach
- cancel
- change
- create
- define
- delete
- discard
- edit
- eliminate
- enlarge
- erase
- exclude
- extrude
- fasten
- fetch
- fix
- include
- insert
- invert
- join
- label
- link
- loft
- merge
- mix
- model
- modify
- reduce
- remove
- rename
- replace
- reshape
- scale
- shape
- stretch
- subtract
- sweep
- truncate
- use

Transformation Verbs

- locate
- move
- orient
- place
- position
- put
- rotate
- translate
- turn

Archiving, Display and Other Verbs

- abort
- bye
- compare
- display
- draw
- end
- execute
- exit
- file
- find
- frame
- hardcopy
- help
- hide
- inquire
- list
- open
- pan
- pause
- plot
- query
- quit
- recall
- render
- run
- save
- shade
- stop
- store
- view
- zoom

Positioning with respect to another component is done by using 'prepositions' and the 'name' of a pre-defined component. Table 19 gives a list of common prepositions that are used commonly in describing relative positions.

[orientation]: This represents the orientation of the component in question. It uses the method mentioned above to compute the actual orientation of the component.

The set of elements must be complete for every command. In certain cases when some elements are not required, their values default to 'not applicable' so that no ambiguity will exist while interpreting the input. This structuring scheme gives flexibility to the type of logical input that can be interpreted by software.

3.2 Overview of UML

With this in mind, an English language interface was developed and can be used to provide a wide variety of functions like modeling (creation and modification of models), viewing (transformations) and interrogation (inquiries for definition, parameters, etc.). The description given below gives an idea about UML construction and how individual pieces contribute to its working. Figure 18 shows the elementary steps in processing string inputs by UML.

Table 19. Commonly Used Prepositions.

- above
- across
- after
- among
- angled
- around
- axial
- back
- before
- below
- beneath
- between
- border
- bottom
- center
- concentric
- eccentric
- edge
- end
- external
- front
- horizontal
- in
- inside
- internal
- into
- left
- lower
- middle
- normal
- offset
- outside
- over
- perpendicular
- radial
- right
- start
- through
- top
- under
- upper
- vertical

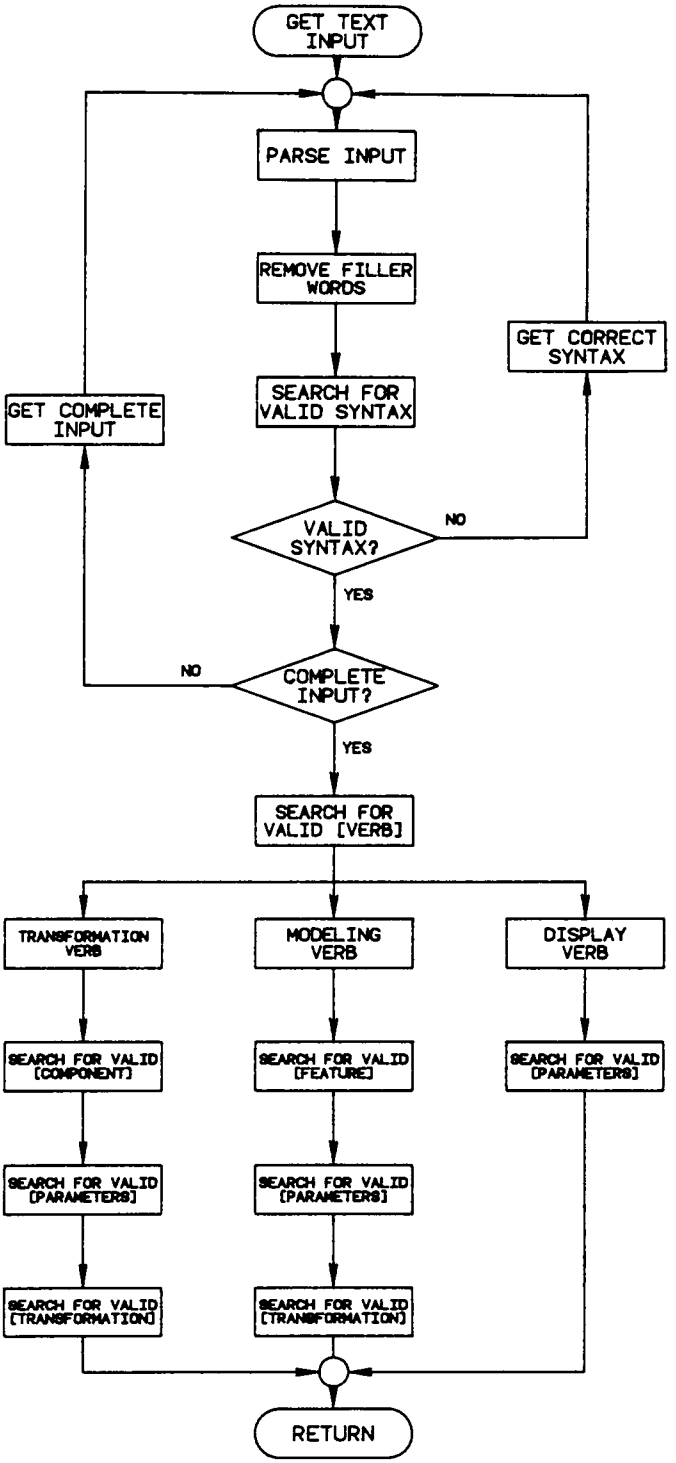


Figure 18. Input Processing by UML.

3.2.1 Word Parser

The word parser is used to break the sentence into its individual elements, for example, text or numbers. Any 'separators' are recognized and are discarded while parsing. The elements are then passed to the next part of the program for further treatment. Table 20 gives the list of separators that can be recognized by UML.

3.2.2 Structure Generator

This portion of UML restructures the parsed input into a form that is more understandable by the interpreter. Technical descriptions of a product need not contain words or phrases that are obvious from the context. Such 'filler words' do not play a constructive role in sentence structures and can be removed in order to simplify the sentence. The next step is to convert the sentence into a understandable grammatical structure or 'representation language'. All words of the sentence are checked and assigned as values to the appropriate element of the structure. This allows the user to enter identical commands in a number of ways rather than just one fixed format. However, it is necessary to remove all ambiguities from the sentence. Any missing information must be compiled by either prompting the user, by using rules and definitions in the data structure, or from previously established references. The complete structured sentence is now passed to the interpreter for decision making. Table 20 lists filler words that are removed from the sentence.

Table 20. List of Filler Words and Separators

Filler Words

- a
- an
- and
- as
- at
- by
- called
- for
- from
- in
- into
- it
- location
- named
- of
- on
- orientation
- out
- parameters
- the
- this
- to
- up
- upto
- with

Separators

- (
-)
- ;
- ,
- :
- #
- @
- {
- }
- ?
- !
- "

3.2.3 Interpreter

This is the most important module of UML. A complete analysis of the command is done in this section and the direction of flow of the program is decided. Each element of the statement structure affects the direction and hence each must be analyzed. The 'verb', which has primary influence on the command, is first analyzed and then classified as a modeling, graphics, inquiry or system function. The control is then passed to the appropriate function module for complete implementation of the command.

3.3 Execution of Input Command

Once structuring has been completed, and each element has been assigned a value, they are individually processed and interpreted. If an element has missing information that is required for successful execution of the command, or the information does not meet the guidelines for the specific command, then the user is prompted for additional or alternate information. In some cases, it may be difficult for a user to decide the value to be assigned. Then, the program has the capability of using rules and references to decide the value to be assigned to the element.

Now, the command is ready to be fully executed. The control is passed to each element utility which in turn governs the further transfer of control. The first element to be analyzed is the 'verb'. It is classified into 'modeling', 'graphics' or 'inquiry' functions. This specifies the set of elements required, and the order in which they

must be processed. In case of a 'modeling' function, the elements that must be processed include the 'component' type and 'name'. If the value of 'component' is 'feature' then all other elements are also needed. A 'graphics' function generally does not affect the model. Hence, only the 'parameters' element is needed. Some commands, like 'rendering', also require the 'name' of a component to which the function must be applied. An 'inquiry' function is normally applied to components and requires the component 'name'. Other elements do not affect this function and hence are not needed.

Each element that is processed allows the program to make partial decisions. Each partial decision is translated into a set of environment dependent calls. A series of such decisions leads to complete implementation of a command.

3.3.1 Sample Conversion of Input by UML

As described earlier, UML consists of three major steps. Application of UML to a feature-based modeling system can be described by means of an example. Consider the following input:

Define a long rod with parameters (20.0, 150.0) at location (0.0 0.0 0.0) and orientation (30.0, 30.0, 30.0)

This input first passes through the parser which removes the separators and returns a string array:

Define a long rod with parameters 20.0 150.0 at location 0.0 0.0 0.0 and orientation 30.0 30.0 30.0

This array is now processed and all filler words that do not form part of the structure are removed:

Define long rod 20.0 150.0 0.0 0.0 0.0 30.0 30.0 30.0

Now the structure elements are given the appropriate values so that:

[verb] = DEFINE
[adjective] = LONG
[component] = FEATURE
[name] = ROD
[parameters] = 20.0 150.0
[location] = 0.0 0.0 0.0
[orientation] = 30.0 30.0 30.0

3.4 Application of UML in Modeling Systems.

UML has been developed to act as a tool that allows modeling in an environment of a user's choice. English ASCII characters can be processed by all computer systems and any hardware. Hence any software, such as UML, which uses English-like language to interact with its users can operate independently of the environment. Since all commands (except graphics display functions) are independent of the modeling environment, it is easy to integrate applications programs that have been customized for specific graphics packages.

An important application of UML in feature-based modeling systems is the ability of the user to interactively create a geometric model representing a feature or to make a series of inquires about it. UML uses the feature library to answer user queries. This allows the user to extract information of mathematical rules, default parameters, etc., and helps decide which feature is best suited for a specific application. It allows the user to pass feature parameters to a modeling utility that can return feature definition in terms of parametric equations, surface points or the

control polyhedron for uniform rational B-spline surfaces. The set of surface points allows integration of other surfaces to the modeler. This allows the user to create models using different kinds of surfaces. UML can also be used with application-specific menu structures. If the hierarchy of the menu structure is known, the user can, by a simple command, get the elements that form a specific menu page. The elements are available as an array of character strings and can be used by the programmer to write an interactive program that uses a graphics support of his choice.

Thus, UML acts as a factor in unification of different modeling environments. It can be used as a stepping stone to develop natural language procedures that allow quick, efficient and complete creation or interrogation of modeling databases. UML presents the best approach for future integration of software with voice recognition systems that translate voice into a string of ASCII characters. This will allow the user to interact with software in a manner that he is best suited and trained for - conversing in simple English.

Chapter 4: Geometric Modeling of Features

In keeping with today's advanced styling requirements, a designer should be able to quickly create a variety of sophisticated shapes. Most shapes can be modeled using straightforward modeling schemes. Some complex shapes require blending of a series of surface profiles. Such complex sculptured surfaces cannot be modeled using normal techniques. This may require 'lofting' and 'sweeping' procedures which would allow a designer to specify complex planar shapes at different positions and orientations that can be lofted or cross-sections that may be swept about complex three dimensional curves.

Of all the methods used in the creation of solid models, B-Rep (Boundary Representation) appears to be the most suitable general method to define a solid model using features. The B-Rep method makes use of the exterior surfaces to create a solid model. Such surfaces should be versatile in nature and easily modeled. Several surfaces are popular in geometric modeling. However, B-spline surfaces have distinct advantages over others. They have local control over the shape of the curves that define the surface and hence complex shapes can be

modeled with ease. Use of rationality further enhances this characteristic. Uniform rational B-spline surfaces will be used as a base for feature representation.

4.1 Modeling of B-spline Surfaces

In order to successfully implement a surface modeler, it is necessary define the basic mathematics for the surface. This has been developed over the years and is explained in detail by Pratt (1979), Mortenson (1985), Riesenfeld (1973), Versprille (1975) and many others. A B-spline surface is defined in terms of its characteristic polyhedron. The shape of the surface approximates the polyhedron. Any quadrilateral patch element on the surface defined by $(m + 1)$ x $(n + 1)$ points, can be expressed by:

$$p(u,w) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} h_{ij}\ P_{ij}\ N_{i,k}(u)\ N_{j,l}(w)}{\sum_{i=0}^{m}\sum_{j=0}^{n} h_{ij}\ N_{i,k}(u)\ N_{j,l}(w)} \tag{4.1}$$

where P_{ij} are the vertices of the defining polyhedron, h_{ij} are the corresponding homogeneous coordinates and $N_{i,k}(u)$ and $N_{j,l}(w)$ are the blending functions (given below) whose degrees are governed by parameters k and l. The blending functions for $N_{i,k}(u)$ are defined recursively by the following expressions:

$$\begin{aligned} N_{i,1}(u) &= 1 \quad \text{if } t_i \le u < t_{i+1} \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{4.2}$$

and

$$N_{i,k}(u) = \frac{(u - t_i)\ N_{i,k-1}(u)}{t_{i+k-1} - t_i} + \frac{(t_{i+k} - u)\ N_{i+1,k-1}(u)}{t_{i+k} - t_{i+1}} \tag{4.3}$$

where t_i are the 'knot values' that relate the parametric variables to the control points. The blending functions for $N_{j,l}(w)$ also have the same form.

This is the general representation for a B-spline surface. If the knot values are uniformly spaced in the range $k \leq i \leq m$ and $l \leq j \leq n$ then the resulting surface is a uniform B-spline surface. If they are not, then the resulting surface is a non-uniform B-spline surface. The homogenous coordinates h_{ij} act as weights on the corresponding control point and have the same effect as having multiple-coincident points. They help in controlling the shape of the surface by either pulling it closer or pushing it further from the control point. If $h_{ij} = 1.0$ for all points, the denominator in equation (4.1) goes to 1.0 and the resulting surface is a non-rational surface.

For this work, we will consider uniform rational B-spline surfaces. Hence the knot values must be equally spaced. Let the values of t_i be given by:

$$\begin{aligned} t_i &= 1 && \text{if} \quad i < k \\ &= i - k + 1 && \text{if} \quad k \leq i \leq m \\ &= m - k + 2 && \text{if} \quad i > m \end{aligned}$$

and let t_j be given by similar expressions. The resulting blending functions act like switches, turning on and off the terms that they control. The symmetries and congruences of their shapes allow us to develop a more convenient and more familiar matrix notation. Choose an interval in i and j so that $k \leq i \leq m$ and $l \leq j \leq n$.

Let t_i and t_j be computed from equation (4.4) so that $t_i = i - 3$ (k = 4, for cubic splines) and $t_j = j - 3$ (l = 4, for cubic splines). Using the recursive expressions in (4.2) and (4.3) and reparameterizing the intervals we get a general matrix form for the surface as:

$$p_{st}(u,w) = \frac{U_k \; M_k \; (hP)_{kl} \; M_l^T \; W_l^T}{U_k \; M_k \; h_{kl} \; M_l^T \; W_l^T} \qquad \begin{array}{l} s \in [1{:}m + 2 - k] \\ t \in [1{:}n + 2 - l] \\ u,w \in [0,1] \end{array} \tag{4.5}$$

For cubic surfaces, $k = l = 4$ so that:

$$U = [\, u^3 \; u^2 \; u \; 1 \,]$$

$$W = [\, w^3 \; w^2 \; w \; 1 \,]$$

$$M = \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & 6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$

$(hP)_{kl}$ is the matrix containing the control polyhedron and the weights, and h_{kl} is the matrix containing the weights. Thus if the points on the control polyhedron are known then the surface can be computed using the above expression. The position of the control points govern the shape of the surface and hence must be chosen judiciously. The process of specifying the control polyhedron to create a surface will be called the forward modeling process. Figure 19 shows a simple example of modeling a B-spline surface using the forward modeling technique. It shows the control polyhedron and the resulting B-spline surface.

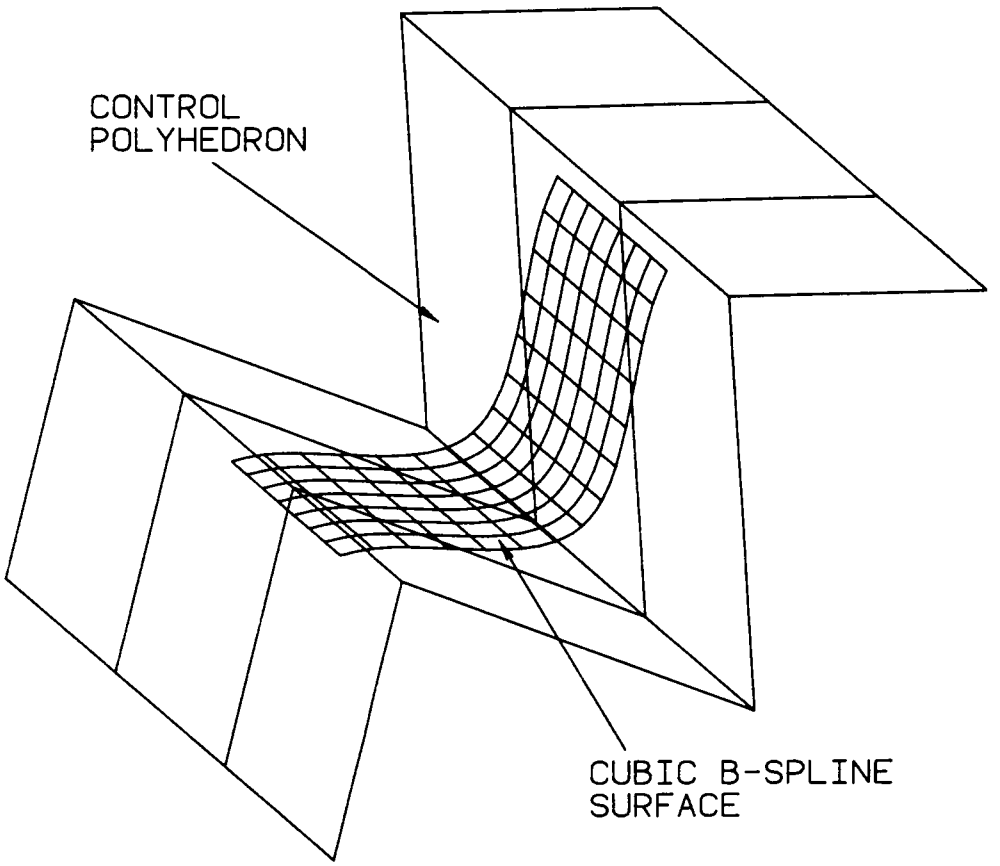


Figure 19. Example of Forward Modeling.

4.2 *Inverse Modeling of B-spline Surfaces*

The ability to model free-form and sculpted surfaces is dependent on the ease with which the set of control points is predicted. The capabilities of the modeler will be enhanced if a designer is allowed to simply specify a finite set of points that lie on the surface. Some surface defining techniques interpolate a given set of points, which means that the surface produced passes exactly through the control points. These techniques have a disadvantage when incorporated into an interactive CAD program. Specifically, we do not get a strong intuitive feel for how to change or control the shape of the surface. For example, if we try to change the shape of a spline-interpolated surface by moving one or more control points, we may produce unexpected and undesirable perturbations and inflections. Alternatively, another approach defines a surface that only approximates or approaches the control points. In such cases it is easier to predict the changes in the shape of the surface and provide better shape control. However, since they do not pass through the control points, it is difficult to control their exact shape and size. Periodic B-splines fall in the latter group: that is, they do not interpolate the end-points on the characteristic polyhedron and seldom pass through other control points. The use of parametric equations to define the geometry of a part requires that the surface pass through the set of computed points. Thus a method must be devised to accurately compute and construct a polyhedron that will ensure that the surface will pass through the specified set of points. Also, in case modifications are required, it is better that the surface shape be controlled in a predictable way by changing only a few simple parameters.

This method will be called the inverse modeling method and can be derived by applying matrix operations on the basic representation for B-spline surfaces. This leads to the following inverse relation:

$$P_{kl} = M_k^{-1} V_k^{-1} p_{st}(u,w) X_l^{-1} (M_l^T)^{-1} \tag{4.6}$$

For, $k = l = 4$ V and X are:

$$V = \begin{bmatrix} u_1^3 & u_1^2 & u_1 & 1 \\ u_2^3 & u_2^2 & u_2 & 1 \\ u_3^3 & u_3^2 & u_3 & 1 \\ u_4^3 & u_4^2 & u_4 & 1 \end{bmatrix} \qquad X = \begin{bmatrix} w_1^3 & w_2^3 & w_3^3 & w_4^3 \\ w_1^2 & w_2^2 & w_3^2 & w_4^2 \\ w_1 & w_2 & w_3 & w_4 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

The values of the parameters u and w, corresponding to each point, can be specified by the user or can be computed internally by equal spacing parameterization. Figure 20 shows an example where the specified points through which the surface must pass, the computed polyhedron and the modeled surface are shown. Using this inverse relation one can now specify the points that the surface patch will interpolate and compute the corresponding control points, which will not be interpolated, needed to generate the patch.

A disadvantage of surfaces that do not directly interpolate control points is that any change in the control points may result in unpredictable changes in the surface. Thus a designer may have little knowledge of the local shape of the surface, making it difficult to model complex shapes. A distinct advantage of using this inverse method is that the designer can now reshape a surface by recomputing the surface points without worrying about the changes in the control points or the transmission of this change to the surface.

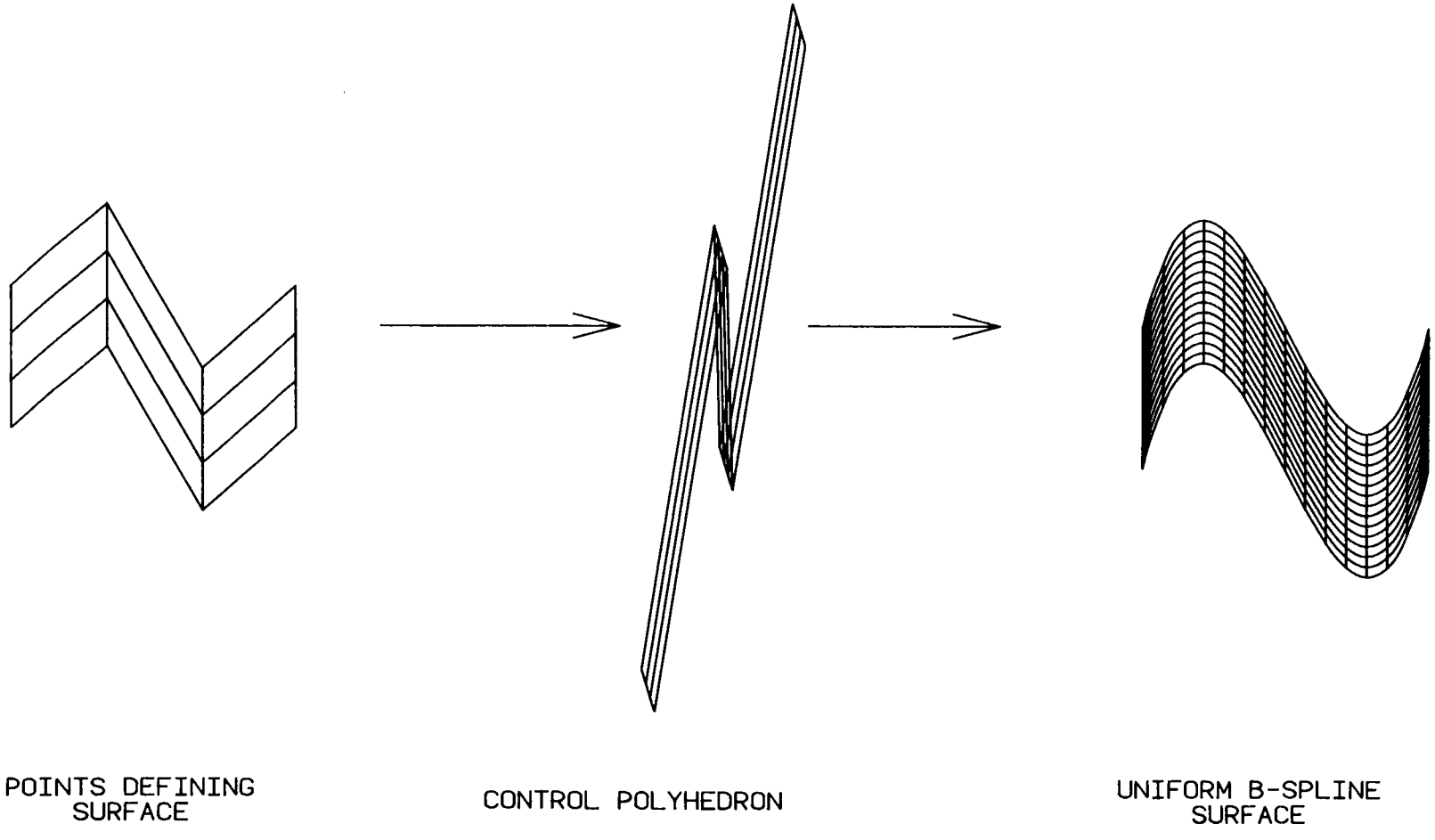


Figure 20. Example of Inverse Modeling.

This procedure can be used very efficiently for feature-based modeling. Since, a designer defines a part in terms of its parameters, they can be used to compute the points lying on the surface of the feature. The inverse relation can then be applied to create a B-spline surface that interpolates the computed points.

The method allows the modeling of single surface patches. In general, a feature will consist of more than one surface patch. Thus it is necessary to break down a feature in a manner that allows easy implementation of inverse modeling. A large patch requires a greater number of points to ensure the accuracy of the shape. This means a larger number of points must to be computed and interpolated. Hence it is necessary to find the optimum shapes for which the inverse method will work accurately and easily. It was found that this method worked best when (if possible) a feature is broken down into surfaces that are rectangular, cylindrical or elliptical in shape. Rectangular shapes are modeled as one patch, whereas cylindrical and elliptical surfaces are split into patches that cover one quadrant or octant as the case may be. This ensures that the symmetric nature of a feature is preserved, and that the surface is represented accurately. Some examples of the basic patches used are shown in Figure 21.

This method was used to model several features. Figures 22 and 23 show hidden surface images of some features that were modeled. Figure 22 shows the surfaces defining an ellipsoid, a neck, a hollow cone, a solid cone, a cone with a cylindrical hole, a counter-bore and a counter-sink. Figure 23 shows surfaces representing a sphere, hemisphere, hollow hemisphere, cube, truncated prism, wedge, cylinder, hollow cylinder and an arch.

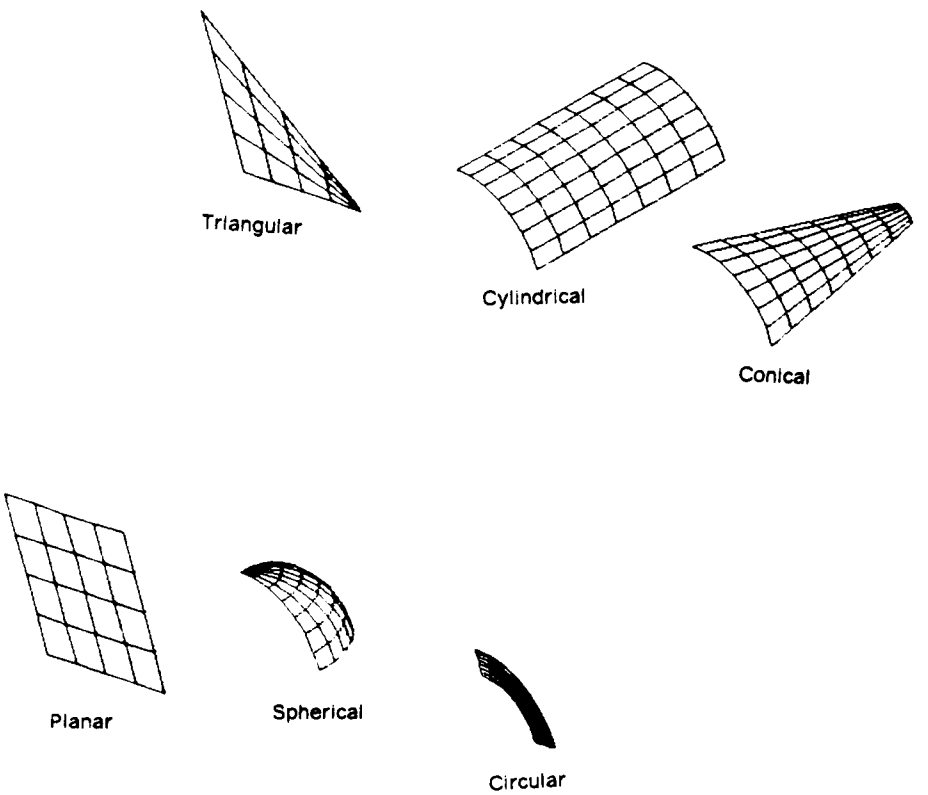


Figure 21. Examples of Patches Using Inverse Solution.

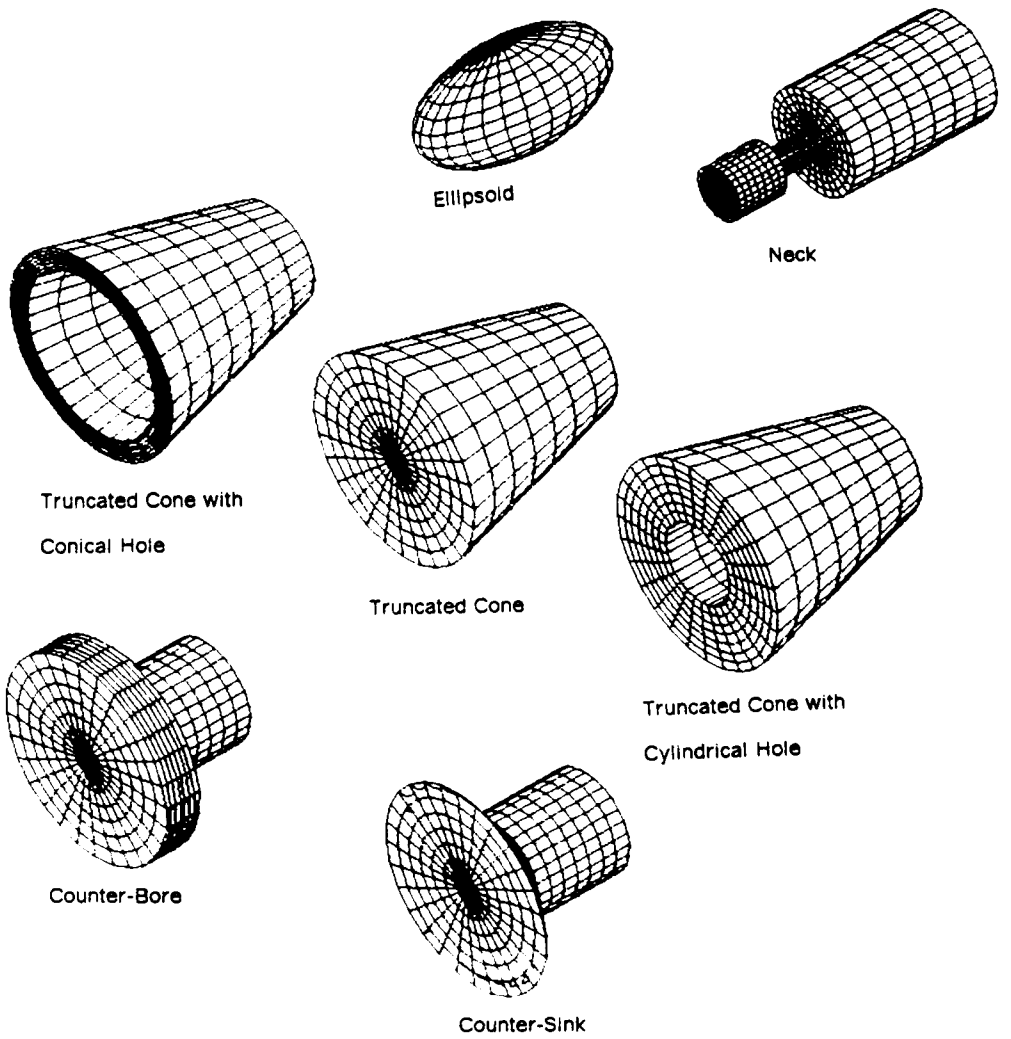


Figure 22. Examples of Features Using Inverse Solution.

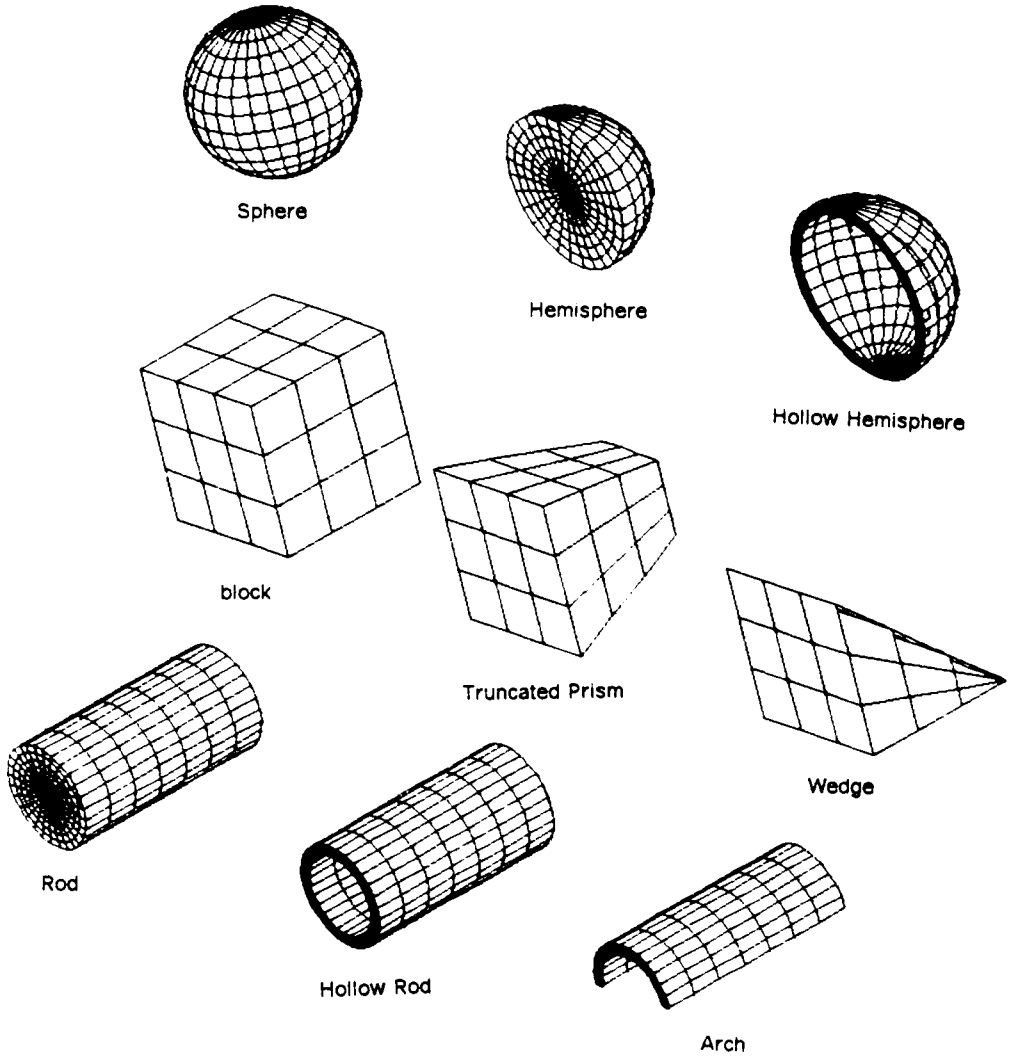


Figure 23. Examples of Features Using Inverse Solution.

Chapter 5: FeatureMod - A Prototype Feature Based Modeler

We can now investigate the possibility of implementing a prototype feature-based modeling system. This system should incorporate the ideas discussed in preceding sections and also meet the basic requirements of a feature modeler. A good system can be developed only after taking an in-depth look at such requirements.

5.1 Requirements of a Feature Modeler

A good feature modeler should address several issues of geometric modeling. It should be capable of operating independently in any computer environment, be coupled to a graphics display system, perform basic functions of modeling and have a data structure that contains complete information of a geometric model. Functional

requirements of a feature-based modeling system include system, graphics and modeling functions. In addition, a data structure that contains information of different elements in the model should be developed. In order to meet these requirements, a brief analysis of such functions is necessary. Archiving and graphics functions are similar for any modeling environment. Modeling functions and the data structure are more important and are discussed as they apply to feature-based modeling.

Archiving Functions: Any modeler should be independent of its operating system. To operate as a complete system it should have the capability of model storage and retrieval without loss of any information. Such functions act as a link between the designer and the data structure. Extensive queries on the data structure should be possible and external links to peripherals like printers and plotters should be established. Thus functions like store, recall, exit, query, list and plot should be supported by the modeler.

Graphics Functions: Today's development environment requires designing at different locations simultaneously. The ability to combine this work will greatly depend on the compatibility of the modeling systems used. Older systems use graphics software that is device dependent and hence is not portable and has compatibility problems with other systems. The need to provide a portable modeling system is the most important reason to have device independence. This can be achieved by the use of ISO standards for three dimensional graphics. A modeler that utilizes graphics software that meets the specifications and follows guidelines of three dimensional graphics standards will be portable and can be used at different sites without difficulty.

Viewing transformations allow a designer to observe an object in different orientations. Hence functions like rotate, zoom and pan functions that provide single or multiple views should be implemented. Other graphics support needed include rendering, hidden surface removal and shading to create realistic images and provide a designer with a better perspective of the object and the status of design. Such functions will also help remove viewing ambiguities that are present in wireframe and surface models.

Modeling Functions: The virtues and drawbacks of a modeler originate in the way that modeling is handled. The ease with which modeling functions are executed will determine the quality of the modeler. Some examples of functions are creation, deletion, modification, positioning, copying, instancing and interrogation. Creation and deletion of features should be handled unambiguously and at the designer's instigation. Modification of existing features that result in local surface redefinition or even a completely new set of parameters should be possible. Positioning by means of translation and rotation transformations will allow global positioning of the feature leaving its geometry and defining dimensional parameters unchanged. Copying is similar to moving, except that the original feature still remains in position and maintains its separate identity. Establishment of object hierarchy allows a designer to apply modeling functions at different levels, for example assemblies, objects and feature instances. It also allows model creation by means of instancing.

Execution of any modeling function leads to changes at all levels of the model and should be transmitted correctly to the data structure. The capability of a modeler to handle extensive interrogation will be governed by the amount of information available in the database. Any analysis to be done on the model will require a great

deal of information from the database. A good data structure should store information that will make such analysis easy and quick. Information about feature parameters, surface definition, location, orientation, attributes and its level in the object hierarchy are some examples of information that should be stored in the database.

Keeping the requirements in mind, a prototype system called FeatureMod has been developed. Written in FORTRAN 77, it uses graPHIGS, the IBM version of PHIGS, for graphics display. The modeling system is operational on the IBM 5080. However, any device that supports graPHIGS can be used for modeling. As mentioned before, FeatureMod uses the previously discussed feature dictionary to create surface models of features using rational B-splines and allows user interaction using menu picks or UML.

5.2 *Overview of FeatureMod*

FeatureMod is a prototype feature-based modeling system that allows the creation of a design by using a vocabulary of familiar geometric features and shape descriptors. Figure 24 shows the data flow diagram of FeatureMod. A user can interact with the modeler through a modeling and graphics interface. This interface allows a user to create features by accessing the feature dictionary and modeling library. Graphics utilities for transformations and rendering can also be accessed through this interface.

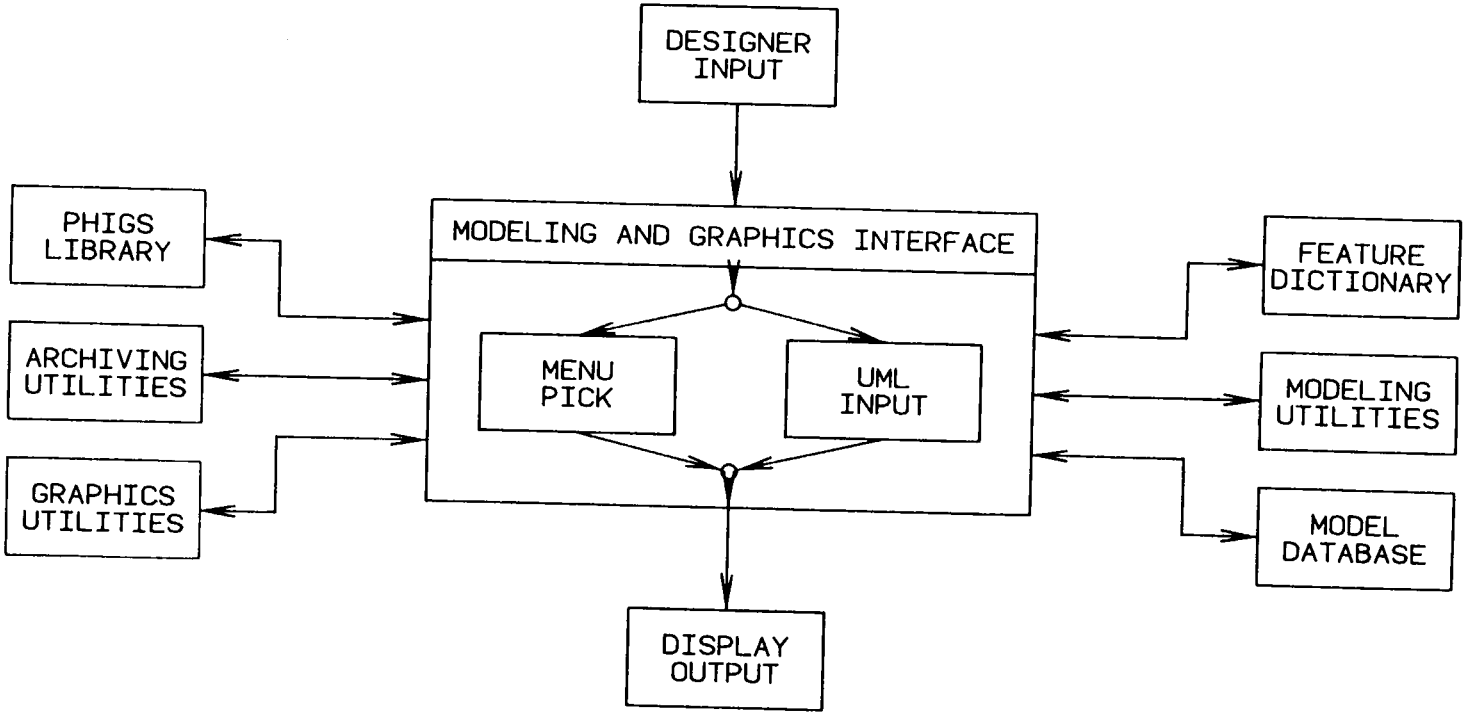


Figure 24. Data Flow Diagram of FeatureMod.

Interaction with the modeling and graphics interface can be achieved by two different methods. The first method utilizes the previously discussed Universal Modeling Language and allows a designer to take a natural language approach towards modeling. The second method is a conventional menu pick method that requires that a user be familiar with the hierarchy of the system.

The graphics library consists of graPHIGS routines which allow display of 3-D graphics primitives such as lines, markers, text, etc. and allows attachment of different attributes to these primitives. The system library contains procedures that allow execution of system functions like store and recall. Transformation library contains routines that will allow a user to display single or multiple orthographic views of an object. Valuators are used to apply window transformations for rotation, pan and zoom. The feature library contains information of different features available for modeling and is used in deciding whether a feature meets the inequality criteria. The modeling library consists of procedures to create a hierarchical feature-based model. The database contains complete information about the geometric model, its attributes and the object hierarchy.

A graphics display, attached to the interface, is mainly used to display the current status of a design and as means of communication between the user and the modeler. The display screen is broken into several areas used for component display, menu display, prompt display, scroll message area, template area, string input and valuator display area. The reserved areas on the screen are shown in Figure 25.

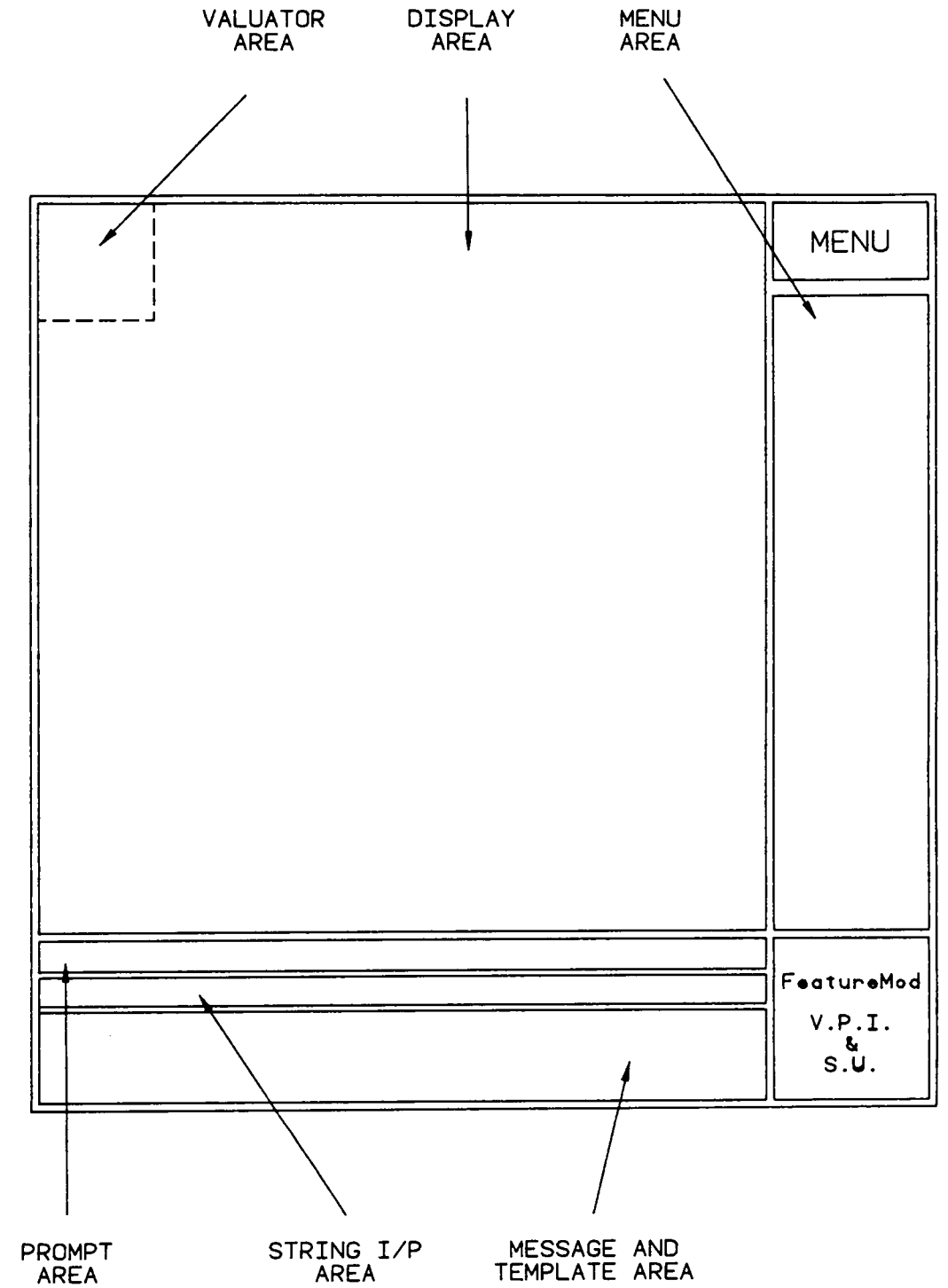


Figure 25. Reserved Areas on Display Screen.

5.3 Object and Menu Hierarchy in FeatureMod

FeatureMod uses a bottom-up approach for model creation. A model consists of assemblies, objects and feature instances. Assemblies occupy the highest levels in the hierarchy and the lowest levels are occupied by features. Thus features can be created in local coordinates and be placed in the parent object's coordinate system. Similarly an object can be placed relatively in an assembly's coordinate space. A representative object hierarchy for a model is shown in Figure 26. A feature can be part of different objects, but an object can be a part of only one assembly. If an object has to be used in more than one assembly, it should be copied before being used again.

The use of menus in the modeler make the establishment of a menu tree necessary. The tree is based on the command sequence that should be executed in order to reach a particular location in it. Thus each menu item is accessible only by a pre-specified path. This means that a user must be familiar with the modeling procedures in order to use the system efficiently. The menu tree is shown in Figure 27.

5.4 Data Storage in FeatureMod

To support easy evaluation of a model the geometric database should represent the model explicitly in terms of features and other data related to them. This section

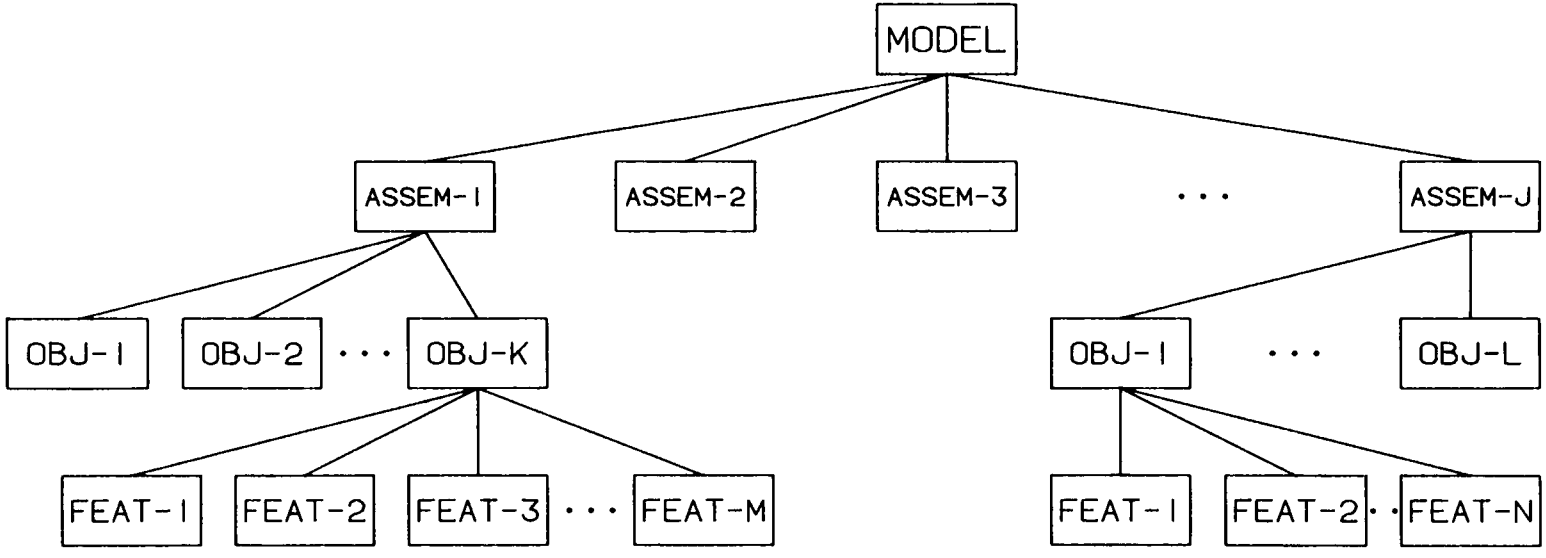


Figure 26. Representative Object Hierarchy in FeatureMod.

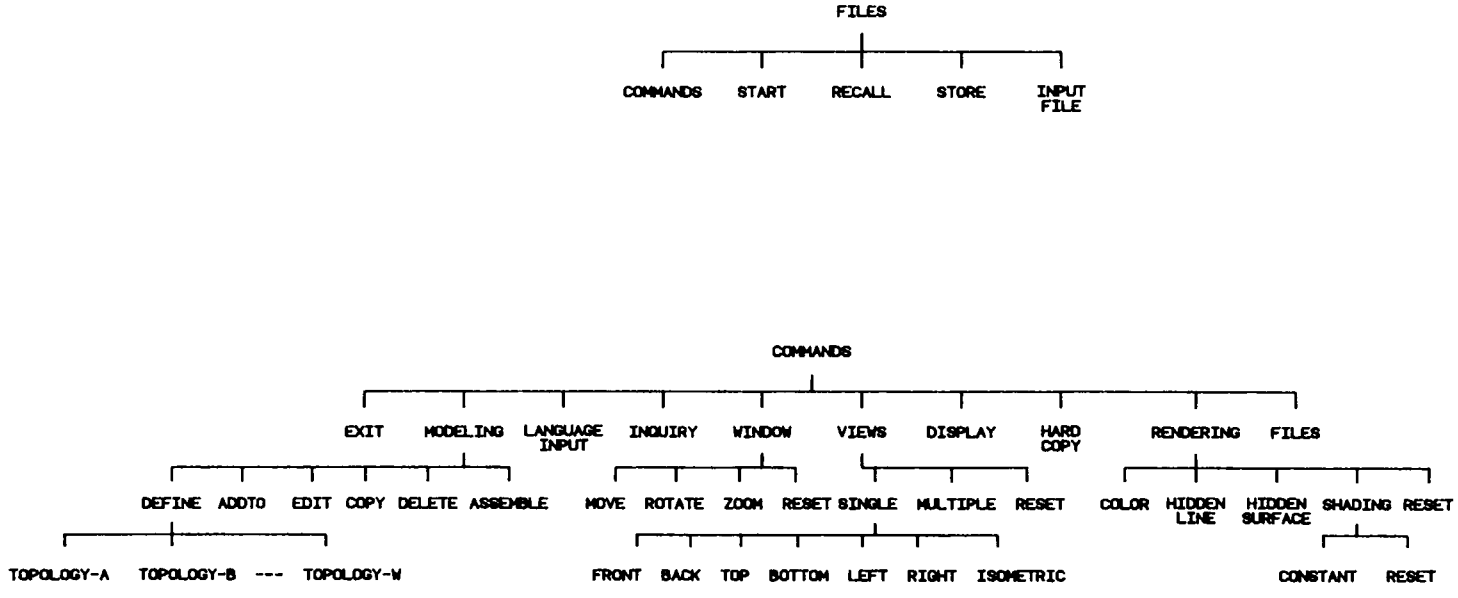


Figure 27. Menu Hierarchy in FeatureMod.

briefly describes how the symbolic representation of the design geometry in FeatureMod provides the needed information. The database contains information about different assemblies, objects and features. Figure 28 gives an idea about the information stored for each component and how they relate to each other.

Assembly: The information consists of assembly number, assembly name, its position and orientation in the global coordinate system and identities of the objects that make up that assembly.

Object: Similar to an assembly, object information consists of object number, object name, parent assembly identity, its position and orientation in the assembly coordinate system and identities of the features that make up that object.

Feature: This is the lowest component in the object hierarchy and hence contains maximum information. Parameters defining the geometry of the features are stored separately for convenience. In order to simplify computation, it stores the control polyhedrons defining the surfaces that make up the feature. In addition it contains information about the feature number, feature name, parent object identity, its position and orientation in the object coordinate system and attributes to be used for computation and display.

5.5 *Design Methodology for FeatureMod*

The procedure required to create a design is simple. The user first has to initiate a model that can later be stored or retrieved. Once initialization is complete the user can now start modeling. First, the user should build an object by using feature instances. Features are created by specifying a feature name from the

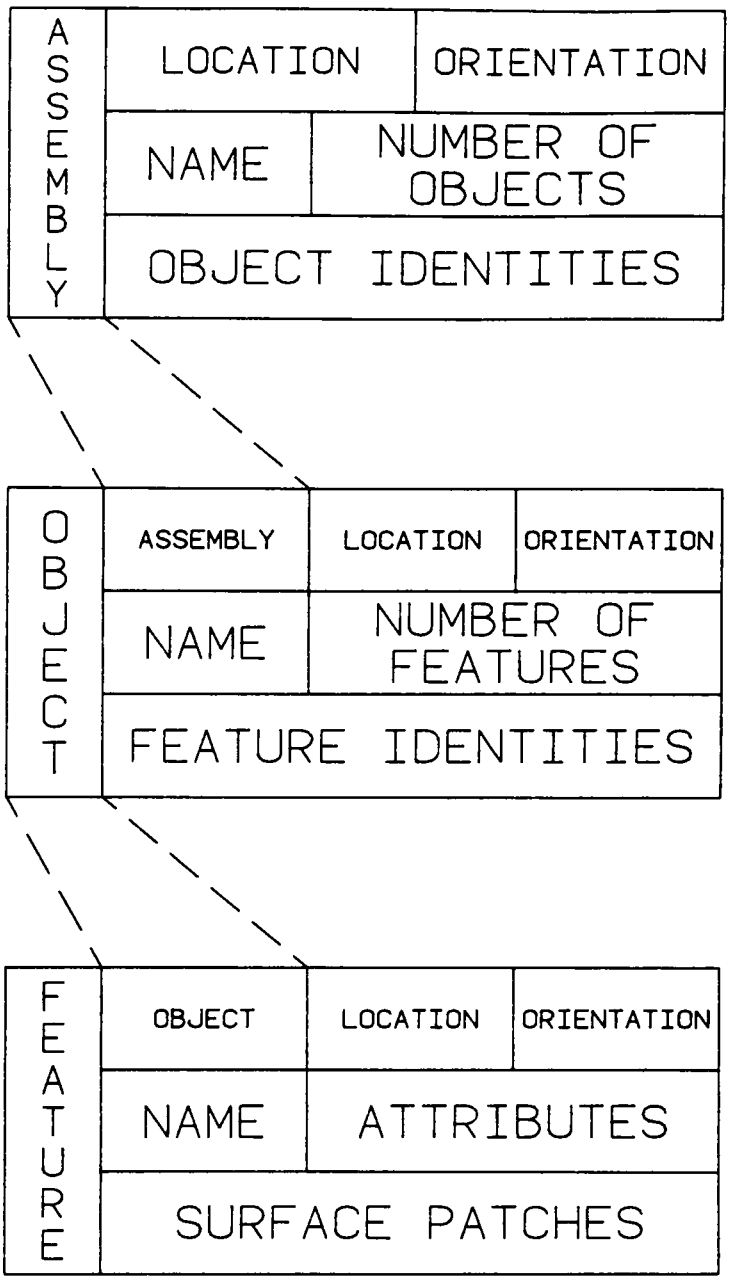
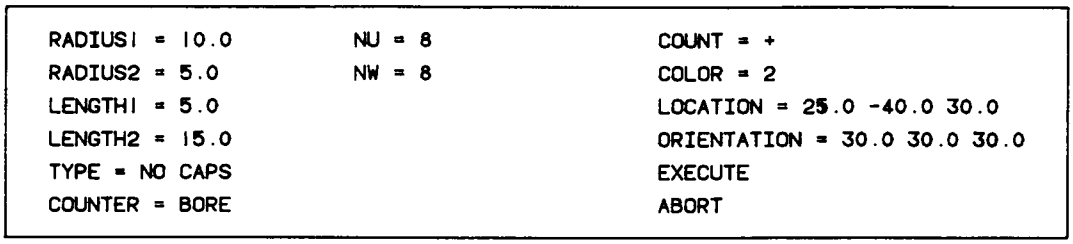


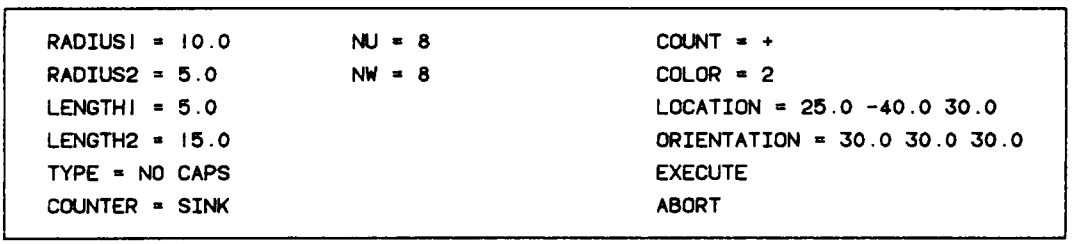
Figure 28. Data Storage in FeatureMod.

dictionary or by picking from the displayed menu. Parameters required for feature definition are obtained by means of templates. A template with default parameters for the feature is displayed in the message area. The user is allowed to change the parameter values within a range specified by inequality constraints. If all parameters are not specified, then the list is completed by using mathematical relations that govern the feature definition. The feature can be positioned by the user and attributes for surface creation and display can be attached to it. In some cases different types of features may be available and can be selected by toggling the menu item. Figure 29a shows a template used for defining a counter-bore. By selecting item 'counter' the same template can now be used for defining a counter-sink. Figure 29b shows the template after toggling. After an object is defined it can be edited by using the 'edit' command or can be added to by the 'addto' command. Once different objects are defined, they can be assembled into one assembly. Assembling is also done with the help of templates which allow positioning of entire objects with respect to the assembly's coordinate axes. As mentioned before, an object can be a part of only one assembly. If an object is already a part of an assembly or is non existent, then the user is warned and is advised to take corrective action. Figure 30 shows example templates used for creating assemblies with warning prompts.

Only one object or assembly is displayed on the screen at a time. This allows the display on the screen to be kept at a minimal level. However, when necessary the entire model can be displayed by using a simple command. Viewing transformations and rendering can be applied to the model at any time for improved visualization of the object.



(A)



(B)

Figure 29. Templates for Creating (a) Counter-bore and (b) Counter-sink.

NON-EXISTENT OBJECT	OBJECT = OBJECT1 LOCATION = 25.0 -40.0 30.0 ORIENTATION = 30.0 30.0 30.0 ASSEMBLY COMPLETE EXECUTE ABORT

OBJECT ALREADY ASSEMBLED	OBJECT = OBJECT1 LOCATION = 25.0 -40.0 30.0 ORIENTATION = 30.0 30.0 30.0 ASSEMBLY COMPLETE EXECUTE ABORT

Figure 30. Sample Templates for Creating Assemblies.

Thus FeatureMod is a working prototype that allows easy creation of complex geometric models and uses high-level representation of the part geometry in the form of features. Used as an interactive CAD environment with color graphics it allows the building of a model by reasonable and logical definitions of features. A dual input system that uses UML and menu picks allow both advanced and entry-level users to use the system efficiently. The combination of advanced surface modeling and parametric programming capabilities allows creation of models that communicate precise geometric information, and, possibly manufacturing information to the production personnel. The ability to simultaneously render image and information makes FeatureMod an elegant tool for enriching and refining the flow of information from engineering to manufacturing. Appendix B gives the entire listing of the routines that make up FeatureMod.

Chapter 6: Case Studies

6.1 Feature Vocabulary

The lists of features, shapes, forms and shape-altering words are combined to form a vocabulary that must match that of a designer. If the list is complete, then any word that a designer uses to describe a shape or form should use a combination of these lists to produce a parametric feature or a synonym of a feature. Thus, it is important that this list be complete.

In other words, a designer who intends to model any part should be able to use these lists, and with proper combinations be able to completely describe the part under consideration. In order to check for the completeness of the feature vocabulary, two parts were used to perform case studies.

6.1.1 Pulley Mount

The first part considered is an assembly used for mounting a shaft and pulley arrangement (Figure 31 and 32). A designer who is given the task of modeling this part might describe it in the following manner:

The part consists of a **thin plate** (used as a base) with two through **holes** and two **tapped-holes**. A **vertical column** is attached to the plate by means of a **weld**. A cantilever **beam** that supports the **pulley** is attached to the column by means of a **deep split ring** to be tightened around the column by means of a **hex-bolt** and **hex-nut** passing through a hole in it. A **rib** provides additional support to the beam. The **shaft** passes through a **tapering hub** attached to the pulley by means of **thick rectangular spokes**. The axial movement of the pulley is restricted by two **collars**, one **screwed** to the shaft and the other forming a part of the cantilever beam. The **top end** of the shaft has a **tapped-hole** for lubrication. The pulley rests on a **conical hub** which is part of the cantilever beam and collar arrangement. Figures 31 and 32 show different views of the part and the (visible) features used to describe the part.

6.1.2 Flow Controlling Valve

The second part considered is a valve which regulates the flow of fluid through an elbow joint (Figure 33). A designer may describe it as follows:

The part consists of two **thick square plates** with rounded corners and a valve **seal** between them. Each plate has four through **holes** which allow the plates to be clamped together by means of **hex-nuts, washers** and **hex-bolts**. The lower plate has **flanges** with **rectangular slots** on two sides. All corners are **rounded**. This is done

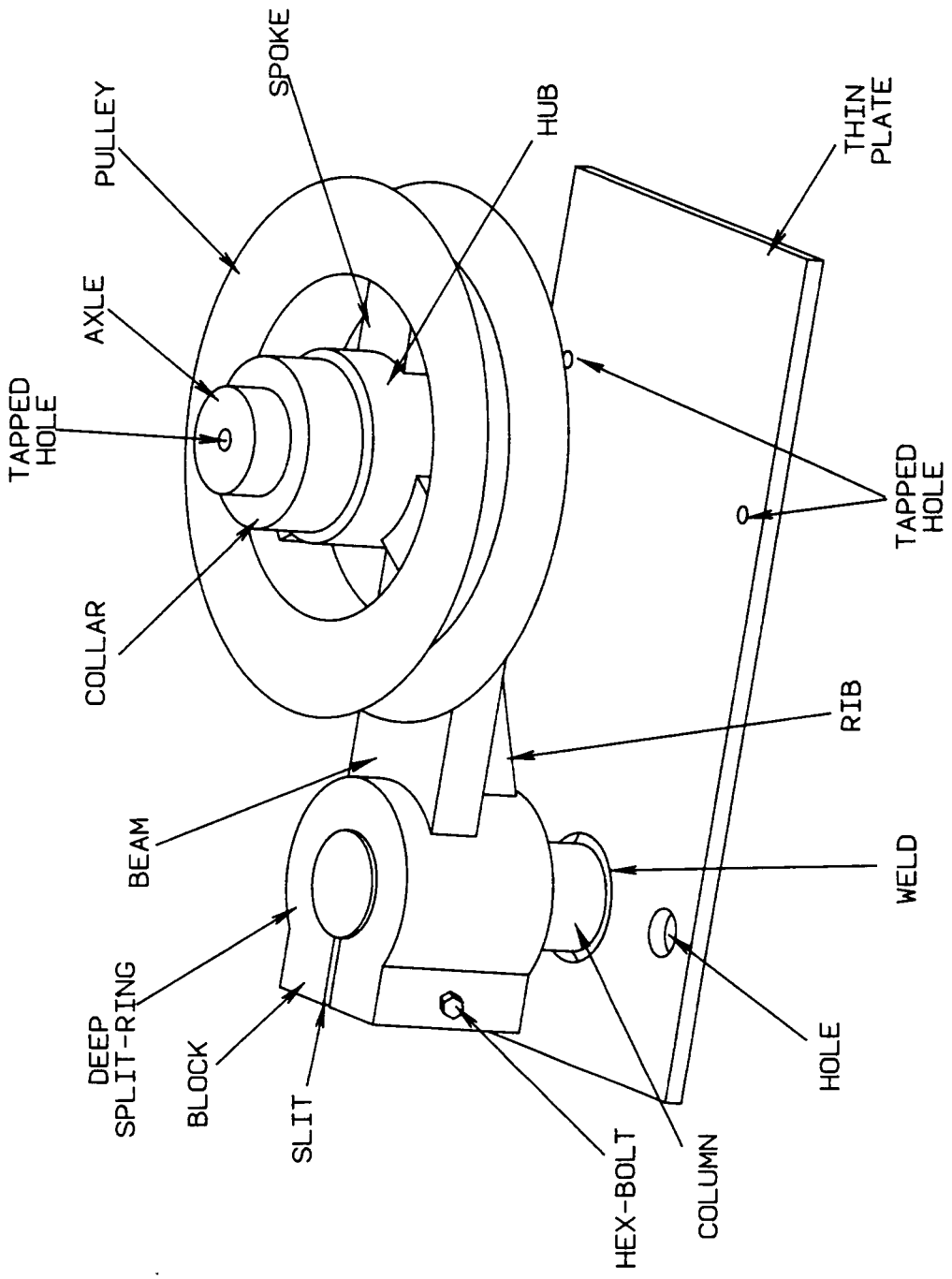


Figure 31. Features Describing a Pulley Mount - I.

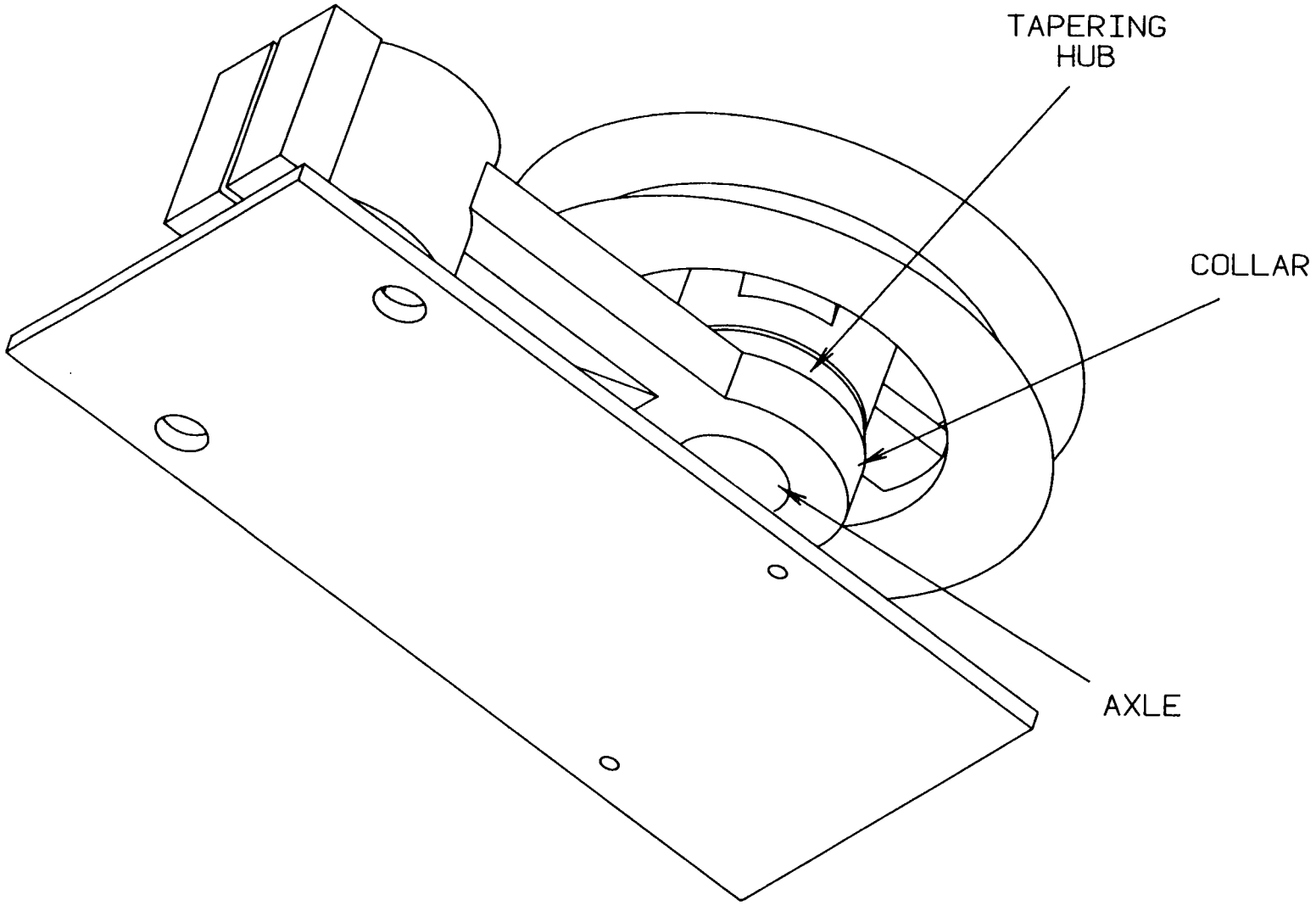


Figure 32. Features Describing a Pulley Mount - II.

by **holes** on either end. The **pipe** is attached to the lower plate by means of a **weld.** The upper plate has a **conical hub** on which sits the **hand wheel** The hand wheel has a conical **dip** on the top and a **cylindrical boss** at the bottom. The flow control **shaft** is **threaded** and goes through a **hole** in the conical hub. Figure 33 shows the part and the (visible) features used to describe the part.

6.2 *FeatureMod Implementation*

The pulley mount was also modeled on the prototype feature-based modeling system that has been developed. The set of features that were identified from the designer's description were used to form the part. Figures 34 through 37 show the modeled part. Since no Boolean operations have been implemented, the model consists of a set of features but is not the final product.

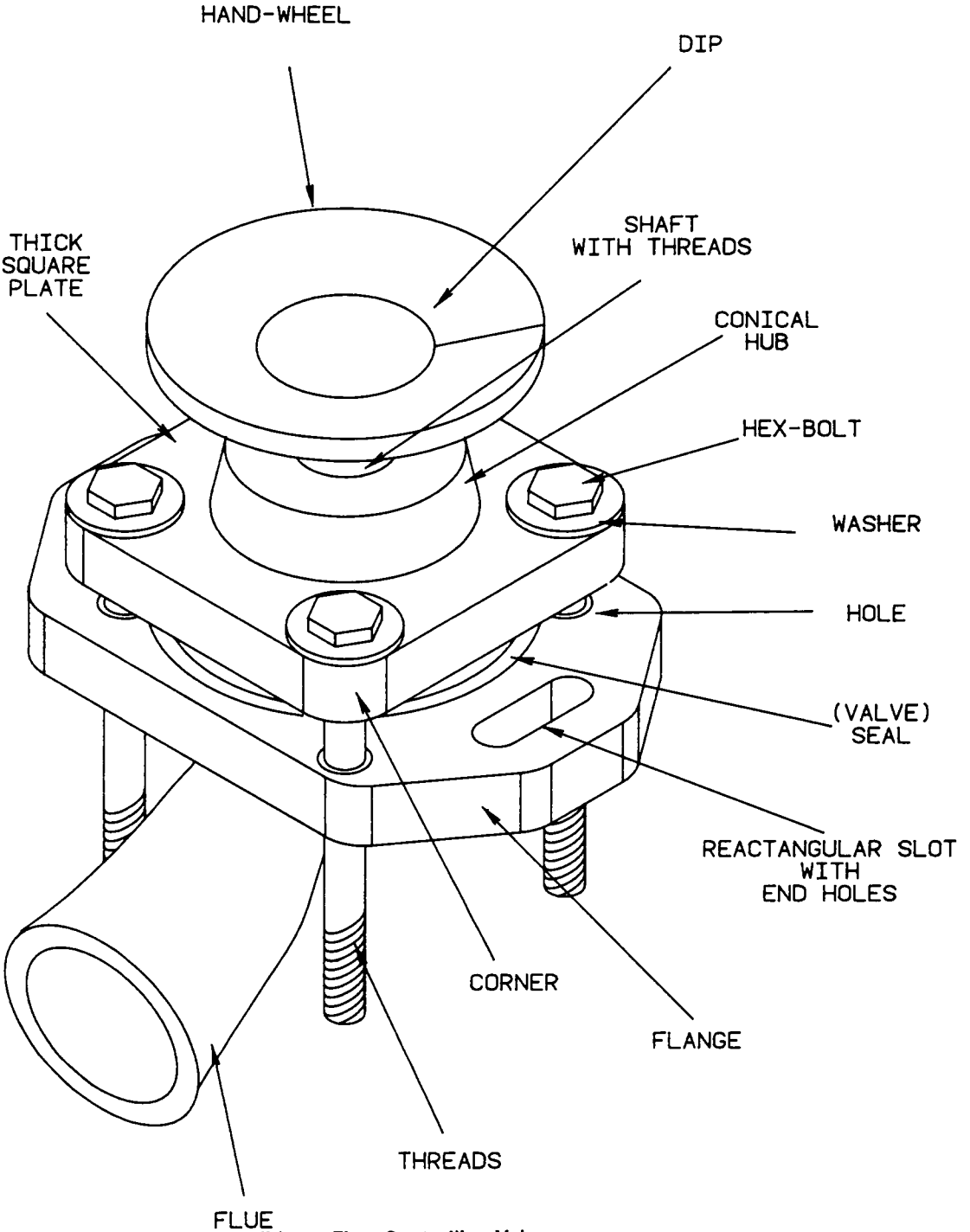


Figure 33. Features Describing a Flow Controlling Valve.

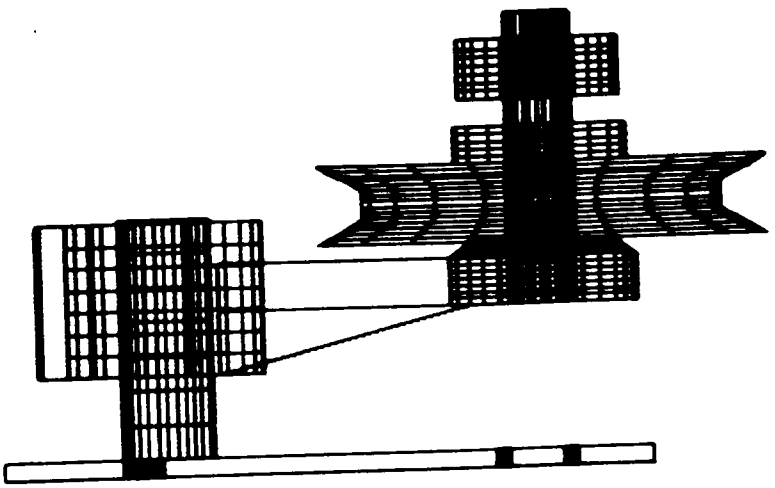


Figure 34. A Pulley Mount Arrangement Created by FeatureMod - Front View.

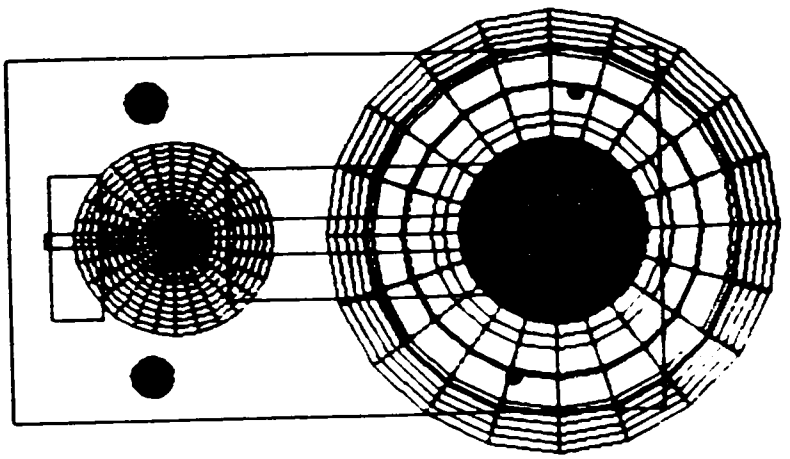


Figure 35. A Pulley Mount Arrangement Created by FeatureMod - Top View.

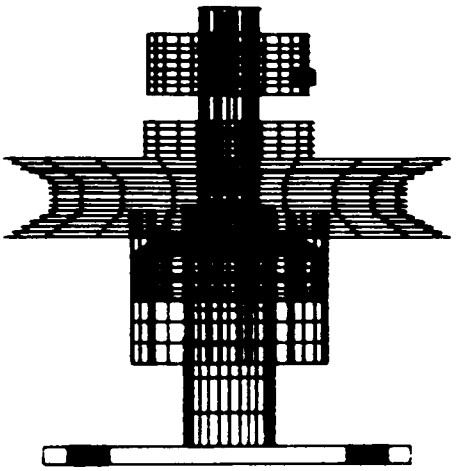


Figure 36. A Pulley Mount Arrangement Created by FeatureMod - Right Side View.

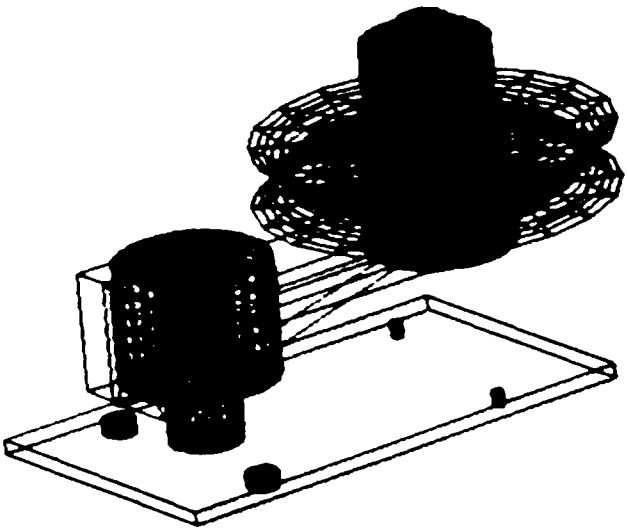


Figure 37. A Pulley Mount Arrangement Created by FeatureMod - Isometric View.

Chapter 7: Conclusions and Recommendations

7.1 Conclusions

There is a great deal of research activity in the area of feature-based modeling. Prior to this, no in-depth study of commonly used design features has been done. A feature-based modeling system can be useful only if it has the capability of representing a large number of features and makes the task of creation quick and easy for the designer.

There is a need to develop methods that not only make the task of feature creation easy, but also make the selection process an intelligent one. The effectiveness of such methods will depend on the number of features that are available for creation and the ease with which a designer can modify them to create different variations. There is also a need to speed up the feature creation procedure. This is mainly governed by the manner in which feature dimensions and other parameters can be specified. If a designer is allowed to use any word describing

shape or form to produce a feature, newer methods for feature selection and creation need to be developed. This clearly precludes the use of conventional menu screens. A natural language approach seems to be necessary. Thus, a method that uses a natural language approach to allow intelligent selection and creation of features seems to be an appropriate step in speeding the design process.

The main objective of this research was to find a suitable approach that bridges the gaps in the existing feature modeling methods. It aims at formulating a generalized procedure for using features as the basis for creation of geometric models and to best exploit the existence of a large vocabulary among designers or production personnel for this purpose. An exhaustive study of English terms used to describe the shapes and forms of geometric features was performed to improve the ease with which design by features can be accomplished. A study of shape altering words that result in additional description or modification of a feature was done. In order to simplify implementation, a method of systematic grouping of features with similar topologies was developed. Features were also classified by shape forms to permit quick access of similar features based on recollection of similar features. A dictionary that allows a designer to better understand the nature and function of a feature was created.

Parametric features were created from groups to allow easy definition, creation and modification. Prototype inequality relations to differentiate between features in the same topology group were established. These rules establish relations between different parameters that define a feature and assist a designer in selecting reasonable values for the parameters. They also assist in creation of a feature through incomplete specification of parameters. A method to compute a set of

parameters using the mathematical inequalities and partial specification of parameters in the set was developed.

The parametric equations and the parameter values decide the points that lie on the surface. Any surface used for representation must interpolate these points. The resulting feature is modeled with rational B-spline surfaces. These surfaces are versatile and allow local control of a shape, but by nature do not interpolate points on the defining control polyhedron. An inverse relation that allows a B-spline surface to interpolate pre-determined points was developed. This approach gives great flexibility in modeling of complex shapes and their surface representation.

To allow a designer to use similar procedures for visualizing and creating features, a natural language approach seemed necessary. A modeling language that allows a designer to use simple English commands to perform different tasks of modeling was developed. It provides a designer with means of easy communication with a modeling system, and hence eases the manner in which he may define a feature.

Finally, a prototype system for designing with features that partially incorporates the above-mentioned work and other requirements for feature-based modeling systems was developed. It allows creation of geometric models using object hierarchy (assemblies, objects and feature instances) and easy placement and aligning of different components. This provides a designer with a flexible tool for using features to create a part.

To conclude, this work provides a designer with a flexible method for feature selection for creation of a part. It provides him/her with an easy method for

specifying commands, since it uses simple English for input. The use of mathematical relations further enhances the ability of a designer to produce reasonable features and also allows incomplete specification of parameters for feature creation.

7.2 Recommendations

The work described in this dissertation is comprehensive but is definitely not complete. The nature of mechanical parts requires that all possible shapes and forms be accounted for. It is obvious that a larger number of topological groups will provide better flexibility in modeling different shapes. There is a need to identify more specialized shapes that can be grouped into newer topologies and to parameterize them. The classification schemes should be further developed to account for any specialized shapes that cannot be grouped currently.

While mathematical rules for some features were established, those for the other features must be compiled. These rules should be flexible and may be defined by a user. The ability to use mathematical relations for computing missing parameters must be further enhanced by developing schemes that can handle non linear relations. The vocabulary of modeling verbs and shape altering words must be enhanced to account for the most complex commands and modifications.

It is recommended that further research be done in processing of natural language input and be used as a base for achieving the ultimate of goal of processing voice input. While uniform rational B-spline surfaces are suitable, it is recommended

that non-uniform rational B-spline surfaces (NURBS) be implemented for added flexibility.

The ability to use a model for mass properties, interference checks and other utilities depends on its completeness. A model is complete only if it is a solid model. Thus, procedures for converting a surface model into a solid must be implemented. Additionally, Boolean operations must be implemented. Boolean operations require the ability to compute the intersection between two solids. The difficulty with which intersections can be found depends on the level at which the problem is to be handled. The lower the level, the easier it is to compute the intersections. An efficient method to compute surface intersections must be implemented to make Boolean operations easier.

References

1. Allen, G., "An Introduction to Solid Modeling", *Computer and Graphics*, Vol. 8, No. 4, pp. 439-447, 1984.

2. Barnhill, R. E., and Boehm, W., *Surfaces in Computer Aided Geometric Design*, North Holland Publishing Company, 1983.

3. Barsky, B. A., *Computer Graphics and Geometric Modeling using Beta-Splines*, Springer-Verlag, Berlin, 1988.

4. Bartels, R. H., Beatty, J. C. and Barsky, B. A., *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann Publishers, Los Altos, California, 1988.

5. Bezier, P., *Mathematical Basis of the UNISURF CAD System*, Butterworths Inc., Boston, 1986.

6. Brodlie, K. W., *Mathematical Methods in Computer Graphics and Design*, Academic Press, London, 1980.

7. Brody, H., "CAD meets CAM", *High Technology*, pp. 12-15, May 1987.

8. Cunningham, J. J. and Dixon, J. R., "Designing with Features: Origin of Features", *Proceedings of ASME International Computers in Engineering Conference and Exhibition*, San Francisco, August 1988.

9. Davidson, P. L., "Computer-Aided Decision Making: An Expert System for the Elaboration of Design Strategies", Computers in Engineering, Vol. 1, 1985.

10. Del Vecchio, A., *Dictionary of Mechanical Engineering*, Littlefield Publishing Company, New Jersey, 1962.

11. Dixon, J. R., and Simmons, M. K., "Creating and Using a Features Data Base", Computers in Mechanical Engineering, pp. 25-33, Nov. 1986.

12. Dixon, J. R., and Dym C. L., "Artificial Intelligence and Geometric Reasoning in Manufacturing Technology", *Applied Mechanical Review*, Vol. 39, No. 9, pp. 1325-1330, 1986.

13. French, T. E., and Svensen C. L., *Mechanical Drawing*, McGraw-Hill Book Company, New York, 1966

14. Gardan, Y., *Numerical Methods for CAD*, Kogan Page Ltd., London, 1985.

15. Giachino, J. W., and Beukema H. J., *Engineering-Technical Drafting and Graphics*, American Technical Society, Chicago, 1964.

16. Gossard, D., Zuffante, R. P., and Sakurai, H., "Representing Dimensions, Tolerances, and Features in MCAE Systems", *IEEE Computer Graphics and Applications*, pp. 51-59, March 1988.

17. Greville, T. N. E., *Theory and Applications of Spline Functions*, Academic Press, London, 1969.

18. Hatvany, J., "CAD - State of the Art and a Tentative Forecast", *Robotics and Computer Integrated Manufacturing*, Vol. 1, No. 1, 1984.

19. Horner, J. G., and Staton A., *Dictionary Of Terms Used In Theory and Practice Of Mechanical Engineering*, Technical Press Inc., London, 1952.

20. Lai, K., "Mechanical Design Simplification Using Function Description Language", *15th North American Manufacturing Research Conference (NAMRC)*, Bethlehem, Pennsylvania, 1987.

21. Lai, K., and Wilson, W. R. D., "FDL - A Language for Function Description and Rationalization in Mechanical Design", *1987 ASME International Computers in Engineering Conference and Exhibition*, New York, August, 1987.

22. Libardi, E. C., Dixon J. R., and Simmons, M. K., "Designing with Features: Design and Analysis of Extrusions as an Example", *Proceedings of Mechanical Design Conference*, Chicago, March 1986.

23. Light, R. L., and Gossard, D., "Modification of Geometric Models through Variational Geometry", *Computer Aided Design*, pp.209-214, 1982.

24. Luby, S. C., Dixon, J. R., and Simmons, M. K., "Designing with Features: Creating and Using a Features Data Base for Evaluation of Manufacturability of Castings", *Computers in Engineering*, Vol. 1, 1986.

25. Mortenson, M. E., *Geometric Modeling*, John Wiley and Sons, New York, 1985.

26. Nayler, J. L., and Nayler, H. F., *Dictionary of Mechanical Engineering*, Newnes Publishers, London, 1967.

27. Nnaji, B. O., and Vishnu, A. K., "A Generalized Shape Descriptor from Wireframe Models on a CAD/CAM system", *Journal of Manufacturing Systems*, Vol. 5, No. 4, pp. 231-241, 1987.

28. Oberg, E., Jones, F. D., and Horton, H. L., *Machinery's Handbook*, Industrial Press Inc., New York, 1988.

29. Parker, S. P., *McGraw-Hill Dictionary of Scientific and Technical Terms*, fourth edition, McGraw-Hill Book Company, New York, 1989.

30. Pratt, M. J., "Synthesis of an Optimal Approach to Form Feature Modeling", *Proceedings of ASME International Computers in Engineering Conference and Exhibition*, San Francisco, August 1988.

31. Pratt, M. J., and Faux, I. D., *Computational Geometry for Design and Manufacture*, John Wiley and Sons, New York, 1979.

32. Piegl, L. and Tiller, W., "Curve and Surface Construction using Rational B-Splines", Computer Aided Design, November 1987.

33. Riesenfeld, R., "Applications of B-Spline Approximation to Geometric Problems of Computer-Aided Design", Ph.D. dissertation, Syracuse University, 1973.

34. Sankar, J., and Myklebust, A., "An Expert System for the Automatic Generation of Geometry Interfaces Between Applications Programs and CAD/CAM Systems", accepted for publication in Computer Aided Design.

35. Schoenberg, I. J., *Approximations with Special Emphasis on Spline Functions*, Academic Press, London, 1969.

36. Serrano, D., and Gossard, D. C., "Combining Mathematical Models with Geometric Models in CAE Systems", Computers in Engineering, Vol. 1, 1986.

37. Shah J. J. and Rogers, M. T., "Feature Based Modeling Shell: Design and Implementation", *Proceedings of ASME International Computers in Engineering Conference and Exhibition*, San Francisco, August 1988.

38. Shah J., Bhatnagar, A. and Hsiao D., "Feature Mapping and Application Shell", *Proceedings of ASME International Computers in Engineering Conference and Exhibition*, San Francisco, August 1988.

39. Takase, H., and Nakajima, N., "A Language for Describing Assembled Machines", *International Symposium on Design and Synthesis*, Tokyo, July, 1984.

40. Tikerpuu, J. and Ullman, D. G., "General Feature-Based Frame Representation for Representing Mechanical Engineering Design Developed from Empirical Data", *Proceedings of ASME International Computers in Engineering Conference and Exhibition*, San Francisco, August 1988.

41. Tiller, W., "Rational B-Splines for Curve and Surface Representation", IEEE Computer Graphics and Applications, pp. 61-69, September 1983.

42. Versprille, K. J., "Computer-Aided Design Applications of Rational B-Spline Approximation Form", Ph.D. dissertation, Syracuse University, 1975.

43. Woo, T. C., "Progress in Shape Modeling", Computer, pp. 40-46, November 1977.

44. Yaslow, S., *Elements of Mechanical Drafting*, Industrial Press Inc., New York, 1969.

Appendix A - Feature Dictionary

FEATURE NAME	:	ABUTMENT
ENGLISH DEFI	:	A surface or mass provided to withstand thrust.
GROUP CLASSI	:	TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS	:	
RELATIONS	:	
DEFINING PARAMETERS	:	L, B1, B2, H
END		
FEATURE NAME	:	AIGUILLE
ENGLISH DEFI	:	A slender form of drill used for boring or drilling a
ENGLISH DEFI	:	blast hole in a rock.
GROUP CLASSI	:	TOPOLOGY GROUP - A FORM GROUP - B1, B5B
SYNONYMS	:	DRILL
RELATIONS	:	
DEFINING PARAMETERS	:	R, L
END		
FEATURE NAME	:	ANGLE
ENGLISH DEFI	:	The figure or space made by the meeting of two straight
ENGLISH DEFI	:	lines or two plane surfaces.
GROUP CLASSI	:	TOPOLOGY GROUP - H FORM GROUP - B2
SYNONYMS	:	BEND, BRACKET, CORNER, ELBOW
RELATIONS	:	
DEFINING PARAMETERS	:	L, B, H, T, R, A
END		
FEATURE NAME	:	ANNULAR-HOLE
ENGLISH DEFI	:	A space between the casing and wall of a hole or between
ENGLISH DEFI	:	a drill pipe and casing.
GROUP CLASSI	:	TOPOLOGY GROUP - C FORM GROUP - B1, C11
SYNONYMS	:	
RELATIONS	:	
DEFINING PARAMETERS	:	R1, R2, L
END		
FEATURE NAME	:	APERTURE
ENGLISH DEFI	:	A small opening or an orifice.
GROUP CLASSI	:	TOPOLOGY GROUP - B FORM GROUP - B1, C11
SYNONYMS	:	INLET, OPENING, ORIFICE, PERFORATION, PUNCTURE
RELATIONS	:	
DEFINING PARAMETERS	:	R, L
END		
FEATURE NAME	:	APEX

ENGLISH DEFI : The highest point or peak.
GROUP CLASSI : TOPOLOGY GROUP - K FORM GROUP - B1, C4
SYNONYMS : PEAK, VERTEX
RELATIONS :
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : APRON
ENGLISH DEFI : A plate serving to cover or protect a machine.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : ARBOR
ENGLISH DEFI : A shaft, spindle or axle in different machines.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C1
SYNONYMS : AXLE, MANDREL, SHAFT, SPINDLE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : ARBOR-HOLE
ENGLISH DEFI : A hole in a revolving cutter or grinding wheel for
ENGLISH DEFI : mounting it on an arbor.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - B1, C11
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : ARC
ENGLISH DEFI : Anything with a shape of an arch, a curve or part of a
ENGLISH DEFI : circle.
GROUP CLASSI : TOPOLOGY GROUP - D FORM GROUP - B3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : ARCH
ENGLISH DEFI : A curved structure spanning an opening.
GROUP CLASSI : TOPOLOGY GROUP - D FORM GROUP - B3, C1
SYNONYMS : b
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : ARM
ENGLISH DEFI : A side of an angle.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : ARRIS
ENGLISH DEFI : A short edge or angle at the junction of two surfaces,
ENGLISH DEFI : especially moldings and raised edges.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : ASTRAGAL
ENGLISH DEFI : A small convex molding.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :

DEFINING PARAMETERS : A, B
END
FEATURE NAME : AXLE
ENGLISH DEFI : A supporting member that carries wheels and rotates with
ENGLISH DEFI : the wheel.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : ARBOR, MANDREL, SHAFT, SPINDLE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : BAFFLE-PLATE
ENGLISH DEFI : A plate that regulates the flow of a fluid as in a steam
ENGLISH DEFI : boiler, flue or gasoline muffler.
GROUP CLASSI : TOPOLOGY GROUP - C, E FORM GROUP - B1, B2, C3, C12
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BALL
ENGLISH DEFI : A spherical or nearly spherical object.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : GLOBE, SPHERE
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : BAND
ENGLISH DEFI : A flat flexible strip of any material often used for
ENGLISH DEFI : binding or securing.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B5A, C3
SYNONYMS : BELT, RIBBON, STRIP
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BAR
ENGLISH DEFI : An elongated piece of metal of simple uniform cross
ENGLISH DEFI : section dimensions, usually rectangular, circular or
ENGLISH DEFI : hexagonal.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS : CORE, ROD
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BARREL
ENGLISH DEFI : A container having a circular lateral cross section that
ENGLISH DEFI : is largest in the middle and ends that are flat.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : BASE
ENGLISH DEFI : The lowest or supporting part of anything.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B1, B2, C3
SYNONYMS : BED, BEDDING
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BASKET
ENGLISH DEFI : A light weight container with perforations.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :

RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : BEAD
ENGLISH DEFI : A small, usually round object.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : BEAM
ENGLISH DEFI : A body whose function is to carry lateral loads and
ENGLISH DEFI : bending moments.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BED
ENGLISH DEFI : The part of a machine having precisely machined ways or
ENGLISH DEFI : bearing surface which supports or aligns other machine
ENGLISH DEFI : parts.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : BASE, BEDDING
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BEDDING
ENGLISH DEFI : That which forms a bed or foundation.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : BASE, BED
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BELT
ENGLISH DEFI : An endless band of flexible material for transmitting
ENGLISH DEFI : power from one wheel or shaft to another.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B5A, C3
SYNONYMS : BAND, RIBBON, STRIP
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BENCH
ENGLISH DEFI : A table for mechanical work.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BEND
ENGLISH DEFI : To assume the form of a curve; crook; bow.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B2, B3
SYNONYMS : ANGLE, BRACKET, CORNER, ELBOW
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : BEVEL
ENGLISH DEFI : An inclination of two surfaces other than perpendicular.
GROUP CLASSI : TOPOLOGY GROUP - V FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, A.R.

END
FEATURE NAME : BILLET
ENGLISH DEFI : A semifinished, short, thick, bar in form of a cylinder or
ENGLISH DEFI : rectangular prism.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BLOCK
ENGLISH DEFI : A solid piece of metal with one or more flat surface.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BOLSTER-PLATE
ENGLISH DEFI : A plate fixed on the bed of a power press to locate and
ENGLISH DEFI : support the die assembly.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : BUTTON, CAP, COVER, DIAL, DISK, HATCH, PLATE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : BORE
ENGLISH DEFI : To cut or drill a hole in or through a work piece.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - B1, B5A, C11,
GROUP CLASSI : TOPOLOGY GROUP - FORM GROUP - C13
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : BOSS
ENGLISH DEFI : Protuberance on a part to add strength, facilitate
ENGLISH DEFI : assembly, provide for fastening, etc.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BOW
ENGLISH DEFI : A part shaped as an arc or a polygon.
GROUP CLASSI : TOPOLOGY GROUP - D FORM GROUP - B3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : BOX
ENGLISH DEFI : A receptacle or a case usually having a lid.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B, C3
SYNONYMS : CASE, CASING, FRAME
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : BRACE
ENGLISH DEFI : A diagonally placed structural member that withstands
ENGLISH DEFI : tension and compression.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END

FEATURE NAME : BRACKET
ENGLISH DEFI : A brace used to strengthen an angle; A piece of metal
ENGLISH DEFI : used to support an object.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B3
SYNONYMS : ANGLE, BEND, CORNER, ELBOW
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : BREAK
ENGLISH DEFI : To separate into pieces or fragments; to crack.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : CRACK, CREVICE, CUT, FISSURE, FRACTURE, RUPTURE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : BROACH
ENGLISH DEFI : The shaping of a part by pushing or pulling a broach
ENGLISH DEFI : across the surface or through an existing hole in the
ENGLISH DEFI : work piece.
GROUP CLASSI : TOPOLOGY GROUP - A, B FORM GROUP - B1, C13
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R, L
END
FEATURE NAME : BUBBLE
ENGLISH DEFI : A liquid globule filled with air or other gases.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : DROP, DROPLET, GLOBULE, TEAR
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : BUCKET
ENGLISH DEFI : A deep cylindrical vessel.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : BULB
ENGLISH DEFI : A rounded protuberance.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : BULGE
ENGLISH DEFI : A protuberant, rounded part.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1, C7
SYNONYMS : BUMP, ENTASIS, SWELL, SWELLING
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : BULLET
ENGLISH DEFI : Any small ball or cylindrical object.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : BUMP
ENGLISH DEFI : A protuberance or an uneven place.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1, C7

SYNONYMS : BULGE, ENTASIS, SWELL, SWELLING
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : BUSHING
ENGLISH DEFI : A metallic lining for a hole designed to insulate or
ENGLISH DEFI : prevent abrasion between moving parts.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A, C14
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : BUTTON
ENGLISH DEFI : A knob or disk serving as a fastening.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C1
SYNONYMS : BOLSTER-PLATE, CAP, COVER, DIAL, DISK, HATCH, PLATE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : CABLE
ENGLISH DEFI : A stranded rope like assembly of wire or fiber.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : WIRE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : CAGE
ENGLISH DEFI : A boxlike structure made up of wires or bars.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : CAN
ENGLISH DEFI : A cylindrical metal vessel or container, usually with an
ENGLISH DEFI : open top or a removable cover.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CANAL
ENGLISH DEFI : A passage or a duct or a tube.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CANTILEVER
ENGLISH DEFI : A long structural member, as a truss, beam or slab lying
ENGLISH DEFI : across the supports with projecting arms in balance.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CAP
ENGLISH DEFI : An object that acts as a cover.
GROUP CLASSI : TOPOLOGY GROUP - A, E, O FORM GROUP - B1, B2, B5A, C1
GROUP CLASSI : TOPOLOGY GROUP - FORM GROUP - C2, C12
SYNONYMS : BOLSTER-PLATE, BUTTON, COVER, DIAL, DISK, HATCH, PLATE
RELATIONS :
DEFINING PARAMETERS : R, L

DEFINING PARAMETERS : L, B1, B2, H
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : CAPSULE
ENGLISH DEFI : A thin covering, container or seal.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1,B5A
SYNONYMS : CARTRIDGE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CARTRIDGE
ENGLISH DEFI : Any small container or casing.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS : CAPSULE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CASE
ENGLISH DEFI : A box for containing something.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B, C3
SYNONYMS : BOX, CASING, FRAME
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : CASING
ENGLISH DEFI : A protective case or covering.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B, C3
SYNONYMS : BOX, CASE, FRAME
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : CAVE
ENGLISH DEFI : A chamber beneath the earth, in a mountain.
GROUP CLASSI : TOPOLOGY GROUP - D FORM GROUP - B1, C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CAVITY
ENGLISH DEFI : A hollow or sunken space; hole.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C2
SYNONYMS : CHANNEL, FURROW, GROOVE, RUT, SLOT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CELL
ENGLISH DEFI : A small compartment, receptacle or cavity.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B
SYNONYMS : CHAMBER, COMPARTMENT
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : CHAMBER
ENGLISH DEFI : An enclosed space or cavity; a sleeve or channel.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B
SYNONYMS : CELL, COMPARTMENT
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : CHAMFER
ENGLISH DEFI : To cut away a corner of; a cut a furrow in.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B2
SYNONYMS :

RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : CHANNEL
ENGLISH DEFI : A tubular passage, groove, furrow or a cavity.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C2, C10
SYNONYMS : CAVITY, FURROW, GROOVE, RUT, SLOT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CHASE
ENGLISH DEFI : To ornament by embossing or engraving.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B5B, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : CHASED-GROOVE
ENGLISH DEFI : A groove or a slot.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B5B
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CHASSIS
ENGLISH DEFI : A flat, rectangular frame on which a body is mounted.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : CHIMNEY
ENGLISH DEFI : A tube, usually of glass or a funnel.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CHORD
ENGLISH DEFI : A line segment which intersects a curve or surface only
ENGLISH DEFI : at the end points of the segment.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : CHUTE
ENGLISH DEFI : An inclined through or a vertical passage.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B5B
SYNONYMS : FLUME
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CLEAT
ENGLISH DEFI : A strip of metal fastened across or projecting from a
ENGLISH DEFI : surface to strengthen, support or provide a grip.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A
SYNONYMS : FIN, RIB
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CLEAVAGE

ENGLISH DEFI : A split or a cleft.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : CLEFT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CLEFT
ENGLISH DEFI : A fissure, divide or rift that separates partially or
ENGLISH DEFI : completely.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : CLEAVAGE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CLEVIS
ENGLISH DEFI : A U-shaped shackle for connecting a rod and a pin.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : CLIP
ENGLISH DEFI : A device that fastens by gripping, clasping or hooking one
ENGLISH DEFI : part to another.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A, C14
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CLOUGH
ENGLISH DEFI : A ravine or a narrow valley.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : COIL
ENGLISH DEFI : A series of concentric rings or spirals.
GROUP CLASSI : TOPOLOGY GROUP - R FORM GROUP - B1, B5B
SYNONYMS : CURL, HELIX, SPIRAL, SPRING
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N
END
FEATURE NAME : COIN
ENGLISH DEFI : A thin piece of metal, usually circular in shape.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : COLLAR
ENGLISH DEFI : A ring placed around an object to restrict its motion,
ENGLISH DEFI : hold it in place, or cover an opening.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : COLLET
ENGLISH DEFI : A split, coned sleeve.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H

END
FEATURE NAME : COLUMN
ENGLISH DEFI : A vertical shaft designed to bear axial loads.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS : PILLAR
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : COMPARTMENT
ENGLISH DEFI : A part or sub-division of an enclosed space.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B
SYNONYMS : CELL, CHAMBER
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : CONE
ENGLISH DEFI : A solid having a circle as its base and tapering evenly
ENGLISH DEFI : on all its surfaces to a point.
GROUP CLASSI : TOPOLOGY GROUP - V FORM GROUP - B1, B5A, C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, A.R.
END
FEATURE NAME : CONIC
ENGLISH DEFI : An object with a shape of a cone.
GROUP CLASSI : TOPOLOGY GROUP - V FORM GROUP - B1, B5A, C4
SYNONYMS : CONICAL
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, A.R.
END
FEATURE NAME : CONICAL
ENGLISH DEFI : An object with a shape of a cone.
GROUP CLASSI : TOPOLOGY GROUP - V FORM GROUP - B1, B5A, C4
SYNONYMS : CONIC
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, A.R.
END
FEATURE NAME : CONSTRICTION
ENGLISH DEFI : Narrowing of a channel or cylindrical member.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : CONVEX
ENGLISH DEFI : Curving outward; as the exterior of a globe.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : CORD
ENGLISH DEFI : A string or small rope; twine.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : ROPE, STRING, TWINE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : CORE
ENGLISH DEFI : The central or innermost part of a ring, similar to a
ENGLISH DEFI : rod or shaft.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3

SYNONYMS : BAR, ROD
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CORNER
ENGLISH DEFI : The point or edge formed due to the meeting of two lines
ENGLISH DEFI : or two surfaces.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B2, B3, C4
SYNONYMS : ANGLE, BEND, BRACKET, ELBOW
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : COTTER
ENGLISH DEFI : A conical key, wedge or pin that is split lengthwise.
GROUP CLASSI : TOPOLOGY GROUP - V FORM GROUP - B1, B5A, C4
SYNONYMS : KEY, WEDGE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, A.R.
END
FEATURE NAME : COUNTER-BORE
ENGLISH DEFI : A depression created by cutting the edge of a hole to
ENGLISH DEFI : allow placing of a component.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B1, C11, C13
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L1, L2
END
FEATURE NAME : COUNTER-SINK
ENGLISH DEFI : A depression created by cutting the edge of a hole to
ENGLISH DEFI : allow placing of a component.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B1, C11, C13
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L1, L2
END
FEATURE NAME : COVER
ENGLISH DEFI : A plate whose primary function is to protect or conceal.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3
GROUP CLASSI : TOPOLOGY GROUP - FORM GROUP - C12
SYNONYMS : BOLSTER-PLATE, BUTTON, CAP, DIAL, DISK, HATCH, PLATE
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CRACK
ENGLISH DEFI : A fissure; to break without separation of parts.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : BREAK, CREVICE, CUT, FISSURE, FRACTURE, RUPTURE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CRATER
ENGLISH DEFI : A large bowl shaped topographic depression with steep
ENGLISH DEFI : sides.
GROUP CLASSI : TOPOLOGY GROUP - L FORM GROUP - B1, C2, C8
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : CRESCENT
ENGLISH DEFI : Any part that has one convex and one concave edge.
GROUP CLASSI : TOPOLOGY GROUP - D FORM GROUP - B3

SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CREVICE
ENGLISH DEFI : A fissure or a crack; cleft.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C11
SYNONYMS : BREAK, CRACK, CUT, FISSURE, FRACTURE, RUPTURE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CROWN
ENGLISH DEFI : The top dome of a surface.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : CUBE
ENGLISH DEFI : A solid bounded by six equal squares and having all its
ENGLISH DEFI : angles right angles.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CUP
ENGLISH DEFI : A small, open vessel often with a handle.
GROUP CLASSI : TOPOLOGY GROUP - O FORM GROUP - B1, C2
SYNONYMS : LADLE, SCOOP
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : CURL
ENGLISH DEFI : To twist into ringlets or curves to form a spiral shape.
GROUP CLASSI : TOPOLOGY GROUP - R FORM GROUP - B1, B5B
SYNONYMS : COIL, HELIX, SPIRAL, SPRING
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N
END
FEATURE NAME : CURVE
ENGLISH DEFI : A line continuously bent, as an arc of a circle.
GROUP CLASSI : TOPOLOGY GROUP - D FORM GROUP - B3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : CUSP
ENGLISH DEFI : A pointed or rounded projection.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : CUT
ENGLISH DEFI : An opening created due to penetration with a sharp edge;
ENGLISH DEFI : gash; pierce.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : BREAK, CRACK, CREVICE, FISSURE, FRACTURE, RUPTURE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : CYLINDER

ENGLISH DEFI : A solid bounded by a cylindrical surface and two parallel
ENGLISH DEFI : planes.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : DEEP
ENGLISH DEFI : Having a depth or dimension.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : DEFLATION
ENGLISH DEFI : To reduce or restrict.
GROUP CLASSI : TOPOLOGY GROUP - L FORM GROUP - B1, C2, C8
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : DENT
ENGLISH DEFI : A small depression made by striking or pressing.
GROUP CLASSI : TOPOLOGY GROUP - L FORM GROUP - B1, C2, C8
SYNONYMS : DEPRESSION
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : DEPRESSION
ENGLISH DEFI : A low or depressed place or surface.
GROUP CLASSI : TOPOLOGY GROUP - L FORM GROUP - B1, C2, C8
SYNONYMS : DENT
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : DEPTH
ENGLISH DEFI : The state or degree of being deep.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : DIAL
ENGLISH DEFI : Any graduated circular plate or face or a rotating disk.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : BOLSTER-PLATE, BUTTON, CAP, COVER, DISK, HATCH, PLATE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : DIAPHRAGM
ENGLISH DEFI : A thin sheet placed between parallel parts of a member to
ENGLISH DEFI : increase its rigidity.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : DILATION
ENGLISH DEFI : To widen or enlarge.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - C1, C7
SYNONYMS :
RELATIONS :

DEFINING PARAMETERS : R, L
END
FEATURE NAME : DIP
ENGLISH DEFI : A hollow or depression.
GROUP CLASSI : TOPOLOGY GROUP - M FORM GROUP - C2, C8
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : DISH
ENGLISH DEFI : An open, concave, usually shallow container.
ENGLISH DEFI : A hollow or depression.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : DISK
ENGLISH DEFI : A fairly flat circular plate.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : BOLSTER-PLATE, BUTTON, CAP, COVER, DIAL, HATCH, PLATE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : DITCH
ENGLISH DEFI : A long narrow trench dug in the ground.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : DOME
ENGLISH DEFI : A roof resembling an inverted cup or hemisphere.
GROUP CLASSI : TOPOLOGY GROUP - O FORM GROUP - B1, C1, C2
SYNONYMS : HEMISPHERE
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : DONUT
ENGLISH DEFI : A small cake having a hole in the center.
GROUP CLASSI : TOPOLOGY GROUP - S FORM GROUP - B1, B5A
SYNONYMS : DOUGHNUT
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : DOUGHNUT
ENGLISH DEFI : A small cake having a hole in the center.
GROUP CLASSI : TOPOLOGY GROUP - S FORM GROUP - B1, B5A
SYNONYMS : DONUT
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : DOWEL
ENGLISH DEFI : A headless cylindrical pin or peg fitted tightly into
ENGLISH DEFI : adjacent holes of two pieces so as to hold them together.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C4
SYNONYMS : NAIL, NEEDLE, PEG, PIN, STUD, TACK
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : DRIFT-PIN
ENGLISH DEFI : A tapered steel pin used to bring rivet holes fair in
ENGLISH DEFI : assembling steel work.

GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : DRILL
ENGLISH DEFI : A tool used for boring holes in hard substances.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : AIGUILLE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : DROP
ENGLISH DEFI : Something resembling a drop in shape and size.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : BUBBLE, DROPLET, GLOBULE, TEAR
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : DROPLET
ENGLISH DEFI : Something resembling a drop in shape and size.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : BUBBLE, DROP, GLOBULE, TEAR
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : DRUM
ENGLISH DEFI : A hollow cylindrical container.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : DUCT
ENGLISH DEFI : Any tube, canal or passage through which a fluid is
ENGLISH DEFI : conveyed.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B2B
SYNONYMS : FLUE, GLAND, PIPE, TUBE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : DUNE
ENGLISH DEFI : A hill of loose sand.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : EDGE
ENGLISH DEFI : A bounding or dividing curve.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B4
SYNONYMS : LINE
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : ELBOW
ENGLISH DEFI : A joint at the bend of an object.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B2
SYNONYMS : ANGLE, BEND, BRACKET, CORNER
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : ELLIPSE

ENGLISH DEFI : An oval shaped curve; a conic section.
GROUP CLASSI : TOPOLOGY GROUP - W FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : ELLIPSOID
ENGLISH DEFI : A solid of which every plane surface is a ellipse or a
ENGLISH DEFI : circle.
GROUP CLASSI : TOPOLOGY GROUP - W FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : EMBOSS
ENGLISH DEFI : A raised pattern on a surface produced by means of a die.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : ENTASIS
ENGLISH DEFI : A slight swelling.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - C1
SYNONYMS : BUMP, BULGE, SWELL, SWELLING
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : FACE
ENGLISH DEFI : The front or principal surface of any object.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : PLANE, SHEET
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FENCE
ENGLISH DEFI : A structure of rails, stakes, strung wire, etc., erected
ENGLISH DEFI : as an enclosure, barrier or boundary.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : FILLERS
ENGLISH DEFI : Either plate or ring fills used to take up space.
GROUP CLASSI : TOPOLOGY GROUP - A, C, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FILLET
ENGLISH DEFI : A rounded filling of the internal angle between two
ENGLISH DEFI : surfaces.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : FILLING
ENGLISH DEFI : Material used to fill cavities.
GROUP CLASSI : TOPOLOGY GROUP - A, C, E FORM GROUP - B1, B2, B5A, C3

SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FILM
ENGLISH DEFI : A thin covering or layer.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : FOIL, FOLD, LEAF, MEMBRANE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FILTER
ENGLISH DEFI : A porous article or material for separating matter.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FIN
ENGLISH DEFI : A projecting plate, structure, appendage or attachment.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B2, C3
SYNONYMS : CLEAT, RIB
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FISSURE
ENGLISH DEFI : A long narrow opening, cleft or furrow.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : BREAK, CRACK, CREVICE, CUT, FRACTURE, RUPTURE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FLAKE
ENGLISH DEFI : A small, thin piece peeled or split off from a surface.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FLANGE
ENGLISH DEFI : A projecting rim or a collar of a mechanical part design
ENGLISH DEFI : to aid attachment or to increase stiffness.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B1, B5A, C11
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L1, L2, B1, B2, R, T
END
FEATURE NAME : FLUE
ENGLISH DEFI : A pipe or a tube through which smoke, hot air, steam is
ENGLISH DEFI : drawn off.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B2, B5A
SYNONYMS : DUCT, GLAND, PIPE, TUBE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : FLUME
ENGLISH DEFI : A chute or a through.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C3

SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FLUTE
ENGLISH DEFI : A groove having a curved section, especially when parallel
ENGLISH DEFI : to the main axis.
GROUP CLASSI : TOPOLOGY GROUP - N FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : FOIL
ENGLISH DEFI : A metal hammered or rolled into thin, pliant sheets.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : FILM, FOLD, LEAF, MEMBRANE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FOLD
ENGLISH DEFI : To wrap up or enclose with a thin covering.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - C9
SYNONYMS : FILM, FOIL, LEAF, MEMBRANE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FRACTURE
ENGLISH DEFI : A break, crack or rupture.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : BREAK, CRACK, CREVICE, CUT, FISSURE, RUPTURE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : FRAME
ENGLISH DEFI : A case or border made to enclose something.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B
SYNONYMS : BOX, CASE, CASING
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : FURROW
ENGLISH DEFI : Any long, narrow, deep depression as a groove, rut or deep
ENGLISH DEFI : wrinkle.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : CAVITY, CHANNEL, GROOVE, RUT, SLOT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : GAP
ENGLISH DEFI : An opening, wide crack or deep notch.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : GASKET
ENGLISH DEFI : A packing, usually in form of a sheet or ring to make a
ENGLISH DEFI : joint leakproof.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A, C14
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END

FEATURE NAME : GAUZE
ENGLISH DEFI : Any tiny, open-mesh material.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : GIB
ENGLISH DEFI : A plate of metal or other material machined to hold
ENGLISH DEFI : other parts in place.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : GLAND
ENGLISH DEFI : A long cylindrical body with a hollow center, generally
ENGLISH DEFI : used for conveying something.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS : DUCT, FLUE, PIPE, TUBE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : GLOBE
ENGLISH DEFI : A sphere.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : BALL, SPHERE
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : GLOBULE
ENGLISH DEFI : A tiny sphere or drop.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : BUBBLE, DROP, DROPLET, TEAR
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : GORGE
ENGLISH DEFI : A narrow deep ravine.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : GRAIN
ENGLISH DEFI : A small rounded prominence.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, B5A, C1, C7
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : GRID
ENGLISH DEFI : An arrangement of regularly placed parallel or
ENGLISH DEFI : intersecting bars.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : GRILL
ENGLISH DEFI : A grating used as a screen.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS :

RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : GROOVE
ENGLISH DEFI : A long narrow indentation or furrow cut into a surface;
ENGLISH DEFI : Any narrow depression, channel or rut.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : CAVITY, CHANNEL, FURROW, RUT, SLOT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : GUIDE
ENGLISH DEFI : Any device that regulates or controls the operations of a
ENGLISH DEFI : part.
GROUP CLASSI : TOPOLOGY GROUP - E, F FORM GROUP - B4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : GUSSET
ENGLISH DEFI : A plate used to connect various members.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B1, B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : GUTTER
ENGLISH DEFI : Any groove or channel.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C2, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : HAND-WHEEL
ENGLISH DEFI : A wheel that is designed especially to be grasped by
ENGLISH DEFI : hand.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, C12
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : HATCH
ENGLISH DEFI : A cover over an opening in a floor or plane.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, C12
SYNONYMS : BOLSTER-PLATE, BUTTON, CAP, COVER, DIAL, DISK, PLATE
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : HELIX
ENGLISH DEFI : A line thread or wire curved as if wound in a single layer
GROUP CLASSI : TOPOLOGY GROUP - R FORM GROUP - B1, B5B
SYNONYMS : COIL, CURL, SPIRAL, SPRING
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N
END
FEATURE NAME : HEM
ENGLISH DEFI : To shut in, enclose or restrict by folding in.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - C12
SYNONYMS :
RELATIONS :
END

FEATURE NAME : HEMISPHERE
ENGLISH DEFI : A half-sphere formed by a plane passing through the center
ENGLISH DEFI : of the sphere.
GROUP CLASSI : TOPOLOGY GROUP - O FORM GROUP - B1, C1, C2
SYNONYMS : DOME
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : HEX-BOLT
ENGLISH DEFI : A pin or rod for holding something in place, usually
ENGLISH DEFI : having a head on one end and threading on the other.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, T
END
FEATURE NAME : HEX-NUT
ENGLISH DEFI : A small block of metal having an internal thread so that
ENGLISH DEFI : it can be fitted with a bolt or screw.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, T
END
FEATURE NAME : HOLE
ENGLISH DEFI : A cavity or aperture in a solid mass or body. Can be
ENGLISH DEFI : blind or through.
GROUP CLASSI : TOPOLOGY GROUP - B, F FORM GROUP - B1, B5A, C11
SYNONYMS : WELL
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : HOLLOW
ENGLISH DEFI : A cavity, valley or empty space in anything.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : HOOD
ENGLISH DEFI : Anything resembling a hood in form or use.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP -N.A.
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : HUB
ENGLISH DEFI : The central part of a wheel, pulley or propeller attached
ENGLISH DEFI : to the driving member.
GROUP CLASSI : TOPOLOGY GROUP - V FORM GROUP - B1, B5A, C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, A.R.
END
FEATURE NAME : HUMP
ENGLISH DEFI : A rounded protuberance.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS : PROMINENCE, PROTUBERANCE, PROTRUSION
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : IMPRESSION

ENGLISH DEFI : A mark made by pressure or engraving.
GROUP CLASSI : TOPOLOGY GROUP - L FORM GROUP - B1, C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : INLET
ENGLISH DEFI : A relatively narrow entrance or orifice for admission of
ENGLISH DEFI : fluid.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - C11
SYNONYMS : APERTURE, OPENING, ORIFICE, PERFORATION, PUNCTURE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : JOURNAL
ENGLISH DEFI : The part of shaft or crank which is supported by and turns
ENGLISH DEFI : in a bearing.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : KEEL
ENGLISH DEFI : The main structural member of a vessel, running fore and
ENGLISH DEFI : aft along the bottom, to which all the crosswise members
ENGLISH DEFI : are solidly fixed.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : KERF
ENGLISH DEFI : A narrow deep cut.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : KEY
ENGLISH DEFI : A wedge, cotter pin used to secure various parts.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B2, C3
SYNONYMS : COTTER, WEDGE
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : KEYHOLE
ENGLISH DEFI : A hole or slot for receiving a key used to fasten.
GROUP CLASSI : TOPOLOGY GROUP - B, F FORM GROUP - B2, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : KEYSEAT
ENGLISH DEFI : A groove or channel for placing a key in any mechanical
ENGLISH DEFI : part.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C10
SYNONYMS : KEYWAY
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : KEYWAY

ENGLISH DEFI : A groove or channel for placing a key in any mechanical
ENGLISH DEFI : part.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C10
SYNONYMS : KEYSEAT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : KNOB
ENGLISH DEFI : A rounded protuberance, bunch or hump.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS : NODE
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : KNURL
ENGLISH DEFI : A roughening or indentation on a turned surface.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - C7
SYNONYMS :
RELATIONS :
END
FEATURE NAME : LADLE
ENGLISH DEFI : A cup-shaped vessel with a long handle.
GROUP CLASSI : TOPOLOGY GROUP - O FORM GROUP - B1, C2
SYNONYMS : CUP, SCOOP
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : LEAF
ENGLISH DEFI : Metal in form of a very thin sheet or plate.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : FILM, FOIL, FOLD, MEMBRANE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : LEDGE
ENGLISH DEFI : A raised edge or a narrow shelf-like projection.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : LEG
ENGLISH DEFI : A support resembling a leg in shape, position or function.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : LINE
ENGLISH DEFI : A slender continuous mark or indentation.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B4
SYNONYMS : EDGE
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : LIP
ENGLISH DEFI : A marginal part or structure resembling a flared edge.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END

FEATURE NAME : LOBE
ENGLISH DEFI : A rounded division, protuberance or part.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : LOOP
ENGLISH DEFI : A ring or bent piece of metal serving as a fastener.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS : RIM, RING
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : LUG
ENGLISH DEFI : A projecting "ear", usually rectangular in cross section.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : MANDREL
ENGLISH DEFI : A shaft inserted through a hole in a component to support
ENGLISH DEFI : the work during machining.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : AXLE, ARBOR, SHAFT, SPINDLE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : MANIFOLD
ENGLISH DEFI : A branch pipe arrangement having several or many openings.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : MASK
ENGLISH DEFI : A covering used to conceal all or part of an object.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : MASS
ENGLISH DEFI : An object having no specific shape but a relatively large
ENGLISH DEFI : size.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : MAT
ENGLISH DEFI : A small, flat piece of material sometimes used for
ENGLISH DEFI : protection or covering.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : MEMBRANE
ENGLISH DEFI : A thin, pliable layer of tissue serving to cover or line
ENGLISH DEFI : an object or part.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B1, B2, B5A, C3

SYNONYMS : FILM, FOIL, FOLD, LEAF
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : MESH
ENGLISH DEFI : A net, nest or screen
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS : NET, NEST, SCREEN
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : MESSENGER
ENGLISH DEFI : A small cylindrical metal weight.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : MOSAIC
ENGLISH DEFI : A design arrangement resembling a network.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : NAIL
ENGLISH DEFI : A slender, usually pointed, fastener with a head designed
ENGLISH DEFI : for insertion by impact.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C4
SYNONYMS : DOWEL, NEEDLE, PEG, PIN, STUD, TACK
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : NECK
ENGLISH DEFI : A groove around a shaft, usually near the end.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, R3, L1, L2, L3
END
FEATURE NAME : NEEDLE
ENGLISH DEFI : A slender, pointed instrument usually of steel used to
ENGLISH DEFI : transmit vibration.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C4
SYNONYMS : DOWEL, NAIL, PEG, PIN, STUD, TACK
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : NEST
ENGLISH DEFI : Something constructed with meshes.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS : MESH, NET, SCREEN
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : NET
ENGLISH DEFI : Something constructed with meshes.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS : MESH, NEST, SCREEN
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : NILE

ENGLISH DEFI : A small projecting point.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1, C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : NIPPLE
ENGLISH DEFI : A short piece of tubing, usually with internal/external
ENGLISH DEFI : threading used to couple pipes.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : NODE
ENGLISH DEFI : A knob, protuberance or swelling.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS : KNOB
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : NOTCH
ENGLISH DEFI : A V-shaped cut in a surface or a rectangular depression.
GROUP CLASSI : TOPOLOGY GROUP - F, M FORM GROUP - B3, C8
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : NOZZLE
ENGLISH DEFI : A tube-like device, usually stream-lined for acceleration
ENGLISH DEFI : and directing fluid flow.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - C11
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : O-RING
ENGLISH DEFI : A circular object with functions similar to that of a
ENGLISH DEFI : gasket.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : OPENING
ENGLISH DEFI : A widening of a crevice or an open space.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - C11
SYNONYMS : APERTURE, INLET, ORIFICE, PERFORATION, PUNCTURE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : ORIFICE
ENGLISH DEFI : An opening into a cavity or aperture.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - B1, C11
SYNONYMS : APERTURE, INLET, OPENING, PERFORATION, PUNCTURE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : OVAL
ENGLISH DEFI : Having a shape of an egg or resembling an ellipsoid.
GROUP CLASSI : TOPOLOGY GROUP - W FORM GROUP - B1, B5A, C1
SYNONYMS :

RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : PACKING
ENGLISH DEFI : Any material used in packing, closing a joint.
GROUP CLASSI : TOPOLOGY GROUP - A, C, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS : PAD, PADDING, WADDING
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PAD
ENGLISH DEFI : A layer of material used as a cushion or for protection.
GROUP CLASSI : TOPOLOGY GROUP - A, C, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS : PACKING, PADDING, WADDING
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PADDING
ENGLISH DEFI : A layer of material used as a cushion or for protection.
GROUP CLASSI : TOPOLOGY GROUP - A, C, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS : PACKING, PAD, WADDING
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PAIL
ENGLISH DEFI : A cylindrical or slightly tapered container.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : PALLET
ENGLISH DEFI : A slender flat piece of wood.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PAN
ENGLISH DEFI : A wide, shallow receptacle.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : PARTING
ENGLISH DEFI : The act of separating or dividing.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PARTITION
ENGLISH DEFI : Something that divides or separates.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :

DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PATCH
ENGLISH DEFI : An small part of a surface not sharing the general
ENGLISH DEFI : character as a whole.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PATTERN
ENGLISH DEFI : A design arrangement resembling a network.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : PEAK
ENGLISH DEFI : The highest point or apex.
GROUP CLASSI : TOPOLOGY GROUP - K FORM GROUP - B1, C4
SYNONYMS : APEX, VERTEX
RELATIONS :
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : PEEN
ENGLISH DEFI : The end of a hammer head, usually shaped like a wedge.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
END
FEATURE NAME : PEG
ENGLISH DEFI : A small pointed or tapered piece to fasten parts.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C4
SYNONYMS : DOWEL, NAIL, NEEDLE, PIN, STUD, TACK
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : PELLET
ENGLISH DEFI : A small round cylinder or ball.
GROUP CLASSI : TOPOLOGY GROUP - A, I FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R
END
FEATURE NAME : PERFORATION
ENGLISH DEFI : A tiny hole created by drilling or piercing operation.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - C11
SYNONYMS : APERTURE, INLET, OPENING, ORIFICE, PUNCTURE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : PERIMETER
ENGLISH DEFI : The boundary of any two dimensional object.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : PILLAR
ENGLISH DEFI : A vertical shaft designed to bear axial loads.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : COLUMN

RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : PIN
ENGLISH DEFI : A cylindrical fastener used to join two members or parts
ENGLISH DEFI : with freedom of angular movement at the joint.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C4
SYNONYMS : DOWEL, NAIL, NEEDLE, PEG, STUD, TACK
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : PIPE
ENGLISH DEFI : A cylindrical tube or a long conducting passage.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS : DUCT, FLUE, GLAND, TUBE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : PIT
ENGLISH DEFI : A wide and deep cavity.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PLANE
ENGLISH DEFI : A surface such that a straight line joining any of its
ENGLISH DEFI : two points wholly lies on the surface.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - C3
SYNONYMS : FACE, SHEET
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PLATE
ENGLISH DEFI : A flat piece of metal of arbitrary minimum thickness and
ENGLISH DEFI : width depending on type of metal.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B2, C3
SYNONYMS : BOLSTER-PLATE, BUTTON, CAP, COVER, DIAL, DISK, HATCH
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PLUG
ENGLISH DEFI : A piece of cylindrical metal used to stop or close a
ENGLISH DEFI : hole.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C12
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : PLUNGER
ENGLISH DEFI : A long rod or piston.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : POCKET
ENGLISH DEFI : Any opening, receptacle or container.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B1, B2, B5A, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H

END
FEATURE NAME : POD
ENGLISH DEFI : A seed vessel or capsule.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, B2, C1, C7
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : POINT
ENGLISH DEFI : Something sharp and tapering such as a point of a needle.
GROUP CLASSI : TOPOLOGY GROUP - K FORM GROUP - B1, C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : POT
ENGLISH DEFI : A round, fairly deep vessel of metal.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : POUCH
ENGLISH DEFI : A small bag, sack or other container.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : PRISM
ENGLISH DEFI : A solid whose sides are equal and parallel plane figures
ENGLISH DEFI : and whose lateral faces are parallelograms.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B2, C3
SYNONYMS : PYRAMID, TETRAHEDRON
RELATIONS :
DEFINING PARAMETERS : L, B, H, A.R.
END
FEATURE NAME : PROBE
ENGLISH DEFI : A small tubing containing the sensing element of
ENGLISH DEFI : electronic equipment.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : PROJECTION
ENGLISH DEFI : A protrusion.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : PROMINENCE
ENGLISH DEFI : Jutting out or projecting.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : PROTRUSION
ENGLISH DEFI : To push or thrust out; to project outward.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS : HUMP, PROMINENCE, PROTUBERANCE

RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : PROTUBERANCE
ENGLISH DEFI : Something that protrudes; a knob; prominence.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, C1
SYNONYMS : HUMP, PROMINENCE, PROTRUSION
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : PULLEY
ENGLISH DEFI : A wheel grooved to receive a rope to increase mechanical
ENGLISH DEFI : advantage.
GROUP CLASSI : TOPOLOGY GROUP - Q FORM GROUP - C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, R3, L1, L2, L3
END
FEATURE NAME : PUNCTURE
ENGLISH DEFI : A small hole made by piercing with a sharp point.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - C11
SYNONYMS : APERTURE, INLET, OPENING, PERFORATION, PUNCTURE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : PYRAMID
ENGLISH DEFI : A solid consisting of a polygonal base and triangular
ENGLISH DEFI : sides with a common vertex.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B2A, C3
SYNONYMS : PRISM, TETRAHEDRON
RELATIONS :
DEFINING PARAMETERS : L, B, H, A.R.
END
FEATURE NAME : RABBET
ENGLISH DEFI : A recess or groove near the edge of one piece of wood, cut
ENGLISH DEFI : so as to receive the edge of another piece.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : RACE
ENGLISH DEFI : A sluice or channel by which to conduct water.
GROUP CLASSI : TOPOLOGY GROUP - D, N FORM GROUP - B1, C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : R, L
END
FEATURE NAME : RAIL
ENGLISH DEFI : A bar of metal resting on supports.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS : TRACK
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : RAISE
ENGLISH DEFI : Elevated.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : A, B

END
FEATURE NAME : RAVINE
ENGLISH DEFI : A small, narrow valley with steeply sloping sides.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : REAM
ENGLISH DEFI : To enlarge or clear out a hole or to enlarge a taper.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - B1, C13
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : RECEPTACLE
ENGLISH DEFI : Anything that serves as to contain or hold anything.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B1, B2, B5A, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : RECESS
ENGLISH DEFI : A depression or indentation in any surface.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : REEL
ENGLISH DEFI : A rotary frame or device used for winding.
GROUP CLASSI : TOPOLOGY GROUP - Q FORM GROUP - C2
SYNONYMS : SPOOL
RELATIONS :
DEFINING PARAMETERS : R1, R2, R3, L1, L2, L3
END
FEATURE NAME : RIB
ENGLISH DEFI : A transverse structural member that gives cross section
ENGLISH DEFI : shape and strength to a portion of an object.
GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : RIBBON
ENGLISH DEFI : A narrow strip of metal with finished edges.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B5A, C3
SYNONYMS : BAND, BELT, STRIP
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : RIDGE
ENGLISH DEFI : An elongated, narrow, steep sided elevation.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : RIFLE
ENGLISH DEFI : A bore hole that follows a spiral course.
GROUP CLASSI : TOPOLOGY GROUP - T FORM GROUP - B1, B5B
SYNONYMS :
RELATIONS :

DEFINING PARAMETERS : R1, L, T, T2, H, N
END
FEATURE NAME : RIM
ENGLISH DEFI : An outer edge or border, sometimes raised or projecting.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS : LOOP, RING
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : RING
ENGLISH DEFI : A tie member of a chain link, usually circular.
GROUP CLASSI : TOPOLOGY GROUP - C, T FORM GROUP - B1, B5A
SYNONYMS : LOOP, RIM
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : R1, L, T, T2, H, N
END
FEATURE NAME : RIVET
ENGLISH DEFI : A short soft metal bolt, having a head on one end, used to
ENGLISH DEFI : join objects.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : ROD
ENGLISH DEFI : A thin round bar of metal or wood.
ENGLISH DEFI : A straight, slim piece of wood, metal or other material.
ENGLISH DEFI : A bar, typically of metal, forming part of a machine.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A
SYNONYMS : BAR, CORE
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : ROLLER
ENGLISH DEFI : A cylindrical device for rolling or rotating.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : ROPE
ENGLISH DEFI : A construction of twisted fibers, intertwined in strands
ENGLISH DEFI : to form a thick cord.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : CORD, STRING, TWINE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : ROUND
ENGLISH DEFI : Having a contour that is circular; spherical; ring-shaped
ENGLISH DEFI : cylindrical.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : ROUNDING
ENGLISH DEFI : Having a contour that is circular; spherical; ring-shaped
ENGLISH DEFI : cylindrical.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :

DEFINING PARAMETERS : R
END
FEATURE NAME : RUPTURE
ENGLISH DEFI : To break apart; separate into parts.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : BREAK, CRACK, CREVICE, CUT, FISSURE, FRACTURE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : RUT
ENGLISH DEFI : A sunken track or groove.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS : CAVITY, CHANNEL, FURROW, GROOVE, SLOT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SCAR
ENGLISH DEFI : A mark, damage or lasting effect resulting from an
ENGLISH DEFI : accident.
GROUP CLASSI : TOPOLOGY GROUP - B, F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SCOOP
ENGLISH DEFI : A cup-shaped vessel with a long handle.
GROUP CLASSI : TOPOLOGY GROUP - O FORM GROUP - B1, B2, B5A, C2
SYNONYMS : CUP, LADLE
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : SCREEN
ENGLISH DEFI : A wire mesh or netting forming a partition.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS : MESH, NEST, NET
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : SCREW
ENGLISH DEFI : A cylindrical body with a helical groove cut into its
ENGLISH DEFI : surface.
GROUP CLASSI : TOPOLOGY GROUP - U FORM GROUP - B1, B5B, C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, L, T, T2, H, N
END
FEATURE NAME : SEAL
ENGLISH DEFI : A device used to fasten, secure or close.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2, B5A, C12
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SEAM
ENGLISH DEFI : The line of junction between parts; to mark with a crack
ENGLISH DEFI : fissure or cut.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B4
SYNONYMS :
RELATIONS :
END
FEATURE NAME : SEAT

ENGLISH DEFI : The place where anything is situated, located or
ENGLISH DEFI : established.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SEPARATION
ENGLISH DEFI : The act or process of separating; division.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SHAFT
ENGLISH DEFI : A cylindrical piece of metal used to carry rotating
ENGLISH DEFI : machine parts.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C1
SYNONYMS : ARBOR, AXLE, MANDREL, SPINDLE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : SHARP
ENGLISH DEFI : Having a keen edge or an acute point.
GROUP CLASSI : TOPOLOGY GROUP - K FORM GROUP - C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : SHEET
ENGLISH DEFI : A piece of metal or other substance hammered, rolled or
ENGLISH DEFI : cut very thin.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : FACE, PLANE
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SHELL
ENGLISH DEFI : A hollow structure or vessel, generally thin and weak.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B1, B2B, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : SHIELD
ENGLISH DEFI : A cover that conceals or protects.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SHIM
ENGLISH DEFI : A thin piece of wood or steel placed under a member to
ENGLISH DEFI : bring it to a desired elevation.
GROUP CLASSI : TOPOLOGY GROUP - A, E FORM GROUP - B1, B2A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SHOULDER
ENGLISH DEFI : Anything that supports, bears up or projects like a
ENGLISH DEFI : shoulder.

GROUP CLASSI : TOPOLOGY GROUP - H FORM GROUP - B2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T, R, A
END
FEATURE NAME : SIEVE
ENGLISH DEFI : A frame of wire mesh.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : SLAB
ENGLISH DEFI : A thick flat piece or slice of material.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SLEEVING
ENGLISH DEFI : A cylindrical tubing designed to fit over another part.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : SLIT
ENGLISH DEFI : A long narrow cut or opening; slash.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SLOT
ENGLISH DEFI : A long narrow groove or opening; cut.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C10
SYNONYMS : CAVITY, CHANNEL, FURROW, GROOVE, RUT
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SOCKET
ENGLISH DEFI : A cavity or opening adapted to receive or hold something.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - B2, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SPHERE
ENGLISH DEFI : A solid having a surface every point on which is
ENGLISH DEFI : equidistant from the center; ball, globe.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : BALL, GLOBE
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : SPIKE
ENGLISH DEFI : A large pointed nail.
GROUP CLASSI : TOPOLOGY GROUP - K FORM GROUP - C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : SPINDLE

ENGLISH DEFI : A rotating rod, axis or shaft, usually slender.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C1
SYNONYMS : ARBOR, AXLE, MANDREL, SHAFT
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : SPIRAL
ENGLISH DEFI : A curve winding like a screw thread or helix.
GROUP CLASSI : TOPOLOGY GROUP - R FORM GROUP - B5B
SYNONYMS : COIL, CURL, HELIX, SPRING
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N
END
FEATURE NAME : SPLINE
ENGLISH DEFI : A long keyway. Sometimes also a flat key.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N
END
FEATURE NAME : SPLIT
ENGLISH DEFI : To separate into two parts, especially equal.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SPLIT-RING
ENGLISH DEFI : A circular band for holding, connecting or hanging
ENGLISH DEFI : that grips by means of a tightening screw.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : SPOKE
ENGLISH DEFI : One of the small radiating bars inserted in the hub of
ENGLISH DEFI : a wheel or pulley to support the rim.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SPOOL
ENGLISH DEFI : A concave cylinder used for winding.
GROUP CLASSI : TOPOLOGY GROUP - Q FORM GROUP - C2
SYNONYMS : REEL
RELATIONS :
DEFINING PARAMETERS : R1, R2, R3, L1, L2, L3
END
FEATURE NAME : SPOT-FACE
ENGLISH DEFI : A finished round spot on a rough surface, to give a good
ENGLISH DEFI : seat to a screw or bolt-head.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : SPRING
ENGLISH DEFI : An elastic body or a contrivance, as a coiled wire.
GROUP CLASSI : TOPOLOGY GROUP - R FORM GROUP - B5B
SYNONYMS : COIL, CURL, HELIX, SPIRAL
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N

END
FEATURE NAME : STICK
ENGLISH DEFI : A slender piece of wood; like a baton or wand.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C1
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : STIFFENER
ENGLISH DEFI : Angle, plate or channel riveted to a member to prevent
ENGLISH DEFI : buckling.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : STOPPER
ENGLISH DEFI : Something that stops up or closes an opening; one that
ENGLISH DEFI : stops or checks a movement.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A, C14
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : STRING
ENGLISH DEFI : A slender line or strip, thinner than a cord and thicker
ENGLISH DEFI : than a thread.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : CORD, ROPE, TWINE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : STRIP
ENGLISH DEFI : A narrow, thin and comparatively long piece of metal.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : BAND, BELT, RIBBON
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : STUD
ENGLISH DEFI : A rivet, boss or nail with a large head; A short rod or
ENGLISH DEFI : bolt threaded at both ends without a head.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B2, C3
SYNONYMS : DOWEL, NAIL, NEEDLE, PEG, PIN, TACK
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : STUFFING
ENGLISH DEFI : A layer of material used for cushion or for protection.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B1, B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : SWELL
ENGLISH DEFI : A local enlargement.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, B5A, C1
SYNONYMS : BUMP, BULGE, ENTASIS, SWELLING
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : SWELLING
ENGLISH DEFI : A local enlargement.
GROUP CLASSI : TOPOLOGY GROUP - J FORM GROUP - B1, B5A, C1

SYNONYMS : BUMP, BULGE, ENTASIS, SWELL
RELATIONS :
DEFINING PARAMETERS : A, B
END
FEATURE NAME : T-BOLT
ENGLISH DEFI : A bolt with a rectangular head used in a T-slot.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : T-SLOT
ENGLISH DEFI : A slot shaped like an inverted-T, to hold or guide pieces.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : TABLE
ENGLISH DEFI : A platform or a plate used for support.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : TACK
ENGLISH DEFI : A small, sharp-pointed, nail, commonly with tapering sides
ENGLISH DEFI : and a flat head.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A, C4
SYNONYMS : DOWEL, NAIL, NEEDLE, PEG, PIN, STUD
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : TANK
ENGLISH DEFI : A large vessel, basin or receptacle for holding a fluid.
GROUP CLASSI : TOPOLOGY GROUP - G FORM GROUP - B2B, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, H, T
END
FEATURE NAME : TAPER
ENGLISH DEFI : A gradual diminution of size in an elongated object.
GROUP CLASSI : TOPOLOGY GROUP - V FORM GROUP - B1, B5A C4
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, A.R.
END
FEATURE NAME : TAPPED-HOLE
ENGLISH DEFI : A hole containing an internal screw thread.
GROUP CLASSI : TOPOLOGY GROUP - T FORM GROUP - B5B, C11
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, L, T, T2, H, N
END
FEATURE NAME : TEAR
ENGLISH DEFI : A drop of liquid; globular.
GROUP CLASSI : TOPOLOGY GROUP - I FORM GROUP - B1, B5A, C1
SYNONYMS : BUBBLE, DROP, DROPLET, GLOBULE
RELATIONS :
DEFINING PARAMETERS : R
END
FEATURE NAME : TETRAHEDRON
ENGLISH DEFI : A polyhedron bounded by four plane triangular faces.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - B2, C2
SYNONYMS : PRISM, PYRAMID

RELATIONS :
DEFINING PARAMETERS : L, B, H, A.R.
END
FEATURE NAME : THICKENING
ENGLISH DEFI : A thickened place or a part.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : THREAD
ENGLISH DEFI : The spiral ridge of a screw.
GROUP CLASSI : TOPOLOGY GROUP - R FORM GROUP - B5B
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N
END
FEATURE NAME : TILE
ENGLISH DEFI : A thin plate used for covering or protecting.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : TORUS
ENGLISH DEFI : The surface of a doughnut.
GROUP CLASSI : TOPOLOGY GROUP - S FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2
END
FEATURE NAME : TRACK
ENGLISH DEFI : A set of rails or a rail.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2, C3
SYNONYMS : RAIL
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : TREPAN
ENGLISH DEFI : To cut an outside annular groove around a hole.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : TUBE
ENGLISH DEFI : A long cylindrical body with a hollow center, generally
ENGLISH DEFI : used for conveying something.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B5A
SYNONYMS : DUCT, FLUE, GLAND, PIPE
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : TUNNEL
ENGLISH DEFI : An underground passage-way.
GROUP CLASSI : TOPOLOGY GROUP - D FORM GROUP - B1, B5A, C2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
END
FEATURE NAME : TWINE
ENGLISH DEFI : A string consisting of two strands twisted together.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : CORD, ROPE, STRING

RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : TWIST
ENGLISH DEFI : Thread or cord made of slightly twisted strands.
GROUP CLASSI : TOPOLOGY GROUP - R FORM GROUP - B5B
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L, N
END
FEATURE NAME : UNDERCUT
ENGLISH DEFI : A cut that leaves an overhanging edge. A cut with inwardly
ENGLISH DEFI : sloping sides.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : VALLEY
ENGLISH DEFI : Any depression or hollow like a valley.
GROUP CLASSI : TOPOLOGY GROUP - F FORM GROUP - C8, C10
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : VANE
ENGLISH DEFI : An arm or blade extending from the rotating shaft; a
ENGLISH DEFI : movable thin plate of metal.
GROUP CLASSI : TOPOLOGY GROUP - E FORM GROUP - B2
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : VEIN
ENGLISH DEFI : A thin, tiny tubular vessel.
GROUP CLASSI : TOPOLOGY GROUP - C, F FORM GROUP - B1, B2, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : VERTEX
ENGLISH DEFI : The highest point of anything; apex.
GROUP CLASSI : TOPOLOGY GROUP - K FORM GROUP - B1, C4
SYNONYMS : APEX, PEAK
RELATIONS :
DEFINING PARAMETERS : R, H, A.R.
END
FEATURE NAME : WADDING
ENGLISH DEFI : A small compact mass of any shaft or flexible substance.
GROUP CLASSI : TOPOLOGY GROUP - A, C, E FORM GROUP - B1, B2, B5A, C3
SYNONYMS : PACKING, PAD, PADDING
RELATIONS :
DEFINING PARAMETERS : R, L
DEFINING PARAMETERS : R1, R2, L
DEFINING PARAMETERS : L, B1, B2, H
END
FEATURE NAME : WASHER
ENGLISH DEFI : A flattened, ring-shaped device used to improve tightness
ENGLISH DEFI : of a screw fastener.
GROUP CLASSI : TOPOLOGY GROUP - C FORM GROUP - B1, B2, B5A, C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R1, R2, L

END
FEATURE NAME : WEB
ENGLISH DEFI : Any structure woven of or as interlaced strands.
GROUP CLASSI : TOPOLOGY GROUP - P FORM GROUP - C3
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : L, B, DL, DB
END
FEATURE NAME : WEDGE
ENGLISH DEFI : A V-shaped piece of solid metal.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS : KEY, COTTER
RELATIONS :
DEFINING PARAMETERS : L, B, H
END
FEATURE NAME : WELD
ENGLISH DEFI : A rounded filling used to unite two pieces of metal.
GROUP CLASSI : TOPOLOGY GROUP - N.A. FORM GROUP - N.A.
SYNONYMS :
RELATIONS :
END
FEATURE NAME : WELL
ENGLISH DEFI : A vertical hole or shaft.
GROUP CLASSI : TOPOLOGY GROUP - B FORM GROUP - B1, B5A
SYNONYMS : HOLE
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : WHEEL
ENGLISH DEFI : A circular frame with a hub at the center and capable of
ENGLISH DEFI : rotating on a central axes for attachment to an axle.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS :
RELATIONS :
DEFINING PARAMETERS : R, L
END
FEATURE NAME : WIRE
ENGLISH DEFI : A strand of ductile metal.
GROUP CLASSI : TOPOLOGY GROUP - A FORM GROUP - B1, B5A
SYNONYMS : CABLE
RELATIONS :
DEFINING PARAMETERS : R, L
END

Appendix B - FeatureMod Program Listing

Index Of Subroutines.

1. FADTO - Allows additions to existing objects and assemblies
2. FASSM - Allows the addition of objects to different assemblies
3. FBCKV - Displays the back view of an object
4. FBMAT - Computes the P-matrix for a periodic cubic B-spline surface and displays the point mesh
5. FBMSH - Computes the mesh of points lying on a periodic cubic B-spline surface
6. FBOTV - Displays the bottom view of an object
7. FBPCH - Creates a B-spline surface patch for specified control points
8. FCAPS - Converts a mixed case string input to all upper case
9. FCOMM - Sets up counters for shading
10. FCSN1 - Computes w-tangents at endpoints of patch for n eq 0.0
11. FCSN2 - Computes w-tangents at endpoints of patch for 0.0 lt n lt 2.0
12. FCSN3 - Computes w-tangents at endpoints of patch for n = 2.0
13. FCSN4 - Computes w-tangents at endpoints of patch for n gt 2.0
14. FCTCP - Maps points on a B-spline curve into its control points
15. FDASM - Defines New Assemblies
16. FDELT - Deletes a specified object or assembly
17. FDINS - Defines new instances and makes them part of object
18. FDISP - Displays a specified object or assembly
19. FDOBJ - Defines New Objects
20. FDRWU - Draws all constant u lines
21. FDRWW - Draws all constant w lines
22. FEDAT - Sets up 2 polygon edge attributes (color, edge type)
23. FFNTV - Displays the front view of an object
24. FGBMT - Computes the elements of B-matrix based on global axes
25. FGTST - Requests a string input from the user
26. FHMAT - Assembles the global B-matrix for bicubic Hermite patch
27. FHMSH - Uses the B-matrix and computes different points on a bicubic Hermite patch

28. FINIT - Initializes all input devices
29. FINPT - Waits for any input from the user (event mode)
30. FINQU - Allows inquiries about features and returns requested information
31. FLBMT - Computes the elements of B-matrix based on local axes
32. FLFTV - Displays the left side view of an object
33. FMENU - Displays different menu pages in the defined area
34. FMPK1 - Processes the pick-id of the item picked from Menu 1 (FILES)
35. FMPK10- Processes the pick-id of the item picked from Menu 10 (DEFINE)
36. FMPK2 - Processes the pick-id of the item picked from Menu 2 (COMMANDS)
37. FMPK3 - Processes the pick-id of the item picked from Menu 3 (MODELING)
38. FMPK4 - Processes the pick-id of the item picked from Menu 4 (RENDERING)
39. FMPK5 - Processes the pick-id of the item picked from Menu 5 (VIEWS)
40. FMPK6 - Processes the pick-id of the item picked from Menu 6 (WINDOWS)
41. FMPK7 - Processes the pick-id of the item picked from Menu 7 (SHADING)
42. FMPK8 - Processes the pick-id of the item picked from Menu 8 (SINGLE WINDOWS)
43. FMPK9 - Processes the pick-id of the item picked from Menu 9 (COLOR)
44. FMSSG - Displays messages and scrolls them in the defined area
45. FMT11 - Multiplies a 1x4 and a 4x1 matrix
46. FMT14 - Multiplies a 1x4 and a 4x4 matrix
47. FMT44 - Multiplies a 4x4 and a 4x4 matrix
48. FMULT - Displays three primary views of an object simultaneously
49. FNORM - Normalizes a vector to unit length
50. FOPEN - Used to initialize graPHIGS
51. FPARS - Parses a string input into a word array
52. FPGAT - Sets up 2 polygon attributes (color,interior style)
53. FPNTS - Computes global B-matrix for object with axis = z-axis
54. FPRMP - Displays prompts to users
55. FPRPK - Processes the menu number to which the item picked belonged.
56. FQURY - Queries number of assemblies and objects and their names
57. FRBMH - Computes the mesh of points lying on a periodic cubic rational B-spline surface
58. FRBMT - Computes the H-matrix and P-matrix for a periodic cubic rational B-spline surface and displays the point mesh
59. FRCLL - Recalls a existing model file
60. FRGTV - Displays the right side view of an object
61. FROTA - Allows viewing transformations on the model
62. FSCAN - Matches a string input with predefined syntax
63. FSCLE - Allows zooming
64. FSCRA - Displays all background areas
65. FSTCP - Maps points on a B-spline surface into its control points
66. FSTOR - Stores all points on patch into polygon forms
67. FSTRG - Seeks an string input from user (request mode)
68. FSUCO - Computes object to be displayed with title screen
69. FTITL - Displays Title screen for "FEATURE"
70. FTOPV - Displays the top view of an object
71. FTRNS - Transforms all points from the global coordinate axes to local axes
72. FUWVT - Computes a 1x4 vector using the vector parameter
73. FWIND - Sets all Windows and Viewports

```
************************************************************************
************************************************************************
**    PROGRAM FEATUREMOD                                              **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS IS THE MAIN PROGRAM FOR THE FEATURE BASED MODELING         **
**    SOFTWARE.                                                       **
**                                                                    **
**    IT IS USED TO DISPLAY THE TITLE PAGE AND THEN THE MENU SCREEN.  **
**    IT PASSES THE COMMAND TO THE REQUIRED SUBROUTINE AFTER AN       **
**    INPUT HAS BEEN SPECIFIED.                                       **
**                                                                    **
**    BY:      ASHIT R. GANDHI                                        **
**    DATE:    01/13/89                                               **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NONE                                                            **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      REAL*4 V(4), SCOL(3)
      INTEGER*4 ICLASS,IDEV,IPKID, ASNUM, OBNUM, INNUM
      CHARACTER STRG*50, PMENU(50)*15

      COMMON/SHADE/SCOL
      COMMON/IDS/ASNUM, OBNUM, INNUM

*SET SCALE FACTORS

      SCOL(1) = 1.0
      SCOL(2) = 0.0
      SCOL(3) = 0.0

*SET COMPONENT ID NUMBERS

      ASNUM = 100
      OBNUM = 1000
      INNUM = 10000

*OPEN GRAPHIGS

      CALL FOPEN

*DEFINE WINDOWS AND VIEWPORTS

      CALL FWIND

*DISPLAY THE TITLE SCREEN

      CALL FTITL

*DISPLAY THE BASIC MENUS

      CALL FSCRA

*INITIALIZE ALL INPUTS

      CALL FINIT

*DISPLAY THE FIRST SCROLL MESSAGES

      DO 50 I = 1,5
         CALL FMSSG(I,STRG)
50    CONTINUE

*DISPLAY FIRST MENU AND MAKE SURE THAT FIRST PICKS ARE START OR RECALL

      MNUM = 1
      CALL FMENU2(MNUM,PMENU)
      CALL GPUPWS(1,2)

      CALL FPRMP(1)
300   CALL FINPT(PMENU,ICLASS,IDEV,IPKID,V,STRG)
      IF (ICLASS .NE. 5)THEN
         STRG = 'INVALID OPTION SELECTED'
         CALL FMSSG(30,STRG)
         GOTO 300
      ELSE
         IF (IPKID .NE. 5 .AND. IPKID .NE. 6)THEN
            STRG = 'INVALID OPTION SELECTED'
            CALL FMSSG(30,STRG)
            GOTO 300
         ELSE
            CALL FPRPK(MNUM,IPKID,PMENU)
            CALL GPUPWS(1,2)
         ENDIF
      ENDIF
```

```
*ONCE FILE HAS BEEN STARTED OR RECALLED ALLOW USER TO SELECT DIFFERENT
*OPTIONS
      CALL GPUPWS(1,2)
200   CALL FPRMP(2)
*GET INPUT FROM THE USER
      CALL FINPT(PMENU,ICLASS,IDEV,IPKID,V,STRG)
*IF INPUT IS VALUATOR
      IF (ICLASS .EQ. 3)THEN
*IF DEVICE NUMBER IS 4 THEN SCALE OTHERWISE ROTATE THE VIEWS
         IF (IDEV .NE. 4)THEN
            CALL FROTA(21,V(1),V(2),V(3),IDEV)
            CALL GPUPWS(1,2)
         ELSE
            CALL FSCLE(V(4))
            CALL GPUPWS(1,2)
         ENDIF
      ENDIF
*IF INPUT IS PICK
      IF (ICLASS .EQ. 5)THEN
         CALL FPRPK(MNUM,IPKID,PMENU)
         CALL GPUPWS(1,2)
      ENDIF
      IF (MNUM .EQ. 0)GOTO 100
      GOTO 200
100   STOP
      END
```

SUBROUTINE BUSH

```
************************************************************************
************************************************************************
**    SUBROUTINE BUSH(R1,R2,THK,LOC,ORI,ICOL,BOOL)                    **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRED FOR       **
**    CREATING A BUSHING USING B-SPLINE SURFACES.                     **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     01/13/89                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    R1 = INNER RADIUS OF BUSHING                                    **
**    R2 = INNER RADIUS OF BUSHING                                    **
**    THK = THICKNESS OF THE BUSHING                                  **
**    LOC = LOCATION OF THE BUSHING                                   **
**    ORI = ORIENTATION OF THE BUSHING                                **
**    ICOL = COLOR FOR THE BUSHING                                    **
**    BOOL = BOOLEAN TYPE                                             **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE BUSH(R1,R2,THK,LOC,ORI,ICOL,BOOL)

      INTEGER*4 ICOL
      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 LOC(3), ORI(3), PT(3)
      REAL*4 R1, R2, THK
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3
```

```
      INNUM = INNUM + 1

      CALL FDINS(INNUM,LTYPE,'BUSHING ',16,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

      ICHK = 1

*CREATE END RINGS

      CALL RING(R1,R2,LOC,ORI,ICOL,BOOL,ICHK,0.)
      PT(1) = THK
      PT(2) = 0.0
      PT(3) = 0.0
      CALL RING(R1,R2,LOC,ORI,ICOL,BOOL,ICHK,THK)

*CREATE INNER AND OUTER CYLINDER

      CALL CYNDR(1,R2,THK,LOC,ORI,ICOL,BOOL,ICHK,0.)
      CALL CYNDR(1,R1,THK,LOC,ORI,ICOL,BOOL,ICHK,0.)

      RETURN
      END
```

SUBROUTINE CONE

```
************************************************************************
************************************************************************
**    SUBROUTINE CONE(IFL,RAD,LEN,AR,LOC,ORI,ICOL,BOOL,ICHK,PLACE)    **
**                                                                     **
**    PROGRAM DESCRIPTION                                              **
**                                                                     **
**    THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRED TO         **
**    CREATE A CONE USING B-SPLINE SURFACES                            **
**                                                                     **
**                                                                     **
**    BY:       ASHIT R. GANDHI                                        **
**    DATE:     11/12/88                                               **
**                                                                     **
**    PARAMETERS USED:                                                 **
**                                                                     **
**    RAD   = RADIUS OF THE CONE (REAL,I/P)                            **
**    LEN   = LENGTH OF THE CONE (REAL,I/P)                            **
**    LOC   = LOCATION OF THE CONE (REAL,I/P)                          **
**    ORI   = EULER ORIENTATION FOR THE CONE (REAL,I/P)                **
**    ICOL  = COLOR FOR THE CONE                                       **
**    BOOL  = BOOLEAN OPERATOR ON THE CONE                             **
**    ICHK  = FLAG FOR NEW FEATURE                                     **
**    PLACE= LOCAL POSITION OF CONE                                    **
**                                                                     **
************************************************************************
************************************************************************
      SUBROUTINE CONE(IFL,RAD,LEN,AR,LOC,ORI,ICOL,BOOL,ICHK,PLACE)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 RAD, LEN, LOC(3), ORI(3), FX(4), FY(4), FZ(4), R(4)
      REAL*4 U(4), W(4), H0(20,20), PT(3), PTS(4,4,3), PNTS(20,20,3)
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA FX/ 0.0, 0.0, 1.0, 1.0/
      DATA FY/ 1.0, 1.0,-1.0,-1.0/
      DATA FZ/ 1.0,-1.0,-1.0, 1.0/
      DATA U/0.0, 0.33333333, 0.66666667, 1.0/
      DATA W/0.0, 0.33333333, 0.66666667, 1.0/
      DATA H0/400*1.0/

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      IF (ICHK .EQ. 0)THEN
         INNUM = INNUM + 1
         IF (IFL .EQ. 1)NSURF = 4
         IF (IFL .EQ. 2 .AND. AR .EQ. 0.0)NSURF = 8
         IF (IFL .EQ. 2 .AND. AR .NE. 0.0)NSURF = 12
         CALL FDINS(INNUM,LTYPE,'CONE    ',NSURF,CNUM,
```

```
     1              OBJECT(CNUM-1000),LOC,ORI)
      ENDIF

      IF (IFL .NE. 1 .AND. IFL .NE. 2)THEN
         WRITE(6,*)'CONE  ====>  FLAG FOR CAPS NOT 1 OR 2'
         WRITE(6,*)'CONE  ====>  FLAG SHOULD BE SET TO 1 OR 2'
         RETURN
      ENDIF

      PI = 3.14159

      DEL = PI/6.0

      R(1) = 1.0
      R(2) = 1.0 - (1.0 - AR)/3.0
      R(3) = 1.0 - 2.0*(1.0 - AR)/3.0
      R(4) = AR

*COMPUTE THE HOLLOW CONE

      DO 300 K = 1,4
         WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
         THETA = 0.0
         DO 200 I = 1,4
            DO 100 J = 1,4
               PT(1) = U(J)*LEN + PLACE
               PT(2) = FY(K)*RAD*SIN(THETA)*R(J)
               PT(3) = FZ(K)*RAD*COS(THETA)*R(J)
               PTS(I,J,1) = PT(1)
               PTS(I,J,2) = PT(2)
               PTS(I,J,3) = PT(3)
100         CONTINUE
            THETA = THETA + DEL
200      CONTINUE
         CALL FSTCP(U,W,PTS,PNTS)
         CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)

         DO 250 I = 1,4
            WRITE(10,*)(PNTS(I,J,1),J=1,4)
            WRITE(10,*)(PNTS(I,J,2),J=1,4)
            WRITE(10,*)(PNTS(I,J,3),J=1,4)
250         WRITE(10,*)(HO(1,J),J=1,4)

300   CONTINUE

*IF FLAG IS HOLLOW CONE THEN RETURN

      IF (IFL .EQ. 1)RETURN

*IF SOLID, COMPUTE THE END CAPS OF THE CONE IF NEEDED

      ND = 3
      IF (AR .EQ. 0.0)ND = 5

      DO 700 IJ =1,4,ND
         DO 600 K = 1,4
            WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
            THETA = 0.0
            DO 500 I = 1,4
               DO 400 J = 1,4
                  PT(1) = FX(IJ)*LEN + PLACE
                  PT(2) = FY(K)*U(J)*RAD*SIN(THETA)*R(IJ)
                  PT(3) = FZ(K)*U(J)*RAD*COS(THETA)*R(IJ)
                  PTS(I,J,1) = PT(1)
                  PTS(I,J,2) = PT(2)
                  PTS(I,J,3) = PT(3)
400            CONTINUE
               THETA = THETA + DEL
500         CONTINUE
            CALL FSTCP(U,W,PTS,PNTS)
            CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)

            DO 550 I = 1,4
            WRITE(10,*)(PNTS(I,J,1),J=1,4)
            WRITE(10,*)(PNTS(I,J,2),J=1,4)
            WRITE(10,*)(PNTS(I,J,3),J=1,4)
550         WRITE(10,*)(HO(1,J),J=1,4)

600      CONTINUE
700   CONTINUE

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE COUNT

```
**********************************************************************
**********************************************************************
**    SUBROUTINE  COUNT(R1,R2,L1,L2,IFL,IFL1,LOCA,ORI,COLOR,BOOL)    **
```

```
**                                                                     **
**     PROGRAM DESCRIPTION                                             **
**                                                                     **
**     THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRED FOR       **
**     CREATING A COUNTER BORE OR A COUNTER SINK USING B-SPLINE        **
**     SURFACES                                                        **
**                                                                     **
**     BY:       ASHIT R. GANDHI                                       **
**     DATE:     11/12/88                                              **
**                                                                     **
**     PARAMETERS USED:                                                **
**                                                                     **
**     R1     = RADIUS AT TOP OF FEATURE                               **
**     R2     = INTERMEDIATE RADIUS OF FEATURE                         **
**     L1     = LENGTH BETWEEN TOP AND INTERMEDIATE RADIUS             **
**     L2     = LENGTH OF FEATURE                                      **
**     IFL    = FLAG INDICATING CAPS FOR COUNTER BORE                  **
**     IFL1   = FLAG INDICATING CAPS FOR COUNTER SINK                  **
**     LOCA   = LOCATION OF THE FEATURE                                **
**     ORI    = ORIENTATION OF THE FEATURE                             **
**     COLOR  = COLOR TO BE GIVEN TO THE FEATURE                       **
**     BOOL   = BOOLEAN OPERATOR FOR THE FEATURE                       **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE COUNT(R1,R2,L1,L2,IFL,IFL1,LOC,ORI,ICOL,BOOL)

      INTEGER*4 ICOL
      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 LOC(3), ORI(3), PT(3)
      REAL*4 R1, R2, L1, L2
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      INNUM = INNUM + 1

      IF (IFL .EQ. 1)THEN
         NSURF = 0
      ELSE
         NSURF = 8
      ENDIF

      IF (IFL1 .EQ. 1)THEN
         NSURF = NSURF + 8
         CALL FDINS(INNUM,LTYPE,'C. BORE ',NSURF,CNUM,
     1              OBJECT(CNUM-1000),LOC,ORI)
      ELSE
         NSURF = NSURF + 12
         CALL FDINS(INNUM,LTYPE,'C. SINK ',NSURF,CNUM,
     1              OBJECT(CNUM-1000),LOC,ORI)
      ENDIF

      ICHK = 1

      IF (R2 .GT. R1)THEN
         R3 = R2
         R2 = R1
         R1 = R3
      ENDIF
      AR = R2/R1

      IF (IFL1 .EQ. 2)THEN
         CALL CONE(1,R1,L1,AR,LOC,ORI,ICOL,BOOL,ICHK,0.)
         CALL CYNDR(1,R2,L2,LOC,ORI,ICOL,BOOL,ICHK,L1)
      ELSE
         CALL CYNDR(1,R1,L1,LOC,ORI,ICOL,BOOL,ICHK,0.)
         CALL CYNDR(1,R2,L2,LOC,ORI,ICOL,BOOL,ICHK,L1)
         CALL RING(R2,R1,LOC,ORI,ICOL,BOOL,ICHK,L1)
      ENDIF

      IF (IFL .EQ. 2)THEN
         CALL RING(0.,R1,LOC,ORI,ICOL,BOOL,ICHK,0.)
         RL = L1 + L2
         CALL RING(0.,R2,LOC,ORI,ICOL,BOOL,ICHK,RL)
      ENDIF

      RETURN
      END
```

```
************************************************************************
************************************************************************
**     SUBROUTINE CYNDR(IFL,RAD,LEN,LOC,ORI,ICOL,BOOL,ICHK,PLACE)     **
**                                                                    **
**     PROGRAM DESCRIPTION                                            **
**                                                                    **
**     THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRED TO       **
**     CREATE A CYLINDER USING B-SPLINE SURFACES                      **
**                                                                    **
**                                                                    **
**     BY:       ASHIT R. GANDHI                                      **
**     DATE:     11/12/88                                             **
**                                                                    **
**     PARAMETERS USED:                                               **
**                                                                    **
**     RAD   = RADIUS OF THE CYLINDER (REAL,I/P)                      **
**     LEN   = LENGTH OF THE CYLINDER (REAL,I/P)                      **
**     LOC   = LOCATION OF THE CYLINDER (REAL,I/P)                    **
**     ORI   = EULER ORIENTATION FOR THE CYLINDER (REAL,I/P)          **
**     ICOL = COLOR FOR THE CYLINDER                                  **
**     BOOL = BOOLEAN OPERATOR ON THE CYLINDER                        **
**     ICHK = FLAG FOR NEW FEATURE                                    **
**     PLACE= LOCATION IN LOCAL AXIS                                  **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE CYNDR(IFL,RAD,LEN,LOC,ORI,ICOL,BOOL,ICHK,PLACE)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 RAD, LEN, LOC(3), ORI(3), FX(4), FY(4), FZ(4)
      REAL*4 U(4), W(4), H0(20,20), PT(3), PTS(4,4,3), PNTS(20,20,3)
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA FX/-1.0, 0.0, 1.0, 2.0/
      DATA FY/ 1.0, 1.0,-1.0,-1.0/
      DATA FZ/ 1.0,-1.0,-1.0, 1.0/
      DATA U/0.0, 0.33333333, 0.66666667, 1.0/
      DATA W/0.0, 0.33333333, 0.66666667, 1.0/
      DATA H0/400*1.0/

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      IF (ICHK .EQ. 0)THEN
         INNUM = INNUM + 1
         IF (IFL .EQ. 1)THEN
            NSURF = 4
         ELSE
            NSURF = 12
         ENDIF
         CALL FDINS(INNUM,LTYPE,'CYLINDER',NSURF,CNUM,
     1              OBJECT(CNUM-1000),LOC,ORI)
      ENDIF

      IF (IFL .NE. 1 .AND. IFL .NE. 2)THEN
         WRITE(6,*)'CYLINDER  ====>  FLAG FOR CAPS NOT 1 OR 2'
         WRITE(6,*)'CYLINDER  ====>  FLAG SHOULD BE SET TO 1 OR 2'
         RETURN
      ENDIF

      PI = 3.14159

      DEL = PI/6.0

*COMPUTE THE HOLLOW CYLINDER

      DO 300 K = 1,4
         WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
         THETA = 0.0
         DO 200 I = 1,4
            DO 100 J = 1,4
               PT(1) = U(J)*LEN + PLACE
               PT(2) = FY(K)*RAD*SIN(THETA)
               PT(3) = FZ(K)*RAD*COS(THETA)
               PTS(I,J,1) = PT(1)
```

```
               PTS(I,J,2) = PT(2)
               PTS(I,J,3) = PT(3)
100         CONTINUE
            THETA = THETA + DEL
200      CONTINUE
         CALL FSTCP(U,W,PTS,PNTS)
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

         DO 250 I = 1,4
            WRITE(10,*)(PNTS(I,J,1),J=1,4)
            WRITE(10,*)(PNTS(I,J,2),J=1,4)
            WRITE(10,*)(PNTS(I,J,3),J=1,4)
250         WRITE(10,*)(H0(1,J),J=1,4)

300   CONTINUE

*IF FLAG IS HOLLOW CYLINDER THEN RETURN

      IF (IFL .EQ. 1)RETURN

*IF SOLID, COMPUTE THE END CAPS OF THE CYLINDER IF NEEDED

      DO 700 L = 2,3
         DO 600 K = 1,4
            WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
            THETA = 0.0
            DO 500 I = 1,4
               DO 400 J = 1,4
                  PT(1) = FX(L)*LEN + PLACE
                  PT(2) = FY(K)*U(J)*RAD*SIN(THETA)
                  PT(3) = FZ(K)*U(J)*RAD*COS(THETA)
                  PTS(I,J,1) = PT(1)
                  PTS(I,J,2) = PT(2)
                  PTS(I,J,3) = PT(3)
400            CONTINUE
               THETA = THETA + DEL
500         CONTINUE
            CALL FSTCP(U,W,PTS,PNTS)
            CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

            DO 550 I = 1,4
            WRITE(10,*)(PNTS(I,J,1),J=1,4)
            WRITE(10,*)(PNTS(I,J,2),J=1,4)
            WRITE(10,*)(PNTS(I,J,3),J=1,4)
550         WRITE(10,*)(H0(1,J),J=1,4)

600      CONTINUE

700   CONTINUE

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE ELPSOD

```
**********************************************************************
**********************************************************************
**    SUBROUTINE ELPSOD(RX,RY,LOC,ORI,ICOL,BOOL)                    **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS ROUTINE COMPUTES THE CONTROL POINTS THAT WOULD BE        **
**    REQUIRED TO CREATE A ELLIPSOID USING B-SPLINE SURFACES        **
**                                                                  **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     11/21/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    RAD   = RADIUS OF THE ELLIPSOID (REAL,I/P)                    **
**    LOC   = LOCATION OF THE ELLIPSOID (REAL,I/P)                  **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE ELPSOD(RX,RY,LOC,ORI,ICOL,BOOL)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 LOC(3), PT(3), PTS(4,4,3), ORI(3)
      REAL*4 U(4), W(4), PNTS(20,20,3), H0(20,20), XS(2), YS(4), ZS(4)
      REAL*4 RX, RY
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
```

```
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA U/0.0,.33333333,.66666667,1.0/
      DATA W/0.0,.33333333,.66666667,1.0/
      DATA HO/400*1.0/
      DATA XS/1.0,-1.0/
      DATA YS/1.0,1.0,-1.0,-1.0/
      DATA ZS/1.0,-1.0,-1.0,1.0/

      PI = 3.14159

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      INNUM = INNUM + 1

      CALL FDINS(INNUM,LTYPE,'ELLIPSOI',8,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

      DELTA = PI/6.0

      DO 600 IK = 1,2
         DO 300 IJ = 1,4
            THETA = 0.0
            DO 200 I = 1,4
               ALPHA = 0.0
               DO 100 J = 1,4
                  PT(1) = RX*COS(THETA)*COS(ALPHA)*XS(IK)
                  PT(2) = RY*SIN(THETA)*YS(IJ)
                  PT(3) = RY*COS(THETA)*SIN(ALPHA)*ZS(IJ)
C                 CALL FTRNS(LOC,ORI,PT)
                  PTS(I,J,1) = PT(1)
                  PTS(I,J,2) = PT(2)
                  PTS(I,J,3) = PT(3)
                  ALPHA = ALPHA + DELTA
100            CONTINUE
               THETA = THETA + DELTA
200         CONTINUE
            CALL FSTCP(U,W,PTS,PNTS)
            CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)

            WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
            DO 250 I = 1,4
               WRITE(10,*)(PNTS(I,J,1),J=1,4)
               WRITE(10,*)(PNTS(I,J,2),J=1,4)
               WRITE(10,*)(PNTS(I,J,3),J=1,4)
               WRITE(10,*)(HO(1,J),J=1,4)
250         CONTINUE
300      CONTINUE
600   CONTINUE

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE FADTO

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FADTO(IER)                                         **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS SUBROUTINE WILL BE USED TO ADD TO A EXISTING OBJECT      **
**    OR ASSEMBLY.                                                  **
**                                                                  **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     12/16/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    IER   :  ERROR CODE                                           **
**                                                                  **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FADTO(IER)

      INTEGER*4 ASNUM, OBNUM, INNUM, CNUM

      CHARACTER ONAME*8, OBJECT(9000)*8, STRG*50, WORD(15)*12
```

```
      CHARACTER ASSEM(900)*8

      COMMON/COMP/ASSEM,OBJECT
      COMMON /IDS/ ASNUM, OBNUM, INNUM
      COMMON /PRTOF/CNUM

*GET NAME OF OBJECT

      IER = 0

      CALL FPRMP(17)

100   CALL FGTST(STRG)

      CALL FPARS(STRG,NW,WORD)
      ONAME = WORD(1)(1:8)

*CHECK TO SEE IF THIS OBJECT ALREADY EXISTS

      IF (ONAME .EQ. 'CANCEL  ')THEN
         IER = 1
         RETURN
      ENDIF

      DO 200 I = 1001,OBNUM
         IF (ONAME .EQ. OBJECT(I-1000))THEN
            STRG = 'ADDING TO OBJECT '//ONAME
            CALL FMSSG(30,STRG)
            NCOMP = I
            GOTO 400
         ENDIF
200   CONTINUE

      DO 300 I = 101,ASNUM
         IF (ONAME .EQ. ASSEM(I-100))THEN
            STRG = 'ADDING TO ASSEMBLY '//ONAME
            CALL FMSSG(30,STRG)
            NCOMP = I
            GOTO 400
         ENDIF
300   CONTINUE

      STRG = 'NON EXISTING COMPONENT. ENTER NEW NAME'
      CALL FMSSG(30,STRG)
      GOTO 100

400   CNUM = NCOMP

      RETURN
      END
```

SUBROUTINE FASSM

```
************************************************************************
************************************************************************
**    SUBROUTINE FASSM(NOBJ,LOCA,ORI)                                 **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING  **
**    AN ASSEMBLY. CORRECT LOCATION AND ORIENTATION OF THE OBJECT     **
**    IS ALSO REQUESTED                                               **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NOBJ   = ID OF OBJECT TO BE ASSEMBLED (INTEGER, O/P)            **
**    LOCA   = LOCATION OF THE OBJECT (REAL, O/P)                     **
**    ORI    = ORIENTATION OF THE OBJECT (REAL, O/P)                  **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FASSM(NOBJ,LOCA,ORI)

      REAL*4 LOCA(3), POS(2), ORI(3)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)

      INTEGER*4 COLOR, PPATH(3), ASNUM, OBNUM, INNUM

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER LCHAR*50, WORD(15)*12, MSG*32, CHAR1*8, CHAR2*8
      CHARACTER*8 ONAME, ASSEM(900), OBJECT(9000), FILE22

      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
```

```
*DEFINE GENERIC INFORMATION

      NOBJ = 0
      ONAME = '          '
      MSG = ' '
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

*CONVERT ALL NUMBERS TO CHARACTERS FOR DISPLAY

      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
      POS(2) = 0.80

      STRG = 'OBJECT = '//ONAME
      CALL GPPKID(1)
      CALL GPAN2(POS,17,STRG)

      POS(1) = 0.62
      POS(2) = 0.67

      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(2)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.54

      STRG = 'ORIENT =   '//CORI1//CORI2//CORI3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.41

      STRG = 'ASSEMBLY COMPLETE'
      CALL GPPKID(4)
      CALL GPAN2(POS,17,STRG)

      POS(1) = 0.62
      POS(2) = 0.28

      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)

      POS(1) = 0.62
```

```
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      CALL GPAN2(POS,32,MSG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION AND PROMPT USER FOR INPUT

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6)IPKID

*IF ITEM PICKED IS "OBJECT" THEN GET NAME OF OBJECT AND COMPUTE THE
*OBJECT NUMBER

1     CALL FGTST(LCHAR)
      NOBJ = 0
      CALL FPARS(LCHAR,NW,WORD)
      ONAME = WORD(1)(1:8)

      DO 200 I = 1,OBNUM-1000
         IF (ONAME .EQ. OBJECT(I))THEN
            NOBJ = I+1000
            MSG = ' '
            REWIND(22)

*IF ASSEMBLY INFORMATION EXISTS THEN CHECK IF OBJECT IS ALREADY A PART
*OF A ASSEMBLY

            IF (IFL22 .EQ. 22)THEN
               DO 150 J = 1,20000
                  READ(22,10,END=175)ICHK,CHAR1,JUNK,CHAR2
                  IF (ICHK .EQ. NOBJ)THEN
                     ONAME = ' '
                     NOBJ = 0
                     MSG  = 'OBJECT PART OF ASSEMBLY '//ASSEM(JUNK-100)
                  ENDIF
                  READ(22,*,END=175)R1,R2,R3,R4,R5,R6
150            CONTINUE
175            BACKSPACE(22)
            ENDIF
            GOTO 50
         ENDIF
200   CONTINUE

*IF NON EXISTING OBJECT, GET A NEW NAME

      IF (NOBJ .EQ. 0)THEN
         ONAME = ' '
         MSG  = 'NON-EXISTENT OBJECT'
      ENDIF
      GOTO 50

*GET LOCATION INFORMATION

2     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &   .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

*GET ORIENTATION INFORMATION

3     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &   .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
```

```
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

*ASSEMBLY IS COMPLETE

4     NOBJ = 0
      GOTO 300

*IF CHOICE IS "EXECUTE" BUT NO OBJECT NAME WAS GIVEN, GET NAME.
*IF ALL INFORMATION HAS BEEN GIVEN PROCEED TO ASSEMBLE

5     IF ( NOBJ. EQ. 0)THEN
         MSG = 'NO OBJECT NAME GIVEN'
         GOTO 50
      ENDIF
      IFL22 = 22
      GOTO 300

*"ABORT" CURRENT OBJECT ASSEMBLY

6     NOBJ = 1

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

300   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
10    FORMAT(1X,2(I4,1X,A8))
      END
```

SUBROUTINE FBCKV

```
************************************************************************
************************************************************************
**    SUBROUTINE FBCKV(NVIEW)                                         **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE WILL DISPLAY THE BACK VIEW OF ANY OBJECT.       **
**                                                                    **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/07/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NVIEW = VIEW IN WHICH THE TOP VIEW IS TO BE DISPLAYED           **
**            (INTEGER,I/P)                                           **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FBCKV(NVIEW)

      REAL*4 MAT(4,4), MAT1(4,4), MAT2(4,4), SCALEF(3)

      COMMON /SCLE/SCALEF

      PI = 3.14159
      DL = PI

*ROTATE THE VIEW ELEMENT ABOUT THE Y-AXIS BY +180 DEGREES

      CALL GPROTY(DL,MAT2)
```

```
*SCALE THE MODEL
      CALL GPSC3(SCALEF,MAT1)
*COMPOSE THE TRANSFORMATION
      CALL GPCMT3(MAT2,MAT1,MAT)
*ASSOCIATE THIS ROTATION TO THE VIEW
      CALL GPVMT3(1,NVIEW,MAT)
      CALL GPVP(1,NVIEW,2,1)
      RETURN
      END
```

SUBROUTINE FBMAT

```
************************************************************************
************************************************************************
**    SUBROUTINE FBMAT(IU,IW,MPTS,NPTS,PNTS)                          **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE WILL ASSEMBLE THE P-MATRIX TO BE USED IN THE    **
**    CREATION OF A B-SPLINE (PERIODIC, CUBIC) SURFACE.               **
**    AFTER ASSEMBLING THE MATRIX IT WILL COMPUTE THE POINTS ON THE   **
**    SURFACE.                                                        **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/03/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    IU     = FLAG INDICATING WHETHER SURFACE IS CLOSED AT U = 0     **
**             (INTEGER,I/P)                                          **
**             0 = OPEN                                               **
**             1 = CLOSED                                             **
**    IW     = FLAG INDICATING WHETHER SURFACE IS CLOSED AT W = 0     **
**             (INTEGER,I/P)                                          **
**             0 = OPEN                                               **
**             1 = CLOSED                                             **
**    MPTS   = NUMBER OF POINTS IN THE M-DIRECTION (INTEGER,I/P)      **
**    NPTS   = NUMBER OF POINTS IN THE N-DIRECTION (INTEGER,I/P)      **
**    PNTS   = ARRAY OF POINTS DEFINING THE SURFACE                   **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FBMAT(IU,IW,MPTS,NPTS,PNTS)

      REAL*4 PMAT(4,4,3), PTS(0:19,0:19,3), PT(20,20,3), PNTS(20,20,3)

*READ IN VALUES FOR THE CONTROL POINTS

      DO 300 I = 0,MPTS-1
         DO 200 J = 0,NPTS-1
            DO 100 K = 1,3
               PTS(I,J,K) = PNTS(I+1,J+1,K)
100         CONTINUE
200      CONTINUE
300   CONTINUE

*CHECK TO SEE WHETHER THE SURFACE IS CLOSED OR OPEN

*IF SURFACE IS OPEN AT U = 0.0 AND OPEN AT W = 0.0

      IF (IU .EQ. 0 .AND. IW .EQ. 0)THEN
         SEND = MPTS - 3
         TEND = NPTS - 3
      ENDIF

*IF SURFACE IS OPEN AT U = 0.0 AND CLOSED AT W = 0.0

      IF (IU .EQ. 0 .AND. IW .EQ. 1)THEN
         SEND = MPTS - 3
         TEND = NPTS
      ENDIF

*IF SURFACE IS CLOSED AT U = 0.0 AND OPEN AT W = 0.0

      IF (IU .EQ. 1 .AND. IW .EQ. 0)THEN
         SEND = MPTS
         TEND = NPTS - 3
      ENDIF

*IF SURFACE IS CLOSED AT U = 0.0 AND CLOSED AT W = 0.0

      IF (IU .EQ. 1 .AND. IW .EQ. 1)THEN
```

```
         SEND = MPTS
         TEND = NPTS
      ENDIF

*COMPUTE ALL ELEMENTS ON THE SURFACE

*COMPUTE ELEMENTS IN THE S-DIRECTION

      DO 800 I = 1,SEND

*COMPUTE ELEMENTS IN THE T-DIRECTION

         DO 700 J = 1,TEND

*SET UP THE P-MATRIX FOR EACH ELEMENT

            DO 600 II = 1,4
               DO 500 JJ = 1,4
                  DO 400 LL = 1,3
                     IX = MOD((I+II-2),MPTS)
                     IY = MOD((J+JJ-2),NPTS)
                     PMAT(II,JJ,LL) = PTS(IX,IY,LL)
400               CONTINUE
500            CONTINUE
600         CONTINUE

*COMPUTE ALL POINTS ON EACH SURFACE ELEMENT

            NU = 8
            NW = 8

            CALL FBMSH(PMAT,NU,NW,PT)

*DRAW THE CONSTANT U AND W LINES

            CALL FDRWU(IP,NP,IS,N,NU,NW,PT)
            CALL FDRWW(IP,NP,IS,N,NU,NW,PT)

      WRITE(10,*)I,J
      DO 650 K = 1,NW
      WRITE(10,11)(PT(L,K,1),L=1,NU)
      WRITE(10,11)(PT(L,K,2),L=1,NU)
650   WRITE(10,11)(PT(L,K,3),L=1,NU)
      WRITE(10,*)' '

700      CONTINUE
800   CONTINUE

      RETURN
11    FORMAT(1X,4(F6.2,1X))
      END
```

SUBROUTINE FBMSH

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FBMSH(PMAT,NU,NW,PT)                               **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS SUBROUTINE USES THE GEOMETRIC COEFFICIENT MATRIX TO      **
**    CALCULATE THE SPECIFIED NUMBER OF POINTS WHICH WOULD LIE ON   **
**    THE PERIODIC CUBIC B-SPLINE SURFACE                           **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     10/31/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    PMAT   = GEOMETRIC COEFFICIENT MATRIX (REAL,I/P)              **
**    NU     = NUMBER OF POINTS TO BE COMPUTED IN U-DIRECTION       **
**             (INTEGER,I/P)                                        **
**    NW     = NUMBER OF POINTS TO BE COMPUTED IN W-DIRECTION       **
**             (INTEGER,I/P)                                        **
**    PT     = COORDINATES OF THE POINTS ON THE SURFACE             **
**             (REAL,O/P)                                           **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FBMSH(PMAT,NU,NW,PT)

      INTEGER*4 NU, NW

      REAL*4 M(4,4), MT(4,4)
      REAL*4 U, VU(4), F(4), W, VW(4), S(4)
      REAL*4 PMAT(4,4,3), PXMAT(4,4), PYMAT(4,4), PZMAT(4,4)
      REAL*4 PT(20,20,3)

      DATA M /-1.0, 3.0,-3.0, 1.0,
     2         3.0,-6.0, 0.0, 4.0,
```

```
     3         -3.0, 3.0, 3.0, 1.0,
     4          1.0, 0.0, 0.0, 0.0/

      DATA MT/-1.0, 3.0,-3.0, 1.0,
     2          3.0,-6.0, 3.0, 0.0,
     3         -3.0, 0.0, 3.0, 0.0,
     4          1.0, 4.0, 1.0, 0.0/

*SPLIT THE P-MATRIX INTO ITS RESPECTIVE ELEMENTS

      DO 200 I = 1,4
         DO 100 J = 1,4
            PXMAT(I,J) = PMAT(I,J,1)
            PYMAT(I,J) = PMAT(I,J,2)
            PZMAT(I,J) = PMAT(I,J,3)
100      CONTINUE
200   CONTINUE

*CALCULATE THE DELTA CHANGES IN U AND W

      DELU = 1.0/FLOAT(NU -1)
      DELW = 1.0/FLOAT(NW -1)

*CALCULATE THE POINTS CORRESPONDING TO DIFFERENT VALUES OF U AND W

      U = 0.0

      DO 500 I = 1,NU
         W = 0.0
         CALL FUWVT(U,VU)
         CALL FMT14(VU,M,F)
         DO 400 J = 1,NW
            CALL FUWVT(W,VW)
            CALL FMT41(MT,VW,S)
            CALL FMT14(F,PXMAT,VU)
            CALL FMT11(VU,S,PT(I,J,1))
            CALL FMT14(F,PYMAT,VU)
            CALL FMT11(VU,S,PT(I,J,2))
            CALL FMT14(F,PZMAT,VU)
            CALL FMT11(VU,S,PT(I,J,3))
            W = W + DELW
            DO 300 K = 1,3
               PT(I,J,K) = PT(I,J,K)/36.0
300         CONTINUE
400      CONTINUE
         U = U + DELU
500   CONTINUE

      RETURN
      END
```

SUBROUTINE FBOTV

```
**********************************************************************
**********************************************************************
**     SUBROUTINE FBOTV(NVIEW)                                      **
**                                                                  **
**     PROGRAM DESCRIPTION                                          **
**                                                                  **
**     THIS SUBROUTINE WILL DISPLAY THE BOTTOM VIEW OF ANY OBJECT.  **
**                                                                  **
**                                                                  **
**                                                                  **
**     BY:       ASHIT R. GANDHI                                    **
**     DATE:     11/07/88                                           **
**                                                                  **
**     PARAMETERS USED:                                             **
**                                                                  **
**     NVIEW = VIEW IN WHICH THE BOTTOM VIEW IS TO BE DISPLAYED     **
**             (INTEGER,I/P)                                        **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FBOTV(NVIEW)

      REAL*4 MAT(4,4), MAT1(4,4), MAT2(4,4), SCALEF(3)

      COMMON /SCLE/ SCALEF

      PI = 3.14159
      DL = -PI/2.0

*ROTATE THE VIEW ELEMENT ABOUT THE X-AXIS BY -90 DEGREES

      CALL GPROTX(DL,MAT2)

*SCALE THE MODEL

      CALL GPSC3(SCALEF,MAT1)
```

```
*COMPOSE THE TRANSFORMATION
      CALL GPCMT3(MAT2,MAT1,MAT)
*ASSOCIATE THIS ROTATION TO THE VIEW
      CALL GPVMT3(1,NVIEW,MAT)
      CALL GPVP(1,NVIEW,2,1)
      RETURN
      END
```

SUBROUTINE FBPCH

```
************************************************************************
************************************************************************
**    SUBROUTINE FBPCH(MPTS,NPTS,IU,IW,PNTS,HO,ICOL,LTYPE)            **
**                                                                     **
**    PROGRAM DESCRIPTION                                              **
**                                                                     **
**    THIS SUBROUTINE WILL DRAW A B-SPLINE SURFACE PATCH FOR           **
**    CONTROL POINTS SPECIFIED BY THE USER                             **
**                                                                     **
**                                                                     **
**    BY:       ASHIT R. GANDHI                                        **
**    DATE:     11/22/88                                               **
**                                                                     **
**    PARAMETERS USED:                                                 **
**                                                                     **
**    MPTS = NUMBER OF POINTS IN M-DIRECTION (INTEGER,I/P)             **
**    NPTS = NUMBER OF POINTS IN N-DIRECTION (INTEGER,I/P)             **
**    IU   = FLAG FOR CLOSING IN U-DIRECTION (INTEGER,I/P)             **
**    IW   = FLAG FOR CLOSING IN W-DIRECTION (INTEGER,I/P)             **
**    PNTS = ARRAY OF CONTROL POINTS DEFINING PATCH (REAL,I/P)         **
**    HO   = HOMOGENOUS COORDINATES (REAL, I/P)                        **
**    ICOL = COLOR ATTRIBUTE TO BE USED (INTEGER, I/P)                 **
**    LTYPE= LINE TYPE TO BE USED (INTEGER, I/P)                       **
************************************************************************
************************************************************************
      SUBROUTINE FBPCH(MPTS,NPTS,IU,IW,PNTS,HO,ICOL,LTYPE)

      INTEGER*4 ASNUM,OBNUM,INNUM
      REAL*4 PNTS(20,20,3), HO(20,20), SCALEF(3)
      COMMON /SCLE/SCALEF
      COMMON/POLY/IPGNR
      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM

*SET COUNTER FOR POLYGON NUMBER

      IPGN = IPGNR
      IF (IPGNR .EQ. 0)THEN
         IPGN = IPGNR
         IPGNR = 1
      ENDIF

*SET DEFAULT SCALE FACTORS

      SCALEF(1) = 1.0
      SCALEF(2) = 1.0
      SCALEF(3) = 1.0

*OPEN CORRECT STRUCTURE AND ATTACH ATTRIBUTES

      CALL GPOPST(INNUM)
      CALL GPPLCI(ICOL)

*COMPUTE THE MATRIX FOR THE SURFACE AND COMPUTE THE POINTS ON IT

      CALL FRBMT(IPGN,IU,IW,MPTS,NPTS,PNTS,HO,ICOL)

      CALL GPCLST

      IPGNR = IPGN

300   RETURN
      END
```

SUBROUTINE FCOMM

```
************************************************************************
************************************************************************
**    SUBROUTINE FCOMM(IPGN,PGN,ICOL,IOPGN)                            **
```

```
**                                                                      **
**     PROGRAM DESCRIPTION                                              **
**                                                                      **
**     THIS ROUTINE SETS UP THE PARAMETERS REQUIRED FOR RENDERING       **
**     A OBJECT OR ASSEMBLY                                             **
**                                                                      **
**     BY:       ASHIT R. GANDHI                                        **
**     DATE:     01/13/89                                               **
**                                                                      **
**     PARAMETERS USED:                                                 **
**                                                                      **
**     IPGN   = POLYGON NUMBER (INTEGER, I/P)                           **
**     PGN    = POLYGON COORDINATES (REAL, I/P)                         **
**     ICOL   = COLOR ATTRIBUTE FOR POLYGON (INTEGER, I/P)              **
**     IOPGN = STARTING ELEMENT FOR PART (INTEGER, I/P)                 **
**                                                                      **
**********************************************************************
**********************************************************************
      SUBROUTINE FCOMM(IPGN,PGN,ICOL,IOPGN)

      REAL PGN(3,20000,5), COL(3), XYZSUB(3,20000,5)
      INTEGER SHDFLG
      INTEGER NPSUB(20000), NPARTS, PRTLMT(2,100), CMPCLR(100)

      COMMON /POLDAT/ NPARTS, PRTLMT, NSUBT, NPSUB, XYZSUB
      COMMON /CPCL/CMPCLR
      COMMON/IDS/ASNUM,OBNUM,INNUM

      DATA NOLD/0/

*COMPUTE THE PART NUMBER

      NPARTS = INNUM - 10000

*IF A NEW PART THEN COMPUTE FIRST ELEMENT NUMBER

      IF (NOLD .NE. NPARTS)PRTLMT(1,NPARTS) = IOPGN

      NOLD = NPARTS

*COMPUTE THE LAST ELEMENT IN THE CURRENT PART

      PRTLMT(2,NPARTS) = IPGN

*ASSIGN COLOR ATTRIBUTE TO CURRENT PART

      CMPCLR(NPARTS) = ICOL

*COMPUTE TOTAL NUMBER OF POLYGONS

      NSUBT = IPGN

*ASSIGN NUMBER OF EDGES TO EACH POLYGON

      DO 100 I = IOPGN,IPGN
         NPSUB(I) = 4
100   CONTINUE

*COMPUTE COORDINATES OF POLYGON VERTICES

      DO 400 I = 1,3
         DO 300 J = IOPGN,IPGN
            DO 200 K = 1,NPSUB(J)
               XYZSUB(I,J,K) = PGN(I,J,K)
200         CONTINUE
300      CONTINUE
400   CONTINUE

      IOPGN = IPGN

      RETURN
      END
```

SUBROUTINE FCTCP

```
**********************************************************************
**********************************************************************
**     SUBROUTINE FCTCP(U,B,P)                                          **
**                                                                      **
**     PROGRAM DESCRIPTION                                              **
**                                                                      **
**     THIS ROUTINE WILL CALCULATE THE CONTROL POINTS NEEDED TO CREATE  **
**     A PERIODIC B-SPLINE CURVE THAT PASSES THROUGH FOUR               **
**     POINTS SPECIFIED BY THE USER.                                    **
**                                                                      **
**     BY:       ASHIT R. GANDHI                                        **
**     DATE:     11/12/88                                               **
**                                                                      **
**     PARAMETERS USED:                                                 **
**                                                                      **
```

```
**     U = ARRAY OF PARAMETER (U) VALUES CORRESPONDING TO INPUT POINTS **
**         (REAL,I/P)                                                  **
**     B = ARRAY OF POINTS THROUGH WHICH THE B-SPLINE CURVE SHOULD PASS**
**         (REAL,I/P)                                                  **
**     P = ARRAY OF CONTROL POINTS NEEDED TO GET THE SPECIFIED CURVE   **
**         (REAL,O/P)                                                  **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FCTCP(U,B,P)

      REAL*4 UM(4,4), M(4,4), B(4,3), P(4,3), VU(4), A(4,4), AINV(4,4)
      REAL*4 B1(4),B2(4),B3(4), U(4)
      REAL*4 P1(4),P2(4),P3(4)

      DATA M /-1.0, 3.0,-3.0, 1.0,
     2         3.0,-6.0, 0.0, 4.0,
     3        -3.0, 3.0, 3.0, 1.0,
     4         1.0, 0.0, 0.0, 0.0/

*FOR EACH VALUE OF U COMPUTE THE PARAMETER VECTOR AND ASSEMBLE THE
*COMPLETE 4X4 MATRIX

      DO 200 I = 1,4
         CALL FUNVT(U(I),VU)
         DO 100 J = 1,4
            UM(I,J) = VU(J)
100      CONTINUE
200   CONTINUE

*MULTIPLY MATRIX(U) AND MATRIX(M) TO GET MATRIX(A) AND THEN SCALE
*MATRIX(A)

      CALL FMT44(UM,M,A)

      DO 400 I = 1,4
         DO 300 J = 1,4
            A(I,J) = A(I,J)*27.0
300      CONTINUE
400   CONTINUE

*COMPUTE THE CLOSED FORM INVERSE OF MATRIX(A) AND SCALE MATRIX(AINV)

      CALL MATINV(A,AINV)

      DO 600 I = 1,4
         DO 500 J = 1,4
            AINV(I,J) = AINV(I,J)*162.0
500      CONTINUE
600   CONTINUE

*ASSIGN APPROPRIATE VALUES OF COORDINATES TO THE ARRAY

      DO 700 I = 1,4
         B1(I) = B(I,1)
         B2(I) = B(I,2)
         B3(I) = B(I,3)
700   CONTINUE

*COMPUTE THE CONTROL POINTS COORDINATES

      CALL FMT41(AINV,B1,P1)
      CALL FMT41(AINV,B2,P2)
      CALL FMT41(AINV,B3,P3)

*ASSIGN APPROPRIATE VALUES OF COORDINATES TO THE ARRAY

      DO 800 I = 1,4
         P(I,1) = P1(I)
         P(I,2) = P2(I)
         P(I,3) = P3(I)
800   CONTINUE

      RETURN
      END
```

SUBROUTINE FDASM

```
*************************************************************************
*************************************************************************
**     SUBROUTINE FDASM (IADTO)                                        **
**                                                                     **
**     PROGRAM DESCRIPTION                                             **
**                                                                     **
**     THIS ROUTINE CREATES A NEW ASSEMBLY.  IT IS ALSO USED TO        **
**     ADD TO A PREVIOUSLY DEFINED ASSEMBLY.                           **
**                                                                     **
**                                                                     **
**     BY:       ASHIT R. GANDHI                                       **
**     DATE:     12/16/88                                              **
```

```
**                                                                     **
**     PARAMETERS USED:                                                **
**                                                                     **
**     IADTO = FLAG INDICATING NEW OR OLD ASSEMBLY (INTEGER, I/P)      **
**                                                                     **
**                                                                     **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FDASM(IADTO)

      INTEGER*4 ASNUM, OBNUM, INNUM, CNUM

      REAL*4 ASMAT(4,4), MAT1(4,4), MAT2(4,4), MAT(4,4)
      REAL*4 OLOC(3), OORI(3)

      CHARACTER ANAME*8, OBJECT(9000)*8, STRG*50, WORD(15)*12
      CHARACTER ASSEM(900)*8

      COMMON/COMP/ASSEM,OBJECT
      COMMON /IDS/ ASNUM, OBNUM, INNUM
      COMMON /PRTOF/CNUM

      DATA ASMAT/1.0,4*0.0,1.0,4*0.0,1.0,4*0.0,1.0/


      IF (IADTO .EQ. 1)GOTO 400

*GET NAME OF OBJECT

      CALL FPRMP(15)

100   CALL FGTST(STRG)

      CALL FPARS(STRG,NW,WORD)
      ANAME = WORD(1)(1:8)

      IF (ANAME .EQ. 'CANCEL  ')RETURN

*CHECK TO SEE IF THIS ASSEMBLY ALREADY EXISTS

      DO 200 I = 101,ASNUM
         IF (ANAME .EQ. ASSEM(I-100))THEN
            STRG = 'ASSEMBLY EXISTS. PLEASE ENTER NEW NAME'
            CALL FMSSG(30,STRG)
            GOTO 100
         ENDIF
200   CONTINUE

      DO 300 I = 1001,OBNUM
         IF (ANAME .EQ. OBJECT(I-1000))THEN
            STRG = 'OBJECT WITH SAME NAME EXISTS. PLEASE ENTER NEW NAME'
            CALL FMSSG(30,STRG)
            GOTO 100
         ENDIF
300   CONTINUE

*IF NEW OBJECT THEN PROCEED

      STRG = 'DEFINED ASSEMBLY '//ANAME
      CALL FMSSG(30,STRG)

*DEFINE STRUCTURE FOR OBJECT

      ASNUM = ASNUM + 1

      ASSEM(ASNUM-100) = ANAME
      WRITE(31,30)ASNUM,ANAME

      CALL GPOPST(ASNUM)
      CALL GPMLX3(ASMAT,1)
      CALL GPCLST
      CNUM = ASNUM

400   NOBJ = 0
      CALL FASSM(NOBJ,OLOC,OORI)

      IF (NOBJ .EQ. 0)RETURN
      IF (NOBJ .EQ. 1)GOTO 400

      CALL FURMT(OORI,MAT1)
      CALL GPTRL3(OLOC,MAT2)
      CALL GPCMT3(MAT1,MAT2,MAT)

      CALL GPOPST(CNUM)
        CALL GPMLX3(MAT,3)
        CALL GPEXST(NOBJ)
      CALL GPCLST

      WRITE(22,10)NOBJ, OBJECT(NOBJ-1000), CNUM, ASSEM(CNUM-100)
      WRITE(22,20)OLOC,OORI
      GOTO 400

10    FORMAT(1X,2(I4,1X,A8))
20    FORMAT(1X,6(F8.3,1X))
30    FORMAT(1X,I3,1X,A8)
```

```
      END
```

SUBROUTINE FDELT

```
************************************************************************
************************************************************************
**    SUBROUTINE FDELT                                                **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE WILL DELETE A SPECIFIED COMPONENT               **
**                                                                    **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     12/18/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NONE                                                            **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FDELT

      INTEGER*4 ASNUM,OBNUM,INNUM
      CHARACTER*8 ASSEM(900), OBJECT(9000), DCOMP
      CHARACTER WORD(15)*12, STRG*50

      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT

*PROMPT USER FOR INPUT

      CALL FPRMP(16)

*GET COMPONENT TO BE DELETED

100   CALL FGTST(STRG)
      CALL FPARS(STRG,NW,WORD)
      DCOMP = WORD(1)(1:8)

*IF "CANCEL" RETURN

      IF (DCOMP .EQ. 'CANCEL  ')RETURN

*COMPUTE ID OF COMPONENT TO BE DELETED

      DO 200 I = 101,ASNUM
         IF (DCOMP .EQ. ASSEM(I-100))THEN
            IDDEL = I
            GOTO 400
         ENDIF
200   CONTINUE

      DO 300 I = 1001,OBNUM
         IF (DCOMP .EQ. OBJECT(I-1000))THEN
            IDDEL = I
            GOTO 400
         ENDIF
300   CONTINUE

*IF NON EXISTING OBJECT, QUERY FOR NEW NAME

      STRG = 'NON-EXISITING COMPONENT. ENTER NEW NAME.'
      CALL FMSSG(30,STRG)
      GOTO 100

400   CALL GPDLST(IDDEL)

*COMPLETED REQUEST MESSAGE

      STRG = 'DELETED COMPONENT '//DCOMP
      CALL FMSSG(30,STRG)

500   RETURN
      END
```

SUBROUTINE FDINS

```
************************************************************************
************************************************************************
```

```
**     SUBROUTINE FDINS(INNUM,LTYPE,INST,NSURF,OBNUM,OBNAME,LOC,DCS)   **
**                                                                     **
**     PROGRAM DESCRIPTION                                             **
**                                                                     **
**     THIS ROUTINE CREATES NEW FEATURE INSTANCES. IT EXECUTES A       **
**     NEW STRUCTURE AND MAKES IT A PART OF THE OBJECT STRUCTURE       **
**                                                                     **
**                                                                     **
**     BY:       ASHIT R. GANDHI                                       **
**     DATE:     12/16/88                                              **
**                                                                     **
**     PARAMETERS USED:                                                **
**                                                                     **
**     INNUM = INSTANCE NUMBER                                         **
**     LTPYE = LINE TYPE                                               **
**     INST  = NAME OF INSTANCE                                        **
**     NSURF = NUMBER OF PATCHES MAKING UP THE INSTANCE                **
**     OBNUM = OBJECT NUMBER THAT INSTANCE IS PART OF                  **
**     OBNAME= NAME OF PARENT OBJECT                                   **
**     LOC   = LOCATION IN OBJECT COORDINATES                          **
**     ORI   = ORIENTATION IN OBJECT COORDINATES                       **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FDINS(INNUM,LTYPE,INST,NSURF,OBNUM,OBNAME,LOC,DCS)

      NAMELIST /INFO/PLACE, ORIENT

      INTEGER*4 OBNUM, INNUM, LTYPE

      REAL*4 LOC(3), DCS(3), MAT1(4,4), MAT2(4,4), MAT3(4,4)
      REAL*4 PLACE(3), ORIENT(3)
      CHARACTER*8 INST, OBNAME

*COMPUTE THE CORRECT LOCATION AND ORIENTATION

      DO 100 I = 1,3
         PLACE(I) = LOC(I)
         ORIENT(I) = DCS(I)
100   CONTINUE

*ESTABLISH THE CORRECT TRANSFORMATION MATRIX

      CALL FURMT(DCS,MAT1)
      CALL GPTRL3(LOC,MAT2)
      CALL GPCMT3(MAT1,MAT2,MAT3)

*EXECUTE STRUCTURE INSIDE OBJECT STRUCTURE

      CALL GPOPST(OBNUM)
      CALL GPMLX3(MAT3,3)
      CALL GPEXST(INNUM)
      CALL GPCLST

      WRITE(10,10)INNUM, LTYPE, INST, NSURF, OBNUM, OBNAME
      WRITE(10,20)LOC,DCS

      RETURN
10    FORMAT(1X,I5,1X,I2,1X,A8,1X,I2,1X,I4,1X,A8)
20    FORMAT(1X,6(F8.3,1X))
      END
```

SUBROUTINE FDISP

```
*************************************************************************
*************************************************************************
**     SUBROUTINE FDISP                                                **
**                                                                     **
**     PROGRAM DESCRIPTION                                             **
**                                                                     **
**     THIS ROUTINE DISPLAYS THE SPECIFIED OBJECT OR ASSEMBLY.         **
**     IF AN UNKNOWN NAME IS GIVEN IT PROMPTS THE USER FOR A NEW NAME. **
**     ALLOWS ONLY ONE COMPONENT TO BE DISPLAYED AT A TIME.            **
**                                                                     **
**     BY:       ASHIT R. GANDHI                                       **
**     DATE:     01/13/89                                              **
**                                                                     **
**     PARAMETERS USED:                                                **
**                                                                     **
**     NONE                                                            **
**                                                                     **
**                                                                     **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FDISP

      INTEGER*4 ASNUM, OBNUM, INNUM

      CHARACTER CNAME*8, STRG*50, WORD(15)*12
```

```
      CHARACTER ASSEM(900)*8, OBJECT(9000)*8

      COMMON/IDS/ASNUM, OBNUM, INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/VIS/NVIS

      DATA NOLD/100000/

      IF (NOLD .EQ. 100000)THEN
         CALL GPOPST(NOLD)
         CALL GPCLST
         CALL GPARV(1,7,NOLD,0)
         CALL GPARV(1,8,NOLD,0)
         CALL GPARV(1,9,NOLD,0)
         CALL GPARV(1,10,NOLD,0)
         CALL GPARV(1,11,NOLD,0)
      ENDIF

*PROMPT USER FOR INPUT

      CALL FPRMP(18)

*GET COMPONENT NAME FROM USER

100   CALL FGTST(STRG)
      CALL FPARS(STRG,NW,WORD)
      CNAME = WORD(1)(1:8)

*IF "CANCEL" RETURN

      IF (CNAME .EQ. 'CANCEL  ')THEN
         RETURN
      ENDIF

*CHECK IF COMPONENT ENTERED IS AN OBJECT

      NCOMP = 0
      DO 200 I = 1001,OBNUM
         IF (CNAME .EQ. OBJECT(I-1000))THEN
            NCOMP = I
            GOTO 400
         ENDIF
200   CONTINUE

*CHECK IF COMPONENT ENTERED IS AN ASSEMBLY

      DO 300 I = 101,ASNUM
         IF (CNAME .EQ. ASSEM(I-100))THEN
            NCOMP = I
            GOTO 400
         ENDIF
300   CONTINUE

*IF COMPONENT ENTERED IS NEITHER, THEN PROMPT FOR NEW NAME

      STRG = 'NON-EXISTENT COMPONENT. ENTER NEW NAME'
      CALL FMSSG(30,STRG)
      GOTO 100

*DISASSOCIATE EXISTING COMPONENT FROM THE SCREEN

400   CALL GPDRV(1,7,NOLD)
      CALL GPDRV(1,8,NOLD)
      CALL GPDRV(1,9,NOLD)
      CALL GPDRV(1,10,NOLD)
      CALL GPDRV(1,11,NOLD)

*ASSOCIATE REQUESTED COMPONENT WITH THE SCREEN

      CALL GPARV(1,7,NCOMP,0)
      CALL GPARV(1,8,NCOMP,0)
      CALL GPARV(1,9,NCOMP,0)
      CALL GPARV(1,10,NCOMP,0)
      CALL GPARV(1,11,NCOMP,0)
      CALL GPVP(1,7,2,1)

*COMPLETED REQUESTED MESSAGE

      STRG = 'DISPLAYED COMPONENT '//CNAME
      CALL FMSSG(30,STRG)

*SET CURRENTLY VISIBLE STRUCTURE IDS

      NOLD = NCOMP
      NVIS = NCOMP

      RETURN
      END
```

```
************************************************************************
************************************************************************
**     SUBROUTINE FDOBJ(IER)                                          **
**                                                                    **
**     PROGRAM DESCRIPTION                                            **
**                                                                    **
**     THIS ROUTINE ALLOWS CREATION AND DEFINITION OF AN OBJECT.      **
**                                                                    **
**                                                                    **
**                                                                    **
**     BY:       ASHIT R. GANDHI                                      **
**     DATE:     12/16/88                                             **
**                                                                    **
**     PARAMETERS USED:                                               **
**                                                                    **
**     IER = ERROR CODE INDICATING SUCCESS                            **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FDOBJ(IER)

      INTEGER*4 ASNUM, OBNUM, INNUM, CNUM

      REAL*4 OBMAT(4,4)

      CHARACTER ONAME*8, OBJECT(9000)*8, STRG*50, WORD(15)*12
      CHARACTER ASSEM(900)*8

      COMMON/COMP/ASSEM,OBJECT
      COMMON /IDS/ ASNUM, OBNUM, INNUM
      COMMON /PRTOF/CNUM
      COMMON/VIS/NVIS

      DATA OBMAT/1.0,4*0.0,1.0,4*0.0,1.0,4*0.0,1.0/
      DATA NOLD/100000/

*GET NAME OF OBJECT

      IER = 0

      CALL FPRMP(15)

100   CALL FGTST(STRG)

      CALL FPARS(STRG,NW,WORD)
      ONAME = WORD(1)(1:8)

*CHECK TO SEE IF THIS OBJECT ALREADY EXISTS

      IF (ONAME .EQ. 'CANCEL  ')THEN
         IER = 1
         RETURN
      ENDIF

      DO 200 I = 1001,OBNUM
         IF (ONAME .EQ. OBJECT(I-1000))THEN
            STRG = 'OBJECT EXISTS. PLEASE ENTER NEW NAME'
            CALL FMSSG(30,STRG)
            GOTO 100
         ENDIF
200   CONTINUE

      DO 300 I = 101,ASNUM
         IF (ONAME .EQ. ASSEM(I-100))THEN
            STRG = 'ASSEMBLY EXISTS. PLEASE ENTER NEW NAME'
            CALL FMSSG(30,STRG)
            GOTO 100
         ENDIF
300   CONTINUE

*IF NEW OBJECT THEN PROCEED

*DEFINE STRUCTURE FOR OBJECT

      OBNUM = OBNUM + 1
      CNUM = OBNUM

      OBJECT(OBNUM-1000) = ONAME
      WRITE(32,10)OBNUM,ONAME

      STRG = 'DEFINED OBJECT '//OBJECT(OBNUM-1000)
      CALL FMSSG(30,STRG)

      CALL GPOPST(OBNUM)
      CALL GPMLX3(OBMAT,1)
      CALL GPCLST

      IF (NOLD .NE. 100000)THEN
```

```
         CALL GPDRV(1,7,NOLD)
         CALL GPDRV(1,8,NOLD)
         CALL GPDRV(1,9,NOLD)
         CALL GPDRV(1,10,NOLD)
         CALL GPDRV(1,11,NOLD)
      ENDIF

      CALL GPARV(1,7,OBNUM,0)
      CALL GPARV(1,8,OBNUM,0)
      CALL GPARV(1,9,OBNUM,0)
      CALL GPARV(1,10,OBNUM,0)
      CALL GPARV(1,11,OBNUM,0)
      CALL GPVP(1,7,2,1)

      NOLD = OBNUM
      NVIS = OBNUM

      RETURN
10    FORMAT(1X,I4,1X,A8)
      END
```

SUBROUTINE FDRWU

```
************************************************************************
************************************************************************
**    SUBROUTINE FDRWU(IP,NP,IS,N,NU,NW,PT)                           **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE DRAWS THE CONSTANT U LINES FOR A PATCH.         **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     10/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**    IP    = THE PATCH NUMBER (INTEGER,INPUT)                        **
**    NP    = TOTAL NUMBER OF PATCHES (INTEGER,INPUT)                 **
**    IS    = THE SECTION NUMBER (INTEGER,INPUT)                      **
**    N     = TOTAL NUMBER OF CROSS SECTIONS (INTEGER,INPUT)          **
**    NU    = TOTAL NUMBER OF CONSTANT U LINES (INTEGER,INPUT)        **
**    NW    = TOTAL NUMBER OF CONSTANT W LINES (INTEGER,INPUT)        **
**    PT    = ARRAY OF POINTS THROUGH WHICH THE LINES WILL BE DRAWN   **
**            (REAL,INPUT)                                            **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FDRWU(IP,NP,IS,N,NU,NW,PT)

      INTEGER*4 IP,IS,NU,NW

      REAL*4 PT(20,20,3), PTS(60)

*DECIDE THE STARTING AND ENDING LINES TO BE DRAWN (TO AVOID REDRAWING)

         ISTRT = 1
         IEND = NU

*CREATE ARRAY FOR GRAPHICAL INPUT

      DO 200 L = ISTRT,IEND
         DO 100 M = 1,NW
            PTS(3*M-2) = PT(L,M,1)
            PTS(3*M-1) = PT(L,M,2)
            PTS(3*M  ) = PT(L,M,3)
100      CONTINUE

*DRAW THE LINE

         CALL GPPL3(NW,3,PTS)
200   CONTINUE

      RETURN
      END
```

SUBROUTINE FDRWW

```
************************************************************************
************************************************************************
**    SUBROUTINE FDRWW(IP,NP,IS,N,NU,NW,PT)                           **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE DRAWS THE CONSTANT W LINES FOR A PATCH.         **
```

```
**                                                                      **
**                                                                      **
**     BY:       ASHIT R. GANDHI                                        **
**     DATE:     10/12/88                                               **
**                                                                      **
**     PARAMETERS USED:                                                 **
**     IP     = THE PATCH NUMBER (INTEGER,INPUT)                        **
**     NP     = TOTAL NUMBER OF PATCHES (INTEGER,INPUT)                 **
**     IS     = THE SECTION NUMBER (INTEGER,INPUT)                      **
**     N      = TOTAL NUMBER OF CROSS SECTIONS (INTEGER,INPUT)          **
**     NU     = TOTAL NUMBER OF CONSTANT U LINES (INTEGER,INPUT)        **
**     NW     = TOTAL NUMBER OF CONSTANT W LINES (INTEGER,INPUT)        **
**     PT     = ARRAY OF POINTS THROUGH WHICH THE LINES WILL BE DRAWN   **
**              (REAL,INPUT)                                            **
**                                                                      **
**************************************************************************
**************************************************************************
      SUBROUTINE FDRWW(IP,NP,IS,N,NU,NW,PT)

      INTEGER*4 IP,IS,NU,NW

      REAL*4 PT(20,20,3), PTS(60)

*DECIDE THE STARTING AND ENDING LINES TO BE DRAWN (TO AVOID REDRAWING)

      IF (IP .EQ. NP)IEND = NW - 1
      ISTRT = 1
      IEND = NW

*CREATE ARRAY FOR GRAPHICAL INPUT

      DO 200 M = ISTRT,IEND
         DO 100 L = 1,NU
            PTS(3*L-2) = PT(L,M,1)
            PTS(3*L-1) = PT(L,M,2)
            PTS(3*L  ) = PT(L,M,3)
100      CONTINUE

*DRAW THE LINE

         CALL GPPL3(NU,3,PTS)

200   CONTINUE

      RETURN
      END
```

SUBROUTINE FEDAT

```
**************************************************************************
**************************************************************************
**     SUBROUTINE FEDAT(IEDFL,IEDLT,IEDCL)                              **
**                                                                      **
**     SUBROUTINE DESCRIPTION                                           **
**                                                                      **
**     THIS SUBROUTINE SETS UP POLYGON EDGE ATTRIBUTES.                 **
**                                                                      **
**     BY:   Ashit R. Gandhi                                            **
**     DATE: 10/07/88                                                   **
**                                                                      **
**     PARAMETERS USED:                                                 **
**                                                                      **
**     IEDFL  : EDGE FLAG  1=OFF, 2=ON                                  **
**     IEDLT  : EDGE LINE TYPE                                          **
**                                                                      **
**************************************************************************
**************************************************************************
      SUBROUTINE FEDAT(IEDFL,IEDLT,IEDCL)

*     SET UP EDGE FLAG

      CALL GPEF(IEDFL)

*     SET EDGE LINE TYPE

      CALL GPELT(IEDLT)

*     SET EDGE LINE COLOR

      CALL GPECI(IEDCL)

      RETURN
      END
```

SUBROUTINE FFNTV

```
************************************************************************
************************************************************************
**     SUBROUTINE FFNTV(NVIEW)                                        **
**                                                                    **
**     PROGRAM DESCRIPTION                                            **
**                                                                    **
**     THIS SUBROUTINE WILL DISPLAY THE TOP VIEW OF ANY OBJECT.       **
**                                                                    **
**                                                                    **
**                                                                    **
**     BY:       ASHIT R. GANDHI                                      **
**     DATE:     11/07/88                                             **
**                                                                    **
**     PARAMETERS USED:                                               **
**                                                                    **
**     NVIEW = VIEW IN WHICH THE TOP VIEW IS TO BE DISPLAYED          **
**             (INTEGER,I/P)                                          **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FFNTV(NVIEW)

      REAL*4 MAT(4,4), MAT1(4,4), MAT2(4,4), SCALEF(3)

      COMMON /SCLE/SCALEF

      PI = 3.14159
      DL = 0.0

*ROTATE THE VIEW ELEMENT ABOUT THE X-AXIS BY +90 DEGREES

      CALL GPROTX(DL,MAT2)

*SCALE THE MODEL

      CALL GPSC3(SCALEF,MAT1)

*COMPOSE THE TRANSFORMATION

      CALL GPCMT3(MAT2,MAT1,MAT)

*ASSOCIATE THIS ROTATION TO THE VIEW

      CALL GPVMT3(1,NVIEW,MAT)

      CALL GPVP(1,NVIEW,2,1)

      RETURN
      END
```

SUBROUTINE FGTST

```
************************************************************************
************************************************************************
**     SUBROUTINE FGTST(STRG)                                         **
**                                                                    **
**     PROGRAM DESCRIPTION                                            **
**                                                                    **
**     THIS SUBROUTINE GETS A STRING INPUT FROM THE USER. THE STRING  **
**     INPUT IS TAKEN FROM AN ACTIVATED AREA RESERVED FOR SUCH INPUT  **
**                                                                    **
**     BY:       ASHIT R. GANDHI                                      **
**     DATE:     11/04/88                                             **
**                                                                    **
**     PARAMETERS USED:                                               **
**                                                                    **
**     STRG = STRING INPUT (CHARACTER, O/P)                           **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FGTST(STRG)

      COMMON CSIZE

      INTEGER*4 L

      REAL*4 CSIZE(3), AREA(6)

      CHARACTER STRG*(*), EMTY*50, PREF*9, MSSG*59

      DATA EMTY/'                                                '/
      STRG = '

*PUT STRING IN REQUEST MODE
```

```
      CALL GPSTMO(1,1,1,2)
*GET STRING
      L=50
100   CALL GPRQST(1,1,50,2,LR,STRG)
      IF (LR .EQ. 0) GOTO 100

*CONVERT STRING TO ALL CAPITALS
      CALL FCAPS(STRG)
      CALL GPVP(1,6,2,1)
*PUT STRING BACK IN EVENT MODE
      CALL GPSTMO(1,1,3,2)
      RETURN
      END
```

SUBROUTINE FINIT

```
************************************************************************
************************************************************************
**    SUBROUTINE FINIT                                                **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE INITIALIZES ALL INPUT DEVICES.                  **
**                                                                    **
**    BY:      ASHIT R. GANDHI                                        **
**    DATE:    10/26/88                                               **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NONE                                                            **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FINIT

      INTEGER*4 PPATH(3)

      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)

      CHARACTER EMTY*50

      DATA EMTY/'                                                  '/

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE
      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

************************************************************************
*        PREPARE FOR PICK INPUT                                        *
************************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

************************************************************************
*        PREPARE FOR STRING INPUT                                      *
************************************************************************

*PLACE STRING DEVICE IN THE REQUEST MODE

      CALL GPSTMO(1,1,1,2)
```

```
*SETUP STRING INPUT AREA

      SAREA(1)=0.007*CSIZE(1)
      SAREA(2)=0.845*CSIZE(1)
      SAREA(3)=0.15*CSIZE(2)
      SAREA(4)=0.17*CSIZE(2)
      SAREA(5)=0.0
      SAREA(6)=CSIZE(3)

*INITALIZE THE STRING DEVICE

      CALL GPINST(1,1,50,EMTY,1,SAREA,50,1,0,EMTY)

*PLACE STRING DEVICE IN THE EVENT MODE

      CALL GPSTMO(1,1,3,2)

*************************************************************************
*        PREPARE FOR VALUATOR INPUT                                     *
*************************************************************************

*DEFINE PICK AREA

      VAREA(1) = 0.0
      VAREA(2) = CSIZE(1)
      VAREA(4) = CSIZE(2)
      VAREA(5) = 0.0
      VAREA(6) = CSIZE(3)

*PLACE VALUATOR DEVICES IN THE EVENT MODE
*INITIALIZE VALUATORS TO GET VALUES FROM 0 TO 360

      DO 100 I = 1,3

        VAREA(3) = CSIZE(2) - 0.01*I

        CALL GPVLMO(1,I,1,1)

        CALL GPINVL(1,I,0.,1,VAREA,0.,360.,0,DATA)

        CALL GPVLMO(1,I,3,2)

100   CONTINUE

        VAREA(3) = CSIZE(2) - 0.01*4

        CALL GPVLMO(1,4,1,1)

        CALL GPINVL(1,4,1.,1,VAREA,0.01,5.,0,DATA)

        CALL GPVLMO(1,4,3,2)
      RETURN
      END
```

SUBROUTINE FINPT

```
*************************************************************************
*************************************************************************
**    SUBROUTINE FINPT(PMENU,ICLASS,IDEV,IPKID,V,STRG)                 **
**                                                                     **
**    PROGRAM DESCRIPTION                                              **
**                                                                     **
**    THIS SUBROUTINE WILL WAIT FOR AN INPUT FROM THE USER. ONCE THE   **
**    INPUT IS GIVEN IT WILL RETURN INFORMATION ABOUT THE KIND OF      **
**    INPUT THAT WAS RECEIVED AND ITS VALUES TO BE FURTHER PROCESSED.  **
**                                                                     **
**    BY:       ASHIT R. GANDHI                                        **
**    DATE:     10/26/88                                               **
**                                                                     **
**    PARAMETERS USED:                                                 **
**                                                                     **
**    PMENU   = MENU LIST THAT IS CURRENTLY DISPLAYED                  **
**              (CHARACTER(50)*15,I/P)                                 **
**    ICLASS  = CLASS OF INPUT DEVICE (INTEGER,O/P)                    **
**    IDEV    = DEVICE NUMBER, IF VALUATOR INPUT (INTEGER,O/P)         **
**    IPKID   = PICK-ID OF ITEM, IF PICKED FROM MENU (INTEGER,O/P)     **
**    V       = ARRAY OF VALUES FROM VALUATOR (REAL(8),O/P)            **
**    STRG    = STRING VALUE, IF STRING INPUT (CHARACTER*50,O/P)       **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FINPT(PMENU,ICLASS,IDEV,IPKID,V,STRG)

      INTEGER*4 CLASS, MODE, ECHO, IPKID, PPATH(3)

      REAL*4 PAREA(6),CSIZE(3),ASIZE(6), DATA(12)
      REAL*4 SAREA(6),VAREA(6),V(8)

      CHARACTER STRG*50, EMTY*50, PREF*9, MSSG*59, PMENU(50)*15
```

```
      CHARACTER MSG*50, WORD(15)*12
      DATA EMTY/'                                                    '/
      STRG = '                                                      '
      CLASS = 1
*SET PICK FILTER TO MAKE CLASS DETECTABLE
      CALL GPPKF(1,1,1,CLASS,1,2)
*AWAIT AN EVENT
200   CALL GPAWEV(1.,1,ICLASS,IDEV)
*********************************************************************
*        ONCE INPUT IS GIVEN GO TO REQUIRED GET-INPUT CALLS         *
*********************************************************************
*IF INPUT CLASS IS "LOCATOR"
      IF (ICLASS .EQ. 1) THEN
         MSSG = 'LOCATOR NOT IMPLEMENTED'
         CALL FMSSG(30,MSSG)
      ENDIF
*********************************************************************
*        IF INPUT CLASS IS "STROKE"                                 *
*********************************************************************
      IF (ICLASS .EQ. 2) THEN
         MSSG = 'STROKE NOT IMPLEMENTED'
         CALL FMSSG(30,MSSG)
      ENDIF
*********************************************************************
*        IF INPUT CLASS IS "VALUATOR"                               *
*********************************************************************
      IF (ICLASS .EQ. 3) THEN
*GET THE VALUE FROM THE VALUATOR
         CALL GPGTVL(VLU)
*UPDATE ARRAY OF VALUES FOR VALUATOR
         V(IDEV) = VLU
      ENDIF
*********************************************************************
*        IF INPUT CLASS IS "CHOICE"                                 *
*********************************************************************
      IF (ICLASS .EQ. 4) THEN
         MSSG = 'CHOICE NOT IMPLEMENTED'
         CALL FMSSG(30,MSSG)
      ENDIF
*********************************************************************
*        IF INPUT CLASS IS "PICK"                                   *
*********************************************************************
      IF (ICLASS .EQ. 5) THEN
*UPDATE WORKSTATION (TRAVERSE ALL VIEWS)
        CALL GPUPWS(1,2)
*GET PICK
        CALL GPGTPK(1,1,PPATH)
*IPKID IS SECOND OF PPATH
        IPKID = PPATH(2)
        IF (IPKID .EQ. 1)GOTO 200
      ENDIF
*********************************************************************
*        IF INPUT CLASS IS "STRING"                                 *
*********************************************************************
      IF (ICLASS .EQ. 6) THEN
         WRITE(6,*)'STRING INPUT'
         L=50
```

```
*GET THE STRING INPUT
         CALL GPGTST(L,LR,STRG)

         IF (LR .EQ. 0)GOTO 200
         CALL FPARS(STRG,NW,WORD)
         CALL FCAPS(WORD)

*ECHO THE STRING INPUT

         IF (WORD(1)(1:5) .EQ. 'HCOPY')THEN
            CALL HRDCPY(1,2)
            CALL GPUPWS(2,2)
            CALL HRDCPY(1,2)
            CALL GPUPWS(2,2)
            STRG = 'HARD COPY GENERATED'
            CALL FMSSG(30,STRG)
         ENDIF

      ENDIF

       RETURN
       END
```

SUBROUTINE FINQU

```
************************************************************************
************************************************************************
**     SUBROUTINE FINQU(FNAME,TYPE)                                   **
**                                                                    **
**     PROGRAM DESCRIPTION                                            **
**     THIS SUBROUTINE ALLOWS A USER TO INQUIRE INFORMATION ON        **
**     DIFFERENT FEATURES. DIFFERENT LEVELS AND TYPES OF INQUIRIES     **
**     ARE POSSIBLE.                                                  **
**                                                                    **
**                                                                    **
**     BY:       ASHIT R. GANDHI                                      **
**     DATE:     01/30/89                                             **
**                                                                    **
**     PARAMETERS USED:                                               **
**                                                                    **
**     FNAME : FEATURE ON WHOM INFORMATION IS REQUESTED. (I/P,CHARACTER**
**     TYPE  : TYPE AND LEVEL OF INQUIRY REQUESTED. (I/P, CHARACTER*3) **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FINQU(FNAME,TYPE)

      CHARACTER FNAME*8, LCHAR*73, HEAD(6)*12, TYPE*3

      DATA HEAD /'FEATURE NAME','ENGLISH DEFI','GROUP CLASSI',
     1           'SYNONYMS    ','MATHEMATICAL','DEFAULT MODE'/

      DATA NHEAD/6/
      DATA NIN,NOUT/8,10/

      IFLAG = 0
      DO 100 I = 1,20000
         READ(NIN,10,END=250)LCHAR
         IF (LCHAR(1:12) .EQ. HEAD(1))THEN
            IF (LCHAR(16:23) .EQ. FNAME)THEN
               IFLAG = 1
               GOTO 200
            ENDIF
         ENDIF
100   CONTINUE
200   WRITE(NOUT,*)LCHAR
250   IF (IFLAG .NE. 1)THEN
         WRITE(NOUT,*)'FEATURE DOES NOT EXIST IN LIBRARY'
         GOTO 500
      ENDIF

      IF (TYPE .NE. 'ALL')THEN
         DO 300 I = 1, 1000
            READ(NIN,10)LCHAR
            IF (TYPE .EQ. LCHAR(1:3))THEN
               IFLAG = 2
               WRITE(NOUT,*)LCHAR
            ENDIF
            IF (LCHAR(1:3) .EQ. 'END')GOTO 450
300      CONTINUE
      ELSE
         DO 400 I = 1, 1000
            READ(NIN,10)LCHAR
            WRITE(NOUT,*)LCHAR
            IFLAG = 2
```

```
          IF (LCHAR(1:3) .EQ. 'END')GOTO 450
400       CONTINUE
          ENDIF

450       IF (IFLAG .NE. 2)THEN
             WRITE(NOUT,*)'REQUESTED INFORMATION DOES NOT EXIST IN LIBRARY'
          ENDIF

500       REWIND(NIN)

          RETURN
10        FORMAT(A73)
          END
```

SUBROUTINE FLFTV

```
**********************************************************************
**********************************************************************
**     SUBROUTINE FLFTV(NVIEW)                                      **
**                                                                  **
**     PROGRAM DESCRIPTION                                          **
**                                                                  **
**     THIS SUBROUTINE WILL DISPLAY THE LEFT SIDE VIEW OF ANY OBJECT.  **
**                                                                  **
**                                                                  **
**                                                                  **
**     BY:       ASHIT R. GANDHI                                    **
**     DATE:     11/07/88                                           **
**                                                                  **
**     PARAMETERS USED:                                             **
**                                                                  **
**     NST    = STRUCTURE ASSOCIATED WITH THE VIEW (INTEGER,I/P)     **
**     NVIEW  = VIEW IN WHICH THE TOP VIEW IS TO BE DISPLAYED        **
**              (INTEGER,I/P)                                       **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FLFTV(NVIEW)

      REAL*4 MAT(4,4), MAT1(4,4), MAT2(4,4), SCLEF(3)

      COMMON/SCLE/SCALEF

      PI = 3.14159
      DL = PI/2.0

*ROTATE THE VIEW ELEMENT ABOUT THE Y-AXIS BY +90 DEGREES

      CALL GPROTY(DL,MAT2)

*SCALE THE MODEL

      CALL GPSC3(SCALEF,MAT1)

*COMPOSE THE TRANSFORMATION

      CALL GPCMT3(MAT2,MAT1,MAT)

*ASSOCIATE THIS ROTATION TO THE VIEW

      CALL GPVMT3(1,NVIEW,MAT)

      CALL GPVP(1,NVIEW,2,1)

      RETURN
      END
```

SUBROUTINE FMENU2

```
**********************************************************************
**********************************************************************
**     SUBROUTINE FMENU2(NUM,PMENU)                                 **
**                                                                  **
**     PROGRAM DESCRIPTION                                          **
**                                                                  **
**     THIS ROUTINE DISPLAYS THE REQUIRED MENU ON THE SCREEN.       **
**     IT RETURNS A STRING OF MENU ITEMS THAT ARE CURRENTLY         **
**     BEING DISPLAYED.                                             **
**                                                                  **
**     BY:       ASHIT R. GANDHI                                    **
**     DATE:     10/12/88                                           **
**                                                                  **
**     PARAMETERS USED:                                             **
**
```

```
**     NUM    = PAGE NUMBER TO BE DISPLAYED                                  **
**     PMENU  = STRING ARRAY OF CURRENT MENU ITEMS                           **
**                                                                           **
*******************************************************************************
*******************************************************************************
      SUBROUTINE FMENU2(NUM,PMENU)

      COMMON CSIZE

      REAL*4 POS(2), CSIZE(6),PAREA(6)
      INTEGER*4 CHOICE,PPATH(3)
      CHARACTER TMENU(50,50)*15, MSG*15, PMENU(50)*15

*                              TITLE TEST,LENGTH
      DATA LNT/15/

      DATA TMENU( 1, 1)/'     FILES       '/
      DATA TMENU( 1,25)/'     -----       '/
      DATA TMENU( 1, 2)/'                 '/
      DATA TMENU( 1,26)/'                 '/
      DATA TMENU( 1, 3)/'COMMANDS         '/
      DATA TMENU( 1,27)/'                 '/
      DATA TMENU( 1, 4)/'                 '/
      DATA TMENU( 1,28)/'                 '/
      DATA TMENU( 1, 5)/'START            '/
      DATA TMENU( 1,29)/'                 '/
      DATA TMENU( 1, 6)/'RECALL           '/
      DATA TMENU( 1,30)/'                 '/
      DATA TMENU( 1, 7)/'                 '/
      DATA TMENU( 1,31)/'                 '/
      DATA TMENU( 1, 8)/'FILE             '/
      DATA TMENU( 1,32)/'                 '/
      DATA TMENU( 1, 9)/'PARAMETERS       '/
      DATA TMENU( 1,33)/'                 '/
      DATA TMENU( 1,10)/'                 '/
      DATA TMENU( 1,34)/'                 '/
      DATA TMENU( 1,11)/'COMMAND FILE     '/
      DATA TMENU( 1,35)/'                 '/
      DATA TMENU( 1,12)/'                 '/
      DATA TMENU( 1,36)/'                 '/
      DATA TMENU( 1,13)/'                 '/
      DATA TMENU( 1,37)/'                 '/
      DATA TMENU( 1,14)/'                 '/
      DATA TMENU( 1,38)/'                 '/
      DATA TMENU( 1,15)/'                 '/
      DATA TMENU( 1,39)/'                 '/
      DATA TMENU( 1,16)/'                 '/
      DATA TMENU( 1,40)/'                 '/
      DATA TMENU( 1,17)/'                 '/
      DATA TMENU( 1,41)/'                 '/
      DATA TMENU( 1,18)/'                 '/
      DATA TMENU( 1,42)/'                 '/
      DATA TMENU( 1,19)/'                 '/
      DATA TMENU( 1,43)/'                 '/
      DATA TMENU( 1,20)/'                 '/
      DATA TMENU( 1,44)/'                 '/
      DATA TMENU( 1,21)/'                 '/
      DATA TMENU( 1,45)/'                 '/
      DATA TMENU( 1,22)/'                 '/
      DATA TMENU( 1,46)/'                 '/
      DATA TMENU( 1,23)/'                 '/
      DATA TMENU( 1,47)/'                 '/
      DATA TMENU( 1,24)/'                 '/
      DATA TMENU( 1,48)/'                 '/

      DATA TMENU( 2, 1)/'   COMMANDS      '/
      DATA TMENU( 2,25)/'   --------      '/
      DATA TMENU( 2, 2)/'                 '/
      DATA TMENU( 2,26)/'                 '/
      DATA TMENU( 2, 3)/'EXIT             '/
      DATA TMENU( 2,27)/'                 '/
      DATA TMENU( 2, 4)/'                 '/
      DATA TMENU( 2,28)/'                 '/
      DATA TMENU( 2, 5)/'MODELING         '/
      DATA TMENU( 2,29)/'                 '/
      DATA TMENU( 2, 6)/'LANGUAGE I/P     '/
      DATA TMENU( 2,30)/'                 '/
      DATA TMENU( 2, 7)/'                 '/
      DATA TMENU( 2,31)/'                 '/
      DATA TMENU( 2, 8)/'RENDERING        '/
      DATA TMENU( 2,32)/'                 '/
      DATA TMENU( 2, 9)/'VIEWS            '/
      DATA TMENU( 2,33)/'                 '/
      DATA TMENU( 2,10)/'WINDOWS          '/
      DATA TMENU( 2,34)/'                 '/
      DATA TMENU( 2,11)/'                 '/
      DATA TMENU( 2,35)/'                 '/
      DATA TMENU( 2,12)/'DISPLAY          '/
      DATA TMENU( 2,36)/'                 '/
      DATA TMENU( 2,13)/'INQUIRE          '/
      DATA TMENU( 2,37)/'                 '/
      DATA TMENU( 2,14)/'H.COPY           '/
      DATA TMENU( 2,38)/'                 '/
      DATA TMENU( 2,15)/'                 '/
      DATA TMENU( 2,39)/'                 '/
      DATA TMENU( 2,16)/'FILES            '/
```

```
DATA TMENU(2,40)/'                '/
DATA TMENU(2,17)/'                '/
DATA TMENU(2,41)/'                '/
DATA TMENU(2,18)/'                '/
DATA TMENU(2,42)/'                '/
DATA TMENU(2,19)/'                '/
DATA TMENU(2,43)/'                '/
DATA TMENU(2,20)/'                '/
DATA TMENU(2,44)/'                '/
DATA TMENU(2,21)/'                '/
DATA TMENU(2,45)/'                '/
DATA TMENU(2,22)/'                '/
DATA TMENU(2,46)/'                '/
DATA TMENU(2,23)/'                '/
DATA TMENU(2,47)/'                '/
DATA TMENU(2,24)/'                '/
DATA TMENU(2,48)/'                '/

DATA TMENU(3, 1)/'    MODELING    '/
DATA TMENU(3,25)/'    --------    '/
DATA TMENU(3, 2)/'                '/
DATA TMENU(3,26)/'                '/
DATA TMENU(3, 3)/'RETURN          '/
DATA TMENU(3,27)/'                '/
DATA TMENU(3, 4)/'                '/
DATA TMENU(3,28)/'                '/
DATA TMENU(3, 5)/'DEFINE          '/
DATA TMENU(3,29)/'                '/
DATA TMENU(3, 6)/'ADDTO           '/
DATA TMENU(3,30)/'                '/
DATA TMENU(3, 7)/'EDIT            '/
DATA TMENU(3,31)/'                '/
DATA TMENU(3, 8)/'                '/
DATA TMENU(3,32)/'                '/
DATA TMENU(3, 9)/'COPY            '/
DATA TMENU(3,33)/'                '/
DATA TMENU(3,10)/'DELETE          '/
DATA TMENU(3,34)/'                '/
DATA TMENU(3,11)/'ASSEMBLE        '/
DATA TMENU(3,35)/'                '/
DATA TMENU(3,12)/'                '/
DATA TMENU(3,36)/'                '/
DATA TMENU(3,13)/'                '/
DATA TMENU(3,37)/'                '/
DATA TMENU(3,14)/'                '/
DATA TMENU(3,38)/'                '/
DATA TMENU(3,15)/'                '/
DATA TMENU(3,39)/'                '/
DATA TMENU(3,16)/'                '/
DATA TMENU(3,40)/'                '/
DATA TMENU(3,17)/'                '/
DATA TMENU(3,41)/'                '/
DATA TMENU(3,18)/'                '/
DATA TMENU(3,42)/'                '/
DATA TMENU(3,19)/'                '/
DATA TMENU(3,43)/'                '/
DATA TMENU(3,20)/'                '/
DATA TMENU(3,44)/'                '/
DATA TMENU(3,21)/'                '/
DATA TMENU(3,45)/'                '/
DATA TMENU(3,22)/'                '/
DATA TMENU(3,46)/'                '/
DATA TMENU(3,23)/'                '/
DATA TMENU(3,47)/'                '/
DATA TMENU(3,24)/'                '/
DATA TMENU(3,48)/'                '/

DATA TMENU(4, 1)/'  RENDERING     '/
DATA TMENU(4,25)/'  ---------     '/
DATA TMENU(4, 2)/'                '/
DATA TMENU(4,26)/'                '/
DATA TMENU(4, 3)/'RETURN          '/
DATA TMENU(4,27)/'                '/
DATA TMENU(4, 4)/'                '/
DATA TMENU(4,28)/'                '/
DATA TMENU(4, 5)/'COLOR           '/
DATA TMENU(4,29)/'                '/
DATA TMENU(4, 6)/'                '/
DATA TMENU(4,30)/'                '/
DATA TMENU(4, 7)/'HIDDEN LINES    '/
DATA TMENU(4,31)/'                '/
DATA TMENU(4, 8)/'HIDDEN SURFACE  '/
DATA TMENU(4,32)/'                '/
DATA TMENU(4, 9)/'                '/
DATA TMENU(4,33)/'                '/
DATA TMENU(4,10)/'SHADING         '/
DATA TMENU(4,34)/'                '/
DATA TMENU(4,11)/'                '/
DATA TMENU(4,35)/'                '/
DATA TMENU(4,12)/'RESET           '/
DATA TMENU(4,36)/'                '/
DATA TMENU(4,13)/'                '/
DATA TMENU(4,37)/'                '/
DATA TMENU(4,14)/'                '/
DATA TMENU(4,38)/'                '/
DATA TMENU(4,15)/'                '/
```

```
DATA TMENU(4,39)/'                '/
DATA TMENU(4,16)/'                '/
DATA TMENU(4,40)/'                '/
DATA TMENU(4,17)/'                '/
DATA TMENU(4,41)/'                '/
DATA TMENU(4,18)/'                '/
DATA TMENU(4,42)/'                '/
DATA TMENU(4,19)/'                '/
DATA TMENU(4,43)/'                '/
DATA TMENU(4,20)/'                '/
DATA TMENU(4,44)/'                '/
DATA TMENU(4,21)/'                '/
DATA TMENU(4,45)/'                '/
DATA TMENU(4,22)/'                '/
DATA TMENU(4,46)/'                '/
DATA TMENU(4,23)/'                '/
DATA TMENU(4,47)/'                '/
DATA TMENU(4,24)/'                '/
DATA TMENU(4,48)/'                '/

DATA TMENU(5, 1)/'     VIEWS      '/
DATA TMENU(5,25)/'     -----      '/
DATA TMENU(5, 2)/'                '/
DATA TMENU(5,26)/'                '/
DATA TMENU(5, 3)/'RETURN          '/
DATA TMENU(5,27)/'                '/
DATA TMENU(5, 4)/'                '/
DATA TMENU(5,28)/'                '/
DATA TMENU(5, 5)/'MOVE            '/
DATA TMENU(5,29)/'                '/
DATA TMENU(5, 6)/'ROTATE          '/
DATA TMENU(5,30)/'                '/
DATA TMENU(5, 7)/'ZOOM            '/
DATA TMENU(5,31)/'                '/
DATA TMENU(5, 8)/'                '/
DATA TMENU(5,32)/'                '/
DATA TMENU(5, 9)/'RESET           '/
DATA TMENU(5,33)/'                '/
DATA TMENU(5,10)/'                '/
DATA TMENU(5,34)/'                '/
DATA TMENU(5,11)/'                '/
DATA TMENU(5,35)/'                '/
DATA TMENU(5,12)/'                '/
DATA TMENU(5,36)/'                '/
DATA TMENU(5,13)/'                '/
DATA TMENU(5,37)/'                '/
DATA TMENU(5,14)/'                '/
DATA TMENU(5,38)/'                '/
DATA TMENU(5,15)/'                '/
DATA TMENU(5,39)/'                '/
DATA TMENU(5,16)/'                '/
DATA TMENU(5,40)/'                '/
DATA TMENU(5,17)/'                '/
DATA TMENU(5,41)/'                '/
DATA TMENU(5,18)/'                '/
DATA TMENU(5,42)/'                '/
DATA TMENU(5,19)/'                '/
DATA TMENU(5,43)/'                '/
DATA TMENU(5,20)/'                '/
DATA TMENU(5,44)/'                '/
DATA TMENU(5,21)/'                '/
DATA TMENU(5,45)/'                '/
DATA TMENU(5,22)/'                '/
DATA TMENU(5,46)/'                '/
DATA TMENU(5,23)/'                '/
DATA TMENU(5,47)/'                '/
DATA TMENU(5,24)/'                '/
DATA TMENU(5,48)/'                '/

DATA TMENU(6, 1)/'    WINDOWS     '/
DATA TMENU(6,25)/'    -------     '/
DATA TMENU(6, 2)/'                '/
DATA TMENU(6,26)/'                '/
DATA TMENU(6, 3)/'RETURN          '/
DATA TMENU(6,27)/'                '/
DATA TMENU(6, 4)/'                '/
DATA TMENU(6,28)/'                '/
DATA TMENU(6, 5)/'SINGLE          '/
DATA TMENU(6,29)/'                '/
DATA TMENU(6, 6)/'MULTIPLE        '/
DATA TMENU(6,30)/'                '/
DATA TMENU(6, 7)/'                '/
DATA TMENU(6,31)/'                '/
DATA TMENU(6, 8)/'RESET           '/
DATA TMENU(6,32)/'                '/
DATA TMENU(6, 9)/'                '/
DATA TMENU(6,33)/'                '/
DATA TMENU(6,10)/'                '/
DATA TMENU(6,34)/'                '/
DATA TMENU(6,11)/'                '/
DATA TMENU(6,35)/'                '/
DATA TMENU(6,12)/'                '/
DATA TMENU(6,36)/'                '/
DATA TMENU(6,13)/'                '/
DATA TMENU(6,37)/'                '/
DATA TMENU(6,14)/'                '/
```

```
DATA TMENU(6,38)/'                '/
DATA TMENU(6,15)/'                '/
DATA TMENU(6,39)/'                '/
DATA TMENU(6,16)/'                '/
DATA TMENU(6,40)/'                '/
DATA TMENU(6,17)/'                '/
DATA TMENU(6,41)/'                '/
DATA TMENU(6,18)/'                '/
DATA TMENU(6,42)/'                '/
DATA TMENU(6,19)/'                '/
DATA TMENU(6,43)/'                '/
DATA TMENU(6,20)/'                '/
DATA TMENU(6,44)/'                '/
DATA TMENU(6,21)/'                '/
DATA TMENU(6,45)/'                '/
DATA TMENU(6,22)/'                '/
DATA TMENU(6,46)/'                '/
DATA TMENU(6,23)/'                '/
DATA TMENU(6,47)/'                '/
DATA TMENU(6,24)/'                '/
DATA TMENU(6,48)/'                '/

DATA TMENU(7, 1)/'    SHADING     '/
DATA TMENU(7,25)/'    -------     '/
DATA TMENU(7, 2)/'                '/
DATA TMENU(7,26)/'                '/
DATA TMENU(7, 3)/'RETURN          '/
DATA TMENU(7,27)/'                '/
DATA TMENU(7, 4)/'                '/
DATA TMENU(7,28)/'                '/
DATA TMENU(7, 5)/'CONSTANT        '/
DATA TMENU(7,29)/'                '/
DATA TMENU(7, 6)/'                '/
DATA TMENU(7,30)/'                '/
DATA TMENU(7, 7)/'                '/
DATA TMENU(7,31)/'                '/
DATA TMENU(7, 8)/'                '/
DATA TMENU(7,32)/'                '/
DATA TMENU(7, 9)/'RESET           '/
DATA TMENU(7,33)/'                '/
DATA TMENU(7,10)/'                '/
DATA TMENU(7,34)/'                '/
DATA TMENU(7,11)/'                '/
DATA TMENU(7,35)/'                '/
DATA TMENU(7,12)/'                '/
DATA TMENU(7,36)/'                '/
DATA TMENU(7,13)/'                '/
DATA TMENU(7,37)/'                '/
DATA TMENU(7,14)/'                '/
DATA TMENU(7,38)/'                '/
DATA TMENU(7,15)/'                '/
DATA TMENU(7,39)/'                '/
DATA TMENU(7,16)/'                '/
DATA TMENU(7,40)/'                '/
DATA TMENU(7,17)/'                '/
DATA TMENU(7,41)/'                '/
DATA TMENU(7,18)/'                '/
DATA TMENU(7,42)/'                '/
DATA TMENU(7,19)/'                '/
DATA TMENU(7,43)/'                '/
DATA TMENU(7,20)/'                '/
DATA TMENU(7,44)/'                '/
DATA TMENU(7,21)/'                '/
DATA TMENU(7,45)/'                '/
DATA TMENU(7,22)/'                '/
DATA TMENU(7,46)/'                '/
DATA TMENU(7,23)/'                '/
DATA TMENU(7,47)/'                '/
DATA TMENU(7,24)/'                '/
DATA TMENU(7,48)/'                '/

DATA TMENU(8, 1)/'     SINGLE     '/
DATA TMENU(8,25)/'     ------     '/
DATA TMENU(8, 2)/'                '/
DATA TMENU(8,26)/'                '/
DATA TMENU(8, 3)/'RETURN          '/
DATA TMENU(8,27)/'                '/
DATA TMENU(8, 4)/'                '/
DATA TMENU(8,28)/'                '/
DATA TMENU(8, 5)/'FRONT           '/
DATA TMENU(8,29)/'                '/
DATA TMENU(8, 6)/'TOP             '/
DATA TMENU(8,30)/'                '/
DATA TMENU(8, 7)/'RIGHT SIDE      '/
DATA TMENU(8,31)/'                '/
DATA TMENU(8, 8)/'ISOMETRIC       '/
DATA TMENU(8,32)/'                '/
DATA TMENU(8, 9)/'BACK            '/
DATA TMENU(8,33)/'                '/
DATA TMENU(8,10)/'BOTTOM          '/
DATA TMENU(8,34)/'                '/
DATA TMENU(8,11)/'LEFT SIDE       '/
DATA TMENU(8,35)/'                '/
DATA TMENU(8,12)/'                '/
DATA TMENU(8,36)/'                '/
DATA TMENU(8,13)/'                '/
```

```
DATA TMENU(8,37)/'                '/
DATA TMENU(8,14)/'                '/
DATA TMENU(8,38)/'                '/
DATA TMENU(8,15)/'                '/
DATA TMENU(8,39)/'                '/
DATA TMENU(8,16)/'                '/
DATA TMENU(8,40)/'                '/
DATA TMENU(8,17)/'                '/
DATA TMENU(8,41)/'                '/
DATA TMENU(8,18)/'                '/
DATA TMENU(8,42)/'                '/
DATA TMENU(8,19)/'                '/
DATA TMENU(8,43)/'                '/
DATA TMENU(8,20)/'                '/
DATA TMENU(8,44)/'                '/
DATA TMENU(8,21)/'                '/
DATA TMENU(8,45)/'                '/
DATA TMENU(8,22)/'                '/
DATA TMENU(8,46)/'                '/
DATA TMENU(8,23)/'                '/
DATA TMENU(8,47)/'                '/
DATA TMENU(8,24)/'                '/
DATA TMENU(8,48)/'                '/

DATA TMENU(9, 1)/'     COLOR      '/
DATA TMENU(9,25)/'     -----      '/
DATA TMENU(9, 2)/'                '/
DATA TMENU(9,26)/'                '/
DATA TMENU(9, 3)/'RETURN          '/
DATA TMENU(9,27)/'                '/
DATA TMENU(9, 4)/'                '/
DATA TMENU(9,28)/'                '/
DATA TMENU(9, 5)/'MAXIMUM RED     '/
DATA TMENU(9,29)/'                '/
DATA TMENU(9, 6)/'MAXIMUM GREEN   '/
DATA TMENU(9,30)/'                '/
DATA TMENU(9, 7)/'MAXIMUM BLUE    '/
DATA TMENU(9,31)/'                '/
DATA TMENU(9, 8)/'MAXIMUM YELLOW  '/
DATA TMENU(9,32)/'                '/
DATA TMENU(9, 9)/'MAGENTA         '/
DATA TMENU(9,33)/'                '/
DATA TMENU(9,10)/'TURQUOISE       '/
DATA TMENU(9,34)/'                '/
DATA TMENU(9,11)/'DARK MAGENTA    '/
DATA TMENU(9,35)/'                '/
DATA TMENU(9,12)/'DARK GREEN      '/
DATA TMENU(9,36)/'                '/
DATA TMENU(9,13)/'DARK BLUE       '/
DATA TMENU(9,37)/'                '/
DATA TMENU(9,14)/'BACKGROUND GREY '/
DATA TMENU(9,38)/'                '/
DATA TMENU(9,15)/'LIGHT GREY      '/
DATA TMENU(9,39)/'                '/
DATA TMENU(9,16)/'PEACOCK GREEN   '/
DATA TMENU(9,40)/'                '/
DATA TMENU(9,17)/'ORANGE          '/
DATA TMENU(9,41)/'                '/
DATA TMENU(9,18)/'BLACK           '/
DATA TMENU(9,42)/'                '/
DATA TMENU(9,19)/'                '/
DATA TMENU(9,43)/'                '/
DATA TMENU(9,20)/'RESET           '/
DATA TMENU(9,44)/'                '/
DATA TMENU(9,21)/'                '/
DATA TMENU(9,45)/'                '/
DATA TMENU(9,22)/'                '/
DATA TMENU(9,46)/'                '/
DATA TMENU(9,23)/'                '/
DATA TMENU(9,47)/'                '/
DATA TMENU(9,24)/'                '/
DATA TMENU(9,48)/'                '/

DATA TMENU(10, 1)/'     DEFINE     '/
DATA TMENU(10,25)/'     ------     '/
DATA TMENU(10, 2)/'                '/
DATA TMENU(10,26)/'                '/
DATA TMENU(10, 3)/'RETURN          '/
DATA TMENU(10,27)/'                '/
DATA TMENU(10, 4)/'                '/
DATA TMENU(10,28)/'                '/
DATA TMENU(10, 5)/'SLAB            '/
DATA TMENU(10,29)/'                '/
DATA TMENU(10, 6)/'SPHERE          '/
DATA TMENU(10,30)/'                '/
DATA TMENU(10, 7)/'CYLINDER        '/
DATA TMENU(10,31)/'                '/
DATA TMENU(10, 8)/'RING            '/
DATA TMENU(10,32)/'                '/
DATA TMENU(10, 9)/'CONE            '/
DATA TMENU(10,33)/'                '/
DATA TMENU(10,10)/'PYRAMID         '/
DATA TMENU(10,34)/'                '/
DATA TMENU(10,11)/'WEDGE           '/
DATA TMENU(10,35)/'                '/
DATA TMENU(10,12)/'ELLIPSOID       '/
```

```
      DATA TMENU(10,36)/'                '/
      DATA TMENU(10,13)/'HEMISPHERE      '/
      DATA TMENU(10,37)/'                '/
      DATA TMENU(10,14)/'BUSHING         '/
      DATA TMENU(10,38)/'                '/
      DATA TMENU(10,15)/'COUNTER         '/
      DATA TMENU(10,39)/'                '/
      DATA TMENU(10,16)/'TUNNEL          '/
      DATA TMENU(10,40)/'                '/
      DATA TMENU(10,17)/'ANGLE           '/
      DATA TMENU(10,41)/'                '/
      DATA TMENU(10,18)/'                '/
      DATA TMENU(10,42)/'                '/
      DATA TMENU(10,19)/'                '/
      DATA TMENU(10,43)/'                '/
      DATA TMENU(10,20)/'                '/
      DATA TMENU(10,44)/'                '/
      DATA TMENU(10,21)/'                '/
      DATA TMENU(10,45)/'                '/
      DATA TMENU(10,22)/'                '/
      DATA TMENU(10,46)/'                '/
      DATA TMENU(10,23)/'                '/
      DATA TMENU(10,47)/'                '/
      DATA TMENU(10,24)/'                '/
      DATA TMENU(10,48)/'                '/

      POS(1)=0.05
      POS(2)=0.95

*EMPTY STRUCTURE BEFORE DRAWING NEW MENU

      CALL GPDLST(6)

      CALL GPOPST(6)

*INSERT CLASS NAME TO ALLOW PICKING MENU ITEM

      CALL GPADCN(1,1)

*SET TEXT COLOR TO TURQUOISE

      CALL GPTXCI(7)

*SET ANNOTATION SIZE SCALE FACTOR

      CALL GPAHSC(0.5)

*SET PICK IDS AND DRAW THE TEXT

      DO 100 I=1,24
         CALL GPPKID(I)
         CALL GPAN2(POS,LNT,TMENU(NUM,I))
         POS(2)=POS(2)-0.015
         CALL GPAN2(POS,LNT,TMENU(NUM,I+24))
         POS(2)=POS(2)-0.02
 100  CONTINUE

*CLOSE STRUCTURE

      CALL GPCLST

*ASSOCIATE THE STRUCTURE 6 WITH A VIEW 4

      CALL GPARV(1,4,6,0)
      CALL GPVP(1,4,2,1)

      DO 200 I = 1,50
200   PMENU(I) = TMENU(NUM,I)

      RETURN
      END
```

SUBROUTINE FMPK1

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FMPK1(MNUM,IPKID,PMENU)                            **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 1, THEN THE CORRECT    **
**    ACTION SHOULD BE TAKEN.                                       **
**                                                                  **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     11/04/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    MNUM  = MENU NUMBER (INTEGER, I/P)                            **
```

```
**          = 1                                                          **
**     IPKID = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)          **
**     PMENU = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                     **
**             (CHARACTER,I/P,O/P)                                        **
**                                                                        **
************************************************************************
************************************************************************
      SUBROUTINE FMPK1(MNUM,IPKID,PMENU)

      INTEGER*4 MNUM, IPKID

      REAL*4 PMENU(50)

      LOGICAL*4 FLCHK

      CHARACTER STRG*50, PRE*30, FTRG*4, ANS*1, LCHAR*72
      CHARACTER*8 FILE10, FILE21, FILE22, FILE31, FILE32

      COMMON/POLY/IPGNR
      COMMON/MODEL/FILE22

      DATA FLFLAG/0/

*GOTO THE CORRECT ID THAT WAS PICKED

1     GOTO(11,12,13,14,15,16,17,18,19,20,21)IPKID

*IF IPKID .GT. 11 RETURN

      RETURN

*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY

11    RETURN

*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY

12    RETURN

*MNUM = 1, IPKID = 3 ITEM PICKED = COMMANDS
*IN THIS CASE JUST DISPLAY MENU NUMBER 2

13    MNUM = 2
      CALL FMENU2(MNUM,PMENU)
      RETURN

*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY

14    RETURN

*MNUM = 1, IPKID = 5 ITEM PICKED = START
*IN THIS CASE ASK FOR STRING INPUT

*CHECK IF A MODEL FILE ALREADY OPEN (FLFLAG = 0, NO FILE OPEN)
*                                  (FLFLAG = 1,    FILE OPEN)

15    IF (FLFLAG .EQ. 1) THEN
         STRG = 'MODEL FILE IS CURRENTLY OPEN'
         CALL FMSSG(30,STRG)
         STRG = 'IF YOU OPEN A NEW FILE, PREVIOUS WORK'
         CALL FMSSG(30,STRG)
         STRG = 'DONE IN THIS SESSION WILL BE LOST'
         CALL FMSSG(30,STRG)
         CALL FPRMP(7)
         CALL FGTST(ANS)
         IF (ANS .NE. 'Y')RETURN
         CLOSE(10)
         CALL GPDLST(20)
         CALL GPDLST(21)
      ENDIF

*IF FILE IS TO BE OPENED GET THE FILE NAME

      CALL FPRMP(3)
151   CALL FGTST(FTRG)

      DO 152 I = 1,4
         IF (FTRG(I:I) .EQ. ' ')FTRG(I:I) = '0'
152   CONTINUE

*CHECK IF THIS FILE ALREADY EXISTS
      FILE10 = FTRG//'0010'
      INQUIRE(FILE=FILE10,EXIST=FLCHK)
      IF (FLCHK)THEN
         STRG = 'MODEL FILE ALREADY EXISTS'
         CALL FMSSG(30,STRG)
         STRG = 'CANNOT OPEN EXISTING FILE'
         CALL FMSSG(30,STRG)
         STRG = 'ENTER NEW NAME'
         CALL FMSSG(30,STRG)
         GOTO 151
      ENDIF

*IF A NEW FILE HAS TO BE CREATED
```

```
      FILE21 = FTRG//'0021'
      FILE22 = FTRG//'0022'
      FILE31 = FTRG//'0031'
      FILE32 = FTRG//'0032'
      OPEN(UNIT=10,FILE=FILE10,STATUS='NEW')
      OPEN(UNIT=21,FILE=FILE21,STATUS='NEW')
      OPEN(UNIT=22,FILE=FILE22,STATUS='NEW')
      OPEN(UNIT=31,FILE=FILE31,STATUS='NEW')
      OPEN(UNIT=32,FILE=FILE32,STATUS='NEW')

*SET FILE FLAG ON

      FLFLAG = 1
      IPGNR = 0

      STRG = 'STARTED MODEL '//FTRG
      CALL FMSSG(30,STRG)

      RETURN

*MNUM = 1, IPKID = 6 ITEM PICKED = RECALL
*IN THIS CASE ASK FOR STRING INPUT

*CHECK IF A MODEL FILE ALREADY OPEN (FLFLAG = 0, NO FILE OPEN)
*                                   (FLFLAG = 1,    FILE OPEN)

16    IF (FLFLAG .EQ. 1) THEN
         STRG = 'MODEL FILE IS CURRENTLY OPEN'
         CALL FMSSG(30,STRG)
         STRG = 'IF YOU RECALL A NEW FILE, PREVIOUS WORK'
         CALL FMSSG(30,STRG)
         STRG = 'DONE IN THIS SESSION WILL BE LOST'
         CALL FMSSG(30,STRG)
         CALL FPRMP(8)
         CALL FGTST(ANS)
         IF (ANS .NE. 'Y')RETURN
         CLOSE(10)
         CALL GPDLST(20)
         CALL GPDLST(21)
      ENDIF

      CALL FPRMP(4)
161   CALL FGTST(FTRG)

      DO 162 I = 1,4
         IF (FTRG(I:I) .EQ. ' ')FTRG(I:I) = '0'
162   CONTINUE

*CHECK IF THIS FILE EXISTS

      FILE10 = FTRG//'0010'
      FILE21 = FTRG//'0021'
      FILE22 = FTRG//'0022'
      FILE31 = FTRG//'0031'
      FILE32 = FTRG//'0032'

      I10 = 0
      INQUIRE(FILE=FILE10,EXIST=FLCHK)
      IF (FLCHK)THEN
         I10 = 1
         OPEN(UNIT=10,FILE=FILE10,STATUS='OLD')
      ELSE
         STRG = 'MODEL FILE DOES NOT EXIST'
         CALL FMSSG(30,STRG)
         STRG = 'CANNOT RECALL NON-EXISTING FILE'
         CALL FMSSG(30,STRG)
         STRG = 'ENTER NEW NAME'
         CALL FMSSG(30,STRG)
         GOTO 161
      ENDIF

      INQUIRE(FILE=FILE21,EXIST=FLCHK)
      IF (FLCHK)THEN
         I21 = 1
         OPEN(UNIT=21,FILE=FILE21,STATUS='OLD')
      ELSE
         I21 = 0
         OPEN(UNIT=21,FILE=FILE21,STATUS='NEW')
      ENDIF

      INQUIRE(FILE=FILE22,EXIST=FLCHK)
      IF (FLCHK)THEN
         I22 = 1
         OPEN(UNIT=22,FILE=FILE22,STATUS='OLD')
      ELSE
         I22 = 0
         OPEN(UNIT=22,FILE=FILE22,STATUS='NEW')
      ENDIF

      INQUIRE(FILE=FILE31,EXIST=FLCHK)
      IF (FLCHK)THEN
         I31 = 1
         OPEN(UNIT=31,FILE=FILE31,STATUS='OLD')
      ELSE
         I31 = 0
         OPEN(UNIT=31,FILE=FILE31,STATUS='NEW')
```

```
      ENDIF

      INQUIRE(FILE=FILE32,EXIST=FLCHK)
      IF (FLCHK)THEN
         I32 = 1
         OPEN(UNIT=32,FILE=FILE32,STATUS='OLD')
      ELSE
         I32 = 0
         OPEN(UNIT=32,FILE=FILE32,STATUS='NEW')
      ENDIF

*IF A OLD FILE HAS TO BE RECALLED

      STRG = 'RECALLING MODEL '//FTRG
      CALL FMSSG(30,STRG)

*USE RECALL PROCEDURE TO BRING UP MODEL

      CALL FRSET
      CALL FRCLL(I10,I21,I22,I31,I32)


      FLFLAG = 1

      STRG = 'MODEL RECALLED'
      CALL FMSSG(30,STRG)

      RETURN

*MNUM = 1, IPKID = 7 IS NOT A POSSIBILITY

17    RETURN

*MNUM = 1, IPKID = 8 ITEM PICKED = FILE
*IN THIS CASE ASK FOR STRING INPUT

*CHECK IF A MODEL FILE ALREADY OPEN (FLFLAG = 0, NO FILE OPEN)
*                                  (FLFLAG = 1,    FILE OPEN)

18    IF (FLFLAG .EQ. 0) THEN
         STRG = 'NO MODEL FILE EXISTING'
         CALL FMSSG(30,STRG)
         STRG = 'CANNOT STORE EMPTY MODEL'
         CALL FMSSG(30,STRG)
         RETURN
      ENDIF

*INQUIRE NAME OF CURRENT MODEL

      INQUIRE(UNIT=10,NAME=FTRG)
      STRG = 'CURRENT MODEL NAME IS '//FTRG
      CALL FMSSG(30,STRG)
      CALL FPRMP(9)
      CALL FGTST(ANS)
      IF (ANS .EQ. 'Y')THEN
         STRG = 'SAVING MODEL UNDER FILE '//FTRG
         CALL FMSSG(30,STRG)
         CALL GPDLST(20)
         CLOSE(10)
         FLFLAG = 0
         STRG = 'MODEL SAVED'
         CALL FMSSG(30,STRG)
         RETURN
      ENDIF

      CALL FPRMP(5)
      FLFLAG = 0
      CALL FGTST(FTRG)
      CALL GPDLST(20)

*USE STORE PROCEDURE TO SAVE UP MODEL
*     CALL FSAVE

      CLOSE(10)

      RETURN

*MNUM = 1, IPKID = 9 ITEM PICKED = PARAMETER

19    STRG = 'PARAMETER OPTION NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)

      RETURN

*MNUM = 1, IPKID = 10 IS NOT A POSSIBILITY

20    RETURN

*MNUM = 1, IPKID = 11 ITEM PICKED = COMMAND FILE

21    CALL FPRMP(6)
      CALL FGTST(FTRG)
      STRG = 'EXECUTING COMMAND FILE '//FTRG
      CALL FMSSG(30,STRG)
```

```
*USE EXECUTE FILE PROCEDURE TO CREATE UP MODEL
*     CALL FEXEC

      STRG = 'COMMAND FILE NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)

      RETURN
1611  FORMAT(A72)
      END
```

SUBROUTINE FMPK10

```
************************************************************************
************************************************************************
**    SUBROUTINE FMPK10(MNUM,IPKID,PMENU)                             **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 10, THEN THE CORRECT     **
**    ACTION SHOULD BE TAKEN.                                         **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/21/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    MNUM   = MENU NUMBER (INTEGER, I/P)                             **
**           = 1                                                      **
**    IPKID = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)       **
**    PMENU = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                  **
**            (CHARACTER,I/P,O/P)                                     **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FMPK10(MNUM,IPKID,PMENU)

      INTEGER*4 MNUM, IPKID

      REAL*4 PMENU(50), LOC(3), ORI(3), LEN

      CHARACTER STRG*50, PRE*20, FTRG*8, ANS*1, BOOL*1

*GOTO THE CORRECT ID THAT WAS PICKED

1     GOTO(11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26)IPKID

*IF IPKID .GT. 8 RETURN

      RETURN

*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY

11    RETURN

*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY

12    RETURN

*MNUM = 1, IPKID = 3 ITEM PICKED = RETURN

13    MNUM = 3
      CALL FMENU2(MNUM,PMENU)

      RETURN

*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY

14    RETURN

*MNUM = 1, IPKID = 5 ITEM PICKED = SLAB
*IN THIS CASE DISPLAY APPROPRIATE SCREEN

15    CALL MSLAB(A,B,C,LOC,ORI,ICOL,BOOL)

      RETURN

*MNUM = 1, IPKID = 6 ITEM PICKED = SPHERE

16    CALL MSPHERE(RAD,LOC,ICOL,BOOL)

      RETURN

*MNUM = 1, IPKID = 7 ITEM PICKED = CYLINDER
*IN THIS CASE DISPLAY APPROPRIATE SCREEN

17    CALL MCYNDR(RAD,LEN,LOC,ORI,ICOL,BOOL)

      RETURN
```

```
*MNUM = 1, IPKID = 8 ITEM PICKED = RING
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
18    CALL MRING(R1,R2,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 9 ITEM PICKED = CONE
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
19    CALL MCONE(RAD,LEN,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 10 ITEM PICKED = PRYAMID
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
20    CALL MPYRMID(A,B,C,AR,IFL,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 11 ITEM PICKED = WEDGE
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
21    CALL MWEDGE(A,B,C,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 12 ITEM PICKED = ELLIPSOID
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
22    CALL MELPSOD(RX,RY,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 13 ITEM PICKED = HEMISPHERE
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
23    CALL MHEMIS(RX,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 14 ITEM PICKED = BUSHING
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
24    CALL MBUSH(R1,R2,THK,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 15 ITEM PICKED = COUNTER
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
25    CALL MCOUNT(R1,R2,L1,L2,IFL,IFL1,LOC,ORI,ICOL,BOOL)
      RETURN
*MNUM = 10, IPKID = 16 ITEM PICKED = TUNNEL
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
26    CALL MGRP4(R1,R2,THK,LOC,ORI,ICOL,BOOL)
      RETURN
      END
```

SUBROUTINE FMPK2

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FMPK2(MNUM,IPKID,PMENU)                            **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 2, THEN THE CORRECT    **
**    ACTION SHOULD BE TAKEN.                                       **
**                                                                  **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     11/05/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    MNUM   = MENU NUMBER (INTEGER, I/P)                           **
**           = 1                                                    **
**    IPKID  = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)    **
**    PMENU  = ARRAY OF ITEMS AVAILABLE FOR THIS MENU               **
**             (CHARACTER,I/P,O/P)                                  **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
      SUBROUTINE FMPK2(MNUM,IPKID,PMENU)
      INTEGER*4 MNUM, IPKID
      REAL*4 PMENU(50)
      CHARACTER STRG*50, PRE*20, FTRG*8, ANS*1
      LOGICAL*4 FLCHK
*GOTO THE CORRECT ID THAT WAS PICKED
1     GOTO(11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26)IPKID
*IF IPKID .GT. 15 RETURN
      RETURN
*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY
11    RETURN
*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY
12    RETURN
*MNUM = 1, IPKID = 3 ITEM PICKED = EXIT
*MAKE SURE ALL WORK HAS BEEN STORED
*IF NOT ASK WHETHER IT IS OK TO EXIT WITHOUT STORING
13    INQUIRE(UNIT=10,NAME=FTRG)
      IF(FTRG .NE. ' ')THEN
         STRG='MODEL '//FTRG//' CURRENTLY OPEN'
         CALL FMSSG(30,STRG)
         STRG='CURRENT MODEL WILL BE LOST IF YOU EXIT'
         CALL FMSSG(30,STRG)
         CALL FPRMP(10)
         CALL FGTST(ANS)
         IF(ANS .NE. 'Y')RETURN
         CALL FCLSE
      ELSE
         CALL FPRMP(10)
         CALL FGTST(ANS)
         IF(ANS .NE. 'Y')RETURN
         CALL FCLSE
      ENDIF
      STRG = 'EXIT NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY
14    RETURN
*MNUM = 1, IPKID = 5 ITEM PICKED = MODELING
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
15    MNUM = 3
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 6 ITEM PICKED = LANGUAGE INPUT
16    STRG = 'NATURAL LANGUAGE INPUT NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
*MNUM = 1, IPKID = 7 IS NOT A POSSIBILITY
17    RETURN
*MNUM = 1, IPKID = 8 ITEM PICKED = RENDERING
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
18    MNUM = 4
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 9 ITEM PICKED = VIEWS
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
19    MNUM = 5
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 10 ITEM PICKED = WINDOWS
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
20    MNUM = 6
```

```
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 11 IS NOT A POSSIBILITY
21    RETURN
*MNUM = 1, IPKID = 12 ITEM PICKED = DISPLAY
22    CALL FDISP
      RETURN
*MNUM = 1, IPKID = 13 ITEM PICKED = INQUIRE
23    CALL FQURY
      RETURN
*MNUM = 1, IPKID = 14 ITEM PICKED = H. COPY
24    CALL HRDCPY(1,2)
      CALL GPUPWS(2,2)
      STRG = 'HARD COPY GENERATED'
      CALL FMSSG(30,STRG)
      RETURN
*MNUM = 1, IPKID = 15 IS NOT A POSSIBILITY
25    RETURN
*MNUM = 1, IPKID = 16 ITEM PICKED = FILES
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
26    MNUM = 1
      CALL FMENU2(MNUM,PMENU)
      RETURN
      END
```

SUBROUTINE FMPK3

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FMPK3(MNUM,IPKID,PMENU)                            **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 3, THEN THE CORRECT    **
**    ACTION SHOULD BE TAKEN.                                       **
**                                                                  **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     11/05/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    MNUM   = MENU NUMBER (INTEGER, I/P)                           **
**           = 1                                                    **
**    IPKID  = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)    **
**    PMENU  = ARRAY OF ITEMS AVAILABLE FOR THIS MENU               **
**             (CHARACTER,I/P,O/P)                                  **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FMPK3(MNUM,IPKID,PMENU)
      INTEGER*4 MNUM, IPKID, CNUM
      REAL*4 PMENU(50)
      CHARACTER STRG*50, PRE*20, FTRG*8
      COMMON/PRTOF/CNUM
*GOTO THE CORRECT ID THAT WAS PICKED
1     GOTO(11,12,13,14,15,16,17,18,19,20,21)IPKID
*IF IPKID .GT. 10 RETURN
      RETURN
*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY
11    RETURN
*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY
```

```
12    RETURN

*MNUM = 1, IPKID = 3 ITEM PICKED = RETURN
*IN THIS CASE DISPLAY APPROPRIATE SCREEN

13    MNUM = 2
      CALL FMENU2(MNUM,PMENU)

      RETURN

*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY

14    RETURN

*MNUM = 1, IPKID = 5 ITEM PICKED = DEFINE
*IN THIS CASE DISPLAY APPROPRIATE SCREEN

15    CALL FDOBJ(IER)
      IF (IER .EQ. 1)RETURN
      MNUM = 10
      CALL FMENU2(MNUM,PMENU)

      RETURN

*MNUM = 1, IPKID = 6 ITEM PICKED = ADDTO

16    STRG = 'ADDTO NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      CALL FADTO(IER)
      IF (IER .EQ. 1)RETURN
      IF(CNUM .GT. 100 .AND. CNUM .LT. 1000)THEN
         CALL FDASM(1)
      ELSEIF(CNUM .GT. 1000 .AND. CNUM .LT. 10000)THEN
         MNUM = 10
         CALL FMENU2(MNUM,PMENU)
      ENDIF

      RETURN

*MNUM = 1, IPKID = 7 ITEM PICKED = EDIT

17    STRG = 'EDIT NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)

      CALL HRDCPY(1,2)
      CALL GPUPWS(2,2)

      RETURN

*MNUM = 1, IPKID = 8 IS NOT A POSSIBILITY

18    RETURN

*MNUM = 1, IPKID = 9 ITEM PICKED = COPY

19    STRG = 'COPY NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)

      RETURN

*MNUM = 1, IPKID = 10 ITEM PICKED = DELETE

20    STRG = 'DELETE NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      CALL FDELT

      RETURN

*MNUM = 1, IPKID = 11 ITEM PICKED = ASSEMBLE

21    STRG = 'ASSEMBLE NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      CALL FDASM(0)

      RETURN
      END
```

SUBROUTINE FMPK4

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FMPK4(MNUM,IPKID,PMENU)                            **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 4, THEN THE CORRECT    **
**    ACTION SHOULD BE TAKEN.                                       **
**                                                                  **
**                                                                  **
```

```
**     BY:       ASHIT R. GANDHI                                          **
**     DATE:     11/05/88                                                 **
**                                                                        **
**     PARAMETERS USED:                                                   **
**                                                                        **
**     MNUM   = MENU NUMBER (INTEGER, I/P)                                **
**            = 1                                                         **
**     IPKID = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)          **
**     PMENU = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                     **
**               (CHARACTER,I/P,O/P)                                      **
**                                                                        **
************************************************************************
************************************************************************
      SUBROUTINE FMPK4(MNUM,IPKID,PMENU)
      INTEGER*4 MNUM, IPKID
      REAL*4 PMENU(50)
      CHARACTER STRG*50, PRE*20, FTRG*8
      COMMON/VIS/NVIS
*GOTO THE CORRECT ID THAT WAS PICKED
1     GOTO(11,12,13,14,15,16,17,18,19,20,21,22)IPKID
*IF IPKID .GT. 12 RETURN
      RETURN
*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY
11    RETURN
*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY
12    RETURN
*MNUM = 1, IPKID = 3 ITEM PICKED = RETURN
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
13    MNUM = 2
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY
14    RETURN
*MNUM = 1, IPKID = 5 ITEM PICKED = COLOR
15    MNUM = 9
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 6 IS NOT A POSSIBILITY
16    RETURN
*MNUM = 1, IPKID = 7 ITEM PICKED = HIDDEN LINES
17    CALL FSHDE(-1)
      RETURN
*MNUM = 1, IPKID = 8 ITEM PICKED = HIDDEN SURFACE
18    CALL FSHDE(-1)
      RETURN
*MNUM = 1; IPKID = 9 IS NOT A POSSIBILITY
19    RETURN
*MNUM = 1, IPKID = 10 ITEM PICKED = SHADING
*IN THIS CASE DISPLAY THE APPROPRIATE SCREEN
20    MNUM = 7
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 11 IS NOT A POSSIBILITY
21    RETURN
*MNUM = 1, IPKID = 12 ITEM PICKED = RESET
22    CALL GPDRV(1,7,21)
      CALL GPARV(1,7,NVIS,0)
      RETURN
```

```
      END
```

SUBROUTINE FMPK5

```
C**********************************************************************
C**********************************************************************
C**    SUBROUTINE FMPK5(MNUM,IPKID,PMENU)                            **
C**                                                                  **
C**    PROGRAM DESCRIPTION                                           **
C**                                                                  **
C**    IF AN ITEM WAS PICKED FROM MENU NUMBER 5, THEN THE CORRECT    **
C**    ACTION SHOULD BE TAKEN.                                       **
C**                                                                  **
C**                                                                  **
C**    BY:      ASHIT R. GANDHI                                      **
C**    DATE:    11/05/88                                             **
C**                                                                  **
C**    PARAMETERS USED:                                              **
C**                                                                  **
C**    MNUM  = MENU NUMBER (INTEGER, I/P)                            **
C**          = 1                                                     **
C**    IPKID = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)     **
C**    PMENU = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                **
C**            (CHARACTER,I/P,O/P)                                   **
C**                                                                  **
C**********************************************************************
C**********************************************************************
      SUBROUTINE FMPK5(MNUM,IPKID,PMENU)
      INTEGER*4 MNUM, IPKID
      REAL*4 PMENU(50)
      CHARACTER STRG*50, PRE*20, FTRG*8
C*GOTO THE CORRECT ID THAT WAS PICKED
1     GOTO(11,12,13,14,15,16,17,18,19)IPKID
C*IF IPKID .GT. 9  RETURN
      RETURN
C*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY
11    RETURN
C*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY
12    RETURN
C*MNUM = 1, IPKID = 3 ITEM PICKED = RETURN
C*IN THIS CASE DISPLAY APPROPRIATE SCREEN
13    MNUM = 2
      CALL FMENU2(MNUM,PMENU)
      RETURN
C*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY
14    RETURN
C*MNUM = 1, IPKID = 5 ITEM PICKED = MOVE
15    STRG = 'MOVE NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
C*MNUM = 1, IPKID = 6 ITEM PICKED = ROTATE
16    STRG = 'ROTATE NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
C*MNUM = 1, IPKID = 7 ITEM PICKED = ZOOM
17    STRG = 'ZOOM NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
C*MNUM = 1, IPKID = 8 IS NOT A POSSIBILITY
18    RETURN
C*MNUM = 1, IPKID = 9 ITEM PICKED = RESET
```

```
19    STRG = 'RESET NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)

      RETURN
      END
```

SUBROUTINE FMPK6

```
************************************************************************
************************************************************************
**    SUBROUTINE FMPK6(MNUM,IPKID,PMENU)                              **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 6, THEN THE CORRECT      **
**    ACTION SHOULD BE TAKEN.                                         **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/05/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    MNUM   = MENU NUMBER (INTEGER, I/P)                             **
**           = 1                                                      **
**    IPKID = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)       **
**    PMENU = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                  **
**            (CHARACTER,I/P,O/P)                                     **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FMPK6(MNUM,IPKID,PMENU)

      INTEGER*4 MNUM, IPKID

      REAL*4 PMENU(50)

      CHARACTER STRG*50, PRE*20, FTRG*8

*GOTO THE CORRECT ID THAT WAS PICKED

1     GOTO(11,12,13,14,15,16,17,18)IPKID

*IF IPKID .GT. 8  RETURN

      RETURN

*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY

11    RETURN

*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY

12    RETURN

*MNUM = 1, IPKID = 3 ITEM PICKED = RETURN
*IN THIS CASE DISPLAY APPROPRIATE SCREEN

13    MNUM = 2
      CALL FMENU2(MNUM,PMENU)

      RETURN

*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY

14    RETURN

*MNUM = 1, IPKID = 5 ITEM PICKED = SINGLE

15    CALL GPVCH(1,8,2,1,1,2,11,2,1,1)
      CALL GPVCH(1,9,2,1,1,2,11,2,1,1)
      CALL GPVCH(1,10,2,1,1,2,11,2,1,1)
      CALL GPVCH(1,11,2,1,1,2,11,2,1,1)
      CALL GPVCH(1,7,2,1,1,2,11,2,1,2)

      MNUM = 8
      CALL FMENU2(MNUM,PMENU)

      RETURN

*MNUM = 1, IPKID = 6 ITEM PICKED = MULTIPLE
*DISPLAY APPROPRIATE SCREEN

16    CALL GPVCH(1,7,2,1,1,2,11,2,1,1)
      CALL GPVCH(1,8,2,1,1,2,11,2,1,2)
      CALL GPVCH(1,9,2,1,1,2,11,2,1,2)
      CALL GPVCH(1,10,2,1,1,2,11,2,1,2)
      CALL GPVCH(1,11,2,1,1,2,11,2,1,2)
      CALL FMULT
```

```
      RETURN
*MNUM = 1, IPKID = 7 IS NOT A POSSIBILITY
17    RETURN
*MNUM = 1, IPKID = 8 ITEM PICKED = RESET
18    STRG = 'RESET NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
      END
```

SUBROUTINE FMPK7

```
************************************************************************
************************************************************************
**    SUBROUTINE FMPK7(MNUM,IPKID,PMENU)                              **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 7, THEN THE CORRECT      **
**    ACTION SHOULD BE TAKEN.                                         **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/05/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    MNUM   = MENU NUMBER (INTEGER, I/P)                             **
**             = 1                                                    **
**    IPKID = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)       **
**    PMENU = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                  **
**            (CHARACTER,I/P,O/P)                                     **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FMPK7(MNUM,IPKID,PMENU)

      INTEGER*4 MNUM, IPKID

      REAL*4 PMENU(50)

      CHARACTER STRG*50, PRE*20, FTRG*8

*GOTO THE CORRECT ID THAT WAS PICKED
1     GOTO(11,12,13,14,15,16,17,18,19)IPKID
*IF IPKID .GT. 9  RETURN
      RETURN
*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY
11    RETURN
*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY
12    RETURN
*MNUM = 1, IPKID = 3 ITEM PICKED = RETUEN
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
13    MNUM = 4
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY
14    RETURN
*MNUM = 1, IPKID = 5 ITEM PICKED = CONSTANT
15    CALL FSHDE(1)
      RETURN
*MNUM = 1, IPKID = 6 ITEM PICKED = PHONG
16    STRG = 'PHONG SHADING NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
*MNUM = 1, IPKID = 7 ITEM PICKED = GOURAUD
```

```
17    STRG = 'GOURAUD SHADING NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)

      RETURN

*MNUM = 1, IPKID = 8 IS NOT A POSSIBILITY

18    RETURN

*MNUM = 1, IPKID = 9 ITEM PICKED = RESET

19    STRG = 'RESET NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)

      RETURN
      END
```

SUBROUTINE FMPK8

```
************************************************************************
************************************************************************
**    SUBROUTINE FMPK8(MNUM,IPKID,PMENU)                              **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 8, THEN THE CORRECT      **
**    ACTION SHOULD BE TAKEN.                                         **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/05/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    MNUM   = MENU NUMBER (INTEGER, I/P)                             **
**           = 1                                                      **
**    IPKID = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)       **
**    PMENU = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                  **
**            (CHARACTER,I/P,O/P)                                     **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FMPK8(MNUM,IPKID,PMENU)

      INTEGER*4 MNUM, IPKID

      REAL*4 PMENU(50)

      CHARACTER STRG*50, PRE*20, FTRG*8

*GOTO THE CORRECT ID THAT WAS PICKED

1     GOTO(11,12,13,14,15,16,17,18,19,20,21)IPKID

*IF IPKID .GT. 11  RETURN

      RETURN

*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY

11    RETURN

*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY

12    RETURN

*MNUM = 1, IPKID = 3 ITEM PICKED = RETURN
*IN THIS CASE DISPLAY APPROPRIATE SCREEN

13    MNUM = 6
      CALL FMENU2(MNUM,PMENU)

      RETURN

*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY

14    RETURN

*MNUM = 1, IPKID = 5 ITEM PICKED = FRONT

15    CALL FFNTV(7)

      RETURN

*MNUM = 1, IPKID = 6 ITEM PICKED = TOP

16    CALL FTOPV(7)

      RETURN
```

```
*MNUM = 1, IPKID = 7 ITEM PICKED = RIGHT SIDE
17    CALL FRGTV(7)
      RETURN
*MNUM = 1, IPKID = 8 ITEM PICKED = ISOMETRIC
18    CALL FISOV(7)
      RETURN
*MNUM = 1, IPKID = 9 ITEM PICKED = BACK
19    CALL FBCKV(7)
      STRG = 'BACK VIEW NOT IMPLEMENTED'
      CALL FMSSG(30,STRG)
      RETURN
*MNUM = 1, IPKID = 10 ITEM PICKED = BOTTOM
20    CALL FBOTV(7)
      RETURN
*MNUM = 1, IPKID = 11 ITEM PICKED = LEFT SIDE
21    CALL FLFTV(7)
      RETURN
      END
```

SUBROUTINE FMPK9

```
************************************************************************
************************************************************************
**    SUBROUTINE FMPK9(MNUM,IPKID,PMENU)                              **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    IF AN ITEM WAS PICKED FROM MENU NUMBER 7, THEN THE CORRECT      **
**    ACTION SHOULD BE TAKEN.                                         **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/05/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    MNUM   = MENU NUMBER (INTEGER, I/P)                             **
**           = 1                                                      **
**    IPKID  = PICK ID OF THE ITEM THAT WAS PICKED (INTEGER,I/P)      **
**    PMENU  = ARRAY OF ITEMS AVAILABLE FOR THIS MENU                 **
**             (CHARACTER,I/P,O/P)                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FMPK9(MNUM,IPKID,PMENU)

      INTEGER*4 MNUM, IPKID

      REAL*4 PMENU(50), COL(3), SCOL(3)

      CHARACTER STRG*50, PRE*20, FTRG*8

      COMMON/SHADE/SCOL

      DATA ICST/2/
      DATA NCOL/1/

*GOTO THE CORRECT ID THAT WAS PICKED

1     GOTO(11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,
     &     28,29,30)IPKID

*IF IPKID .GT. 9  RETURN

      RETURN

*MNUM = 1, IPKID = 1 IS NOT A POSSIBILITY

11    RETURN

*MNUM = 1, IPKID = 2 IS NOT A POSSIBILITY

12    RETURN

*MNUM = 1, IPKID = 3 ITEM PICKED = RETURN
*IN THIS CASE DISPLAY APPROPRIATE SCREEN
```

```
13    MNUM = 4
      CALL FMENU2(MNUM,PMENU)
      RETURN
*MNUM = 1, IPKID = 4 IS NOT A POSSIBILITY
14    RETURN
*MNUM = 1, IPKID = 5 ITEM PICKED = MAXIMUM RED
15    COL(1) = 1.0
      COL(2) = 0.0
      COL(3) = 0.0
      GOTO 35
*MNUM = 1, IPKID = 6 ITEM PICKED = MAXIMUM GREEN
16    COL(1) = 0.0
      COL(2) = 1.0
      COL(3) = 0.0
      GOTO 35
*MNUM = 1, IPKID = 7 ITEM PICKED = MAXIMUM BLUE
17    COL(1) = 0.0
      COL(2) = 0.0
      COL(3) = 1.0
      GOTO 35
*MNUM = 1, IPKID = 8 ITEM PICKED = MAXIMUM YELLOW
18    COL(1) = 1.0
      COL(2) = 1.0
      COL(3) = 0.0
      GOTO 35
*MNUM = 1, IPKID = 9 ITEM PICKED = MAGENTA
19    COL(1) = 1.0
      COL(2) = 0.0
      COL(3) = 1.0
      GOTO 35
*MNUM = 1, IPKID = 10 ITEM PICKED = TURQUOISE
20    COL(1) = 0.0
      COL(2) = 1.0
      COL(3) = 1.0
      GOTO 35
*MNUM = 1, IPKID = 11 ITEM PICKED = DARK MAGENTA
21    COL(1) = 0.3
      COL(2) = 0.0
      COL(3) = 0.3
      GOTO 35
*MNUM = 1, IPKID = 12 ITEM PICKED = DARK GREEN
22    COL(1) = 0.0
      COL(2) = 0.2
      COL(3) = 0.0
      GOTO 35
*MNUM = 1, IPKID = 13 ITEM PICKED = DARK BLUE
23    COL(1) = 0.0
      COL(2) = 0.0
      COL(3) = 0.5
      GOTO 35
*MNUM = 1, IPKID = 14 ITEM PICKED = BACKGROUND GREY
24    COL(1) = 0.25
      COL(2) = 0.25
      COL(3) = 0.25
      GOTO 35
*MNUM = 1, IPKID = 15 ITEM PICKED = LIGHT GREY
25    COL(1) = 0.4
      COL(2) = 0.5
      COL(3) = 0.5
      GOTO 35
*MNUM = 1, IPKID = 16 ITEM PICKED = PEACOCK GREEN
26    COL(1) = 0.0
      COL(2) = 1.0
      COL(3) = 0.6
      GOTO 35
*MNUM = 1, IPKID = 17 ITEM PICKED = ORANGE
```

```
27    COL(1) = 1.0
      COL(2) = 0.3
      COL(3) = 0.3
      GOTO 35

*MNUM = 1, IPKID = 18 ITEM PICKED = BLACK

28    COL(1) = 0.0
      COL(2) = 0.0
      COL(3) = 0.0
      GOTO 35

*MNUM = 1, IPKID = 19 IS NOT A POSSIBILITY

29    RETURN

*MNUM = 1, IPKID = 20 ITEM PICKED = PURE WHITE

30    COL(1) = 1.0
      COL(2) = 1.0
      COL(3) = 1.0

35    CALL GPCR(1,ICST,NCOL,COL)

      SCOL(1) = COL(1)
      SCOL(2) = COL(2)
      SCOL(3) = COL(3)

      RETURN
      END
```

SUBROUTINE FMSSG

```
************************************************************************
************************************************************************
**    SUBROUTINE FMSSG(NUM)                                           **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL ECHO ALL OPTIONS SELECTED AND ANY STRING      **
**    INPUT.                                                          **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     10/08/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NUM  =  NUMBER FOR CORRESPONDING MESSAGE (INTEGER, I/P)         **
**    MSG  =  STRING FOR MSG IF NOT EXISTIENT IN THE DATA             **
**            (CHARACTER,I/P)                                         **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FMSSG(NUM,MSG)

      REAL POS(2)

      CHARACTER MSGG(30)*72, TXT(9)*72, MSG*(*)

*DATA FOR MESSAGE LENGTH

      DATA LNT/50/
      DATA TXT/9*' '/

*DATA FOR MESSAGES TO BE DISPLAYED

      DATA MSGG(1)/'WELCOME TO "FEATURE"                                '/
      DATA MSGG(2)/'A FEATURE BASED GEOMETRIC MODELING SYSTEM           '/
      DATA MSGG(3)/'VERSION 1.0                                         '/
      DATA MSGG(4)/'DEVELOPED AT                                        '/
      DATA MSGG(5)/'VIRGINIA POLYTECHNIC INSTITUTE & STATE UNIVERSITY'/
      DATA MSGG(6)/'STARTING FILE......                                 '/
      DATA MSGG(7)/'RESTARTING FILE....                                 '/
      DATA MSGG(8)/'DEFINING OBJECT....                                 '/
      DATA MSGG(9)/'ADDTO OBJECT.......                                 '/
      DATA MSGG(10)/'ONLY DEFINE AND ADDTO ACTIVE AT PRESENT...         '/
      DATA MSGG(11)/'EDIT,MODIFY AND ERASE NOT ACTIVE.........          '/
      DATA MSGG(12)/'OBJECT DEFINED....                                 '/
      DATA MSGG(13)/'ADDING TO OBJECT......                             '/
      DATA MSGG(14)/'OBJECT ADDED TO...                                 '/
      DATA MSGG(15)/'FEATURE LIST 1 ...                                 '/
      DATA MSGG(16)/'EXITING FROM THE FEATURE MENU 1 ....               '/
      DATA MSGG(17)/'FEATURE BOX ......                                 '/
      DATA MSGG(18)/'ONLY FEATURE BOX IS ACTIVE.....                    '/
      DATA MSGG(19)/'FEATURE LIST 2 ...                                 '/
      DATA MSGG(20)/'EXITING FROM THE FEATURE MENU 2 ....               '/
      DATA MSGG(21)/'FEATURE LIST 2 NOT SUPPORTED ......                '/
      DATA MSGG(22)/'EXITING BOX.....                                   '/
      DATA MSGG(23)/'DIMENSIONING OBJECT.......                         '/
```

```
      DATA MSGG(24)/'MOVING OBJECT.....                              '/
      DATA MSGG(25)/'MOVING OBJECT NOT SUPPORTED                     '/
      DATA MSGG(26)/'RETURN                                          '/
*DEFINE TEXT LOCATION
      POS(1)=0.02
      POS(2)=0.85
* SCROLL ALL TEXT
      MSGG(30) = MSG
      TXT(1)=TXT(2)
      TXT(2)=TXT(3)
      TXT(3)=TXT(4)
      TXT(4)=TXT(5)
      TXT(5)=TXT(6)
      TXT(6)=TXT(7)
      TXT(7)=TXT(8)
      TXT(8)=TXT(9)
      TXT(9)=MSGG(NUM)
*OPEN STRUCTURE FOR MESSAGE ITEMS
      CALL GPEST(5)
*OPEN STRUCTURE 5
      CALL GPOPST(5)
*SET TEXT COLOR TO YELLOW
      CALL GPTXCI(5)
*SET ANNOTATION TEXT SIZE SCALE FACTOR
      CALL GPAHSC(0.50)
*DRAW ALL TEXT
      DO 100 I=1,8
         CALL GPAN2(POS,LNT,TXT(I))
         POS(2)=POS(2)-0.1
 100  CONTINUE
*SET TEXT COLOR TO GREEN
      CALL GPTXCI(3)
*DRAW TEXT
      CALL GPAN2(POS,LNT,TXT(9))
*CLOSE STRUCTURE
      CALL GPCLST
*ASSOCIATE THE STRUCTURE 5 WITH A VIEW 5
      CALL GPARV(1,5,5,0)
*SET VIEW PRIORITY
      CALL GPVP(1,5,2,1)
*UPDATE WORKSTATION
      CALL GPUPWS(1,2)
      RETURN
      END
```

SUBROUTINE FMT11

```
************************************************************************
************************************************************************
**    SUBROUTINE FMT11(A,B,C)                                         **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE MULTIPLIES A 1X4 MATRIX WITH A 4X1 MATRIX TO GIVE  **
**    A SINGLE SCALAR ANSWER.                                         **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     10/08/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    A    = 1X4 MATRIX (REAL,I/P)                                    **
**    B    = 4X1 MATRIX (REAL,I/P)                                    **
```

```
**     C    = SCALAR SOLUTION (REAL,I/P)                                    **
**                                                                          **
******************************************************************************
******************************************************************************
      SUBROUTINE FMT11(A,B,C)

      REAL*4 A(4), B(4), C

      C = 0.0

*SUMMATION OF MULTIPLICATIONS

      DO 200 I = 1,4
         C = A(I)*B(I) + C
200   CONTINUE

      RETURN
      END
```

SUBROUTINE FMT14

```
******************************************************************************
******************************************************************************
**     SUBROUTINE FM14(U,A,B)                                               **
**                                                                          **
**     PROGRAM DESCRIPTION                                                  **
**                                                                          **
**     THIS ROUTINE MULTIPLIES A 1X4 MATRIX WITH A 4X4 MATRIX TO GIVE       **
**     A 1X4 VECTOR SOLUTION.                                               **
**                                                                          **
**     BY:       ASHIT R. GANDHI                                            **
**     DATE:     10/08/88                                                   **
**                                                                          **
**     PARAMETERS USED:                                                     **
**                                                                          **
**     U    = 1X4 MATRIX (REAL,I/P)                                         **
**     A    = 4X4 MATRIX (REAL,I/P)                                         **
**     B    = 1X4 MATRIX SOLUTION (REAL,I/P)                                **
**                                                                          **
******************************************************************************
******************************************************************************
      SUBROUTINE FMT14(U,A,B)

      REAL   U(4), A(4,4), B(4)

*SUMMATION OF MULTIPLICATIONS

      DO 200 I = 1,4
         B(I) = 0.0
         DO 100 J = 1,4
            B(I) = A(J,I)*U(J) + B(I)
100      CONTINUE
200   CONTINUE

      RETURN
      END
```

SUBROUTINE FMT44

```
******************************************************************************
******************************************************************************
**     SUBROUTINE FMT44(A,B,C)                                              **
**                                                                          **
**     PROGRAM DESCRIPTION                                                  **
**                                                                          **
**     THIS ROUTINE MULTIPLIES A 4X4 MATRIX WITH A 4X1 MATRIX TO GIVE       **
**     A 4X1 VECTOR SOLUTION.                                               **
**                                                                          **
**     BY:       ASHIT R. GANDHI                                            **
**     DATE:     11/07/88                                                   **
**                                                                          **
**     PARAMETERS USED:                                                     **
**                                                                          **
**     A    = 4X4 MATRIX (REAL,I/P)                                         **
**     B    = 4X4 MATRIX (REAL,I/P)                                         **
**     C    = 4X4 MATRIX SOLUTION (REAL,O/P)                                **
**                                                                          **
******************************************************************************
******************************************************************************
      SUBROUTINE FMT44(A,B,C)

      REAL A(4,4), B(4,4), C(4,4)

*SUMMATION OF MULTIPLICATIONS
```

```
      DO 300 I = 1,4
         DO 200 J = 1,4
            C(I,J) = 0.0
            DO 100 K = 1,4
               C(I,J) = A(I,K)*B(K,J) + C(I,J)
100         CONTINUE
200      CONTINUE
300   CONTINUE

      RETURN
      END
```

SUBROUTINE FMULT

```
************************************************************************
************************************************************************
**    SUBROUTINE FMULT                                                **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE WILL ALLOW THE DISPLAY OF MULTIPLE VIEWS OF     **
**    ANY OBJECT.                                                     **
**                                                                    **
**                                                                    **
**    BY:      ASHIT R. GANDHI                                        **
**    DATE:    11/07/88                                               **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NONE                                                            **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FMULT

*APPLY VIEW TRANSFORMATION TO EACH VIEW

      CALL FFNTV(8)
      CALL FTOPV(9)
      CALL FRGTV(10)
      CALL FISOV(11)

      RETURN
      END
```

SUBROUTINE FNORM

```
************************************************************************
************************************************************************
**    SUBROUTINE FNORM(A,B,C)                                         **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE WILL NORMALIZE A GIVEN VECTOR TO A VECTOR       **
**    WITH PREDEFINED MAGNITUDE.                                      **
**                                                                    **
**    BY:      ASHIT R. GANDHI                                        **
**    DATE:    10/08/88                                               **
**                                                                    **
**    PARÁMETERS USED:                                                **
**                                                                    **
**    A = X-ELEMENT OF THE VECTOR (REAL,INPUT,OUTPUT)                 **
**    B = Y-ELEMENT OF THE VECTOR (REAL,INPUT;OUTPUT)                 **
**    C = Z-ELEMENT OF THE VECTOR (REAL,INPUT;OUTPUT)                 **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FNORM(A,B,C)

      REAL*4 A,B,C,RMAG

      RMAG = SQRT(A**2 + B**2 + C**2)

      IF (RMAG .EQ. 0.0)THEN
         RETURN
      ENDIF

      A = A/RMAG
      B = B/RMAG
      C = C/RMAG
```

```
      RETURN
      END
```

SUBROUTINE FOPEN

```
************************************************************************
************************************************************************
**    SUBROUTINE FOPEN                                                **
**                                                                    **
**    SUBROUTINE DESCRIPTION                                          **
**                                                                    **
**    THIS SUBROUTINE ALLOWS THE USER TO INITIALIZE PHIGS. IT OPENS   **
**    PHIGS, WORKSTATION AND LOADS THE COLOR TABLE.                   **
**                                                                    **
**    BY:   ASHIT R. GANDHI                                           **
**    DATE: 10/07/88                                                  **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    ERFILE : NAME OF ERROR FILE                                     **
**    WSTYPE : WORKSTATION TYPE                                       **
**    CONNID : WORKSTATION CONNECTION ID                              **
**    ICST   : START INDEX INTO COLOR TABLE                           **
**    NCOL   : TOTAL NUMBER OF COLORS TO BE LOADED                    **
**    COLORS : ARRAY OF R,G,B VALUES FOR COLORS LOADED                **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FOPEN

      COMMON CSIZE

      REAL*4 COLORS(48), CSIZE(3)
      INTEGER*4 ICST,NCOL,ASIZE(3)
      CHARACTER*8 ERFILE,WSTYPE,CONNID

*                    R   G   B
*                                          MAXIMUM BLACK
      DATA COLORS/ 0., 0., 0.,
*                                          MAXIMUM WHITE
     1             1., 1., 1.,
*                                          MAXIMUM RED
     2             1., 0., 0.,
*                                          MAXIMUM GREEN
     3             0., 1., 0.,
*                                          MAXIMUM BLUE
     4             0., 0., 1.,
*                                          MAXIMUM YELLOW
     5             1., 1., 0.,
*                                          MAGENTA
     6             1., 0., 1.,
*                                          TURQUOISE
     7             0., 1., 1.,
*                                          DARK MAGENTA
     8              .3, 0., .3,
*                                          DARK GREEN
     9             0., .2, 0.,
*                                          DARK BLUE
     0             0., 0., .5,
*                                          BACKGROUND GREY
     1             0., 0., 0.,
*                                          LIGHT GREY
     2            .40,.50,.50,
*                                          PEACOCK GREEN
     3             0., 1., .6,
*                                          ORANGE
     4             1., .3, .3,
*                                          COLOR FOR MODEL
     *             0., 1., .6/

      DATA ICST/0/
      DATA NCOL/16/

      DATA ERFILE/'SYSPRINT'/
      DATA WSTYPE/'5080    '/
      DATA CONNID/'IBM5080 '/

*     OPEN PHIGS

      CALL GPOPPH(ERFILE,0)

*     OPEN WORKSTATION

      CALL GPOPWS(1,CONNID,WSTYPE)

*     LOAD COLOR TABLE

      CALL GPCR(1,ICST,NCOL,COLORS)

*     INQUIRE DISPLAY SURFACE SIZE
```

```
      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

      RETURN
      END
```

SUBROUTINE FPARS

```
**********************************************************************
**********************************************************************
**                                                                  **
**    SUBROUTINE FPARS(LCHAR,NW,WORD)                               **
**                                                                  **
**  THIS SUBROUTINE WILL TAKE A LINE, AND SPLIT IT INTO WORDS THAT  **
**  ARE A PART OF IT.  ADDITIONALLY IT ALSO GIVES THE NUMBER OF     **
**  WORDS THAT FORM THAT LINE.                                      **
**                                                                  **
**                        BY:  ASHIT R. GANDHI                      **
**                        DATE:    07/15/88                         **
**                                                                  **
**  PARAMETERS USED:                                                **
**                                                                  **
**  LCHAR - 72 CHARACTER STRING LINE  (INPUT)                       **
**  NW    - NUMBER OF WORDS FORMING THE LINE (OUTPUT)               **
**  WORD  - ARRAY OF WORDS THAT FORM THE LINE (OUTPUT)              **
**                                                                  **
**  VARIABLES USED:                                                 **
**                                                                  **
**  FC    - FIRST CHARACTER OF THE WORD STRING                      **
**  LC    - LAST CHARACTER OF THE WORD STRING                       **
**  IW    - FLAG TO INDICATE STATE OF WORD                          **
**            0 - END OF WORD/BLANK SPACE                           **
**            1 - CONTINUOUS WORD                                   **
**                                                                  **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FPARS(LCHAR,NW,WORD)
      CHARACTER LCHAR*72, CHAR*1, WORD(15)*12, SEPA(13)*1
      INTEGER FC(20),LC(20)

      DATA SEPA /'(',')',';',',',':','$','@','{','}','?','!','"',' '/
      DATA NSEPA/13/

      NW = 0
      IW = 0
      DO 200 I = 1,72
         CHAR = LCHAR(I:I)
         DO 100 J = 1,NSEPA
            IF (CHAR .EQ. SEPA(J))THEN
               LCHAR(I:I) = ' '
               GOTO 200
            ENDIF
100      CONTINUE
200   CONTINUE

      DO 300 I = 1,72
         CHAR = LCHAR(I:I)
         IF (CHAR .NE. ' ' .AND. IW .EQ. 0)THEN
            NW = NW + 1
            FC(NW) = I
            IW = 1
         ENDIF
         IF (CHAR .EQ. ' ' .AND. IW .EQ. 1)THEN
            LC(NW) = I
            IW = 0
            WORD(NW) = LCHAR(FC(NW):LC(NW))
         ENDIF
300   CONTINUE

      RETURN
      END
```

SUBROUTINE FPGAT

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FPGAT(IPGC,IPGS)                                   **
**                                                                  **
**    SUBROUTINE DESCRIPTION                                        **
**                                                                  **
**    THIS SUBROUTINE SETS UP POLYGON ATTRIBUTES.                   **
**                                                                  **
**    BY:   Ashit R. Gandhi                                         **
```

```
**     DATE: 10/07/88                                                  **
**                                                                     **
**     PARAMETERS USED:                                                **
**                                                                     **
**     IPGC    : POLYGON COLOR INDEX                                   **
**     IPGS    : POLYGON INTERIOR STYLE INDEX                          **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FPGAT(IPGC,IPGS)

*     SET UP COLOR INDEX

      CALL GPICI(IPGC)

*     SET INTERIOR STYLE

      CALL GPIS(IPGS)

      RETURN
      END
```

SUBROUTINE FPRMP

```
*************************************************************************
*************************************************************************
**     SUBROUTINE FPRMP(NUM)                                           **
**                                                                     **
**     PROGRAM DESCRIPTION                                             **
**                                                                     **
**     THIS SUBROUTINE PROMPT MESSAGES IN THE PRE-DEFINED AREA.        **
**                                                                     **
**     BY:       ASHIT R. GANDHI                                       **
**     DATE:     10/08/88                                              **
**                                                                     **
**     PARAMETERS USED:                                                **
**                                                                     **
**     NUM = MESSAGE NUMBER (INTEGER, I/P)                             **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FPRMP(NUM)

      INTEGER*4 LNT,NUM

      REAL*4 POS(2)

      CHARACTER PROMPT(30)*50

*LENGTH OF PROMPT MESSAGES

      DATA LNT/50/

*PROMPT MESSAGES

      DATA PROMPT(1)/'SELECT START OR RECALL TO BEGIN                '/
      DATA PROMPT(2)/'SELECT OPTION                                  '/
      DATA PROMPT(3)/'ENTER NAME OF THE FILE TO START                '/
      DATA PROMPT(4)/'ENTER NAME OF THE FILE TO RECALL               '/
      DATA PROMPT(5)/'ENTER NAME OF THE FILE TO STORE                '/
      DATA PROMPT(6)/'ENTER NAME OF THE COMMAND FILE TO BE EXECUTED  '/
      DATA PROMPT(7)/'DO YOU WANT TO OPEN A NEW FILE? (Y/N)          '/
      DATA PROMPT(8)/'DO YOU WANT TO RECALL A MODEL FILE? (Y/N)      '/
      DATA PROMPT(9)/'SAVE UNDER SAME NAME? (Y/N)                    '/
      DATA PROMPT(10)/'DO YOU STILL WANT TO EXIT? (Y/N)              '/
      DATA PROMPT(11)/'ARE YOU SURE? Y/N                             '/
      DATA PROMPT(12)/'INVALID OPTION SELECTED                       '/
      DATA PROMPT(13)/'MODEL HAS NOT BEEN SAVED. OK TO EXIT ? (Y/N)  '/
      DATA PROMPT(14)/'SELECT PROPER PARAMETERS AND THEN EXECUTE     '/
      DATA PROMPT(15)/'ENTER NAME OF OBJECT TO DEFINE                '/
      DATA PROMPT(16)/'ENTER NAME OF COMPONENT TO BE DELETED         '/
      DATA PROMPT(17)/'ENTER NAME OF COMPONENT TO ADDTO              '/
      DATA PROMPT(18)/'ENTER NAME OF COMPONENT TO DISPLAY            '/

*DEFINE TEXT LOCATION

      POS(1)= .03
      POS(2)= .32

*OPEN STRUCTURE FOR PROMPT ITEMS

      CALL GPEST(4)

*OPEN STRUCTURE 4

      CALL GPOPST(4)

*SET TEXT COLOR TO TURQUOISE

      CALL GPTXCI(7)
```

```
*SET ANNOTATION SIZE SCALE FACTOR
      CALL GPAHSC(0.08)
*DRAW TEXT
      CALL GPAN2(POS,LNT,PROMPT(NUM))
*CLOSE STRUCTURE
      CALL GPCLST
*ASSOCIATE THE STRUCTURE 4 WITH A VIEW 3
      CALL GPARV(1,3,4,0)
*SET VIEW PRIORITY
      CALL GPVP(1,3,2,1)
*UPDATE WORKSTATION
      CALL GPUPWS(1,2)
      RETURN
      END
```

SUBROUTINE FPRPK

```
************************************************************************
************************************************************************
**    SUBROUTINE FPRPK(MNUM,IPKID,PMENU)                              **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE CALLS THE APPROPRIATE MENU ROUTINE TO PROCESS   **
**    THE ITEM THAT WAS PICKED.                                       **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/04/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    MNUM  = MENU NUMBER THAT WAS LAST DISPLAYED (INTEGER,I/P)       **
**    IPKID = PICK ID OF THE ITME THAT WAS PICKED (INTEGER,I/P)       **
**    PMENU = ARRAY OF ITEMS PRESENT ON THE LAST MENU DISPLAYED       **
**            (CHARACTER,I/P)                                         **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FPRPK(MNUM,IPKID,PMENU)
      INTEGER*4 MNUM, IPKID
      CHARACTER STRG*50, PMENU(50)*15
*GO TO CORRECT MENU SELECTION AREA
      GOTO(1,2,3,4,5,6,7,8,9,10)MNUM
*IF MNUM .GT. 10 RETURN
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 1
1     CALL FMPK1(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 2
2     CALL FMPK2(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 3
3     CALL FMPK3(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 4
4     CALL FMPK4(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 5
```

```
5     CALL FMPK5(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 6
6     CALL FMPK6(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 7
7     CALL FMPK7(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 8
8     CALL FMPK8(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 9
9     CALL FMPK9(MNUM,IPKID,PMENU)
      RETURN
*IF LAST MENU DISPLAYED IN NUMBER 10
10    CALL FMPK10(MNUM,IPKID,PMENU)
      RETURN
      END
```

SUBROUTINE FQURY

```
************************************************************************
************************************************************************
**    SUBROUTINE FQURY                                                **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE QUERIES THE NUMBER OF OBJECTS AND ASSEMBLIES       **
**    EXISTING IN THE MODEL. ALSO GIVES CORRESPONDING NAME FOR        **
**    THE COMPONENT                                                   **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     01/13/89                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NONE                                                            **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FQURY

      INTEGER*4 ASNUM, OBNUM, INNUM
      CHARACTER*8 ASSEM(900), OBJECT(9000)
      CHARACTER QUERY*50(5), CA*4
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT

*COMPUTE NUMBER OF OBJECTS AND ASSEMBLIES PRESENT IN THE MODEL

      NASM = ASNUM - 100
      NOBJ = OBNUM - 1000

*PRINT REQUESTED INFORMATION AND ASSEMBLY NAMES

      WRITE(CA,'(I4)')NASM
      QUERY(1) = 'NUMBER OF EXISTING ASSEMBLIES: '//CA
      IF (ASNUM .GT. 100)THEN
         DO 100 I = 1,ASNUM-100
            WRITE(6,*)I,ASSEM(I)
100      CONTINUE
      ENDIF

*PRINT REQUESTED INFORMATION AND OBJECT NAMES

      WRITE(CA,'(I4)')NOBJ
      QUERY(2) = 'NUMBER OF EXISTING OBJECTS: '//CA
      IF (OBNUM .GT. 1000)THEN
         DO 200 I = 1,OBNUM-1000
            WRITE(6,*)I,OBJECT(I)
200      CONTINUE
```

```
      ENDIF
*DISPLAY REQUESTED INFROMATION TO SCREEN
      DO 300 I = 1,2
         CALL FMSSG(30,QUERY(I))
300   CONTINUE
      RETURN
      END
```

SUBROUTINE FRBMH

```
************************************************************************
************************************************************************
**    SUBROUTINE FRBMH(PMAT,HMAT,NU,NW,PT)                            **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE USES THE GEOMETRIC COEFFICIENT MATRIX TO        **
**    CALCULATE THE SPECIFIED NUMBER OF POINTS WHICH WOULD LIE ON     **
**    THE PERIODIC CUBIC RATIONAL B-SPLINE SURFACE                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/07/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    PMAT  = GEOMETRIC COEFFICIENT MATRIX (REAL,I/P)                 **
**    HMAT  = WEIGHT MATRIX (REAL,I/P)                                **
**    NU    = NUMBER OF POINTS TO BE COMPUTED IN U-DIRECTION          **
**            (INTEGER,I/P)                                           **
**    NW    = NUMBER OF POINTS TO BE COMPUTED IN W-DIRECTION          **
**            (INTEGER,I/P)                                           **
**    PT    = COORDINATES OF POINTS ON THE COMPUTED SURFACE           **
**            (REAL,O/P)                                              **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FRBMH(PMAT,HMAT,NU,NW,PT)

      INTEGER*4 NU, NW

      REAL*4 M(4,4), MT(4,4)
      REAL*4 U, VU(4), F(4), W, VW(4), S(4)
      REAL*4 PMAT(4,4,3), PXMAT(4,4), PYMAT(4,4), PZMAT(4,4)
      REAL*4 PT(20,20,3), HMAT(4,4), H(20,20), HMAT1(4,4), PMT(4,4)
      REAL*4 FX(4), FY(4), FZ(4), FH(4)

      DATA M /-1.0, 3.0,-3.0, 1.0,
     2         3.0,-6.0, 0.0, 4.0,
     3        -3.0, 3.0, 3.0, 1.0,
     4         1.0, 0.0, 0.0, 0.0/

      DATA MT/-1.0, 3.0,-3.0, 1.0,
     2         3.0,-6.0, 3.0, 0.0,
     3        -3.0, 0.0, 3.0, 0.0,
     4         1.0, 4.0, 1.0, 0.0/

*SPLIT THE P-MATRIX INTO ITS RESPECTIVE ELEMENTS

      DO 200 I = 1,4
         DO 100 J = 1,4
            PXMAT(I,J) = PMAT(I,J,1)
            PYMAT(I,J) = PMAT(I,J,2)
            PZMAT(I,J) = PMAT(I,J,3)
100      CONTINUE
200   CONTINUE

*CALCULATE THE DELTA CHANGES IN U AND W

      DELU = 1.0/FLOAT(NU -1)
      DELW = 1.0/FLOAT(NW -1)

*CALCULATE THE POINTS CORRESPONDING TO DIFFERENT VALUES OF U AND W

      CALL FMT44(M,PXMAT,PMT)
      CALL FMT44(PMT,MT,PXMAT)
      CALL FMT44(M,PYMAT,PMT)
      CALL FMT44(PMT,MT,PYMAT)
      CALL FMT44(M,PZMAT,PMT)
      CALL FMT44(PMT,MT,PZMAT)
      CALL FMT44(M,HMAT,PMT)
      CALL FMT44(PMT,MT,HMAT1)

      U = 0.0

      DO 500 I = 1,NU
         W = 0.0
         CALL FUWVT(U,VU)
```

```
         CALL FMT14(VU,PXMAT,FX)
         CALL FMT14(VU,PYMAT,FY)
         CALL FMT14(VU,PZMAT,FZ)
         CALL FMT14(VU,HMAT1,FH)
         DO 400 J = 1,NW
            CALL FUWVT(W,VW)
            CALL FMT11(FX,VW,PT(I,J,1))
            CALL FMT11(FY,VW,PT(I,J,2))
            CALL FMT11(FZ,VW,PT(I,J,3))
            CALL FMT11(FH,VW,H(I,J))
            W = W + DELW
            DO 300 K = 1,3
               PT(I,J,K) = PT(I,J,K)/H(I,J)
300         CONTINUE
400      CONTINUE
         U = U + DELU
500   CONTINUE

      RETURN
      END
```

SUBROUTINE FRBMT

```
**********************************************************************
**********************************************************************
**     SUBROUTINE FRBMT(IPGN,IU,IW,MPTS,NPTS,PNTS,HO,ICOL)          **
**                                                                  **
**     PROGRAM DESCRIPTION                                          **
**                                                                  **
**     THIS SUBROUTINE WILL ASSEMBLE THE P-MATRIX TO BE USED IN THE **
**     CREATION OF A RATIONAL B-SPLINE (PERIODIC, CUBIC) SURFACE.   **
**     AFTER ASSEMBLING THE MATRIX IT WILL COMPUTE THE POINTS ON THE **
**     SURFACE.                                                     **
**                                                                  **
**                                                                  **
**     BY:       ASHIT R. GANDHI                                    **
**     DATE:     11/03/88                                           **
**                                                                  **
**     PARAMETERS USED:                                             **
**                                                                  **
**     IU     = FLAG INDICATING WHETHER SURFACE IS CLOSED AT U = 0  **
**              (INTEGER,I/P)                                       **
**              0 = OPEN                                            **
**              1 = CLOSED                                          **
**     IW     = FLAG INDICATING WHETHER SURFACE IS CLOSED AT W = 0  **
**              (INTEGER,I/P)                                       **
**              0 = OPEN                                            **
**              1 = CLOSED                                          **
**     MPTS   = NUMBER OF POINTS IN THE M-DIRECTION (INTEGER,I/P)   **
**     NPTS   = NUMBER OF POINTS IN THE N-DIRECTION (INTEGER,I/P)   **
**     PNTS   = ARRAY OF POINTS DEFINING THE SURFACE (REAL,I/P)     **
**     HO     = ARRAY OF WEIGHTS ASSIGNED TO EACH POINT (REAL,O/P)  **
**     ICOL   = COLOR TO BE ASSIGNED TO INSTANCE                    **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FRBMT(IPGN,IU,IW,MPTS,NPTS,PNTS,HO,ICOL)

      INTEGER*4  ASNUM,OBNUM,INNUM
      REAL*4 PMAT(4,4,3), PTS(0:19,0:19,3), PT(20,20,3), PNTS(20,20,3)
      REAL*4 HO(20,20),HMAT(4,4),H1(0:19,0:19),PGN(3,20000,5)
      REAL*4 PLOC(3), PORI(3)

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/REND/PLOC,PORI

      IF (IPGN .EQ. 0)THEN
         IOPGN = 1
      ELSE
         IOPGN = IPGN + 1
      ENDIF

*READ IN VALUES FOR THE RATIONAL CONTROL POINTS

      DO 300 I = 0,MPTS-1
         DO 200 J = 0,NPTS-1
            H1(I,J) = HO(I+1,J+1)
            DO 100 K = 1,3
               PTS(I,J,K) = HO(I+1,J+1)*PNTS(I+1,J+1,K)
100         CONTINUE
200      CONTINUE
300   CONTINUE

*CHECK TO SEE WHETHER THE SURFACE IS CLOSED OR OPEN

*IF SURFACE IS OPEN AT U = 0.0 AND OPEN AT W = 0.0

      IF (IU .EQ. 0 .AND. IW .EQ. 0)THEN
         SEND = MPTS - 3
```

```
         TEND = NPTS - 3
      ENDIF
*IF SURFACE IS OPEN AT U = 0.0 AND CLOSED AT W = 0.0
      IF (IU .EQ. 0 .AND. IW .EQ. 1)THEN
         SEND = MPTS - 3
         TEND = NPTS
      ENDIF
*IF SURFACE IS CLOSED AT U = 0.0 AND OPEN AT W = 0.0
      IF (IU .EQ. 1 .AND. IW .EQ. 0)THEN
         SEND = MPTS
         TEND = NPTS - 3
      ENDIF
*IF SURFACE IS CLOSED AT U = 0.0 AND CLOSED AT W = 0.0
      IF (IU .EQ. 1 .AND. IW .EQ. 1)THEN
         SEND = MPTS
         TEND = NPTS
      ENDIF
*COMPUTE ALL ELEMENTS ON THE SURFACE
*COMPUTE ELEMENTS IN THE S-DIRECTION
      DO 800 I = 1,SEND
*COMPUTE ELEMENTS IN THE T-DIRECTION
         DO 700 J = 1,TEND
*SET UP THE H-MATRIX AND P-MATRIX FOR EACH ELEMENT
            DO 600 II = 1,4
               DO 500 JJ = 1,4
                  IX = MOD((I+II-2),MPTS)
                  IY = MOD((J+JJ-2),NPTS)
                  HMAT(II,JJ) = H1(IX,IY)
                  DO 400 LL = 1,3
                     PMAT(II,JJ,LL) = PTS(IX,IY,LL)
400               CONTINUE
500            CONTINUE
600         CONTINUE
*COMPUTE ALL POINTS ON EACH SURFACE ELEMENT
            IF (NU .EQ. 0)NU = 8
            IF (NW .EQ. 0)NW = 8
            CALL FRBMH(PMAT,HMAT,NU,NW,PT)
*DRAW THE CONSTANT U AND W LINES
            CALL FDRWU(IP,NP,IS,N,NU,NW,PT)
            CALL FDRWW(IP,NP,IS,N,NU,NW,PT)
*STORE POLYGON DATA
            CALL FSTOR(IPGN,I,J,NU,NW,PT,PGN)
700      CONTINUE
800   CONTINUE
*SET UP RENDERING PARAMETERS
      CALL FCOMM(IPGN,PGN,ICOL,IOPGN)
      RETURN
11    FORMAT(1X,8(F6.2,1X))
      END
```

SUBROUTINE FRCLL

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FRCLL(I10,I21,I22,I31,I32)                         **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS ROUTINE WILL RECALL A EXISTING MODEL.                    **
**                                                                  **
**    BY:      ASHIT R. GANDHI                                      **
**    DATE:    01/13/89                                             **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    I10  : FLAG FOR EXISTENCE OF UNIT = 10                        **
```

```
**     I21  : FLAG FOR EXISTENCE OF UNIT = 21                                **
**     I22  : FLAG FOR EXISTENCE OF UNIT = 22                                **
**     I31  : FLAG FOR EXISTENCE OF UNIT = 31                                **
**     I32  : FLAG FOR EXISTENCE OF UNIT = 32                                **
**                                                                           **
*******************************************************************************
*******************************************************************************
      SUBROUTINE FRCLL(I10,I21,I22,I31,I32)

      INTEGER*4 ASNUM, OBNUM, INNUM

      REAL*4 PNTS(20,20,3), H0(20,20)
      REAL*4 SCALEF(3), LOC(3), ORI(3), MAT1(4,4), MAT2(4,4), MAT3(4,4)
      REAL*4 PLOC(3), PORI(3)

      CHARACTER*8 ASSEM(900), OBJECT(9000), INST, ADUM, ODUM

      COMMON/SCLE/SCALEF
      COMMON/POLY/IPGNR
      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/REND/PLOC,PORI

*INITIALIZE POLYGON NUMBER AND SCALE PARAMETERS

      IPGNR = 0
      IPGN = IPGNR
      SCALEF(1) = 1.0
      SCALEF(2) = 1.0
      SCALEF(3) = 1.0

      WRITE(6,*)'ENTER FRCLL'
      CALL TIMEON

*IF UNIT 31 DOES NOT EXIST SKIP THE NEXT PART

      IF (I31 .EQ. 0)THEN
         NASM = 100
         GOTO 200
      ENDIF

*READ ASSEMBLY NAME AND NUMBER INFORMATION

      DO 100 I = 1,900
         READ(31,10,END=200)NASM,ASSEM(I)
100   CONTINUE

200   BACKSPACE(31)
      ASNUM = NASM

*IF UNIT 21 DOES NOT EXIST SKIP THE NEXT PART

      IF (I21 .EQ. 0)THEN
         GOTO 400
      ENDIF

*READ ASSEMBLY LOCATION AND ORIENTATION INFORMATION

      DO 300 I = 1,ASNUM
         READ(21,10,END=400)NASM,ASSEM(I)
         READ(21,40,END=400)LOC,ORI
         CALL FURMT(ORI,MAT1)
         CALL GPTRL3(LOC,MAT2)
         CALL GPCMT3(MAT1,MAT2,MAT3)
         CALL GPOPST(NASM)
         CALL GPGLX3(MAT3,3)
         CALL GPCLST
300   CONTINUE

400   BACKSPACE(21)

*IF UNIT 32 DOES NOT EXIST SKIP THE NEXT PART

      IF (I32 .EQ. 0)THEN
         NOBJ = 1000
         GOTO 600
      ENDIF

*READ OBJECT NUMBER AND NAME INFORMATION

      DO 500 I = 1,9000
         READ(32,20,END=600)NOBJ,OBJECT(I)
         CALL GPOPST(NOBJ)
         CALL GPCLST
500   CONTINUE

600   BACKSPACE(32)
      OBNUM = NOBJ

*IF UNIT 22 DOES NOT EXIST SKIP THE NEXT PART

      IF (I22 .EQ. 0)THEN
         GOTO 800
      ENDIF
```

```
*READ OBJECT LOCATION AND ORIENTATION WITHIN AN ASSEMBLY

      DO 700 I = 1,OBNUM
         READ(22,30,END=800)NOBJ,ODUM,NASM,ADUM
         READ(22,40,END=800)LOC,ORI
         CALL FURMT(ORI,MAT1)
         CALL GPTRL3(LOC,MAT2)
         CALL GPCMT3(MAT1,MAT2,MAT3)
         CALL GPOPST(NASM)
         CALL GPMLX3(MAT3,3)
         CALL GPEXST(NOBJ)
         CALL GPCLST
700   CONTINUE

800   BACKSPACE(22)

*IF UNIT 32 DOES NOT EXIST SKIP THE NEXT PART

      IF (I10 .EQ. 0)THEN
         NINS = 0
         GOTO 1200
      ENDIF

*READ INSTANCE INFORMATION
*INFORMATION ON INSTANCE NUMBER, TYPE, PARENT OBJECT, SURFACES, COLOR,
*CONTROL POINTS AND WEIGHT INFORMATION

      DO 1100 I = 1,90000
         READ(10,50,END=1200)NINS, LTYPE, INST, NSURF, NOBJ, ODUM
         READ(10,40,END=1200)PLOC,PORI
         INNUM = NINS
         CALL FURMT(PORI,MAT1)
         CALL GPTRL3(PLOC,MAT2)
         CALL GPCMT3(MAT1,MAT2,MAT3)
         CALL GPOPST(NOBJ)
         CALL GPMLX3(MAT3,3)
         CALL GPEXST(NINS)
         CALL GPCLST
         DO 1000 J = 1, NSURF
            READ(10,*,END=1200)NINS,MPTS,NPTS,IU,IW,ICOL,NU,NW
            DO 900 K = 1,NPTS
               READ(10,*,END=1200)(PNTS(L,K,1),L=1,MPTS)
               READ(10,*,END=1200)(PNTS(L,K,2),L=1,MPTS)
               READ(10,*,END=1200)(PNTS(L,K,3),L=1,MPTS)
               READ(10,*,END=1200)(HO(L,K),L=1,MPTS)
900         CONTINUE
            CALL GPOPST(NINS)
            CALL GPPLCI(ICOL)
            CALL FRBMT(IPGN,IU,IW,MPTS,NPTS,PNTS,HO,ICOL)
            CALL GPCLST
            IPGNR = IPGN
1000     CONTINUE
         WRITE(6,*)'INSTANCE NO. = ',I
1100  CONTINUE

1200  BACKSPACE(10)

      RETURN
10    FORMAT(1X,I3,1X,A8)
20    FORMAT(1X,I4,1X,A8)
30    FORMAT(1X,2(I4,1X,A8))
40    FORMAT(1X,6(F8.3,1X))
50    FORMAT(1X,I5,1X,I2,1X,A8,1X,I2,1X,I4,1X,A8)
      END
```

SUBROUTINE FRGTV

```
************************************************************************
************************************************************************
**    SUBROUTINE FRGTV(NVIEW)                                         **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE WILL DISPLAY THE RIGHT SIDE VIEW OF ANY OBJECT. **
**                                                                    **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/07/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NST    = STRUCTURE ASSOCIATED WITH THE VIEW (INTEGER,I/P)       **
**    NVIEW = VIEW IN WHICH THE TOP VIEW IS TO BE DISPLAYED           **
**            (INTEGER,I/P)                                           **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FRGTV(NVIEW)
```

```
      REAL*4 MAT(4,4), MAT1(4,4), MAT2(4,4), SCALEF(3)
      COMMON/SCLE/SCALEF
      PI = 3.14159
      DL = -PI/2.0
*ROTATE THE VIEW ELEMENT ABOUT THE Y-AXIS BY -90 DEGREES
      CALL GPROTY(DL,MAT2)
*SCALE THE MODEL
      CALL GPSC3(SCALEF,MAT1)
*COMPOSE THE TRANSFORMATION
      CALL GPCMT3(MAT2,MAT1,MAT)
*ASSOCIATE THIS ROTATION TO THE VIEW
      CALL GPVMT3(1,NVIEW,MAT)
      CALL GPVP(1,NVIEW,2,1)
      RETURN
      END
```

SUBROUTINE FROTA

```
************************************************************************
************************************************************************
**    SUBROUTINE FROTA(NST,X,Y,Z,IDEV)                                **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE APPLIES THE CORRECT ROTATION REQUESTED ON THE   **
**    VIEW DISPLAYING  A SINGLE WINDOW                                **
**                                                                    **
**                                                                    **
**    BY:      ASHIT R. GANDHI                                        **
**    DATE:    11/25/88                                               **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NST  = NOT REQUIRED                                             **
**    X    = ROTATION ABOUT THE X AXIS (REAL,I/P)                     **
**    Y    = ROTATION ABOUT THE Y AXIS (REAL,I/P)                     **
**    Z    = ROTATION ABOUT THE Z AXIS (REAL,I/P)                     **
**    IDEV = CORRESPONDING DEVICE NUMBER (INTEGER,I/P)                **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FROTA(NST,X,Y,Z,IDEV)

      REAL MAT1(4,4), X, Y, Z, MAT(4,4), RX, RY, RZ, PI
      REAL OLDMAT(4,4)
      REAL AJUNK1(4), AJUNK2(6), AJUNK3(3)

      DATA MAT/1.0,4*0.0,1.0,4*0.0,1.0,4*0.0,1.0/
      DATA MAT1/1.0,4*0.0,1.0,4*0.0,1.0,4*0.0,1.0/
      DATA OX,OY,OZ/0.,0.,0./

      PI = 3.14159

*QUERY THE OLD TRANSFORMATION MATRIX

      CALL GPQCVX(1,7,IER,OLDMAT,AJUNK1,AJUNK2,JUNK1,AJUNK3,AJUNK4,
     1            AJUNK5,AJUNK6, JUNK2, JUNK3, JUNK4, JUNK5, JUNK6,
     2            JUNK7, JUNK8, JUNK9)

*ROTATION REQUESTED IS ABOUT THE X AXIS

      IF (IDEV .EQ. 1)THEN
         RX = PI*X/180.0
         DX = RX - OX
         OX = RX
         CALL GPROTX(DX,MAT1)
         CALL GPCMT3(MAT1,OLDMAT,MAT)
         GOTO 100
      ENDIF

*ROTATION REQUESTED IS ABOUT THE Y AXIS

      IF (IDEV .EQ. 2)THEN
         RY = PI*Y/180.0
         DY = RY - OY
         OY = RY
         CALL GPROTY(DY,MAT1)
         CALL GPCMT3(MAT1,OLDMAT,MAT)
```

```
         GOTO 100
      ENDIF

*ROTATION REQUESTED IS ABOUT THE Z AXIS

      IF (IDEV .EQ. 3)THEN
         RZ = PI*Z/180.0
         DZ = RZ - OZ
         OZ = RZ
         CALL GPROTZ(DZ,MAT1)
         CALL GPCMT3(MAT1,OLDMAT,MAT)
      ENDIF

*UPDATE THE VIEW TRANSFORMATION

100   CALL GPVMT3(1,7,MAT)

      RETURN
      END
```

SUBROUTINE FSCAN

```
************************************************************************
************************************************************************
**    SUBROUTINE FSCAN (LCHAR)                                        **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS PROGRAM ALLOWS THE USER TO INPUT A STRING FILE THAT        **
**    WILL BE PROCESSED AND SENDS THE COMMAND FOR FURTHER ACTION      **
**    TO FEATUREMOD                                                   **
**                                                                    **
**    BY:      ASHIT R. GANDHI                                        **
**    DATE:    01/13/89                                               **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    LCHAR : CHARACTER STRING INPUT BY THE USER                      **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FSCAN(LCHAR)

      CHARACTER LCHAR*72, WORD(15)*12, NAME*12, PREF*12
      CHARACTER*4 VERB(36),TERM(119),FILL(24),POSI(12),SURF(6)

      DATA NV,NT,NF,NP,NS/36,119,24,12,6/

      DATA VERB /'ADD ','COMP','CONC','CONN','COPY','CREA','DISP',
     1           'DRAW','EDIT','ENTE','ERAS','FIND','FRAM','HELP',
     1           'INCL','JOIN','LABE','LIST','MIX ','NAME','PAUS',
     1           'PLAC','PLOT','POSI','RECA','RENA','REPL','RUN ',
     1           'SAVE','SCAL','SET ','STOP','SUBT','USE ','VIEW',
     1           'ZOOM'/

      DATA TERM/'ANGL','ARC ','ARCH','AXLE','BAR ','BEAM','BEAR','BEND',
     1          'BEVE','BLOC','BOLT','BORE','BOSS','BOX ','BRAC','BRAC',
     1          'BUSH','BUTT','CAP ','CASI','CAVI','CHAM','CHAN','CHAS',
     1          'CHAS','CHOR','CLEA','CLEV','CLIP','CLOS','COLL','COLU',
     1          'CONE','CONN','CORN','COUN','COUN','CRAC','CROW','CUBE',
     1          'CUP ','CYLI','DEPR','DEPT','DISK','DOWE','DRIL','DROP',
     1          'ELLI','FACE','FAST','FILL','FIN ','FLAN','FURR','GAP ',
     1          'GASK','GRIP','GROO','HAND','HELI','HEMI','HOLE','HOOK',
     1          'JOIN','JOUR','JUG ','KERF','KEY ','KEYS','KNOB','KNUR',
     1          'LUG ','MESH','NECK','NEED','NOTC','NUT ','PAD ','PEEN',
     1          'PIN ','PIPE','PLAT','POCK','PROT','REEL','RIB ','RING',
     1          'RIVE','ROD ','ROUN','SCRE','SHAF','SLAB','SLEE','SLIT',
     1          'SLOT','SPHE','SPIK','SPIN','SPLI','SPRI','STOR','TAP ',
     1          'TCON','THRE','TOLE','TOOT','TORU','TREP','TUBE','TWIS',
     1          'UNDE','VALL','WASH','WEB ','WEDG','WELD','WHEE'/

      DATA FILL/'A   ','AN  ','AND ','AS  ','AT  ','CALL','FOR ','FROM',
     1          'IN  ','INTO','IT  ','LOCA','NAME','OF  ','ON  ','ORIE',
     1          'OUT ','PARA','THE ','THIS','TO  ','UP  ','UPTO','WITH'/

      DATA POSI/'BACK','BOTT','CENT','END ','FRON','HORI','INSI','LEFT',
     1          'RIGH','STAR','TOP ','VERT'/

      DATA SURF/'CONC','CONV','CORN','EDGE','FACE','SMOO'/

      NIN = 5
      NOUT = 6

*CONVERT STRING INPUT TO UPPER CASE

      CALL FCAPS(LCHAR)

*ERASE THE WORD ARRAY
```

```
50     DO 800 I = 1,12
          WORD(I) = ' '
800    CONTINUE

*PARSE THE LINE INPUT INTO A WORD ARRAY

       CALL FPARS(LCHAR,NW,WORD)

*CHECK FOR "STOP"

       IF (WORD(1) .EQ. 'STOP')GOTO 99

*SCAN FOR FILLER WORDS

       K = 0
       DO 200 I = 1,NW
          DO 100 J = 1,NF
             IF(WORD(I)(1:4) .EQ. FILL(J))THEN
                K = K + 1
                WORD(I) = ' '
             ENDIF
100       CONTINUE
200    CONTINUE

*REMOVE FILLER WORDS

250    DO 400 I = 1,NW-K
          IF(WORD(I) .EQ. ' ')THEN
             DO 300 J = I,NW
                WORD(J) = WORD(J+1)
300          CONTINUE
          ENDIF
          IF (WORD(I) .EQ. ' ')GOTO 250
400    CONTINUE

       NW = NW - K
       NWR = NW

       CALL FMODEL(LCHAR,NW,WORD)

       RETURN

*SCAN FIRST WORD FOR VERB, IF NO MATCH EXISTS GIVE ERROR

       IV = 0
       DO 500 J = 1,NV
          IF(WORD(1)(1:4) .EQ. VERB(J))THEN
             IV = J
             WORD(1) = VERB(J)
          ENDIF
500    CONTINUE
       IF (IV .EQ. 0)THEN
          LCHAR = WORD(1)//' ?'
          CALL FMSSG(30,LCHAR)
          LCHAR = 'ERROR: NO VERB MATCH'
          CALL FMSSG(30,LCHAR)
          LCHAR = 'RE-ENTER COMMAND STRING OR USE ANOTHER MODE OF I/P'
          CALL FMSSG(30,LCHAR)
          GOTO 99
       ENDIF

       NW = NW - 1

*SCAN SECOND WORD FOR GEOMETRICAL TERM

       IT = 0
       DO 600 J = 1,NT
          IF(WORD(2)(1:4) .EQ. TERM(J))THEN
             IT = J
             WORD(2) = TERM(J)
          ENDIF
600    CONTINUE

       IF (IT .EQ. 0)THEN
          LCHAR = 'ERROR: NO TERM MATCH'
          CALL FMSSG(30,LCHAR)
          LCHAR = WORD(2)//'?'
          CALL FMSSG(30,LCHAR)
          LCHAR = WORD(1)//'WHAT ?'
          CALL FMSSG(30,LCHAR)
          READ(NIN,30)WORD(2)
       ENDIF

*SCAN THIRD WORD FOR FEATURE NAME

625    NAME = WORD(3)
       CALL FNAME(NAME,*650)
       GOTO 675
650    LCHAR = 'ERROR: DUPLICATE NAME'
       CALL FMSSG(30,LCHAR)
       LCHAR = 'GIVE NEW NAME FOR '//WORD(2)
       CALL FMSSG(30,LCHAR)
       READ(NIN,30)WORD(3)
       GOTO 625
```

```
*SCAN FOURTH WORD FOR POSITION TERM
675     IP = 0
        DO 700 J = 1,NP
           IF(WORD(4)(1:4) .EQ. POSI(J))THEN
              IP = J
              WORD(4) = POSI(J)
           ENDIF
700     CONTINUE
        IF (IP .EQ. 0)THEN
           LCHAR = 'ERROR: NO POSITION MATCH'
           CALL FMSSG(30,LCHAR)
           LCHAR = WORD(4)//'?'
           CALL FMSSG(30,LCHAR)
           LCHAR = WORD(1)//WORD(2)//'WHERE ?'
           CALL FMSSG(30,LCHAR)
           READ(NIN,30)WORD(4)
        ENDIF

        IF(NWR .LT. 4)NWR = 4
        WRITE(NOUT,*)(WORD(I),I = 1,NWR)


99      RETURN
10      FORMAT(A72)
20      FORMAT(15(1X,A12))
30      FORMAT(A4)

        END

        SUBROUTINE FNAME(NAME)

        INTEGER FNUM
        CHARACTER NAME*12, OBJNME(200)*4

        DATA FNUM/1/

        IF(NAME(1:4) .EQ. '    ')RETURN1
        DO 100 I = 1,FNUM-1
100     IF(NAME(1:4) .EQ. OBJNME(I))RETURN1

        WRITE(6,*)FNUM,'NUM'
        OBJNME(FNUM) = NAME(1:4)

        FNUM = FNUM + 1

        RETURN
        END
```

SUBROUTINE FSCLE

```
************************************************************************
************************************************************************
**      SUBROUTINE FSCLE (SCALE)                                      **
**                                                                    **
**      PROGRAM DESCRIPTION                                           **
**                                                                    **
**      THIS ROUTINE ALLOWS A USER TO ZOOM INTO AN OBJECT THAT IS     **
**      ASSIGNED TO VIEW 7.                                           **
**                                                                    **
**                                                                    **
**      BY:       ASHIT R. GANDHI                                     **
**      DATE:     01/13/89                                            **
**                                                                    **
**      PARAMETERS USED:                                              **
**                                                                    **
**      SCALE = SCALE VALUE TO BE APPLIED TO ZOOM                     **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
        SUBROUTINE FSCLE (SCALE)

        REAL SCALE, SCLRAT(3)

        REAL OLDMAT(4,4), MAT1(4,4), MAT2(4,4)
        REAL AJUNK1(4), AJUNK2(6), AJUNK3(3)

        REAL SCALEF(3)

        COMMON /SCLE/ SCALEF

*COMPUTE NEW SCALE FACTOR

        SCLRAT(1) = SCALE/SCALEF(1)
        SCLRAT(2) = SCALE/SCALEF(2)
        SCLRAT(3) = SCALE/SCALEF(3)
```

```
*COMPUTE NEW SCALE MATRIX
      CALL GPSC3(SCLRAT, MAT1)
*INQUIRE CURRENT VIEWING MATRIX
      CALL GPQCVX (1, 7, IER, OLDMAT, AJUNK1, AJUNK2, JUNK1, AJUNK3,
     > AJUNK4, AJUNK5, AJUNK6, JUNK2, JUNK3, JUNK4, JUNK5, JUNK6,
     > JUNK7, JUNK8, JUNK9)
*COMPOSE MATRIX TO GET NEW MATRIX
      CALL GPCMT3(OLDMAT, MAT1, MAT2)
      SCALEF(1) = SCALE
      SCALEF(2) = SCALE
      SCALEF(3) = SCALE
*APPLY TRANSFORMATION TO VIEW
      CALL GPVMT3(1,7, MAT2)
      RETURN
      END
```

SUBROUTINE FSCRA

```
************************************************************************
************************************************************************
**    SUBROUTINE FSCRA                                                **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE DRAWS THE BACKGROUND SCREEN FOR "FEATURE"       **
**                                                                    **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     10/08/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NONE                                                            **
**                                                                    **
**                                                                    **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FSCRA

      REAL*4 BOX1(8),BOX2(8),BOX3(8),BOX4(8),BOX5(8),BOX6(8),BOX7(8)
      REAL*4 POS1(2),POS2(2),POS3(2),POS4(2),POS5(2)

      CHARACTER TXT1*4,TXT2*6,TXT3*1,TXT4*4,TXT5*7

*BOX FOR "MENU" AREA
      DATA BOX1/70.,89.,99.,89.,99.,99.,70.,99./

*BOX FOR TITLE WORDING BOX
      DATA BOX2/70.,-99.,99.,-99.,99.,-71.,70.,-71./

*BOX FOR SCROLL AREA
      DATA BOX3/-99.,-99.,69.,-99.,69.,-71.,-99.,-71./

*BOX FOR INPUT STRING AREA
      DATA BOX4/-99.,-70.,69.,-70.,69.,-65.,-99.,-65./

*BOX FOR PROMPT AREA
      DATA BOX5/-99.,-64.,69.,-64.,69.,-59.,-99.,-59./

*MAIN SCREEN BOX
      DATA BOX6/-99.,-58.,69.,-58.,69.,99.,-99.,99./

*BOX FOR MENU
      DATA BOX7/70.,-70.,99.,-70.,99.,88.,70.,88./

*DATA FOR TEXT AND TEXT LOCATIONS
      DATA POS1/85.,94./
      DATA TXT1/'MENU'/
      DATA POS2/85.,-84./
      DATA TXT2/'V.P.I.'/
      DATA POS3/85.,-88./
      DATA TXT3/'&'/
      DATA POS4/85.,-92./
      DATA TXT4/'S.U.'/
      DATA POS5/84.,-75./
      DATA TXT5/'FEATURE'/

      CALL GPOPST(3)
```

```
*SET UP POLYGON ATTRIBUTES

      CALL FPGAT(11,2)
      CALL FEDAT(2,1,1)

*DRAW POLYGONS REPRESENTING DIFFERENT AREAS ON THE SCREEN

      CALL GPPG2(1,4,2,BOX1)
      CALL GPPG2(1,4,2,BOX2)
      CALL GPPG2(1,4,2,BOX3)
      CALL GPPG2(1,4,2,BOX4)
      CALL GPPG2(1,4,2,BOX5)
      CALL GPPG2(1,4,2,BOX6)
      CALL GPPG2(1,4,2,BOX7)

*SET UP TEXT ATTRIBUTES AND DRAW THEM

      CALL GPTXPR(3)
      CALL GPTXAL(3,4)
      CALL GPCHH(5.)
      CALL GPTXCI(5)
      CALL GPACFO(1,1,11)
      CALL GPTXFO(11)
      CALL GPTX2(POS1,4,TXT1)
      CALL GPTXCI(3)
      CALL GPCHH(2.5)
      CALL GPTX2(POS2,6,TXT2)
      CALL GPTX2(POS3,1,TXT3)
      CALL GPTX2(POS4,4,TXT4)
      CALL GPCHH(3.5)
      CALL GPTX2(POS5,7,TXT5)

      CALL GPCLST

*ASSOCIATE STRUCTURE TO VIEW AND UPDATE WORKSTATION

      CALL GPARV(1,2,3,0.)
      CALL GPVP(1,2,1,1)
      CALL GPUPWS(1,2)

      RETURN
      END
```

SUBROUTINE FSTCP

```
************************************************************************
************************************************************************
**    SUBROUTINE FSTCP(U,W,B,P)                                       **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL CALCULATE THE CONTROL POINTS NEEDED TO CREATE **
**    A PERIODIC BICUBIC B-SPLINE SURFACE THAT PASSES THROUGH SIXTEEN **
**    POINTS SPECIFIED BY THE USER.                                   **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    U = ARRAY OF PARAMETER (U) VALUES CORRESPONDING TO INPUT POINTS **
**        (REAL,I/P)                                                  **
**    W = ARRAY OF PARAMETER (W) VALUES CORRESPONDING TO INPUT POINTS **
**        (REAL,I/P)                                                  **
**    B = ARRAY OF POINTS THROUGH WHICH THE B-SPLINE CURVE SHOULD PASS**
**        (REAL,I/P)                                                  **
**    P = ARRAY OF CONTROL POINTS NEEDED TO GET THE SPECIFIED CURVE   **
**        (REAL,O/P)                                                  **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE FSTCP(U,W,B,P)

      REAL*4 M(4,4), MT(4,4)
      REAL*4 U(4), VU(4), W(4), VW(4), UM(4,4), WMT(4,4)
      REAL*4 AU(4,4), AUINV(4,4), AW(4,4), AWINV(4,4)
      REAL*4 B(4,4,3), P(20,20,3)

      REAL*4 B1(4,4),B2(4,4),B3(4,4)
      REAL*4 P1(4,4),P2(4,4),P3(4,4),P4(4,4)

      DATA M /-1.0, 3.0,-3.0, 1.0,
     2         3.0,-6.0, 0.0, 4.0,
     3        -3.0, 3.0, 3.0, 1.0,
     4         1.0, 0.0, 0.0, 0.0/

      DATA MT/-1.0, 3.0,-3.0, 1.0,
     2         3.0,-6.0, 3.0, 0.0,
     3        -3.0, 0.0, 3.0, 0.0,
     4         1.0, 4.0, 1.0, 0.0/
```

```
*FOR EACH VALUE OF U AND W COMPUTE THE PARAMETER VECTOR AND ASSEMBLE THE
*COMPLETE 4X4 MATRICES

      DO 200 I = 1,4
         CALL FUWVT(U(I),VU)
         CALL FUWVT(W(I),VW)
         DO 100 J = 1,4
            UM(I,J) = VU(J)
            WMT(J,I) = VW(J)
100      CONTINUE
200   CONTINUE

*MULTIPLY MATRIX(UM) AND MATRIX(M) TO GET MATRIX(AU) AND THEN SCALE
*MATRIX(AU)
*MULTIPLY MATRIX(MT) AND MATRIX(MW) TO GET MATRIX(AW) AND THEN SCALE
*MATRIX(AW)

      CALL FMT44(UM,M,AU)
      CALL FMT44(MT,WMT,AW)

      DO 400 I = 1,4
         DO 300 J = 1,4
            AU(I,J) = AU(I,J)*27.0
            AW(I,J) = AW(I,J)*27.0
300      CONTINUE
400   CONTINUE

*COMPUTE THE CLOSED FORM INVERSE OF MATRIX(AU) AND SCALE MATRIX(AUINV)
*COMPUTE THE CLOSED FORM INVERSE OF MATRIX(AW) AND SCALE MATRIX(AWINV)

      CALL MATINV(AU,AUINV)
      CALL MATINV(AW,AWINV)

      DO 600 I = 1,4
         DO 500 J = 1,4
            AUINV(I,J) = AUINV(I,J)*162.0
            AWINV(I,J) = AWINV(I,J)*162.0
500      CONTINUE
600   CONTINUE

*ASSIGN APPROPRIATE VALUES OF COORDINATES TO THE ARRAY

      DO 800 I = 1,4
         DO 700 J = 1,4
            B1(I,J) = B(I,J,1)
            B2(I,J) = B(I,J,2)
            B3(I,J) = B(I,J,3)
700      CONTINUE
800   CONTINUE

*COMPUTE THE CONTROL POINTS COORDINATES

      CALL FMT44(AUINV,B1,P4)
      CALL FMT44(P4,AWINV,P1)
      CALL FMT44(AUINV,B2,P4)
      CALL FMT44(P4,AWINV,P2)
      CALL FMT44(AUINV,B3,P4)
      CALL FMT44(P4,AWINV,P3)

*ASSIGN APPROPRIATE VALUES OF COORDINATES TO THE ARRAY

      DO 1000 I = 1,4
         DO 900 J = 1,4
            P(I,J,1) = P1(I,J)
            P(I,J,2) = P2(I,J)
            P(I,J,3) = P3(I,J)
900      CONTINUE
1000  CONTINUE

      RETURN
      END
```

SUBROUTINE FSTOR

```
************************************************************************
************************************************************************
**    SUBROUTINE FSTOR(IPGN,I,J,NU,NW,PT,PGN)                         **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE WILL TAKE ALL POINTS EXISTING ON A PATCH AND    **
**    STORE THEM SO THAT A POLYGON COULD PASS THROUGH THEM.           **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     10/24/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**
```

```
**     I    = THE PATCH NUMBER (INTEGER,I/P)                              **
**     J    = SURFACE  NUMBER (INTEGER,I/P)                               **
**     NU   = NUMBER OF LINES IN THE U DIRECTION (INTEGER,I/P)            **
**     NW   = NUMBER OF LINES IN THE W DIRECTION (INTEGER,I/P)            **
**     PT   = POINTS IN THE PATCH (REAL(NU,NW,3),I/P)                     **
**     PGN = POINTS REPRESENTING THE POLYGON (REAL(3,20000,5),O/P)        **
**                                                                        **
************************************************************************
************************************************************************
      SUBROUTINE FSTOR(IPGN,I,J,NU,NW,PT,PGN)

      INTEGER*4 I,J,NU,NW

      REAL*4 PT(20,20,3), PGN(3,20000,5), PTS(3)
      REAL*4 PLOC(3), PORI(3)

      COMMON/REND/PLOC,PORI

*COMPUTE 4 END POINTS FOR EACH POLYGON

      DO 600 K = 1,NU - 1

*FIRST TRAVERSE THE W DIRECTION TO CREATE (NW-1) POLYGONS

         DO 500 L = 1,NW - 1

*INCREMENT POLYGON NUMBER

            IPGN = IPGN + 1

*COMPUTE THE FIRST TWO POINTS DEFINING THE POLYGON

            DO 200 I1 = 1,2

               IK = K + I1 - 1
               IL = L

*TRAVERSE X,Y AND Z COORDINATES

               PTS(1) = PT(IL,IK,1)
               PTS(2) = PT(IL,IK,2)
               PTS(3) = PT(IL,IK,3)
               CALL FTRNS(PLOC,PORI,PTS)
               DO 100 I2 = 1,3
                  PGN(I2,IPGN,I1) = PTS(I2)
100            CONTINUE

200         CONTINUE

*COMPUTE THE LAST TWO POINTS DEFINING THE POLYGON

            DO 400 I1 = 3,4

               IK = K - I1 + 4
               IL = L + 1

*TRAVERSE X,Y AND Z COORDINATES

               PTS(1) = PT(IL,IK,1)
               PTS(2) = PT(IL,IK,2)
               PTS(3) = PT(IL,IK,3)
               CALL FTRNS(PLOC,PORI,PTS)
               DO 300 I2 = 1,3
                  PGN(I2,IPGN,I1) = PTS(I2)
300            CONTINUE

400         CONTINUE

500      CONTINUE

600   CONTINUE

      RETURN
      END
```

SUBROUTINE FSTRG

```
************************************************************************
************************************************************************
**     SUBROUTINE FSTRG(STRG)                                         **
**                                                                    **
**     PROGRAM DESCRIPTION                                            **
**                                                                    **
**     THIS SUBROUTINE GETS A STRING INPUT FROM THE USER. THE STRING  **
**     INPUT IS TAKEN FROM AN ACTIVATED AREA RESERVED FOR SUCH INPUT  **
**                                                                    **
**     BY:       ASHIT R. GANDHI                                      **
**     DATE:     10/08/88                                             **
**                                                                    **
**     PARAMETERS USED:                                               **
```

```
**                                                                      **
**     STRG = STRING INPUT (CHARACTER, O/P)                             **
**                                                                      **
**********************************************************************
**********************************************************************
      SUBROUTINE FSTRG(STRG)

      COMMON CSIZE

      INTEGER*4 L

      REAL*4 CSIZE(3), AREA(6)

      CHARACTER STRG*50, EMTY*50, PREF*9, MSSG*59

      DATA EMTY/'                                                  '/
      STRG = '

*SETUP STRING INPUT AREA

      AREA(1)=0.007*CSIZE(1)
      AREA(2)=0.845*CSIZE(1)
      AREA(3)=0.15*CSIZE(2)
      AREA(4)=0.17*CSIZE(2)
      AREA(5)=0.0
      AREA(6)=CSIZE(3)

*INITIALIZE THE STRING DEVICE

      CALL GPSTMO(1,1,1,2)

*INITALIZE THE STRING DEVICE

      CALL GPINST(1,1,50,EMTY,1,AREA,50,1,0,EMTY)

*PLACE STRING DEVICE IN THE REQUEST MODE

      CALL GPSTMO(1,1,3,2)

*PROCESS THE STRING EVENT

*AWAIT EVENT

 3    CALL GPAWEV(1000.,I,ICLA,IDEV)

*GET STRING

      L=50
      CALL GPGTST(L,LR,STRG)
      IF(STRG.EQ.EMTY)GOTO 3

*RETURN STRING

      PREF = 'STRING = '
      MSSG = PREF//STRG
      CALL FMSSG(30,MSSG)
      CALL GPVP(1,6,2,1)

      RETURN
      END
```

SUBROUTINE FTITL

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FTITL                                                **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS SUBROUTINE SETS UP AND DRAWS THE TITLE SCREEN FOR FEATURE. **
**                                                                    **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     10/08/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    NONE                                                            **
**                                                                    **
**                                                                    **
**                                                                    **
**********************************************************************
**********************************************************************
      SUBROUTINE FTITL

      INTEGER*4 LNT1, LNT2, LNT3

      REAL*4 POS1(2), POS2(2), POS3(2), PTS(12)
```

```
      CHARACTER TXT1*7, TXT2*13, TXT3*18

      DATA POS1/0.,13./
      DATA LNT1/7/
      DATA TXT1/'FEATURE'/
      DATA POS2/0.,-10./
      DATA LNT2/13/
      DATA TXT2/'FEATURE BASED'/
      DATA POS3/0.,-14./
      DATA LNT3/18/
      DATA TXT3/'GEOMETRIC MODELING'/

      DATA PTS/-19.7,-19.7,0.,19.7,-19.7,0.,19.7,19.7,0.,-19.7,19.7,0./

*OPEN STRUCTURE

      CALL GPOPST(1)

*SET UP BACKGROUND

      CALL FPGAT(11,2)
      CALL FEDAT(2,1,1)
      CALL GPPG3(1,4,3,PTS)

*SET UP TEXT ATTRIBUTES

      CALL GPTXPR(3)
      CALL GPTXAL(3,4)
      CALL GPCHH(4.75)
      CALL GPTXCI(7)
      CALL GPACFO(1,1,11)
      CALL GPTXFO(11)

*DRAW TEXT NO. 1

      CALL GPTX2(POS1,LNT1,TXT1)

*SET UP ATTRIBUTES FOR OTHER TEXT

      CALL GPCHH(1.5)
      CALL GPTXCI(2)
      CALL GPACFO(1,1,3)
      CALL GPTXFO(3)

*DRAW TEXT NOS. 2 AND 3

      CALL GPTX2(POS2,LNT2,TXT2)
      CALL GPTX2(POS3,LNT3,TXT3)

*CLOSE STRUCTURE

      CALL GPCLST

*ASSOCIATE STRUCTURE TO VIEW

      CALL GPARV(1,1,1,0.)

*DRAW TITLE SUPER CONICS

      DO 300 J = 1,1
         DO 100 I = 1,26
            RN = (I - 1)/12.5*2.0
            CALL GPDLST(2)
            CALL FSUCO(RN)
            CALL GPARV(1,1,2,0.)
            CALL GPUPWS(1,2)
100      CONTINUE
         DO 200 I = 26,1,-1
            RN = (I - 1)/12.5*2.0
            CALL GPDLST(2)
            CALL FSUCO(RN)
            CALL GPARV(1,1,2,0.)
            CALL GPUPWS(1,2)
200      CONTINUE
300   CONTINUE

*DEACTIVATE THE VIEW

      CALL GPVCH(1,1,2,1,1,2,11,2,1,1)

      RETURN
      END
```

SUBROUTINE FTOPV

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FTOPV(NVIEW)                                       **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
```

```
**                                                                     **
**    THIS SUBROUTINE WILL DISPLAY THE TOP VIEW OF ANY OBJECT.         **
**                                                                     **
**                                                                     **
**                                                                     **
**    BY:       ASHIT R. GANDHI                                      **
**    DATE:     11/07/88                                             **
**                                                                     **
**    PARAMETERS USED:                                               **
**                                                                     **
**    NVIEW = VIEW IN WHICH THE TOP VIEW IS TO BE DISPLAYED          **
**            (INTEGER,I/P)                                          **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FTOPV(NVIEW)

      REAL*4 MAT(4,4), MAT1(4,4), MAT2(4,4), SCALEF(3)

      COMMON /SCLE/SCALEF

      PI = 3.14159
      DL = PI/2.0

*ROTATE THE VIEW ELEMENT ABOUT THE X-AXIS BY +90 DEGREES

      CALL GPROTX(DL,MAT2)

*SCALE THE MODEL

      CALL GPSC3(SCALEF,MAT1)

*COMPOSE THE TRANSFORMATION

      CALL GPCMT3(MAT2,MAT1,MAT)

*ASSOCIATE THIS ROTATION TO THE VIEW

      CALL GPVMT3(1,NVIEW,MAT)

      CALL GPVP(1,NVIEW,2,1)

      RETURN
      END
```

SUBROUTINE FTRNS

```
*************************************************************************
*************************************************************************
**    SUBROUTINE FTRNS(LOC,DCS,PTS)                                    **
**                                                                     **
**    PROGRAM DESCRIPTION                                              **
**                                                                     **
**    THIS SUBROUTINE WILL TRANSFORM THE COORDINATES OF A POINT IN     **
**    THE X-O-Y PLANE TO THE CORRESPONDING POINT ON AN ARBITRARY       **
**    PLANE.  THE VARIABLE CONTAINING THE OLD LOCATION OF THE POINT    **
**    IS RETURNED WITH THE NEW VALUES.                                 **
**                                                                     **
**    BY:       ASHIT R. GANDHI                                        **
**    DATE:     10/08/88                                               **
**                                                                     **
**    PARAMETERS USED:                                                 **
**                                                                     **
**    LOC = LOCATION OF THE ORIGIN OF NEW PLANE (REAL(3),I/P)          **
**    DCS = ANGULAR ROTATIONS OF PLANE WITH RESPECT TO THE AXES.       **
**          (INTGER(3),I/P)                                            **
**    PTS = LOCATION OF THE POINT ON THE OLD (NEW) PLANE.              **
**          (REAL(3),I/P,O/P)                                          **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FTRNS(LOC,DCS,PTS)

      REAL*4 DCS(3)

      REAL*4 LOC(3), PTS(3), R(3,3), V(3), ROT(3)

      PI = 3.14159

*CONVERT THE ROTATIONS FROM ANGLE TO RADIANS

      DO 100 I = 1,3
         ROT(I) = DCS(I)*PI/180.
100   CONTINUE

*CALCULATE THE UNIVERSAL ROTATION MATRIX

      CZ = COS(ROT(3))
      SZ = SIN(ROT(3))
      CY = COS(ROT(2))
```

```
      SY = SIN(ROT(2))
      CX = COS(ROT(1))
      SX = SIN(ROT(1))

      R(1,1) =   CY*CZ
      R(1,2) =   -CY*SZ
      R(1,3) =   SY
      R(2,1) =   SX*SY*CZ + CX*SZ
      R(2,2) =   -SX*SY*SZ + CX*CZ
      R(2,3) =   -SX*CY
      R(3,1) =   -CX*SY*CZ + SX*SZ
      R(3,2) =   CX*SY*SZ + SX*CZ
      R(3,3) =   CX*CY

*MATRIX MULTIPLICATION TO GET NEW ROTATED COORDINATES

      DO 300 I = 1,3
         V(I) = 0.0
         DO 200 J = 1,3
            V(I) = V(I) + R(I,J)*PTS(J)
200      CONTINUE
300   CONTINUE

*CALCULATE THE TRANSLATIONS AND RETURN VALUES TO OLD VARIABLE

         DO 400 I = 1,3
            PTS(I) = V(I) + LOC(I)
400      CONTINUE

      RETURN
      END
```

SUBROUTINE FUWVT

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FUWVT(RV,V)                                        **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS ROUTINE CREATES THE U AND W VECTORS FOR DIFFERENT VALUES  **
**    OF THE PARAMETER                                              **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     10/08/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    RV   = SCALER VALUE OF PARAMETER (REAL,I/P)                   **
**    V    = 4X1 MATRIX (REAL,O/P)                                  **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE FUWVT(RV,V)

      REAL RV, V(4)

      V(1) = RV**3
      V(2) = RV**2
      V(3) = RV
      V(4) = 1.0

      RETURN
      END
```

SUBROUTINE FWIND

```
**********************************************************************
**********************************************************************
**    SUBROUTINE FWIND                                              **
**                                                                  **
**    SUBROUTINE DESCRIPTION                                        **
**                                                                  **
**    THIS SUBROUTINE SETS UP THE WINDOWS AND VIEWPORTS.            **
**                                                                  **
**    BY:   ASHIT R. GANDHI                                         **
**    DATE: 10/07/88                                                **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    WIN1    : ARRAY CONTAINING WINDOW COORDINATES                 **
**    WIN2    : ARRAY CONTAINING WINDOW COORDINATES                 **
**    WIN3    : ARRAY CONTAINING WINDOW COORDINATES                 **
**    WIN4    : ARRAY CONTAINING WINDOW COORDINATES                 **
```

```
**     WIN5   : ARRAY CONTAINING WINDOW COORDINATES                    **
**     WIN6   : ARRAY CONTAINING WINDOW COORDINATES                    **
**     WIN7   : ARRAY CONTAINING WINDOW COORDINATES                    **
**     VPT1   : ARRAY CONTAINING VIEWPORT COORDINATES                  **
**     VPT2   : ARRAY CONTAINING VIEWPORT COORDINATES                  **
**     VPT3   : ARRAY CONTAINING VIEWPORT COORDINATES                  **
**     VPT4   : ARRAY CONTAINING VIEWPORT COORDINATES                  **
**     VPT5   : ARRAY CONTAINING VIEWPORT COORDINATES                  **
**     VPT6   : ARRAY CONTAINING VIEWPORT COORDINATES                  **
**     VPT7   : ARRAY CONTAINING VIEWPORT COORDINATES                  **
**     PT     : ARRAY CONTAINING COORDINATES OF VIEWING                **
**                                                                     **
*************************************************************************
*************************************************************************
      SUBROUTINE FWIND

      REAL*4  WIN1(4), VPT1(6), PT1(3), WIN2(4), WIN3(4), VPT3(4)
      REAL*4  WIN4(4), VPT4(4), WIN5(4), VPT5(4), WIN6(4), VPT6(4)
      REAL*4  WIN7(4), VPT7(6), WIN8(4), VPT8(6), WIN9(4), VPT9(6)
      REAL*4  WIN10(4), VPT10(6), WIN11(4), VPT11(6)

*WINDOW AND VIEWPORT FOR TITLE SCREEN

      DATA WIN1/-20.0,20.0,-20.0,20.0/
      DATA VPT1/0.,1.,0.,1.,0.,1./
      DATA PT1 /0.,0.,300./

*WINDOW AND VIEWPORT FOR SCREEN "A"

      DATA WIN2/-100.,100.,-100.,100./

*WINDOW AND VIEWPORT FOR PROMPT AREA

      DATA WIN3/0.,1.,0.,1./
      DATA VPT3/.005,.845,.18,.205/

*WINDOW AND VIEWPORT FOR STRING INPUT AREA

      DATA WIN6/0.,1.,0.,1./
      DATA VPT6/.005,.845,.15,.175/

*WINDOW AND VIEWPORT FOR SCROLL AREA

      DATA WIN5/0.,1.,0.,1./
      DATA VPT5/.005,.845,.005,.145/

*WINDOW AND VIEWPORT FOR MENU AREA

      DATA WIN4/0.,1.,0.,1./
      DATA VPT4/.85,.995,.15,.940/

*WINDOW AND VIEWPORT FOR DISPLAY AREA

      DATA WIN7/-100.,100.,-100.,100./
      DATA VPT7/.005,.845,.21,.995,0.,1./

*WINDOW AND VIEWPORT FOR MULTIPLE DISPLAY AREA

*FRONT VIEW

      DATA WIN8/-100.,100.,-100.,100./
      DATA VPT8/.005,.425,.21,.6025,0.,1./

*TOP VIEW

      DATA WIN9/-100.,100.,-100.,100./
      DATA VPT9/.005,.425,.6025,.995,0.,1./

*RIGHT SIDE VIEW

      DATA WIN10/-100.,100.,-100.,100./
      DATA VPT10/.425,.845,.21,.6025,0.,1./

*ISOMETRIC VIEW

      DATA WIN11/-100.,100.,-100.,100./
      DATA VPT11/.425,.845,.6025,.995,0.,1./

*SET UP WINDOWS AND VIEWPORTS

      CALL GPVMP3(1,1,WIN1,VPT1,1,PT1,20,200.0,-200.0)
      CALL GPVMP3(1,2,WIN2,VPT1,1,PT1,20,200.0,-200.0)
      CALL GPVMP2(1,3,WIN3,VPT3)
      CALL GPVMP2(1,4,WIN4,VPT4)
      CALL GPVMP2(1,5,WIN5,VPT5)
      CALL GPVMP2(1,6,WIN6,VPT6)
      CALL GPVMP3(1,7,WIN7,VPT7,1,PT1,20,200.0,-200.0)
      CALL GPVMP3(1,8,WIN8,VPT8,1,PT1,20,200.0,-200.0)
      CALL GPVMP3(1,9,WIN9,VPT9,1,PT1,20,200.0,-200.0)
      CALL GPVMP3(1,10,WIN10,VPT10,1,PT1,20,200.0,-200.0)
      CALL GPVMP3(1,11,WIN11,VPT11,1,PT1,20,200.0,-200.0)

*ACTIVATE THE VIEWS

      CALL GPVCH(1,1,1,1,1,2,11,2,1,2)
      CALL GPVCH(1,2,1,1,1,2,11,2,1,2)
```

```
      CALL GPVCH(1,3,1,1,1,2,11,2,1,2)
      CALL GPVCH(1,4,1,1,1,2,11,2,1,2)
      CALL GPVCH(1,5,1,1,1,2,11,2,1,2)
      CALL GPVCH(1,6,1,1,1,2,11,2,1,2)
      CALL GPVCH(1,7,2,1,1,2,11,2,1,2)
      CALL GPVCH(1,8,2,1,1,2,11,2,1,2)
      CALL GPVCH(1,9,2,1,1,2,11,2,1,2)
      CALL GPVCH(1,10,2,1,1,2,11,2,1,2)
      CALL GPVCH(1,11,2,1,1,2,11,2,1,2)

      RETURN
      END
```

SUBROUTINE HEMIS

```
************************************************************************
************************************************************************
**    SUBROUTINE HEMIS(RX,IFL,LOC,ORI,ICOL,BOOL)                      **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE COMPUTES THE CONTROL POINTS THAT WOULD BE          **
**    REQUIRED TO CREATE A HEMISPHERE USING B-SPLINE SURFACES         **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/21/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    RX    = RADIUS OF THE HEMISPHERE                                **
**    IFL   = FLAG FOR NEW FEATURE                                    **
**    ORI   = ORIENTATION OF HEMISPHERE                               **
**    LOC   = LOCATION OF THE SPHERE (REAL,I/P)                       **
**    ICOL  = COLOR FOR HEMISPSHERE  (REAL,I/P)                       **
**    BOOL  = BOOLEAN TYPE FOR HEMISPHERE                             **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE HEMIS(RX,IFL,LOC,ORI,ICOL,BOOL)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 LOC(3), PT(3), PTS(4,4,3), ORI(3)
      REAL*4 U(4), W(4), PNTS(20,20,3), H0(20,20), XS(2), YS(4), ZS(4)
      REAL*4 RX, RY
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA U/0.0,.33333333,.66666667,1.0/
      DATA W/0.0,.33333333,.66666667,1.0/
      DATA H0/400*1.0/
      DATA XS/1.0,-1.0/
      DATA YS/1.0,1.0,-1.0,-1.0/
      DATA ZS/1.0,-1.0,-1.0,1.0/

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      PI = 3.14159

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      INNUM = INNUM + 1

      IF (IFL .EQ. 2)THEN
         NSURF = 8
      ELSE
         NSURF = 4
      ENDIF

      CALL FDINS(INNUM,LTYPE,'HEMISPHE',NSURF,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

      DELTA = PI/6.0

      DO 600 IK = 1,1
         DO 300 IJ = 1,4
            THETA = 0.0
```

```
            DO 200 I = 1,4
               ALPHA = 0.0
               DO 100 J = 1,4
                  PT(1) = RX*COS(THETA)*COS(ALPHA)*XS(IK)
                  PT(2) = RX*SIN(THETA)*YS(IJ)
                  PT(3) = RX*COS(THETA)*SIN(ALPHA)*ZS(IJ)
                  PTS(I,J,1) = PT(1)
                  PTS(I,J,2) = PT(2)
                  PTS(I,J,3) = PT(3)
                  ALPHA = ALPHA + DELTA
100            CONTINUE
               THETA = THETA + DELTA
200         CONTINUE
            CALL FSTCP(U,W,PTS,PNTS)
            CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

            WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
            DO 250 I = 1,4
               WRITE(10,*)(PNTS(I,J,1),J=1,4)
               WRITE(10,*)(PNTS(I,J,2),J=1,4)
               WRITE(10,*)(PNTS(I,J,3),J=1,4)
               WRITE(10,*)(H0(1,J),J=1,4)
250         CONTINUE
300      CONTINUE
600   CONTINUE

      ICHK = 1
      IF (IFL .EQ. 2)CALL RING(0.0,RX,LOC,ORI,ICOL,BOOL,ICHK,0.)

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE MBUSH

```
**********************************************************************
**********************************************************************
**    SUBROUTINE MBUSH(R1,R2,THK,LOCA,ORI,COLOR,BOOL)                 **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING  **
**    A BUSHING IN ITS CORRECT LOCATION AND ORIENTATION               **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    R1     = INNER RADIUS                                           **
**    R2     = OUTER RADIUS                                           **
**    THK    = THICKNESS OF THE BUSHING                               **
**    LOCA   = LOCATION OF THE BUSHING                                **
**    ORI    = ORIENTATION OF THE BUSHING                             **
**    COLOR  = COLOR TO BE GIVEN TO THE BUSHING                       **
**    BOOL   = BOOLEAN OPERATOR FOR THE BUSHING                       **
**                                                                    **
**********************************************************************
**********************************************************************
      SUBROUTINE MBUSH(R1,R2,THK,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      REAL*4 R1, R2
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2
      CHARACTER CR1*7, CR2*7, CTHK*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      R1 = 1.0
      R2 = 2.0
      THK = 1.0
      NU = 4
      NW = 4
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL    = '+'
      COLOR   =  2
```

```
*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE
      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)
*************************************************************************
*        PREPARE FOR PICK INPUT                                         *
*************************************************************************
*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,1,2)
*DEFINE PICK AREAS AND INITIALIZE PICKS
      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*OPEN STRUCTURE
50    CALL GPEST(7)
      CALL GPOPST(7)
      CALL GPADCN(1,2)
*SET TEXT COLOR TO YELLOW
      CALL GPTXCI(5)
*SET ANNOTATION TEXT SIZE SCALE FACTOR
      CALL GPAHSC(0.80)
*CONVERT NUMBERS TO TEXT
      WRITE(CR1,'(F7.2)')R1
      WRITE(CR2,'(F7.2)')R2
      WRITE(CTHK,'(F7.2)')THK
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)
*DRAW ALL TITLE TEXT
      POS(1) = 0.62
      POS(2) = 0.80
      STRG = 'BUSH  = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)
      POS(1) = 0.62
      POS(2) = 0.67
      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)
      POS(1) = 0.62
      POS(2) = 0.54
      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)
      POS(1) = 0.62
      POS(2) = 0.41
      STRG = 'ORIENT =   '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)
      POS(1) = 0.62
      POS(2) = 0.28
      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)
      POS(1) = 0.62
```

```
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU     = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW     = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'R1     = '//CR1
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'R2     = '//CR2
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.54

      STRG = 'THICK  = '//CTHK
      CALL GPPKID(11)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8,9,10,11)IPKID

1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &   .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
```

```
4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &   .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

5     CALL BUSH(R1,R2,THK,LOCA,ORI,COLOR,BOOL)

6     GOTO 200

7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF(INU. GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF(INW. GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

9     CALL FGTST(CR1)
      IF (CR1(1:1) .GE. '0' .AND. CR1(1:1) .LE. '9')THEN
         READ(CR1,*)R1
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

10    CALL FGTST(CR2)
      IF (CR2(1:1) .GE. '0' .AND. CR2(1:1) .LE. '9')THEN
         READ(CR2,*)R2
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

11    CALL FGTST(CTHK)
      IF (CTHK(1:1) .GE. '0' .AND. CTHK(1:1) .LE. '9')THEN
         READ(CTHK,*)THK
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
      END
```

SUBROUTINE MCONE

```
**********************************************************************
**********************************************************************
**    SUBROUTINE MCONE(RS,LEN,LOCA,ORI,COLOR,BOOL)                  **
**                                                                  **
```

```
**     PROGRAM DESCRIPTION                                                    **
**                                                                            **
**     THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING         **
**     A CONE IN ITS CORRECT LOCATION AND ORIENTATION                         **
**                                                                            **
**                                                                            **
**     BY:       ASHIT R. GANDHI                                              **
**     DATE:     11/12/88                                                     **
**                                                                            **
**     PARAMETERS USED:                                                       **
**                                                                            **
**     RS     = RADIUS OF THE CONE                                          **
**     LEN    = LENGTH OF THE CONE                                          **
**     LOCA   = LOCATION OF THE CONE                                        **
**     ORI    = ORIENTATION OF THE CONE                                     **
**     COLOR  = COLOR TO BE GIVEN TO THE CONE                               **
**     BOOL   = BOOLEAN OPERATOR FOR THE CONE                               **
**                                                                            **
********************************************************************************
********************************************************************************
      SUBROUTINE MCONE(RS,LEN,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      REAL*4 RS, LEN, AR
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2, CTYPE*7
      CHARACTER CRS*7, CLEN*7, CAR*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      RS = 1.0
      LEN = 5.0
      AR = 0.0
      CTYPE = 'NO CAPS'
      IFL = 1
      NU = 4
      NW = 4
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL   = '+'
      COLOR  =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

********************************************************************************
*       PREPARE FOR PICK INPUT                                                 *
********************************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

*DRAW ALL TITLE TEXT
```

```
POS(1) = 0.62
POS(2) = 0.80

STRG = 'CONE  = '//BOOL
CALL GPPKID(1)
CALL GPAN2(POS,9,STRG)

WRITE(CRS,'(F7.2)')RS
WRITE(CLEN,'(F7.2)')LEN
WRITE(CAR,'(F7.2)')AR
WRITE(CNU,'(I2)')NU
WRITE(CNW,'(I2)')NW
WRITE(CLOCA1,'(F7.2)')LOCA(1)
WRITE(CLOCA2,'(F7.2)')LOCA(2)
WRITE(CLOCA3,'(F7.2)')LOCA(3)
WRITE(CORI1,'(F7.2)')ORI(1)
WRITE(CORI2,'(F7.2)')ORI(2)
WRITE(CORI3,'(F7.2)')ORI(3)

POS(1) = 0.62
POS(2) = 0.67

WRITE(CCOLOR,'(I2)')COLOR
STRG = 'COLOR = '//CCOLOR
CALL GPPKID(2)
CALL GPAN2(POS,29,STRG)

POS(1) = 0.62
POS(2) = 0.54

STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
CALL GPPKID(3)
CALL GPAN2(POS,32,STRG)

POS(1) = 0.62
POS(2) = 0.41

STRG = 'ORIENT =   '//CORI1//CORI2//CORI3
CALL GPPKID(4)
CALL GPAN2(POS,32,STRG)

POS(1) = 0.62
POS(2) = 0.28

STRG = 'EXECUTE'
CALL GPPKID(5)
CALL GPAN2(POS,7,STRG)

POS(1) = 0.62
POS(2) = 0.15

STRG = 'ABORT'
CALL GPPKID(6)
CALL GPAN2(POS,5,STRG)

POS(1) = 0.32
POS(2) = 0.80

STRG = 'NU     = '//CNU
CALL GPPKID(7)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.32
POS(2) = 0.67

STRG = 'NW     = '//CNW
CALL GPPKID(8)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.80

STRG = 'RADIUS = '//CRS
CALL GPPKID(9)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.67

STRG = 'LENGTH = '//CLEN
CALL GPPKID(10)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.54

STRG = 'A_RATIO ='//CAR
CALL GPPKID(11)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.41

STRG = 'TYPE   = '//CTYPE
CALL GPPKID(12)
CALL GPAN2(POS,16,STRG)
```

```
      CALL GPCLST
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,5)
*ASSOCIATE MENU TO VIEW 5
      CALL GPARV(1,5,7,0)
*UPDATE WORKSTATION
      CALL FPRMP(14)
      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)
*AWAIT FOR A PICK
100   CALL GPAWEV(1.,1,ICLASS,IDEV)
      IF (ICLASS .NE. 5)GOTO 100
      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)
      GOTO (1,2,3,4,5,6,7,8,9,10,11,12)IPKID
1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50
2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF
3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &    .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &    .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
5     CALL CONE(IFL,RS,LEN,AR,LOCA,ORI,COLOR,BOOL,0,0.)
6     GOTO 200
7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF ( INU .GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF ( INW .GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
9     CALL FGTST(CRS)
      IF (CRS(1:1) .GE. '0' .AND. CRS(1:1) .LE. '9')THEN
         READ(CRS,*)RS
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
10    CALL FGTST(CLEN)
```

```
      IF (CLEN(1:1) .GE. '0' .AND. CLEN(1:1) .LE. '9')THEN
         READ(CLEN,*)LEN
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

11    CALL FGTST(CAR)
      IF (CAR(1:1) .GE. '0' .AND. CAR(1:1) .LE. '9')THEN
         READ(CAR,*)AR
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

12    IF (CTYPE .EQ. 'NO CAPS' )THEN
         CTYPE = ' CAPPED'
         IFL = 2
         GOTO 50
      ELSE
         CTYPE = 'NO CAPS'
         IFL = 1
         GOTO 50
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
      END
```

SUBROUTINE MCOUNT

```
************************************************************************
************************************************************************
**    SUBROUTINE MCOUNT(R1,R2,L1,L2,IFL,IFL1,LOCA,ORI,COLOR,BOOL)     **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING  **
**    A COUNTER BORE OR A COUNTER SINK IN ITS CORRECT LOCATION         **
**    AND ORIENTATION                                                 **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    R1    = RADIUS AT TOP OF FEATURE                                **
**    R2    = INTERMEDIATE RADIUS OF FEATURE                          **
**    L1    = LENGTH BETWEEN TOP AND INTERMEDIATE RADIUS              **
**    L2    = LENGTH OF FEATURE                                       **
**    IFL   = FLAG INDICATING CAPS FOR COUNTER BORE                   **
**    IFL1  = FLAG INDICATING CAPS FOR COUNTER SINK                   **
**    LOCA  = LOCATION OF THE FEATURE                                 **
**    ORI   = ORIENTATION OF THE FEATURE                              **
**    COLOR = COLOR TO BE GIVEN TO THE FEATURE                        **
**    BOOL  = BOOLEAN OPERATOR FOR THE FEATURE                        **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE MCOUNT(R1,R2,L1,L2,IFL,IFL1,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3), R1, R2, L1, L2
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
```

```
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2
      CHARACTER CR1*7,CR2*7,CL1*7,CL2*7,CTYPE*7, CNU*7, CNW*7, CTYPE1*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      R1  = 2.0
      R2  = 1.0
      L1  = 1.0
      L2  = 2.0
      NU = 6
      NW = 6
      CTYPE = 'NO CAPS'
      IFL = 1
      CTYPE1 = 'BORE    '
      IFL1 = 1
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL    = '+'
      COLOR   =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

*************************************************************************
*        PREPARE FOR PICK INPUT                                         *
*************************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
      POS(2) = 0.80

      STRG = 'COUNT = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)

      WRITE(CR1,'(F7.2)')R1
      WRITE(CR2,'(F7.2)')R2
      WRITE(CL1,'(F7.2)')L1
      WRITE(CL2,'(F7.2)')L2
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)

      POS(1) = 0.62
      POS(2) = 0.67
```

```
      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)

      POS(1) = 0.62
      POS(2) = 0.54

      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.41

      STRG = 'ORIENT =   '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.28

      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)

      POS(1) = 0.62
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU     = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW     = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'RAD1   = '//CR1
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'RAD2   = '//CR2
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.54

      STRG = 'LEN1   = '//CL1
      CALL GPPKID(11)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.41

      STRG = 'LEN2   = '//CL2
      CALL GPPKID(12)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.28

      STRG = 'TYPE   =  '//CTYPE
      CALL GPPKID(13)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.15

      STRG = 'COUNT =  '//CTYPE1
      CALL GPPKID(14)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,5)
*ASSOCIATE MENU TO VIEW 5
```

```
      CALL GPARV(1,5,7,0)
*UPDATE WORKSTATION
      CALL FPRMP(14)
      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)
*AWAIT FOR A PICK
100   CALL GPAWEV(1.,1,ICLASS,IDEV)
      IF (ICLASS .NE. 5)GOTO 100
      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)
      GOTO (1,2,3,4,5,6,7,8,9,10,11,12,13,14)IPKID
1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50
2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF
3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &   .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &   .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
5     CALL COUNT(R1,R2,L1,L2,IFL,IFL1,LOCA,ORI,COLOR,BOOL)
6     GOTO 200
7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU .GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW .GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
9     CALL FGTST(CR1)
      IF (CR1(1:1) .GE. '0' .AND. CR2(1:1) .LE. '9')THEN
         READ(CR1,*)R1
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
10    CALL FGTST(CR2)
      IF (CR2(1:1) .GE. '0' .AND. CR2(1:1) .LE. '9')THEN
         READ(CR2,*)R2
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
11    CALL FGTST(CL1)
```

```
      IF (CL1(1:1) .GE. '0' .AND. CL1(1:1) .LE. '9')THEN
         READ(CL1,*)L1
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
12    CALL FGTST(CL2)
      IF (CL2(1:1) .GE. '0' .AND. CL2(1:1) .LE. '9')THEN
         READ(CL2,*)L2
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
13    IF(CTYPE .EQ. 'NO CAPS')THEN
         IFL = 2
         CTYPE = ' CAPPED'
         GOTO 50
      ELSE
         IFL = 1
         CTYPE = 'NO CAPS'
         GOTO 50
      ENDIF
14    IF(CTYPE1 .EQ. 'BORE    ')THEN
         IFL1 = 2
         CTYPE1 = 'SINK    '
         GOTO 50
      ELSE
         IFL1 = 1
         CTYPE1 = 'BORE    '
         GOTO 50
      ENDIF
*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON
200   CALL GPPKMO(1,1,1,2)
*DEFINE PICK AREAS AND INITIALIZE PICKS
      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,7)
*ASSOCIATE MENU TO VIEW 5
      CALL GPARV(1,5,5,0)
      RETURN
      END
```

SUBROUTINE MCYNDR

```
************************************************************************
************************************************************************
**    SUBROUTINE MCYNDR(RS,LEN,LOCA,ORI,COLOR,BOOL)                   **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING  **
**    A CYLINDER IN ITS CORRECT LOCATION AND ORIENTATION              **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    RS    = RADIUS OF THE CYLINDER                                  **
**    LOCA  = LOCATION OF THE CYLINDER                                **
**    ORI   = ORIENTATION OF THE CYLINDER                             **
**    COLOR = COLOR TO BE GIVEN TO THE CYLINDER                       **
**    BOOL  = BOOLEAN OPERATOR FOR THE CYLINDER                       **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE MCYNDR(RS,LEN,LOCA,ORI,COLOR,BOOL)
```

```
      REAL*4 LOCA(3), POS(2), RS, LEN, ORI(3)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CRS*7, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CLEN*7, CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2, CTYPE*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      RS = 1.0
      LEN = 5.0
      NU = 4
      NW = 4
      CTYPE = 'NO  CAP'
      IFL = 1
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL   = '+'
      COLOR  =  2


*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

*****************************************************************
*        PREPARE FOR PICK INPUT                                 *
*****************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

*CONVERT NUMBERS TO TEXT

      WRITE(CRS,'(F7.2)')RS
      WRITE(CLEN,'(F7.2)')LEN
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
      POS(2) = 0.80

      STRG = 'CYNDR = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)

      POS(1) = 0.62
      POS(2) = 0.67
```

```
      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)

      POS(1) = 0.62
      POS(2) = 0.54

      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.41

      STRG = 'ORIENT =    '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.28

      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)

      POS(1) = 0.62
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU      = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW      = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'RADIUS = '//CRS
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'LENGTH = '//CLEN
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.54

      STRG = 'TYPE    = '//CTYPE
      CALL GPPKID(11)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8,9,10,11)IPKID
```

```
1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &   .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &   .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

5     ICHK = 0
      CALL CYNDR(IFL,RS,LEN,LOCA,ORI,COLOR,BOOL,ICHK,0.)

6     GOTO 200

7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU. GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW. GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

9     CALL FGTST(CRS)
      IF (CRS(1:1) .GE. '0' .AND. CRS(1:1) .LE. '9')THEN
         READ(CRS,*)RS
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

10    CALL FGTST(CLEN)
      IF (CLEN(1:1) .GE. '0' .AND. CLEN(1:1) .LE. '9')THEN
         READ(CLEN,*)LEN
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

11    IF (CTYPE .EQ. 'NO  CAP' )THEN
         CTYPE = '  CAPPED'
         IFL = 2
         GOTO 50
      ELSE
         CTYPE = 'NO  CAP'
         IFL = 1
         GOTO 50
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
```

```
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,7)
*ASSOCIATE MENU TO VIEW 5
      CALL GPARV(1,5,5,0)
      RETURN
      END
```

SUBROUTINE MELPSOD

```
************************************************************************
************************************************************************
**    SUBROUTINE MELPSOD(RX,RY,LOCA,ORI,COLOR,BOOL)                   **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING  **
**    A ELLIPSOID IN ITS LOCATION AND ORIENTATION                     **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    RX   = RADIUS OF THE ELLIPSOID IN THE X-DIRECTION               **
**    RY   = RADIUS OF THE ELLIPSOID IN THE Y-DIRECTION               **
**    LOCA  = LOCATION OF THE ELLIPSOID                               **
**    ORI   = ORIENTATION OF THE ELLIPSOID                            **
**    COLOR = COLOR FOR THE ELLIPSOID                                 **
**    BOOL  = BOOLEAN TYPE FOR ELLIPSOID                              **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE MELPSOD(RX,RY,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), ORI(3), POS(2)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      REAL*4 RX,RY,RZ
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER BOOL*1, CCOLOR*2, CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER CRX*7, CRY*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      RX = 1.0
      RY = 1.0
      NU = 6
      NW = 6
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL   = '+'
      COLOR  =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

************************************************************************
*       PREPARE FOR PICK INPUT                                         *
************************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
```

```
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

      WRITE(CRX,'(F7.2)')RX
      WRITE(CRY,'(F7.2)')RY
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
      POS(2) = 0.80

      STRG = 'ELPSOD = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,10,STRG)

      POS(1) = 0.62
      POS(2) = 0.67

      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)

      POS(1) = 0.62
      POS(2) = 0.54

      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.41

      STRG = 'ORIENT   = '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.28

      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)

      POS(1) = 0.62
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU     = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW     = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80
```

```
      STRG = 'RAD X  = '//CRX
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'RAD Y  = '//CRY
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8,9,10)IPKID

1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &    .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &    .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

5     CALL ELPSOD(RX,RY,LOCA,ORI,COLOR,BOOL)

6     GOTO 200

7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU .GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW .GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
```

```
      ENDIF
9     CALL FGTST(CRX)
      IF (CRX(1:1) .GE. '0' .AND. CRX(1:1) .LE. '9')THEN
         READ(CRX,*)RX
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
10    CALL FGTST(CRY)
      IF (CRY(1:1) .GE. '0' .AND. CRY(1:1) .LE. '9')THEN
         READ(CRY,*)RY
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON
200   CALL GPPKMO(1,1,1,2)
*DEFINE PICK AREAS AND INITIALIZE PICKS
      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,7)
*ASSOCIATE MENU TO VIEW 5
      CALL GPARV(1,5,5,0)
      RETURN
      END
```

SUBROUTINE MHEMIS

```
************************************************************************
************************************************************************
**    SUBROUTINE MHEMIS(RX,LOCA,ORI,COLOR,BOOL)                       **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING  **
**    A SPHERE AND ITS LOCATION                                       **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    RS   = RADIUS OF THE HEMISPHERE                                 **
**    LOCA  = LOCATION OF THE HEMISPHERE                              **
**    ORI   = ORIENTATION OF THE HEMISPHERE                           **
**    LOCA  = LOCATION OF THE HEMISPHERE                              **
**    COLOR = COLOR FOR THE HEMISPHERE                                **
**    BOOL  = BOOLEAN TYPE FOR HEMISPHERE                             **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE MHEMIS(RX,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), ORI(3), POS(2)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      REAL*4 RX,RY,RZ
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER BOOL*1, CCOLOR*2, CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER CRX*7, CRY*7, CNU*7, CNW*7

*DEFINE GENERIC MODEL

      RX = 1.0
      RY = 1.0
      NU = 6
```

```
      NW = 6
      CRY = 'NO CAP '
      IFL = 1
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL   = '+'
      COLOR  =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

*********************************************************************
*        PREPARE FOR PICK INPUT                                     *
*********************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

      WRITE(CRX,'(F7.2)')RX
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
      POS(2) = 0.80

      STRG = 'HSPHRE = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,10,STRG)

      POS(1) = 0.62
      POS(2) = 0.67

      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)

      POS(1) = 0.62
      POS(2) = 0.54

      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.41

      STRG = 'ORIENT   = '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.28
```

```
      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)

      POS(1) = 0.62
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU     = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW     = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'RADIUS = '//CRX
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'TYPE   = '//CRY
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8,9,10)IPKID

1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &    .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

4     CALL FGTST(CORI)
```

```
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &   .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

5     CALL HEMIS(RX,IFL,LOCA,ORI,COLOR,BOOL)

6     GOTO 200

7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU .GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW .GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

9     CALL FGTST(CRX)
      IF (CRX(1:1) .GE. '0' .AND. CRX(1:1) .LE. '9')THEN
         READ(CRX,*)RX
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

10    IF (CRY .EQ. 'NO CAP ')THEN
         CRY = 'CAPPED'
         IFL = 2
         GOTO 50
      ELSE
         CRY = 'NO CAP'
         IFL = 1
         GOTO 50
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
      END
```

SUBROUTINE MPYRMID

```
**********************************************************************
**********************************************************************
**    SUBROUTINE MPYRMID(A,B,C,AR,IFL,LOCA,ORI,COLOR,BOOL)          **
**                                                                   **
**    PROGRAM DESCRIPTION                                            **
**                                                                   **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING **
**    A PYRAMID IN ITS CORRECT LOCATION AND ORIENTATION              **
**                                                                   **
**                                                                   **
**    BY:       ASHIT R. GANDHI                                      **
```

```
**     DATE:    11/12/88                                                  **
**                                                                        **
**     PARAMETERS USED:                                                   **
**                                                                        **
**     A      = LENGTH OF THE PYRAMID IN THE X-DIRECTION                  **
**     B      = LENGTH OF THE PYRAMID IN THE Y-DIRECTION                  **
**     C      = LENGTH OF THE PYRAMID IN THE Z-DIRECTION                  **
**     LOCA   = LOCATION OF THE PYRAMID                                   **
**     ORI    = ORIENTATION OF THE PYRAMID                                **
**     COLOR  = COLOR TO BE GIVEN TO THE PYRAMID                          **
**     BOOL   = BOOLEAN OPERATOR FOR THE PYRAMID                          **
**                                                                        **
************************************************************************
************************************************************************
      SUBROUTINE MPYRMID(A,B,C,AR,IFL,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3), A, B, C, AR
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2
      CHARACTER CA*7, CB*7, CC*7, CAR*7, CTYPE*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      A    = 1.0
      B    = 1.0
      C    = 1.0
      AR   = 0.0
      NU = 2
      NW = 2
      CTYPE = 'NO CAPS'
      IFL = 1
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL     = '+'
      COLOR    =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

************************************************************************
*        PREPARE FOR PICK INPUT                                        *
************************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
      POS(2) = 0.80

      STRG = 'PRISM = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)
```

```
WRITE(CA,'(F7.2)')A
WRITE(CB,'(F7.2)')B
WRITE(CC,'(F7.2)')C
WRITE(CAR,'(F7.2)')AR
WRITE(CNU,'(I2)')NU
WRITE(CNW,'(I2)')NW
WRITE(CCOLOR,'(I2)')COLOR
WRITE(CLOCA1,'(F7.2)')LOCA(1)
WRITE(CLOCA2,'(F7.2)')LOCA(2)
WRITE(CLOCA3,'(F7.2)')LOCA(3)
WRITE(CORI1,'(F7.2)')ORI(1)
WRITE(CORI2,'(F7.2)')ORI(2)
WRITE(CORI3,'(F7.2)')ORI(3)

POS(1) = 0.62
POS(2) = 0.67

STRG = 'COLOR = '//CCOLOR
CALL GPPKID(2)
CALL GPAN2(POS,29,STRG)

POS(1) = 0.62
POS(2) = 0.54

STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
CALL GPPKID(3)
CALL GPAN2(POS,32,STRG)

POS(1) = 0.62
POS(2) = 0.41

STRG = 'ORIENT =    '//CORI1//CORI2//CORI3
CALL GPPKID(4)
CALL GPAN2(POS,32,STRG)

POS(1) = 0.62
POS(2) = 0.28

STRG = 'EXECUTE'
CALL GPPKID(5)
CALL GPAN2(POS,7,STRG)

POS(1) = 0.62
POS(2) = 0.15

STRG = 'ABORT'
CALL GPPKID(6)
CALL GPAN2(POS,5,STRG)

POS(1) = 0.32
POS(2) = 0.80

STRG = 'NU      = '//CNU
CALL GPPKID(7)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.32
POS(2) = 0.67

STRG = 'NW      = '//CNW
CALL GPPKID(8)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.80

STRG = 'LENGTH = '//CA
CALL GPPKID(9)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.67

STRG = 'BREATH = '//CB
CALL GPPKID(10)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.54

STRG = 'HEIGHT = '//CC
CALL GPPKID(11)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.41

STRG = 'A RATIO ='//CAR
CALL GPPKID(12)
CALL GPAN2(POS,16,STRG)

POS(1) = 0.02
POS(2) = 0.28

STRG = 'TYPE  =  '//CTYPE
```

```
      CALL GPPKID(13)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8,9,10,11,12,13)IPKID

1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &    .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &    .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

5     CALL PYRAMID(A,B,C,AR,IFL,LOCA,ORI,COLOR,BOOL)

6     GOTO 200

7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU .GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW .GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

9     CALL FGTST(CA)
      IF (CA(1:1) .GE. '0' .AND. CA(1:1) .LE. '9')THEN
         READ(CA,*)A
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
```

```
10    CALL FGTST(CB)
      IF (CB(1:1) .GE. '0' .AND. CB(1:1) .LE. '9')THEN
         READ(CB,*)B
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

11    CALL FGTST(CC)
      IF (CC(1:1) .GE. '0' .AND. CC(1:1) .LE. '9')THEN
         READ(CC,*)C
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

12    CALL FGTST(CAR)
      IF (CAR(1:1) .GE. '0' .AND. CAR(1:1) .LE. '9')THEN
         READ(CAR,*)AR
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

13    IF(CTYPE .EQ. 'NO CAPS')THEN
         IFL = 2
         CTYPE = ' CAPPED'
         GOTO 50
      ELSE
         IFL = 1
         CTYPE = 'NO CAPS'
         GOTO 50
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
      END
```

SUBROUTINE MRING

```
**********************************************************************
**********************************************************************
**    SUBROUTINE MRING(R1,R2,LOCA,ORI,COLOR,BOOL)                   **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING **
**    A RING IN ITS CORRECT LOCATION AND ORIENTATION                **
**                                                                  **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     11/12/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    R1     = INNER RADIUS FOR RING                                **
**    R2     = OUTER RADIUS FOR RING                                **
**    LOCA   = LOCATION OF RING                                     **
**    ORI    = ORIENTATION OF RING                                  **
**    COLOR  = COLOR TO BE GIVEN TO RING                            **
**    BOOL   = BOOLEAN OPERATOR FOR RING                            **
**                                                                  **
**********************************************************************
**********************************************************************
```

```
      SUBROUTINE MRING(R1,R2,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      REAL*4 R1, R2
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2, CNU*7, CNW*7
      CHARACTER CR1*7, CR2*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      R1 = 1.0
      R2 = 2.0
      NU = 4
      NW = 4
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL   = '+'
      COLOR  =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

*************************************************************************
*       PREPARE FOR PICK INPUT                                          *
*************************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

      WRITE(CR1,'(F7.2)')R1
      WRITE(CR2,'(F7.2)')R2
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
      POS(2) = 0.80

      STRG = 'RING  = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)

      POS(1) = 0.62
      POS(2) = 0.67

      STRG = 'COLOR = '//CCOLOR
```

```
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)

      POS(1) = 0.62
      POS(2) = 0.54

      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.41

      STRG = 'ORIENT =    '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.28

      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)

      POS(1) = 0.62
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU      = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW      = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'R1      = '//CR1
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'R2      = '//CR2
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8,9,10)IPKID

1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
```

```
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &    .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &    .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

5     ICHK = 0
      CALL RING(R1,R2,LOCA,ORI,COLOR,BOOL,ICHK,0.)

6     GOTO 200

7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF ( INU .GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF ( INW .GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

9     CALL FGTST(CR1)
      IF (CR1(1:1) .GE. '0' .AND. CR1(1:1) .LE. '9')THEN
         READ(CR1,*)R1
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

10    CALL FGTST(CR2)
      IF (CR2(1:1) .GE. '0' .AND. CR2(1:1) .LE. '9')THEN
         READ(CR2,*)R2
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
      END
```

SUBROUTINE MSLAB

```
************************************************************************
************************************************************************
**     SUBROUTINE MSLAB(A,B,C,LOCA,ORI,COLOR,BOOL)                    **
**                                                                    **
**     PROGRAM DESCRIPTION                                            **
**                                                                    **
**     THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING  **
**     A SLAB IN ITS CORRECT LOCATION AND ORIENTATION                 **
**                                                                    **
**                                                                    **
**     BY:       ASHIT R. GANDHI                                      **
**     DATE:     11/12/88                                             **
**                                                                    **
**     PARAMETERS USED:                                               **
**                                                                    **
**     A      = LENGTH OF THE SLAB                                    **
**     B      = BREADTH OF THE SLAB                                   **
**     C      = HEIGHT OF THE SLAB                                    **
**     LOCA   = LOCATION OF THE SLAB                                  **
**     ORI    = ORIENTATION OF THE SLAB                               **
**     COLOR  = COLOR TO BE GIVEN TO THE SLAB                         **
**     BOOL   = BOOLEAN OPERATOR FOR THE SLAB                         **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE MSLAB(A,B,C,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3), A, B, C
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2
      CHARACTER CA*7, CB*7, CC*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      A  = 1.0
      B   = 1.0
      C   = 1.0
      NU = 2
      NW = 2
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL    = '+'
      COLOR   =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

************************************************************************
*       PREPARE FOR PICK INPUT                                         *
************************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW
```

```
      CALL GPTXCI(5)
*SET ANNOTATION TEXT SIZE SCALE FACTOR
      CALL GPAHSC(0.80)
      WRITE(CA,'(F7.2)')A
      WRITE(CB,'(F7.2)')B
      WRITE(CC,'(F7.2)')C
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)
*DRAW ALL TITLE TEXT
      POS(1) = 0.62
      POS(2) = 0.80
      STRG = 'SLAB   = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)
      POS(1) = 0.62
      POS(2) = 0.67
      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)
      POS(1) = 0.62
      POS(2) = 0.54
      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)
      POS(1) = 0.62
      POS(2) = 0.41
      STRG = 'ORIENT =    '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)
      POS(1) = 0.62
      POS(2) = 0.28
      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)
      POS(1) = 0.62
      POS(2) = 0.15
      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)
      POS(1) = 0.32
      POS(2) = 0.80
      STRG = 'NU      = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)
      POS(1) = 0.32
      POS(2) = 0.67
      STRG = 'NW      = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)
      POS(1) = 0.02
      POS(2) = 0.80
      STRG = 'LENGTH = '//CA
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)
      POS(1) = 0.02
      POS(2) = 0.67
      STRG = 'BREATH = '//CB
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)
      POS(1) = 0.02
      POS(2) = 0.54
      STRG = 'HEIGHT = '//CC
      CALL GPPKID(11)
```

```
      CALL GPAN2(POS,16,STRG)
      CALL GPCLST
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,5)
*ASSOCIATE MENU TO VIEW 5
      CALL GPARV(1,5,7,0)
*UPDATE WORKSTATION
      CALL FPRMP(14)
      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)
*AWAIT FOR A PICK
100   CALL GPAWEV(1.,1,ICLASS,IDEV)
      IF (ICLASS .NE. 5)GOTO 100
      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)
      GOTO (1,2,3,4,5,6,7,8,9,10,11)IPKID
1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50
2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF
3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &    .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &    .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
5     CALL SLAB(A,B,C,LOCA,ORI,COLOR,BOOL)
6     GOTO 200
7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU .GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW .GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
9     CALL FGTST(CA)
      IF (CA(1:1) .GE. '0' .AND. CA(1:1) .LE. '9')THEN
         READ(CA,*)A
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
```

```
10    CALL FGTST(CB)
      IF (CB(1:1) .GE. '0' .AND. CB(1:1) .LE. '9')THEN
         READ(CB,*)B
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

11    CALL FGTST(CC)
      IF (CC(1:1) .GE. '0' .AND. CC(1:1) .LE. '9')THEN
         READ(CC,*)C
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
      END
```

SUBROUTINE MSPHERE

```
**********************************************************************
**********************************************************************
**    SUBROUTINE MSPHERE(RS,LOCA,COLOR,BOOL)                        **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING **
**    A SPHERE AND ITS LOCATION                                     **
**                                                                  **
**                                                                  **
**    BY:       ASHIT R. GANDHI                                     **
**    DATE:     11/12/88                                            **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    RS   = RADIUS OF THE SPHERE                                   **
**    LOCA  = LOCATION OF THE SPHERE                                **
**    COLOR = COLOR TO BE ASSIGNED TO SPHERE                        **
**    BOOL  = BOOLEAN OPERATOR FOR SPHERE                           **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE MSPHERE(RS,LOCA,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), RS
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CRS*7, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER BOOL*1, CCOLOR*2, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      RS = 1.0
      NU = 6
      NW = 6
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      BOOL    = '+'
```

```
      COLOR   =  2
*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE
      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)
*************************************************************************
*        PREPARE FOR PICK INPUT                                         *
*************************************************************************
*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,1,2)
*DEFINE PICK AREAS AND INITIALIZE PICKS
      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*OPEN STRUCTURE
50    CALL GPEST(7)
      CALL GPOPST(7)
      CALL GPADCN(1,2)
*SET TEXT COLOR TO YELLOW
      CALL GPTXCI(5)
*SET ANNOTATION TEXT SIZE SCALE FACTOR
      CALL GPAHSC(0.80)
*CONVERT NUMBERS TO TEXT
      WRITE(CRS,'(F7.2)')RS
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
*DRAW ALL TITLE TEXT
      POS(1) = 0.62
      POS(2) = 0.80
      STRG = 'SPHERE = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,10,STRG)
      POS(1) = 0.62
      POS(2) = 0.67
      WRITE(CCOLOR,'(I2)')COLOR
      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)
      POS(1) = 0.62
      POS(2) = 0.54
      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)
      POS(1) = 0.62
      POS(2) = 0.41
      STRG = 'EXECUTE'
      CALL GPPKID(4)
      CALL GPAN2(POS,7,STRG)
      POS(1) = 0.62
      POS(2) = 0.28
      STRG = 'ABORT'
      CALL GPPKID(5)
      CALL GPAN2(POS,5,STRG)
      POS(1) = 0.32
      POS(2) = 0.80
      STRG = 'NU = '//CNU
      CALL GPPKID(6)
      CALL GPAN2(POS,8,STRG)
```

```
      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW = '//CNW
      CALL GPPKID(7)
      CALL GPAN2(POS,8,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'RADIUS = '//CRS
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISASSOCIATE SCROLL MESSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8)IPKID

1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &   .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

4     CALL SPHERE(RS,LOCA,COLOR,BOOL)

5     GOTO 200

6     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU .LE. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

7     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW .LE. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CRS)
      IF (CRS(1:1) .GE. '0' .AND. CRS(1:1) .LE. '9')THEN
         READ(CRS,*)RS
         GOTO 50
```

```
      ELSE
         GOTO 100
      ENDIF
*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON
200   CALL GPPKMO(1,1,1,2)
*DEFINE PICK AREAS AND INITIALIZE PICKS
      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,7)
*ASSOCIATE MENU TO VIEW 5
      CALL GPARV(1,5,5,0)
      RETURN
      END
```

SUBROUTINE MTUNNEL

```
************************************************************************
************************************************************************
**    SUBROUTINE MTUNNEL(R1,R2,THK,LOCA,ORI,COLOR,BOOL)                **
**                                                                     **
**    PROGRAM DESCRIPTION                                              **
**                                                                     **
**    THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING   **
**    A TUNNEL IN ITS CORRECT LOCATION AND ORIENTATION                 **
**                                                                     **
**                                                                     **
**    BY:       ASHIT R. GANDHI                                        **
**    DATE:     11/12/88                                               **
**                                                                     **
**    PARAMETERS USED:                                                 **
**                                                                     **
**    R1     = INNER RADIUS OF THE TUNNEL                              **
**    R2     = OUTER RADIUS OF THE TUNNEL                              **
**    THK    = THICKNESS OF THE TUNNEL                                 **
**    LOCA   = LOCATION OF THE TUNNEL                                  **
**    ORI    = ORIENTATION OF THE TUNNEL                               **
**    COLOR  = COLOR TO BE GIVEN TO THE TUNNEL                         **
**    BOOL   = BOOLEAN OPERATOR FOR THE TUNNEL                         **
**                                                                     **
************************************************************************
************************************************************************
      SUBROUTINE MTUNNEL(R1,R2,THK,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3)
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      REAL*4 R1, R2
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2
      CHARACTER CR1*7, CR2*7, CTHK*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      R1 = 1.0
      R2 = 2.0
      THK = 1.0
      NU = 6
      NW = 6
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL   = '+'
      COLOR  =  2
```

```
*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE
      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)
*************************************************************************
*        PREPARE FOR PICK INPUT                                         *
*************************************************************************
*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,1,2)
*DEFINE PICK AREAS AND INITIALIZE PICKS
      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*OPEN STRUCTURE
50    CALL GPEST(7)
      CALL GPOPST(7)
      CALL GPADCN(1,2)
*SET TEXT COLOR TO YELLOW
      CALL GPTXCI(5)
*SET ANNOTATION TEXT SIZE SCALE FACTOR
      CALL GPAHSC(0.80)
*CONVERT NUMBERS TO TEXT
      WRITE(CR1,'(F7.2)')R1
      WRITE(CR2,'(F7.2)')R2
      WRITE(CTHK,'(F7.2)')THK
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)
*DRAW ALL TITLE TEXT
      POS(1) = 0.62
      POS(2) = 0.80
      STRG = 'BUSH  = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)
      POS(1) = 0.62
      POS(2) = 0.67
      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)
      POS(1) = 0.62
      POS(2) = 0.54
      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)
      POS(1) = 0.62
      POS(2) = 0.41
      STRG = 'ORIENT =   '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)
      POS(1) = 0.62
      POS(2) = 0.28
      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)
      POS(1) = 0.62
```

```
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU      = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW      = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'R1      = '//CR1
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'R2      = '//CR2
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.54

      STRG = 'THICK   = '//CTHK
      CALL GPPKID(11)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,7,0)

*UPDATE WORKSTATION

      CALL FPRMP(14)

      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)

*AWAIT FOR A PICK

100   CALL GPAWEV(1.,1,ICLASS,IDEV)

      IF (ICLASS .NE. 5)GOTO 100

      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)

      GOTO (1,2,3,4,5,6,7,8,9,10,11)IPKID

1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50

2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF

3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &   .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
```

```
4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &   .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

5     ICHK = 1
      CALL FGRP4(R1,R2,THK,LOCA,ORI,COLOR,BOOL,ICHK,0.)

6     GOTO 200

7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF(INU. GT. 1 .AND. INU .LT. 20)NU = INU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF(INW. GT. 1 .AND. INW .LT. 20)NW = INW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

9     CALL FGTST(CR1)
      IF (CR1(1:1) .GE. '0' .AND. CR1(1:1) .LE. '9')THEN
         READ(CR1,*)R1
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

10    CALL FGTST(CR2)
      IF (CR2(1:1) .GE. '0' .AND. CR2(1:1) .LE. '9')THEN
         READ(CR2,*)R2
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

11    CALL FGTST(CTHK)
      IF (CTHK(1:1) .GE. '0' .AND. CTHK(1:1) .LE. '9')THEN
         READ(CTHK,*)THK
         GOTO 50
      ELSE
         GOTO 100
      ENDIF

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

200   CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,7)

*ASSOCIATE MENU TO VIEW 5

      CALL GPARV(1,5,5,0)

      RETURN
      END
```

SUBROUTINE MWEDGE

```
**********************************************************************
**********************************************************************
**    SUBROUTINE MWEDGE(A,B,C,LOCA,ORI,COLOR,BOOL)                  **
```

```
**                                                                      **
**     PROGRAM DESCRIPTION                                              **
**                                                                      **
**     THIS ROUTINE WILL DISPLAY MENU FOR GETTING PARAMETERS DEFINING   **
**     A WEDGE IN ITS CORRECT LOCATION AND ORIENTATION                  **
**                                                                      **
**                                                                      **
**     BY:       ASHIT R. GANDHI                                        **
**     DATE:     11/12/88                                               **
**                                                                      **
**     PARAMETERS USED:                                                 **
**                                                                      **
**     A      = LENGTH OF THE WEDGE                                     **
**     B      = BREADTH OF THE WEDGE                                    **
**     C      = HEIGHT OF THE WEDGE                                     **
**     LOCA   = LOCATION OF THE WEDGE                                   **
**     ORI    = ORIENTATION OF THE WEDGE                                **
**     COLOR  = COLOR TO BE GIVEN TO THE WEDGE                          **
**     BOOL   = BOOLEAN OPERATOR FOR THE WEDGE                          **
**                                                                      **
**********************************************************************
**********************************************************************
      SUBROUTINE MWEDGE(A,B,C,LOCA,ORI,COLOR,BOOL)

      REAL*4 LOCA(3), POS(2), ORI(3), A, B, C
      REAL*4 ASIZE(6),CSIZE(3),DATA(12)
      REAL*4 PAREA(6),SAREA(6),VAREA(6)
      INTEGER*4 COLOR, PPATH(3)

      CHARACTER STRG*32, CLOCA1*7, CLOCA2*7, CLOCA3*7, CLOCA*26
      CHARACTER CORI1*7, CORI2*7, CORI3*7, CORI*26
      CHARACTER BOOL*1, CCOLOR*2
      CHARACTER CA*7, CB*7, CC*7, CNU*7, CNW*7

      COMMON/PATCH/NU,NW

*DEFINE GENERIC MODEL

      A  = 1.0
      B   = 1.0
      C   = 1.0
      NU = 2
      NW = 2
      LOCA(1) = 0.0
      LOCA(2) = 0.0
      LOCA(3) = 0.0
      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0
      BOOL    = '+'
      COLOR   =  2

*INQUIRE ACTUAL MAXIMUM DISPLAY SURFACE SIZE

      CALL GPQADS(1,ERRIND,UNITS,CSIZE,ASIZE)

**********************************************************************
*        PREPARE FOR PICK INPUT                                      *
**********************************************************************

*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,1,2)

*DEFINE PICK AREAS AND INITIALIZE PICKS

      PAREA(1)=0.005*CSIZE(1)
      PAREA(2)=0.845*CSIZE(1)
      PAREA(3)=0.005*CSIZE(2)
      PAREA(4)=0.145*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)

*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON

      CALL GPPKMO(1,1,3,2)

*OPEN STRUCTURE

50    CALL GPEST(7)
      CALL GPOPST(7)

      CALL GPADCN(1,2)

*SET TEXT COLOR TO YELLOW

      CALL GPTXCI(5)

*SET ANNOTATION TEXT SIZE SCALE FACTOR

      CALL GPAHSC(0.80)

*DRAW ALL TITLE TEXT

      POS(1) = 0.62
```

```
      POS(2) = 0.80

      STRG = 'WEDGE = '//BOOL
      CALL GPPKID(1)
      CALL GPAN2(POS,9,STRG)

      WRITE(CA,'(F7.2)')A
      WRITE(CB,'(F7.2)')B
      WRITE(CC,'(F7.2)')C
      WRITE(CNU,'(I2)')NU
      WRITE(CNW,'(I2)')NW
      WRITE(CCOLOR,'(I2)')COLOR
      WRITE(CLOCA1,'(F7.2)')LOCA(1)
      WRITE(CLOCA2,'(F7.2)')LOCA(2)
      WRITE(CLOCA3,'(F7.2)')LOCA(3)
      WRITE(CORI1,'(F7.2)')ORI(1)
      WRITE(CORI2,'(F7.2)')ORI(2)
      WRITE(CORI3,'(F7.2)')ORI(3)

      POS(1) = 0.62
      POS(2) = 0.67

      STRG = 'COLOR = '//CCOLOR
      CALL GPPKID(2)
      CALL GPAN2(POS,29,STRG)

      POS(1) = 0.62
      POS(2) = 0.54

      STRG = 'LOCATION = '//CLOCA1//CLOCA2//CLOCA3
      CALL GPPKID(3)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.41

      STRG = 'ORIENT =    '//CORI1//CORI2//CORI3
      CALL GPPKID(4)
      CALL GPAN2(POS,32,STRG)

      POS(1) = 0.62
      POS(2) = 0.28

      STRG = 'EXECUTE'
      CALL GPPKID(5)
      CALL GPAN2(POS,7,STRG)

      POS(1) = 0.62
      POS(2) = 0.15

      STRG = 'ABORT'
      CALL GPPKID(6)
      CALL GPAN2(POS,5,STRG)

      POS(1) = 0.32
      POS(2) = 0.80

      STRG = 'NU      = '//CNU
      CALL GPPKID(7)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.32
      POS(2) = 0.67

      STRG = 'NW      = '//CNW
      CALL GPPKID(8)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.80

      STRG = 'LENGTH = '//CA
      CALL GPPKID(9)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.67

      STRG = 'BREATH = '//CB
      CALL GPPKID(10)
      CALL GPAN2(POS,16,STRG)

      POS(1) = 0.02
      POS(2) = 0.54

      STRG = 'HEIGHT = '//CC
      CALL GPPKID(11)
      CALL GPAN2(POS,16,STRG)

      CALL GPCLST

*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5

      CALL GPDRV(1,5,5)

*ASSOCIATE MENU TO VIEW 5
```

```
      CALL GPARV(1,5,7,0)
*UPDATE WORKSTATION
      CALL FPRMP(14)
      ICLASS = 2
      CALL GPPKF(1,1,1,ICLASS,1,1)
*AWAIT FOR A PICK
100   CALL GPAWEV(1.,1,ICLASS,IDEV)
      IF (ICLASS .NE. 5)GOTO 100
      CALL GPGTPK(1,1,PPATH)
      IPKID = PPATH(2)
      GOTO (1,2,3,4,5,6,7,8,9,10,11)IPKID
1     CALL FGTST(BOOL)
      IF (BOOL .NE. '+' .AND. BOOL .NE. '-' .AND. BOOL .NE. '*')THEN
         GOTO 100
      ENDIF
      GOTO 50
2     CALL FGTST(CCOLOR)
      IF (CCOLOR(1:1) .GE. '0' .AND. CCOLOR(1:1) .LE. '9')THEN
         READ(CCOLOR,*)COLOR
         IF (COLOR .GE. 1 .AND. COLOR .LE. 6)THEN
            GOTO 50
         ELSE
            GOTO 100
         ENDIF
      ELSE
         GOTO 100
      ENDIF
3     CALL FGTST(CLOCA)
      IF ((CLOCA(1:1) .GE. '0' .AND. CLOCA(1:1) .LE. '9')
     &   .OR. (CLOCA(1:1) .EQ. '+' .OR. CLOCA(1:1) .EQ. '-'))THEN
         READ(CLOCA,*)LOCA(1),LOCA(2),LOCA(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
4     CALL FGTST(CORI)
      IF ((CORI(1:1) .GE. '0' .AND. CORI(1:1) .LE. '9')
     &   .OR. (CORI(1:1) .EQ. '+' .OR. CORI(1:1) .EQ. '-'))THEN
         READ(CORI,*)ORI(1),ORI(2),ORI(3)
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
5     CALL WEDGE(A,B,C,LOCA,ORI,COLOR,BOOL)
6     GOTO 200
7     CALL FGTST(CNU)
      IF (CNU(1:1) .GE. '0' .AND. CNU(1:1) .LE. '9')THEN
         READ(CNU,*)INU
         IF (INU .GT. 1 .AND. INU .LT. 20)INU = NU
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
8     CALL FGTST(CNW)
      IF (CNW(1:1) .GE. '0' .AND. CNW(1:1) .LE. '9')THEN
         READ(CNW,*)INW
         IF (INW .GT. 1 .AND. INW .LT. 20)INW = NW
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
9     CALL FGTST(CA)
      IF (CA(1:1) .GE. '0' .AND. CA(1:1) .LE. '9')THEN
         READ(CA,*)A
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
10    CALL FGTST(CB)
      IF (CB(1:1) .GE. '0' .AND. CB(1:1) .LE. '9')THEN
         READ(CB,*)B
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
11    CALL FGTST(CC)
```

```
      IF (CC(1:1) .GE. '0' .AND. CC(1:1) .LE. '9')THEN
         READ(CC,*)C
         GOTO 50
      ELSE
         GOTO 100
      ENDIF
*PLACE PICK IN THE REQUEST MODE AND TURN ECHO SWITCH ON
200   CALL GPPKMO(1,1,1,2)
*DEFINE PICK AREAS AND INITIALIZE PICKS
      PAREA(1)=0.85*CSIZE(1)
      PAREA(2)=0.995*CSIZE(1)
      PAREA(3)=0.15*CSIZE(2)
      PAREA(4)=0.94*CSIZE(2)
      PAREA(5)=0.0
      PAREA(6)=CSIZE(3)
      CALL GPINPK(1,1,0,PPATH,1,PAREA,0,DATA,1)
*PLACE PICK IN THE EVENT MODE AND TURN ECHO SWITCH ON
      CALL GPPKMO(1,1,3,2)
*DISSASSOCIATE SCROLL MESSSAGES FROM VIEW 5
      CALL GPDRV(1,5,7)
*ASSOCIATE MENU TO VIEW 5
      CALL GPARV(1,5,5,0)
      RETURN
      END
```

SUBROUTINE PYRAMID

```
************************************************************************
************************************************************************
**    SUBROUTINE PYRAMID(L,B,H,AR,IFL,LOCA,ORI,COLOR,BOOL)            **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRED FOR       **
**    CREATING A PYRAMID USING B-SPLINE SURFACES                      **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    L      = LENGTH OF THE PYRAMID IN THE X-DIRECTION               **
**    B      = LENGTH OF THE PYRAMID IN THE Y-DIRECTION               **
**    H      = LENGTH OF THE PYRAMID IN THE Z-DIRECTION               **
**    AR     = ASPECT RATIO OF THE PYRAMID                            **
**    LOCA   = LOCATION OF THE PYRAMID                                **
**    ORI    = ORIENTATION OF THE PYRAMID                             **
**    COLOR  = COLOR TO BE GIVEN TO THE PYRAMID                       **
**    BOOL   = BOOLEAN OPERATOR FOR THE PYRAMID                       **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE PYRAMID(L,B,H,AR,IFL,LOC,ORI,ICOL,BOOL)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM
      .
      .
      REAL*4 L, B, H, LOC(3), ORI(3), H0(20,20), AR
      REAL*4 F2(2), F4(4), X(4), Y(4), Z(4), PTS(3), PNTS(20,20,3)
      REAL*4 SPTS(4,4,3), U(4), W(4), LOCA(3), ORIA(3)
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU, NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA F2/1.0,-1.0/
      DATA F4/-1.0,0.0,1.0,2.0/
      DATA H0/400*1.0/
      DATA U/0.0, 0.3333333, 0.66666667, 1.0/
      DATA W/0.0, 0.3333333, 0.66666667, 1.0/

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
```

```
         PORI(I) = ORI(I)
50    CONTINUE

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      INNUM = INNUM + 1

      IF (IFL .EQ. 1) THEN
         NSURF = 4
      ELSE
         IF (AR .EQ. 0.0)THEN
            NSURF = 5
         ELSE
            NSURF = 6
         ENDIF
      ENDIF

      CALL FDINS(INNUM,LTYPE,'PYRAMID ',NSURF,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

*CREATE THE PATCHES FOR Z = CONSTANT

      X(1)  = 0.0
      X(2)  = L/3.0
      X(3)  =  2.0*L/3.0
      X(4)  =  L

      Z0 = H
      DZ0 = -H/3.0
      ZL = H*(1.0 + AR)/2.0
      DZL = -H*AR/3.0

      Y0 = 0.0
      DY0 = B/3.0
      YL = B*(1.0 - AR)/2.0
      DYL = B*AR/3.0

      WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
      DO 200 I = 1,4
         DO 100 K = 1,4
            Y(K) = Y0 + (YL-Y0)*(K-1)/3.0
            Z(K) = Z0 + (ZL-Z0)*(K-1)/3.0
            PTS(1) = X(K)
            PTS(2) = Y(K)
            PTS(3) = Z(K)
            SPTS(I,K,1) = PTS(1)
            SPTS(I,K,2) = PTS(2)
            SPTS(I,K,3) = PTS(3)
100      CONTINUE
         Y0 = Y0 + DY0
         YL = YL + DYL
200   CONTINUE
      CALL FSTCP(U,W,SPTS,PNTS)
      DO 250 I = 1,4
      WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
250   WRITE(10,*)(H0(1,KL),KL=1,4)
      CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

      Z0 = 0.0
      DZ0 = H/3.0
      ZL = H*(1.0 - AR)/2.0
      DZL = H*AR/3.0

      Y0 = 0.0
      DY0 = B/3.0
      YL = B*(1.0 - AR)/2.0
      DYL = B*AR/3.0

      WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
      DO 400 I = 1,4
         DO 300 K = 1,4
            Y(K) = Y0 + (YL-Y0)*(K-1)/3.0
            Z(K) = Z0 + (ZL-Z0)*(K-1)/3.0
            PTS(1) = X(K)
            PTS(2) = Y(K)
            PTS(3) = Z(K)
            SPTS(I,K,1) = PTS(1)
            SPTS(I,K,2) = PTS(2)
            SPTS(I,K,3) = PTS(3)
300      CONTINUE
         Y0 = Y0 + DY0
         YL = YL + DYL
400   CONTINUE
      CALL FSTCP(U,W,SPTS,PNTS)
      DO 450 I = 1,4
      WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
450   WRITE(10,*)(H0(1,KL),KL=1,4)
      CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

      Z0 = 0.0
```

```
      DZO = H/3.0
      ZL = H*(1.0 - AR)/2.0
      DZL = H*AR/3.0

      YO = 0.0
      DYO = B/3.0
      YL = B*(1.0 - AR)/2.0
      DYL = B*AR/3.0

      WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
      DO 600 I = 1,4
         DO 500 K = 1,4
            Y(K) = YO + (YL-YO)*(K-1)/3.0
            Z(K) = ZO + (ZL-ZO)*(K-1)/3.0
            PTS(1) = X(K)
            PTS(2) = Y(K)
            PTS(3) = Z(K)
            SPTS(I,K,1) = PTS(1)
            SPTS(I,K,2) = PTS(2)
            SPTS(I,K,3) = PTS(3)
500      CONTINUE
         ZO = ZO + DZO
         ZL = ZL + DZL
600   CONTINUE
      CALL FSTCP(U,W,SPTS,PNTS)
      DO 650 I = 1,4
      WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
650   WRITE(10,*)(HO(1,KL),KL=1,4)
      CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)

      ZO = 0.0
      DZO = H/3.0
      ZL = H*(1.0 - AR)/2.0
      DZL = H*AR/3.0

      YO = B
      DYO = -B/3.0
      YL = B*(1.0 + AR)/2.0
      DYL = -B*AR/3.0

      WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
      DO 800 I = 1,4
         DO 700 K = 1,4
            Y(K) = YO + (YL-YO)*(K-1)/3.0
            Z(K) = ZO + (ZL-ZO)*(K-1)/3.0
            PTS(1) = X(K)
            PTS(2) = Y(K)
            PTS(3) = Z(K)
            SPTS(I,K,1) = PTS(1)
            SPTS(I,K,2) = PTS(2)
            SPTS(I,K,3) = PTS(3)
700      CONTINUE
         ZO = ZO + DZO
         ZL = ZL + DZL
800   CONTINUE
      CALL FSTCP(U,W,SPTS,PNTS)
      DO 850 I = 1,4
      WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
      WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
850   WRITE(10,*)(HO(1,KL),KL=1,4)
      CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)

      IF (IFL .EQ. 1)RETURN

*CREATE THE PATCHES FOR X = CONSTANT

      Y(1)  = -B
      Y(2)  =  0.0
      Y(3)  =  B
      Y(4)  =  2.0*B

      WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
      DO 1000 I = 1,4
         DO 900 K = 1,4
            X(K) = 0.0
            Z(K) = F4(I)*H
            PTS(1) = X(K)
            PTS(2) = Y(K)
            PTS(3) = Z(K)
            PNTS(I,K,1) = PTS(1)
            PNTS(I,K,2) = PTS(2)
            PNTS(I,K,3) = PTS(3)
900      CONTINUE
         WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
         WRITE(10,*)(HO(1,KL),KL=1,4)
1000  CONTINUE
      CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)

      IF (AR. EQ. 0.0)RETURN

      Y(1)  = -B*AR
```

```
      Y(2)  =  0.0
      Y(3)  =  B*AR
      Y(4)  =  2.0*B*AR
      LOCA(1) = L
      LOCA(2) = B*(1-AR)/2.0
      LOCA(3) = H*(1-AR)/2.0

      WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
      DO 1200 I = 1,4
         DO 1100 K = 1,4
            X(K) = 0.0
            Z(K) = F4(I)*H*AR
            PTS(1) = X(K)
            PTS(2) = Y(K)
            PTS(3) = Z(K)
            CALL FTRNS(LOCA,ORIA,PTS)
            PNTS(I,K,1) = PTS(1)
            PNTS(I,K,2) = PTS(2)
            PNTS(I,K,3) = PTS(3)
1100     CONTINUE
         WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
         WRITE(10,*)(H0(1,KL),KL=1,4)
1200  CONTINUE
      CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE RING

```
*********************************************************************
*********************************************************************
**    SUBROUTINE RING(R1,R2,LOC,ORI,ICOL,BOOL,ICHK,PLACE)          **
**                                                                  **
**    PROGRAM DESCRIPTION                                           **
**                                                                  **
**    THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRED TO      **
**    CREATE A RING USING B-SPLINE SURFACES                         **
**                                                                  **
**                                                                  **
**    BY:      ASHIT R. GANDHI                                      **
**    DATE:    11/21/88                                             **
**                                                                  **
**    PARAMETERS USED:                                              **
**                                                                  **
**    R1    = INNER RADIUS OF RING (REAL,I/P)                       **
**    R2    = OUTER RADIUS OF RING (REAL,I/P)                       **
**    LOC   = LOCATION OF THE RING (REAL,I/P)                       **
**    ORI   = EULER ORIENTATION FOR THE RING (REAL,I/P)             **
**    ICOL  = COLOR FOR FEATURE                                     **
**    BOOL  = BOOLEAN TYPE FOR FEATURE                              **
**    ICHK  = FLAG FOR NEW FEATURE                                  **
**    PLACE= LOCATION OF FEATURE IN LOCAL AXIS                      **
**                                                                  **
*********************************************************************
*********************************************************************
      SUBROUTINE RING(R1,R2,LOC,ORI,ICOL,BOOL,ICHK,PLACE)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 R1, R2, LOC(3), ORI(3), FX(4), FY(4), FZ(4)
      REAL*4 U(4), W(4), H0(20,20), PT(3), PTS(4,4,3), PNTS(20,20,3)
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA FX/-1.0, 0.0, 1.0, 2.0/
      DATA FY/ 1.0, 1.0,-1.0,-1.0/
      DATA FZ/ 1.0,-1.0,-1.0, 1.0/
      DATA U/0.0, 0.33333333, 0.66666667, 1.0/
      DATA W/0.0, 0.33333333, 0.66666667, 1.0/
      DATA H0/400*1.0/

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

*MAKE SURE THAT OUTER RADIUS IS GREATER THAN INNER RADIUS

      IF (R1 .GT. R2)THEN
```

```
         WRITE(6,*)'RING  ====>  INNER RADIUS IS GREATER THAN'
         WRITE(6,*)'RING  ====>  OUTER RADIUS'
         WRITE(6,*)'RING  ====>  EXCHANGING RADII'
         R3 = R1
         R1 = R2
         R2 = R3
      ENDIF

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      IF (ICHK .EQ. 0) THEN
         INNUM = INNUM + 1
         CALL FDINS(INNUM,LTYPE,'RING    ',4,CNUM,
     1              OBJECT(CNUM-1000),LOC,ORI)
      ENDIF

      PI = 3.14159

      DEL = PI/6.0

*COMPUTE THE RING

      DO 600 K = 1,4
         WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
         THETA = 0.0
         DO 500 I = 1,4
            DO 400 J = 1,4
               PT(1) = 0.0 + PLACE
               PT(2) = FY(K)*(R1 + U(J)*(R2-R1))*SIN(THETA)
               PT(3) = FZ(K)*(R1 + U(J)*(R2-R1))*COS(THETA)
               PTS(I,J,1) = PT(1)
               PTS(I,J,2) = PT(2)
               PTS(I,J,3) = PT(3)
400         CONTINUE
            THETA = THETA + DEL
500      CONTINUE
         CALL FSTCP(U,W,PTS,PNTS)

         DO 550 I = 1,4
            WRITE(10,*)(PNTS(I,J,1),J=1,4)
            WRITE(10,*)(PNTS(I,J,2),J=1,4)
            WRITE(10,*)(PNTS(I,J,3),J=1,4)
            WRITE(10,*)(H0(1,J),J=1,4)
550      CONTINUE
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

600   CONTINUE

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE SLAB

```
************************************************************************
************************************************************************
**    SUBROUTINE  SLAB(L,B,H,LOCA,ORI,COLOR,BOOL)                     **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRES           **
**    FOR CREATING A SLAB USING B-SPLINE SURFACES.                    **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    L     = LENGTH OF THE SLAB                                      **
**    B     = BREADTH OF THE SLAB                                     **
**    H     = HEIGHT OF THE SLAB                                      **
**    LOCA  = LOCATION OF THE SLAB                                    **
**    ORI   = ORIENTATION OF THE SLAB                                 **
**    COLOR = COLOR TO BE GIVEN TO THE SLAB                           **
**    BOOL  = BOOLEAN OPERATOR FOR THE SLAB                           **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE SLAB(L,B,H,LOC,ORI,ICOL,BOOL)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 L, B, H, LOC(3), ORI(3), H0(20,20)
      REAL*4 F2(2), F4(4), X(4), Y(4), Z(4), PTS(3), PNTS(20,20,3)
      REAL*4 PLOC(3), PORI(3)
      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8
      COMMON/PATCH/NU,NW
```

```
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA F2/0.0,1.0/
      DATA F4/-1.0,0.0,1.0,2.0/
      DATA H0/400*1.0/

      INNUM = INNUM + 1

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      CALL FDINS(INNUM,LTYPE,'SLAB     ',6,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

*CREATE THE PATCHES FOR Z = CONSTANT

      X(1)  = -L
      X(2)  = -0.0
      X(3)  =  L
      X(4)  =  2.0*L

      DO 300 J = 1,2
         WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
         DO 200 I = 1,4
            DO 100 K = 1,4
               Y(K) = F4(I)*B
               Z(K) = F2(J)*H
               PTS(1) = X(K)
               PTS(2) = Y(K)
               PTS(3) = Z(K)
               PNTS(I,K,1) = PTS(1)
               PNTS(I,K,2) = PTS(2)
               PNTS(I,K,3) = PTS(3)
100         CONTINUE
            WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
            WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
            WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
            WRITE(10,*)(H0(1,KL),KL=1,4)
200      CONTINUE
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)
300   CONTINUE

*CREATE THE PATCHES FOR Y = CONSTANT

      X(1)  = -L
      X(2)  = -0.0
      X(3)  =  L
      X(4)  =  2.0*L

      DO 600 J = 1,2
         WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
         DO 500 I = 1,4
            DO 400 K = 1,4
               Y(K) = F2(J)*B
               Z(K) = F4(I)*H
               PTS(1) = X(K)
               PTS(2) = Y(K)
               PTS(3) = Z(K)
               PNTS(I,K,1) = PTS(1)
               PNTS(I,K,2) = PTS(2)
               PNTS(I,K,3) = PTS(3)
400         CONTINUE
            WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
            WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
            WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
            WRITE(10,*)(H0(1,KL),KL=1,4)
500      CONTINUE
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)
600   CONTINUE

*CREATE THE PATCHES FOR X = CONSTANT

      Y(1)  = -B
      Y(2)  =  0.0
      Y(3)  =  B
      Y(4)  =  2.0*B

      DO 900 J = 1,2
         WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
         DO 800 I = 1,4
            DO 700 K = 1,4
               X(K) = F2(J)*L
               Z(K) = F4(I)*H
               PTS(1) = X(K)
               PTS(2) = Y(K)
               PTS(3) = Z(K)
               PNTS(I,K,1) = PTS(1)
```

```
               PNTS(I,K,2) = PTS(2)
               PNTS(I,K,3) = PTS(3)
700         CONTINUE
            WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
            WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
            WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
            WRITE(10,*)(H0(1,KL),KL=1,4)
800      CONTINUE
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)
900   CONTINUE

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE SPHERE

```
************************************************************************
************************************************************************
**    SUBROUTINE SPHERE(RAD,LOC,ICOL,BOOL)                            **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE COMPUTES THE CONTROL POINTS THAT WOULD BE          **
**    REQUIRED TO CREATE A SPHERE USING B-SPLINE SURFACES             **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/21/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    RAD  = RADIUS OF THE SPHERE (REAL,I/P)                          **
**    LOC  = LOCATION OF THE SPHERE (REAL,I/P)                        **
**    ICOL = COLOR FOR SPHERE (INTEGER,I/P)                           **
**    BOOL = BOOLEAN TYPE FOR SPHERE (CHARACTER,I/P)                  **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE SPHERE(RAD,LOC,ICOL,BOOL)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 RAD, LOC(3), PT(3), PTS(4,4,3), ORI(3), LOCB(3)
      REAL*4 U(4), W(4), PNTS(20,20,3), H0(20,20), YS(2)
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA U/0.0,.33333333,.66666667,1.0/
      DATA W/0.0,.33333333,.66666667,1.0/
      DATA H0/400*1.0/
      DATA YS/1.0,-1.0/
      DATA LOCB/3*0.0/

      PI = 3.14159

      INNUM = INNUM + 1

      ORI(1) = 0.0
      ORI(2) = 0.0
      ORI(3) = 0.0

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      CALL FDINS(INNUM,LTYPE,'SPHERE  ',8,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      ORI(1) = 0.0
      ORI(3) = 0.0
      DELTA = PI/6.0

      DO 600 IK = 1,2
         THETA = 0.0
         ORI(2) = 0.0
         DO 200 I = 1,4
            ALPHA = 0.0
```

```
            DO 100 J = 1,4
               PT(1) = RAD*COS(THETA)*COS(ALPHA)
               PT(2) = YS(IK)*RAD*SIN(THETA)
               PT(3) = RAD*COS(THETA)*SIN(ALPHA)
               CALL FTRNS(LOCB,ORI,PT)
               PTS(I,J,1) = PT(1)
               PTS(I,J,2) = PT(2)
               PTS(I,J,3) = PT(3)
               ALPHA = ALPHA + DELTA
100         CONTINUE
            THETA = THETA + DELTA
200      CONTINUE
         CALL FSTCP(U,W,PTS,PNTS)
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

         WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
         DO 250 I = 1,4
            WRITE(10,*)(PNTS(I,J,1),J=1,4)
            WRITE(10,*)(PNTS(I,J,2),J=1,4)
            WRITE(10,*)(PNTS(I,J,3),J=1,4)
            WRITE(10,*)(H0(1,J),J=1,4)
250      CONTINUE

         ORI(2) = 90.0
         DO 500 K = 1,3
            DO 400 I = 1,4
               DO 300 J = 1,4
                  PT(1) = PTS(I,J,1) - LOCB(1)
                  PT(2) = PTS(I,J,2) - LOCB(2)
                  PT(3) = PTS(I,J,3) - LOCB(3)
                  CALL FTRNS(LOCB,ORI,PT)
                  PTS(I,J,1) = PT(1)
                  PTS(I,J,2) = PT(2)
                  PTS(I,J,3) = PT(3)
300            CONTINUE
400         CONTINUE
            CALL FSTCP(U,W,PTS,PNTS)
            CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

            WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
            DO 450 I = 1,4
               WRITE(10,*)(PNTS(I,J,1),J=1,4)
               WRITE(10,*)(PNTS(I,J,2),J=1,4)
               WRITE(10,*)(PNTS(I,J,3),J=1,4)
               WRITE(10,*)(H0(1,J),J=1,4)
450         CONTINUE
500      CONTINUE
600   CONTINUE

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

SUBROUTINE TUNNEL

```
************************************************************************
************************************************************************
**    SUBROUTINE  TUNNEL(R1,R2,THK,LOCA,ORI,COLOR,BOOL,ICHK,PLACE)    **
**                                                                    **
**    PROGRAM DESCRIPTION                                             **
**                                                                    **
**    THIS ROUTINE WILL COMPUTE THE CONTROL POINTS FOR                **
**    CREATING A TUNNEL USING B-SPLINE SURFACES                       **
**                                                                    **
**                                                                    **
**    BY:       ASHIT R. GANDHI                                       **
**    DATE:     11/12/88                                              **
**                                                                    **
**    PARAMETERS USED:                                                **
**                                                                    **
**    R1     = INNER RADIUS OF THE TUNNEL                             **
**    R2     = OUTER RADIUS OF THE TUNNEL                             **
**    THK    = THICKNESS OF THE TUNNEL                                **
**    LOCA   = LOCATION OF THE TUNNEL                                 **
**    ORI    = ORIENTATION OF THE TUNNEL                              **
**    COLOR  = COLOR TO BE GIVEN TO THE TUNNEL                        **
**    BOOL   = BOOLEAN OPERATOR FOR THE TUNNEL                        **
**    ICHK   = FLAG FOR NEW FEATURE                                   **
**    PLACE  = LOCATION OF FEATURE IN LOCAL AXIS                      **
**                                                                    **
************************************************************************
************************************************************************
      SUBROUTINE TUNNEL(R1,R2,LEN,LOC,ORI,ICOL,BOOL,ICHK,PLACE)

      INTEGER*4 ICOL
      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 LOC(3), ORI(3), PT(3), PTS(4,4,3)
      REAL*4 R1, R2, LEN
      REAL*4 PLOC(3), PORI(3), PLACE
```

```
      REAL*4 PNTS(20,20,3), HO(20,20), FX(4), FY(4), FZ(4), U(4), W(4)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA FX/-1.0, 0.0, 1.0, 2.0/
      DATA FY/ 1.0, 1.0,-1.0,-1.0/
      DATA FZ/ 1.0,-1.0,-1.0, 1.0/
      DATA U/0.0, 0.33333333, 0.66666667, 1.0/
      DATA W/0.0, 0.33333333, 0.66666667, 1.0/
      DATA HO/400*1.0/

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

*MAKE SURE THAT OUTER RADIUS IS GREATER THAN INNER RADIUS

      IF (R1 .GT. R2)THEN
         WRITE(6,*)'RING  ====>  INNER RADIUS IS GREATER THAN'
         WRITE(6,*)'RING  ====>  OUTER RADIUS'
         WRITE(6,*)'RING  ====>  EXCHANGING RADII'
         R3 = R1
         R1 = R2
         R2 = R3
      ENDIF

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      INNUM = INNUM + 1

      CALL FDINS(INNUM,LTYPE,'TUNNEL  ',10,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

      PI = 3.14159

      DEL = PI/6.0

*COMPUTE THE HOLLOW CYLINDER

      DO 400 II = 1,2
         RAD = R1
         IF (II .EQ. 2)RAD = R2
         DO 300 K = 1,2
            WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
            THETA = 0.0
            DO 200 I = 1,4
               DO 100 J = 1,4
                  PT(1) = U(J)*LEN + PLACE
                  PT(2) = FY(K)*RAD*SIN(THETA)
                  PT(3) = FZ(K)*RAD*COS(THETA)
                  PTS(I,J,1) = PT(1)
                  PTS(I,J,2) = PT(2)
                  PTS(I,J,3) = PT(3)
100            CONTINUE
               THETA = THETA + DEL
200         CONTINUE
            CALL FSTCP(U,W,PTS,PNTS)
            CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)

            DO 250 I = 1,4
               WRITE(10,*)(PNTS(I,J,1),J=1,4)
               WRITE(10,*)(PNTS(I,J,2),J=1,4)
               WRITE(10,*)(PNTS(I,J,3),J=1,4)
250            WRITE(10,*)(HO(1,J),J=1,4)

300      CONTINUE

400   CONTINUE

*COMPUTE THE RING

      DO 900 II = 1,2
         PLACE1 = 0.
         IF (II.EQ. 2)PLACE1 = LEN
         DO 800 K = 1,2
            WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
            THETA = 0.0
            DO 600 I = 1,4
               DO 500 J = 1,4
                  PT(1) = 0.0 + PLACE1
                  PT(2) = FY(K)*(R1 + U(J)*(R2-R1))*SIN(THETA)
                  PT(3) = FZ(K)*(R1 + U(J)*(R2-R1))*COS(THETA)
                  PTS(I,J,1) = PT(1)
                  PTS(I,J,2) = PT(2)
                  PTS(I,J,3) = PT(3)
500            CONTINUE
```

```
               THETA = THETA + DEL
600            CONTINUE
               CALL FSTCP(U,W,PTS,PNTS)

               DO 700 I = 1,4
                  WRITE(10,*)(PNTS(I,J,1),J=1,4)
                  WRITE(10,*)(PNTS(I,J,2),J=1,4)
                  WRITE(10,*)(PNTS(I,J,3),J=1,4)
                  WRITE(10,*)(H0(1,J),J=1,4)
700            CONTINUE
               CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)

800         CONTINUE

900      CONTINUE

*DRAW THE BASE SURFACES

         DELZ = (R2 - R1)/3.0

         DO 1300 II = 1,2
            WRITE(10,10)INNUM,'4 4 0 0',ICOL,NU,NW
            DO 1100 J = 1,4
               Z = R1
               DO 1000 I = 1,4
                  PTS(I,J,1) = U(J)*LEN
                  PTS(I,J,2) = 0.0
                  PTS(I,J,3) = FZ(II)*Z
                  Z = Z + DELZ
1000           CONTINUE
1100        CONTINUE
            CALL FSTCP(U,W,PTS,PNTS)
            DO 1200 I = 1,4
               WRITE(10,*)(PNTS(I,J,1),J=1,4)
               WRITE(10,*)(PNTS(I,J,2),J=1,4)
               WRITE(10,*)(PNTS(I,J,3),J=1,4)
               WRITE(10,*)(H0(1,J),J=1,4)
1200        CONTINUE
            CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)
1300     CONTINUE

         RETURN
10       FORMAT(1X,I5,1X,A7,3(1X,I2))
         END
```

SUBROUTINE WEDGE

```
**********************************************************************
**********************************************************************
**     SUBROUTINE WEDGE(L,B,H,LOCA,ORI,COLOR,BOOL)                  **
**                                                                  **
**     PROGRAM DESCRIPTION                                          **
**                                                                  **
**     THIS ROUTINE WILL COMPUTE THE CONTROL POINTS REQUIRED FOR    **
**     CREATING A WEDGE USING B-SPLINE SURFACES.                    **
**                                                                  **
**                                                                  **
**     BY:       ASHIT R. GANDHI                                    **
**     DATE:     11/12/88                                           **
**                                                                  **
**     PARAMETERS USED:                                             **
**                                                                  **
**     L      = LENGTH OF THE WEDGE                                 **
**     B      = BREADTH OF THE WEDGE                                **
**     H      = HEIGHT OF THE WEDGE                                 **
**     LOCA   = LOCATION OF THE WEDGE                               **
**     ORI    = ORIENTATION OF THE WEDGE                            **
**     COLOR  = COLOR TO BE GIVEN TO THE WEDGE                      **
**     BOOL   = BOOLEAN OPERATOR FOR THE WEDGE                      **
**                                                                  **
**********************************************************************
**********************************************************************
      SUBROUTINE WEDGE(L,B,H,LOC,ORI,ICOL,BOOL)

      INTEGER*4 ASNUM, OBNUM, INNUM, LTYPE, CNUM

      REAL*4 L, B, H, LOC(3), ORI(3), H0(20,20), FU(4), U(4), W(4)
      REAL*4 FZ(2), F4(4), X(4), Y(4), Z(4), PTS(3), PNTS(20,20,3)
      REAL*4 SPTS(4,4,3)
      REAL*4 PLOC(3), PORI(3)

      CHARACTER BOOL*1, ASSEM(900)*8, OBJECT(9000)*8

      COMMON/PATCH/NU,NW
      COMMON/IDS/ASNUM,OBNUM,INNUM
      COMMON/COMP/ASSEM,OBJECT
      COMMON/PRTOF/CNUM
      COMMON/REND/PLOC,PORI

      DATA F2/0.0,1.0/
```

```
      DATA F4/-1.0,0.0,1.0,2.0/
      DATA H0/400*1.0/
      DATA FU/1.0,0.6666667,0.3333333,0.0/
      DATA U/0.0,0.33333333,0.6666667,1.0/
      DATA W/0.0,0.33333333,0.6666667,1.0/

      DO 50 I = 1,3
         PLOC(I) = LOC(I)
         PORI(I) = ORI(I)
50    CONTINUE

      IF (BOOL .EQ. '+')LTYPE = 1
      IF (BOOL .EQ. '-')LTYPE = 2
      IF (BOOL .EQ. '*')LTYPE = 3

      INNUM = INNUM + 1

      CALL FDINS(INNUM,LTYPE,'WEDGE    ',5,CNUM,
     1           OBJECT(CNUM-1000),LOC,ORI)

*CREATE THE PATCHES FOR Z = CONSTANT

      X(1)  = 0.0
      X(2)  = L/3.0
      X(3)  = 2.0*L/3.0
      X(4)  = L

      DO 300 J = 1,2
         WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
         DO 200 I = 1,4
            DO 100 K = 1,4
               Y(K) = B*FU(I)*U(K)
               Z(K) = F2(J)*H
               PTS(1) = X(I)
               PTS(2) = Y(K)
               PTS(3) = Z(K)
               SPTS(I,K,1) = PTS(1)
               SPTS(I,K,2) = PTS(2)
               SPTS(I,K,3) = PTS(3)
100         CONTINUE
200      CONTINUE
         CALL FSTCP(U,W,SPTS,PNTS)
         DO 250 I = 1,4
         WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
250      WRITE(10,*)(H0(1,KL),KL=1,4)
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)
300   CONTINUE

*CREATE THE PATCHES FOR Y = CONSTANT

      X(1)  = 0.0
      X(2)  = L/3.0
      X(3)  = 2.0*L/3.0
      X(4)  = L

      DO 600 J = 1,2
         WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
         DO 500 I = 1,4
            DO 400 K = 1,4
               Y(K) = F2(J)*B*FU(I)
               Z(K) = U(K)*H
               PTS(1) = X(I)
               PTS(2) = Y(K)
               PTS(3) = Z(K)
               SPTS(I,K,1) = PTS(1)
               SPTS(I,K,2) = PTS(2)
               SPTS(I,K,3) = PTS(3)
400         CONTINUE
500      CONTINUE
         CALL FSTCP(U,W,SPTS,PNTS)
         DO 550 I = 1,4
         WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
         WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
550      WRITE(10,*)(H0(1,KL),KL=1,4)
         CALL FBPCH(4,4,0,0,PNTS,H0,ICOL,LTYPE)
600   CONTINUE

*CREATE THE PATCHES FOR X = CONSTANT

      Y(1)  = 0.0
      Y(2)  = B/3.0
      Y(3)  = 2.0*B/3.0
      Y(4)  = B

      DO 900 J = 1,1
         WRITE(10,10)INNUM,'4 4 0 0',ICOL, NU, NW
         DO 800 I = 1,4
            DO 700 K = 1,4
               X(K) = 0.0
               Z(K) = FU(I)*H
               PTS(1) = X(K)
               PTS(2) = Y(K)
               PTS(3) = Z(K)
```

```
                SPTS(I,K,1) = PTS(1)
                SPTS(I,K,2) = PTS(2)
                SPTS(I,K,3) = PTS(3)
700          CONTINUE
800       CONTINUE
          CALL FSTCP(U,W,SPTS,PNTS)
      DO 850 I =1,4
          WRITE(10,*)(PNTS(I,KL,1),KL=1,4)
          WRITE(10,*)(PNTS(I,KL,2),KL=1,4)
          WRITE(10,*)(PNTS(I,KL,3),KL=1,4)
850       WRITE(10,*)(HO(1,KL),KL=1,4)
          CALL FBPCH(4,4,0,0,PNTS,HO,ICOL,LTYPE)
900   CONTINUE

      RETURN
10    FORMAT(1X,I5,1X,A7,3(1X,I2))
      END
```

Appendix C - FeatureMod User's Guide

Overview

This guide is designed to document the use of **FeatureMod** by answering the following questions:

- How is **FeatureMod** software accessed?
- What requirements must be met to run **FeatureMod**?
- What functions can **FeatureMod** perform and how are they performed?

The information contained in this guide is organized by functional topic such as how to run **FeatureMod**, how to create and retrieve model assemblies, how to generate shaded images, etc. The reader is assumed to be familiar with the logon procedure for VPI&SU's VM3 system and have access to a VM3 userid.

Functional Capabilities of FeatureMod

In the current version of FeatureMod a user can perform the following functions:

- interactive creation of a geometric model using features
- display model geometry as a three dimensional surface image
- rotate and scale the three dimensional image using valuator dials
- display multiple views of the model
- produce hidden surface image of the model assembly
- produce a shaded image of the model assembly
- change the red, blue and green fractions of the first displayable workstation color
- interactively store and retrieve models

All of the above functions are discussed in the "FeatureMod Functions" section of this user's guide.

FeatureMod Run-time Requirements

The following are the requirements to run **FeatureMod** on VPI&SU's VM3 system.

Userid Requirements

To run **FeatureMod** on a VPI&SU VM3 userid, the userid must have:

- access to at least 12 megabytes of virtual storage (memory)
- 25 loader tables defined

File Requirements

The following files are need to successfully run **FeatureMod:**

Feature Exec
is the command language program used to run FeatureMod

Feature Text
is the text file containing all modules that form FeatureMod

Hardware Requirement

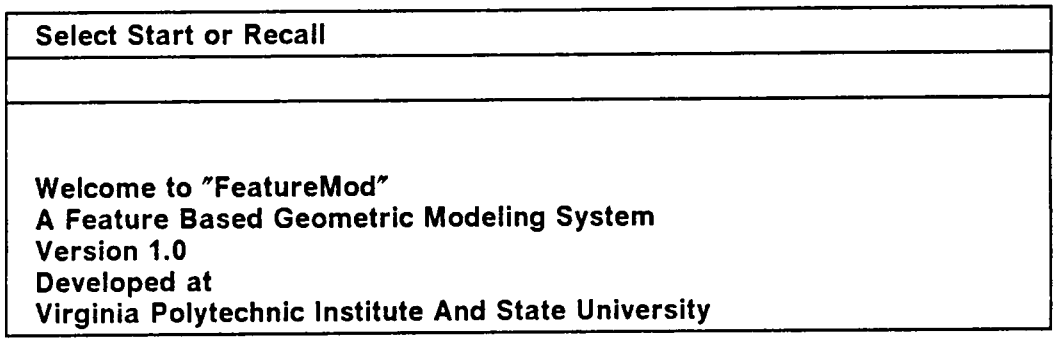
FeatureMod currently runs only on the IBM 5080 workstation. Thus an IBM 5080 must be logically attached to your userid.

Running FeatureMod

If you have met all the requirements listed in the previous section you can run **FeatureMod** by typing the following command:

FEATURE

FEATURE is a command language program which performs all the actions necessary to link, load and run the FeatureMod software. The FeatureMod main menu screen should appear on the IBM 5080 screen. Depending on the current load on the system it may take up to one minute for this screen to appear. When the menu does appear the following messages will be printed in the grey message area at the bottom of the screen:



The first step is to start a new model file or to recall a previously created model file.

FeatureMod Functions

Overview

The following pages contain details of how to use the functions currently available in FeatureMod. Under each function the following format is used to explain the function's use:

Purpose
explains the purpose of the function.

Menu Path
shows the nesting of the menu items which lead to the desired function.

Description
describes the function's use including an example of the FeatureMod message, prompt and input areas as shown below:

Prompt Area
Input Area
message line 9 **message line 8** **message line 7** **message line 6** **message line 5** **message line 4** **message line 3** **message line 2** **MESSAGE LINE 1 (CURRENT MESSAGE)**

Example
provides an example of the function's use.

Starting A New Model

Purpose
To activate a new file for interactive model creation

Menu Path
FILES (START)

Description
When you select the "START" menu item, you will be asked to enter the name for the new model as shown below:

Enter Name Of Model To Be Started
filename
Started Model filename

You must type in a file name with maximum four characters. If less than four characters are used zeros will be post-concatenated to the name. The file name is actually the CMS *filetype*. The CMS filename and filemode are "FILE" and "A1" respectively.

If the model name you entered already exists on your userid, you will receive a "CANNOT START EXISTING MODEL" message and will be asked to reenter a new filename. If the file is not found then you will receive the message "STARTED MODEL filename".

Example
Enter "TRY" to start a new model file. You should get the "STARTED MODEL TRY0" prompt.

Reading Existing Model Files

Purpose
To load a previously defined model file.

Menu Path
FILES (RECALL)

Description
When you select the "RECALL" menu item, you will be asked to enter the name of a previously created model file as shown below:

Enter Name Of Model To Be Recalled
filename
Recalling Model filename **Model Recalled**

You must type in a file name. The file name is actually the CMS *filetype*. The CMS filename and filemode are "FILE" and "A1" respectively.

If the model name you entered does not exist on your userid, you will receive a "CANNOT RECALL NON-EXISTENT MODEL" message and will be asked to reenter a new filename. If the file is found then you will receive the message "RECALLING MODEL filename".

Example
Enter "PAWN" to recall the model that contains the geometry for a chess pawn. After recalling the model you should get the "MODEL RECALLED" prompt.

Defining An Object

Purpose

To allow the definition of an object name and its creation by using feature instances

Menu Path

COMMANDS (MODELING (DEFINE (SLAB / CYLINDER / ...)))

Description

When you select the "DEFINE" menu item, you will be asked to enter the name for the new object. The process for object definition is as shown below:

Enter Name Of Objected To Be Defined
objectname
Defined Object "objectname"

Once the name is input, it is checked for previously existing assemblies or objects. If the component already exists then you will receive a "OBJECT ALREADY EXISTS" message and will be asked to reenter a new object name. The command can be cancelled by entering "CANCEL". Once the object is defined, a new menu page will allow the selection and creation of different features that will form part of the defined object.

Example

Enter "MOUNT" to define a new object. You should get the "DEFINED OBJECT MOUNT" prompt.

Defining An Assembly

Purpose

To allow the definition of an assembly name and its creation by using existing objects.

Menu Path

COMMANDS (MODELING (ASSEMBLE))

Description

When you select the "ASSEMBLE" menu item, you will be asked to enter the name for the new assembly. The process for assembly definition is as shown below:

Enter Name Of Assembly To Be Defined
assemname
Defined Assembly "assemname"

Once the name is input, it is checked for previously existing assemblies or objects. If the component already exists then you will receive a "ASSEMBLY ALREADY EXISTS" message and will be asked to reenter a new assembly name. The command can be cancelled by entering "CANCEL". Once the assembly is defined, a template will allow the selection of different objects that will form part of the defined assembly.

Example

Enter "SEAT" to define a new assembly. You should get the "DEFINED ASSEMBLY SEAT" prompt.

Adding To an Object or Assembly

Purpose

To allow addition to existing objects or assemblies.

Menu Path

COMMANDS (MODELING (ADDTO))

Description

When you select the "ADDTO" menu item, you will be asked to enter the name for the component that must be added to. The process for adding to a component is as shown below:

Enter Name Of Component To Be Added To
compname
Adding To "compname"

Once the name is input, it is checked for existing assemblies or objects. If the component does not exist then you will receive a "COMPONENT DOES NOT EXIST" message and will be asked to reenter a new component name. The command can be cancelled by entering "CANCEL". Once the required name is obtained the appropriate menu page or template is displayed.

Example

Enter "SEAT" to delete the previously defined assembly. You should get the "DELETED ASSEMBLY SEAT" prompt.

Deleting an Object or Assembly

Purpose

To allow deletion of objects or assemblies.

Menu Path

COMMANDS (MODELING (DELETE))

Description

When you select the "DELETE" menu item, you will be asked to enter the name for the component that must be deleted. The process for component deletion is as shown below:

Enter Name Of Component To Be Deleted
compname
Deleted "compname"

Once the name is input, it is checked for existing assemblies or objects. If the component does not exist then you will receive a "COMPONENT DOES NOT EXIST" message and will be asked to reenter a new component name. The command can be cancelled by entering "CANCEL". Once the required name is obtained the appropriate component is removed from the object hierarchy and deleted from the screen.

Example

Enter "SEAT" to add to the previously defined assembly. You should get the "ADDING TO ASSEMBLY SEAT" prompt.

Displaying an Object or Assembly

Purpose

To allow the display of a single object or assembly.

Menu Path

COMMANDS (DISPLAY)

Description

When you select the "DISPLAY" menu item, you will be asked to enter the name for the component that must be displayed. The process for assembly definition is as shown below:

Enter Name Of Component To Displayed
compname
Displayed "compname"

Once the name is input, it is checked for existing assemblies or objects. If the component does not exist then you will receive a "COMPONENT DOES NOT EXIST" message and will be asked to reenter a new component name. The command can be cancelled by entering "CANCEL". Once the required name is obtained the appropriate component is displayed.

Example

Enter "SEAT" to add to the previously defined assembly. You should get the "DISPLAYED ASSEMBLY SEAT" prompt.

Model Inquiry

Purpose

To allow the inquiry of total number of objects and assemblies in the model and their names

Menu Path

COMMANDS (INQUIRE)

Description

When you select the "INQUIRE" menu item, you will get the total number of objects and assemblies in the message area. Their names and structure ids are displayed on the text terminal. The message area is as shown below:

Select Option
Number Of Assemblies: 2 **Number Of Objects: 4**

Information of the objects that form a particular assembly is also displayed on the text terminal.

Example

The above message indicates that the model has 2 assemblies and 4 objects.

Displaying Multiple Views

Purpose
To allow the display of four primary views of a component

Menu Path
COMMANDS (WINDOWS (MULTIPLE))

Description
When you select the "MULTIPLE" menu item, you will get the four primary views of the component that is currently being displayed.

Displaying Single Views

Purpose
To allow the display of primary views of a component

Menu Path
COMMANDS (WINDOWS (SINGLE (FRONT / TOP / ....)))

Description
When you select the menu item, you will get the appropriate primary view of the component that is currently being displayed. Once the required view has been selected the viewing matrix is reset to the viewing parameters specified for that view.

Displaying A Hidden Surface Image

Purpose

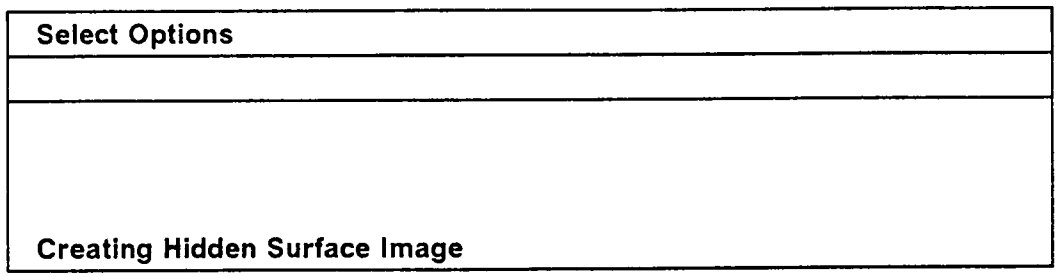
To allow the display of a hidden surface image of the entire model.

Menu Path

COMMANDS (RENDERING (HIDDEN SURFACE))

Description

When you select the "HIDDEN SURFACE" menu item, processing for image generation will begin. The message screen will be as shown below:



Once the image has been generated you will receive a "HIDDEN SURFACE IMAGE CREATED". To reset the image use menu item "RESET".

Displaying A Shaded Image

Purpose

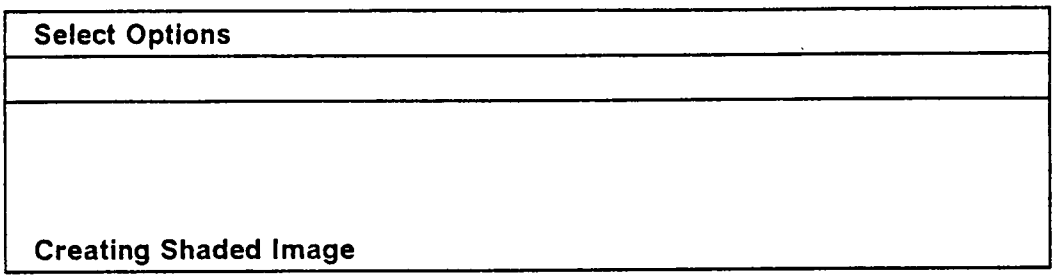
To allow the display of a shaded image of the entire model.

Menu Path

COMMANDS (RENDERING (SHADING (CONSTANT)))

Description

When you select the "CONSTANT" menu item, processing for image generation will begin. The message screen will be as shown below:



Once the image has been generated you will receive a "SHADED IMAGE CREATED". To reset the image use menu item "RESET".

Performing Three Dimensional Viewing Transformations

Purpose

To rotate and zoom the three dimensional image of the model

Menu Path

Any menu that is not requesting numeric or string input

Description

The workstation valuator dials are used to perform viewing transformations and are mapped as follows:

- dial 1 - global x-axis rotation
- dial 2 - global y-axis rotation
- dial 3 - global z-axis rotation
- dial 4 - view scale

If multiple views of the model are displayed then the viewing transformations are applied to the current view.

Appendix D - Feature Inquiry Program

Purpose

To allow a user to query the feature dictionary for information about individual features.

Description

This program requires three files for successful execution:

- **INFO EXEC** is the command language program
- **INFO TEXT** is the module containing commands that can be processed
- **INFO DATA** contains information about individual features

Running the Program

You can run the program by typing **INFO**. Once execution starts you will get a prompt asking for the feature you wish to inquire about. Once you have entered the name of the feature, you will be prompted for the kind of information that you need. The following words may be use to get the required information:

- **FEATURE** provides the name of the feature.
- **ENGLISH** provides the definition of the feature.
- **GROUP** provides the classification of the feature according to topology and shape-form
- **SYNONYM** provides a list of synonym features.
- **RELATIONS** provides a list of mathematical relations that govern the feature parameter ratios.
- **DEFINING** provides a list of parameters that define the feature.
- **END** to exit from the program.

The program requires only the first three characters of the keywords.

The vita has been removed from the scanned document