The Geometric Modelling System DMI.

Report LiSI-90-36

Pere Brunet, Robert Juan, Pau Planas, Enric Torres
Dept. de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya

THE GEOMETRIC MODELLING SYSTEM DMI

1 GENERAL DESCRIPTION

Generally speaking, the DMI System is a set of three closely coherent parts:

- an interface facilitating the user's working with the system. This interface contains a command interpreter, an information input/output manager for the DMI System, and a window manager which gives the system its outward appearance;

- a data structure being able to store the information necessary for the 3D characterization of objects;

- a set of modules carrying out a wide range of 2D and 3D operations and computations with the define objects inthe data structure

The DMI System uses a boundary representation (BR) of the objects with an approximation by means of plane faces. Thus, the topological and geometrical information of the defined solids is available in any case. As a solid modelling system, the DMI System allows the rendering of non-plane surfaces, but always through a polyhedral approximation. The exactness of the approximation is a parameter which can be adjusted by the user. The generation of the solids can basically be performed in three different ways:

- the creation of modelled objects by sweeping 2D polygons which have been defined with the 2D subsystem, following a direction in the space (parallel, vectorial, conic or circular sweeping);

- the construction of objects by means of their characteristic parameters (geometric solids, as, e. g., sphere, cylinder or cone, or also derived solids, as, e. g., valves, nozzles, tanks, etc.);

- 3D solid/solid operations among any of those mentioned above which also construct modelled objects.

2 THE PHASES OF A DESIGN PROCESS WITH THE SYSTEM

The design process is divided into five working phases:

1 - Creation
2 - Modification
3 - Visualization

4 - Calculation of geometric and volumetric properties
5 - Connection with other systems

The first design phase is the CREATION or generation of 3D objects using the modelling techniques provided by the system.

Once the object has been created, the system allows the MODIFICATION of both its form and its external characteristics, position in the space, etc.

The CREATION of an object can be performed not only by using the previously mentioned techniques but also by performing MODIFICATIONS of other previously generated objects.

The objects which have already been generated with the system, and even those which are still in the generation phase, can be displayed with different visualization techniques providing the user with an image of his design with different realism levels.

The system includes a serie of options which allow the calculation of GEOMETRIC AND VOLUMETRIC PROPERTIES of the generated objects.

Finally, the system provides the CONNECTION of its designs with specific OTHER SYSTEMS which develop work being complementary to solid modelling.

2.1 OBJECT CREATION: MULTIPLE SWEEPING TECHNIQUE

In solid modelling there are several techniques of generating objects in the space. The one used in this system is the sweeping technique, characterized by its easy use and by the broad range of forms it allows to generate.

The sweeping consists of a solid's generation as a result of moving a plane polygon in the space.

The multiple sweeping technique is composed of two phases: During the first one the initial sweeping is effected; a 3D object is obtained. In the second phase plane polygons are defined based on the faces of the object; by sweeping those polygons, holes and protusions on the solid are produced, figures 1, 2 and 3.

In the initial sweeping phase, a polygon is defined based on any plane of the space, the so-called working plane. In two dimensions and using the system's 2D editor, the edition of one or several polygons is performed based on this working plane. Finally, the solid object is generated as a result "of sweeping" the space by displacing the polygons.

2.2 GEOMETRIC MODIFICATIONS OF THE OBJECTS

Modifications of objects can be understood as a set of operations which can be performed on previously created objects. Summarized, the system allows the realization of the following modifications:

Geometric modifications:

- Section of one or more objects
- Splitting one object into two or more objects
- Union (glueing) of two objects
- Elimination of a sweep
- Modification of the scale
- Translation of objects in the space
- Rotation of objects in the space
- Boolean operations between two objects

Non-geometric modifications:

- Name of the object
- Colour of the object
- Colour of one face of one object
- Flag of cylinder/prism
- Level of an object
- Density of an object
- Attribute of an object

Every modification is performed based on the set of objects being selected in that very moment (see section 4.1).

In particular, the most important operations are:

- **Geometric transformations:** An object can be properly translated, rotated or scaled in any direction.

- **Copy:** Once the geometry of an object is defined in the accumulators, it is easy to obtain as many objects as desired.

- **Symmetry:** By defining an object and a symmetry plane, the reflected image of the first solid is obtained.

- **Mould:** The complementary of an object or set of objects is generated.

- **Holes and/or protusions:** Within any face of the object it is possible to define holes and protusions by sweeping polygons, figure 2.

- **Glue:** Starting from two objects with coincident faces,a third object is generated by glueing them.

- **Sections:** A set of objects is splitted by a plane. The resulting objects are included in the system.

- **Boolean operations:** Operations of union, difference and intersection are carried out by means of an intermediate octree model. This requires the conversion among the BR and octree models in both directions, before and after the realization of the Boolean operation [1, 2, 3].

Some modifications are performed based on one object (e. g., Undo Sweeping), others based on two (e. g., Glueing), and others based on as many objects as desired (e. g., Change of Colour). In any case, the system warns the user if the number of selected objects is not correct.

2.3 COMPUTATION OF GEOMETRIC PROPERTIES

The system allows to determine the geometric properties of the selected objects, as, e. g., the distance between two points, the area and perimeter of a polygon, and the angle between two directions.

2.4 COMPUTATION OF VOLUMETRIC PROPERTIES

The systems allows to determine the volumetric properties of the selected objects:

- Volume
- Mass
- Coordinates of the gravity centre
- Moments of inertia with respect to the three coordinate axes
- Products of inertia with respect to the three coordinate axes
- Complete surface of the faces
- Complete length of the edges

The command PROPVO performs the calculations as a function of the density of the selected objects, automatically providing their volumetric properties. It is also posible to modify the object's density.

The achieved results correspond to the set of all the selected objects.

3 OBJECT VISUALIZATION

The DMI System produces images of the objects and elements which form part of the design.

The system disposes of a set of commands for the scenes' modification, ranging from varying the observation and illumination positions to using different degrees of realism.

The point the observer looks at is called "pan centre". When drawing the scene, the system makes the pan centre coinciding with the centre of the graphics screen.

The user himself can specify where the observer shall be situated in the space.

The scale with which the set of objects is wanted to be seen is determined by the "zoom factor". This factor allows to extend or reduce the drawing. The part of the drawing which does not fit in the screen will be eliminated by the system.

The different characteristics of the visualization are grouped in the following form:

- visualization mode
- visualization system
- visualization status
- visualization parameters

The visualization mode references the set of objects to be visualized, the visualization system determines the representation of these objects, the visualization status defines the system's behaviour with regard to zoom and perspective, and the parameters modify certain concrete visualization aspects.

3.1 VISUALIZATION MODE

As I will be explained in section 4, all objects existing in the system have an associated "status" which can be "INACTIVATED", "ACTIVATED" or "SELECTED".

The visualization mode allows to specify which objects are to be rendered as a function of their "status". This provides the visualization of only the selected, of only the activated, or of all objects.

The commands for changing the visualization mode are:

VEU-TOT
VEU-ACT
VEU-SEL

VEU-TOT makes all the objects visible. The selected ones appear in their colour, the activated ones in grey, and the inactivated objects in grey and outlined. Only by executing this command the inactivated objects can be visualized.

VEU-ACT makes the activated objects visible (the selected ones are a subset of the activated ones and therefore also visible). The activated objects are displayed in grey, and the selected ones in their particular colour.

VEU-SEL makes only the currently selected objects visible, in their own colour.

The selected objects are always visible, the activated ones only in "VEU-TOT" and "VEU-ACT" mode, and the inactivated ones only in "VEU-TOT" mode.

Visualization of inactivated objects is only performed when executing the command VEU-TOT. Since these objects are not to be found in the Ram storage, they disappear from the screen when a system parameter is modified, and the visualization command has to be invoked again. It is impossible to use the inactivated objects as a reference for the selection of points, faces or edges.

3.2 VISUALIZATION SYSTEMS

An important tool in the system is the possibility to use different visualization techniques with several degrees of realism (obviously, the more realism, the more computational costs), so that it is always possible to work with the most adequate one(Figures 4, 5, 6 and 7). All the methods explained in the following visualize the approximate model of plane faces:

- **Wireframe visualization:** The simplest one. It draws the edges of the solid, and due to its low computation cost it is suitable to to be used during editing sessions.

- **Visualization with hidden lines:** It assumes a higher degree of realism, since it eliminates those segments of the edges which are invisible from the observer's position (Figures 2 and 3).

- **Visualization with plane facets:** Using an algorithm of binary space partition (BSP) [6, 7] it generates the illumination of the plane faces. This method has the advantage that although includes a time-costly preprocessing, as many visualizations from different viewpoints as desired can be done without further computation.

- **Smoothed visualization:** It uses a z-buffer method with Gouraud smoothing (Figure 5).

Independently from the selected method, the user can modify the system's visualization parameters in every moment, the most important ones being the direction of observation, the distance between the observer and the scene, the direction of the illumination and the centre of the image.

The commands to select the visualization systems without illumination are the following:

FIL
FILC
FILT
VSIMPLI
ELO
DELO;

and to select the systems with illumination:

BSP
SUA.

4 MANAGEMENT OF SETS OF OBJECTS

Every design which has been realized with the DMI System can be formed by as many objects as desired, provided that its capacity is not exceeded.

Since, generally, the operations and modifications are performed based on a subset of the total of the objects, the system allows their classification in several status. By attaching to each object the adequate status, the user is enabled to choose the objects on which he wants to perform the operations.

4.1 OBJECT STATUS

The existence of different status allows the user to classify the objects according to their use (operations and modifications). The object status can be modified by the user in every moment.

In order to provide the user with a permanent awareness of the status of every object, they are differently visualized.

One of the following statuses can be assigned to every object: INACTIVATED, ACTIVATED and SELECTED.

SELECTED OBJECTS: They are always visible on the screen in their particular colour. Modification operations can only be performed with selected objects.

ACTIVATED OBJECTS: Activated objects are not affected by a modification command. The function of these objects is to serve as a reference during the design process, since their elements (points, edges, ...) can be taken as geometric data.

INACTIVATED OBJECTS: They cannot be modified and only be visualized by means of the command "VEU-TOT". Generally, the user inactivates those objects which he ceases to use in his design during a certain period of time.

The activated and the selected objects are stored in the main memory, the inactivated objects on the disk, storing only their non-geometric information (attributes) in memory.

Each of the three status assume the preceding one: the activated objects are a subset of the total, the selected objects a subset of the activated ones.

The status change of an object has to be realized by means of the selection and activation commands. Or by the inverse commands for non-selection and inactivation.

5 THE DATA STRUCTURE OF THE DMI SYSTEM

It is vital to understand how the system manages the data belonging to the solids. In fact, two information levels can be distinguished: the first one handles the object like a global entity which forms part of a certain design within the system, whereas the other one is merely descriptive, with the specification of its topology and geometry. Each level is contained in a different data structure.

Each routine of the system can freely access these structures, without the existence of any restriction in the working philosophy defined in the DMI System. Therefore we have to take care that this database is defined in the file GESTIO.INC within the directory SDMI.FONTS, and, thus, the routine supposed to access this information has to make an INCLUDE FORTRAN of this file. Beforehand it is also necessary to include the file PARAM.INC to be found in the directory DRVVAX which defines all the characteristic dimensions of the data structure.

5.1 GRAPHICAL DATABASE (GDB)

The kernel of DMI System's data structure is the GDB. It stores different information which characterizes the objects created by the system, thus allowing a faster management. Each object, once being defined, occupies one entry (line) of the GDB. The line of the GDB belonging to an permanent; it only disappears if the object is erased or the work is completed.

With regard to the GDB we may consider the following sections:

List of Objects (BD*): Contains the information belonging to all designed objects.

List of Solids (CS,CO): Contains the information belonging to the solids in the design. The difference between the concepts of objects and solids in the DMI System has to be stressed. Actually, different objects can share geometrically identical solids, differing only in the applied geometric transformation (translation, rotation).

List of Parameters (BDPARI,BDPARR): Contains the parameters which define the parametric objects in the system.

List of Activated Objects (ACT): List of those objects of the BD* which have the status of being "activated".

List of Selected Objects (SEL): List of those objects of the BD* which have the status of being "selected".

Every object, which exists in the design to be performed in the system, is present in the object list, independently from its state of activation and selection. The names of all vectors and matrices building this list start with "BD". For each object of the system one line of the list is kept. Each line is divided in several fields, some of which have always significance, whereas others are only used if the object can be parameterized.

5.1.1 List of Objects of the GDB

The stored information for each object is as follows:

Field	Type	Description
BDNOM	CHARACTER	Name of the object.
BDTIP	INTEGER	Kind of object (corresponding to its management): 1: Created during the present execution. 2: Object from a previous design. 3: Object from a library.

BDTIPSOL	INTEGER	Kind of object (on a geometric level): TO* are parameters defined in the file G:CONST.INC TOMODE: Modelled solids. TOPARA: Parametric solids. TOPLAN: Flat objects without thickness. TOPAUX: Auxiliary polygons. TOREUN: Set object (includes elementary objects). TOCANO: Pipeline object. TOPREU: Logical linking points. TOSUPE: Surfaces.
BDAPFIT	CHARACTER	Name of file where the object is referenced.
BDAPOB	CHARACTER	Name of object in the previously referenced file.
BDAOBJ	INTEGER	Object position in the temporary file of inactivated objects.
BDTRG	REAL(18)	Geometric transformation and extreme coordinates. In each line of the matrix BDTRG the 12 significant coefficients of the matrix of geometric object transformation are stored, as well as its maximum and minimum coordinates. The transformation matrix is shown as follows:

$$[XYZ1]' = [XYZ1] \cdot \begin{bmatrix} A & B & C & 0 \\ D & E & F & 0 \\ H & I & J & 0 \\ L & M & N & 1 \end{bmatrix}$$

These coefficients are stored in **BDTRG** as follows:

$A =$ **BDTRG**$(,1)$	$B =$ **BDTRG**$(,2)$	$C =$ **BDTRG**$(,3)$
$D =$ **BDTRG**$(,4)$	$E =$ **BDTRG**$(,5)$	$F =$ **BDTRG**$(,6)$
$H =$ **BDTRG**$(,7)$	$I =$ **BDTRG**$(,8)$	$J =$ **BDTRG**$(,9)$
$L =$ **BDTRG**$(,10)$	$M =$ **BDTRG**$(,11)$	$N =$ **BDTRG**$(,12)$

$XMIN =$ **BDTRG**$(,13)$	$XMAX =$ **BDTRG**$(,16)$
$YMIN =$ **BDTRG**$(,14)$	$YMAX =$ **BDTRG**$(,17)$
$ZMIN =$ **BDTRG**$(,15)$	$ZMAX =$ **BDTRG**$(,18)$

BDCOL	INTEGER	Object colour.
BDATRI	CHARACTER	Alphanumeric attribute.
BDDENS	REAL	Density.
BDUNIT	INTEGER	Object units for the coordinates.
BDREUNIO	INTEGER	Set object pointer.
BDCANONA	INTEGER	Object pipeline pointer.
BDNIV	INTEGER	Object level.
BDPRCODI	INTEGER	Code of parametric object.
BDPRAREP	INTEGER	Active schematic representation.

BDPRNREP	INTEGER	Number of schematic representations.
BDPRAPI	INTEGER	Pointer to vectors of integer parameters.
BDCOS	INTEGER	Pointer to the solid list of the GDB.

The dimension of the GDB is determined by the parameter DNU, defined in the file PARAM.INC. Likewise, the variable global FP(1) indicates in every moment the number of objects existing in the GDB.

5.1.2 List of solids of the GDB

The information belonging to the solids (each solid can be shared by more than one object) is stored in a vector ("CS") and a matrix ("CO"). As mentioned before, every object of the GDB references its solid on the field BDCOS, which contains a pointer to the solid's line:

CS	CHARACTER*6	Name of the solid.
CO	INTEGER	Number of objects sharing the solid.

The dimension of the list of solids of the GDB is determined by the parameter DCO, defined in the file PARAM.INC. The global variable FP(2) indicates in every moment the number of different geometric solids being presented in a picture.

5.1.3 List of Parameters

The DMI System allows several methods to generate solids among which it has the capability of instantiating primitives with specific parameters. In the GDB there are two vectors to store the characteristics of these parametric objects:

BDPARI	INTEGER	Array of INTEGER type.
BDPARR	REAL	Array of REAL type.

The dimension of the list of parameters has a default value of 70 times the maximum number of objects the GDB (DNU) may contain. The variables NPARI and NPARR indicate in every moment the number of occupied components, respectively, in the previous arrays.

5.1.4 List of activated objects of the GDB

In this list, implemented in the integer vector "ACT", pointers (line index in the GDB) to all activated objects of the system are stored. The dimension of the list of activated objects of the GDB is determined by the parameter DNU, defined in the file PARAM.INC. The variable "NACT" indicates the number of elements that ACT contains.

5.1.5 List of selected objects

In this list, implemented in the integer vector "SEL", pointers (line index in the GDB) to all selected objects of the system are stored.

The dimension of the list of selected objects of the GDB is determined by the parameter DNU, defined in the file PARAM.INC. The variable "NSEL" indicates the number of elements "SEL" contains.

5.2 ACCUMULATORS

General information about all the objects of the system is stored in the GDB independently from its state of activation and/or selection.

On the other hand, the working philosophy of the DMI System imposes that only the activated objects (and with them the selected ones) can be manipulated. Therefore the system stores its topological and geometrical information in the so-called "accumulators". This information is hierarchically encoded in the following format:

Accumulator of solids (ACCO*): Information belonging to the solids of the activated objects.

Accumulator of solids (ACCA*): Information belonging to the faces of the activated objects.

Accumulator of polygons (ACPO*): Information belonging to the polygons of the activated objects. Actually, the faces of a solid can be defined by several polygons, as, e. g., when plane faces have holes.

Accumulator of edges (ACAR): Information belonging to the edges of the activated objects.

Accumulator of points (ACPU): Information belonging to the points of the activated objects.

Accumulator of visualization points (ACPUVI): Information belonging to the points of the activated objects after having performed visualization transformations.

List of selected objects (ACCOSEL): List of those objects of the accumulator which are selected.

List of visualized objects (ACCOVIS): List of those objects of the accumulator which have to be redrawn.

5.2.1 Hierarchical structure of information

Accumulator of solids: Here the information belonging to the activated objects is stored: topology (pointer to other data) and attributes (colour).

For each solid the following information is stored:

ACCOBD: Pointer to the line the solid occupies in the GDB.

ACCOPC: Pointer to the first face of the solid (pointer to ACCA*).

ACCONC: Number of faces, in ACCA*, of the solid.

ACCOCO: Colour of the solid.

ACCOM3: Pointer to model 3 of the solid. (This model represents the solid after having performed visualization transformations, 3D clipping and perspective transformations.)

ACCOPP, ACCOUP: Limits of the solid's points within the accumulator ACPU.

Accumulator of faces: For each face, geometric (plane equation) and topological (pointer to the polygon) information is stored, as well as its colour and a flag indicating the kind of face concerned.

ACCANO(4,): Components XYZ of the normal vector of the face, and fourth component of the plane equation.

ACCACO: Colour of the face.

ACCAPT: Pointer to the first polygon of the face.

ACCAFL: Vector INTEGER*2 of flags of the face. The bits are numbered from 0 (less significant) to 15 (more significant).

The bits with significance are the following ones:

msb															lsb
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*		*												*	*

bit number 0:	1 if the face is cylindrical- 0 if the face is not cylindrical-
bit number 1:	1 if the face has been generated by the BSP- 0 if the face belongs to the solid-
bit number 13:	1 if the face defines a linking point- 0 if the face does not define a linking point-
bit number 15:	1 if the face is to be eliminated in the next compactation- 0 if the face is not to be eliminated in the next compactation-

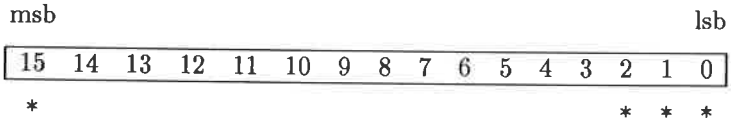
Accumulator of polygons: Mostly an object's face has one singular polygon defining its contour.

However, in those cases where holes or protusions are defined on a face, the number of associated polygons is bigger than one: There is an initial polygon defining the face's contour, and as many polygons as holes or protusions. These polygons are the ones which define the perimeter of the hole or the base of the protusion.

Each face points to its first polygon by means of ACCAPT. If there should exist more polygons, these point to each other by means of the field ACPOSP (next polygon). The last polygon of the face contains ACPOSP=0.

The information stored for each polygon is:

ACPOFL: Flag of the polygon. The bits are numbered from 0(less significant) to 15 (more significant).



bit number 0:	1 if the polygon is inside (hole)- 0 if the polygon is the face's contour-
bit number 1:	1 if it is the root of a solid's perforation- 0 if it is the root of a solid's protuberance- (only makes sense if the bit 0 is on 1)-
bit number 2:	1 if it comes from a DUP3D and if there is no need to redraw it- 0 if it does not come from a DUP3D-
bit number 15:	1 if it has to be eliminated in the next compactation or sweep deletion- 0 if it has not to be eliminated-

ACPOPA: Pointer to the first edge of the polygon (in ACAR).

ACPONA: Number of edges of the polygon in ACAR.

ACPOSP: Pointer to the next polygon (is worth 0 if there is no next one).

Accumulator of edges: Each polygon of an object has a defined arranged sequence of edges, each of which points to the coordinates X, Y, Z of its extreme point, storing the points ACPU in the accumulator.

The cyclic ordering of the edges of a polygon is, on the face of the object, counterclockwise for the first polygon of each face, and clockwise for the rest (bases of holes and protusions).

The number of edges of each polygon is the same as its number of vertices, e. g., a rectangular polygon has ACPONA=4. The last point, pointed to by the last edge, is linked with the first point.

Each edge contains as unique information the pointer to its point of origin.

ACAR: Pointer to the initial point of the edge.

Accumulator of points: For each point, its components are stored, as well as the result of the visualization transformation:

ACPU(3,): Coordinates X, Y, Z of the points.

ACPUVI(3,): Coordinates XP, YP, ZP resulting from applying the visualization transformation to the corresponding point of the ACPU.

The accumulator's dimensions are defined by a series of PARAMETER contained in the file PARAM.INC:

DACCO: Dimension of objects of the accumulator.

DACCO2: Dimension of objects of model 3 accumulator (ACCOBD, ACCOPC, ACCONC).

DACCA: Dimension of faces of the accumulator.

DACPO: Dimension of polygons of the accumulator.

DACAR: Dimension of edges of the accumulator.

DACPU: Dimension of points of the accumulator.

There are also some variables indicating the number of filled lines in the different vectors and matrices which form the accumulators:

FPAC(iac,1): Filled lines in the accumulator of faces.

FPAC(iac,2): Filled lines in the accumulator of polygons.

FPAC(iac,3): Filled lines in the accumulator of edges.

FPAC(iac,4): Filled lines in the accumulator of points.

FPAC(iac,5): Filled lines in the accumulator of solids.

"iac" may be 1 or 2, corresponding to the accumulator concerned.

5.3 CLASSIFICATION ACCORDING TO THE CREATION

As already mentioned above, the system allows the generation of solids of different nature. The variable BDTIPSOL of the GDB indicates the category corresponding to each object according to the following criteria:

- **BDTIPSOL=1:** Corresponds to modelled objects, i. e., generated from sweeping technique, or from other solids by means of some 3D operation. The characterization of the geometry of those objects requires that if a design is saved in a file, the content of the accumulators must be saved, too.

- **BDTIPSOL=2:** Primitive objects, instantiated with the corresponding characteristic parameters. The DMI System disposes of specific generation modules, each of which corresponding to a defined parametric object kind. This speciality simplifies to a large extent the management of these objects, since, e. g., the disk does not need to observe the complete geometry being present in the accumulators; it is sufficient to store the corresponding parameters. When we try to retrieve some of these objects from the disk or to reactivate them, it is only necessary to call the appropriate generation module which then takes the parameter values and generates the object's geometry directly with the accumulator.

- **BDTIPSOL=3:** Objects generated by sweeping, but with thickness 0, used to represent, e. g., sheets. Except for the particularity that they only consist of two faces, they have a management being absolutely identical to the modelled objects.

- **BDTIPSOL=4:** Auxiliary polygons used in different situations in the system (management of pipes, construction of references, ...). In fact, these objects are no solids but they profit from the accumulators by avoiding the unnecessary duplication of data structures.

- **BDTIPSOL=5:** Corresponds to the so-called set objects. These objects are per definition of a logical kind; they contain a set of objects of other categories, allowing the joint manipulation of clusters, Figure 4. Thus, they are the only kind of objects of the DMI System which are not directly linked with an accumulator geometry. The objects belonging to the set include a pointer to the "parent" by means of the field BDREUNIO; they lose all their individual features which can only be retrieved by breaking

up the logical link. Therefore, any attempt to separately select a "child" object, automatically selects the whole set. The sets' management is reduced in order to appropriately manipulate the gathered objects, according to the instructions provided for them.

6 PARAMETRIC OBJECTS

The system provides the automatic generation of simple objects (prisms, cylinders, pyramids, spheres, ...) defined by a set of geometric parameters.

With regard to the other objects, the parametric objects have the advantage that they can easily be generated and modified. Thus, the generation is realized by specifying the object's parameters, and the modification by varying their value (reparameterization).

6.1 DESCRIPTION OF THE PARAMETRIC OBJECTS

Every parametric object is defined by an attribute, a set of geometric parameters, and a reference point.

The attribute of each parametric object is automatically assigned by the system at the moment of its generation. The attributes of the different parametric objects which can be defined are the following:

Parametric Object	Attribute
1. Cylinder	CILINDRO
2. Cone	CONO
3. Cone trunk	TRONC_CON
4. Sphere	ESFERA
5. Spherical segment	CASQUETE
6. Parallelepiped	PARALEPIP
7. Straight prism	PRISMA
8. Regular pyramid	PIRAMIDE
9. Simple valve	VALV_SIMP
10. Complex valve	VALV_COMP
11. Ring	ANILLO
12. Nozzle	TOBERA
13. Wedge	CUNYA
14. Trestle	CABALLETE
15. Circular section elbow	CODO

16. Container	RECIPIENTE
17. Straight T (derivation)	T RECTA
18. Slanted T (derivation)	T 90
19. Rectangular section elbow	CODORECT
20. Rectangular section duct	CONDUCTO
21. Straight pyramid trunk	TRPIRECTA
22. Slanted pyramid trunk	TRPIINCL
23. Profile in "H"	PERFILH
24. Gear	ENGRANAJE
25. Screw	TORNILLO

The geometric parameters define the object's dimensions, while the reference point is used when generating the object for its positioning in the space.

The geometric parameters and the reference point of the different objects are the following:

1. Cylinder (Figure 6-a)

    diameter of the cylinder
    height of the cylinder

All the lateral faces of the cylinder are generated with the entity of cylindrical face, allowing to visualize them as if they were curved surfaces in the systems for visualization with smoothing or for the elimination of hidden lines.

2. Cone

    diameter of the cone base
    height of the cone

All the lateral faces of the cone are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

3. Cone trunk

    diameter of the cone base
    height of the cone (distance between base and vertice of the cone)
    height of the cone trunk

All the lateral faces of the cone trunk are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

4. Sphere

diameter of the sphere

All the lateral faces of the sphere are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

5. Spherical segment

diameter of the sphere
height of the segment

All the lateral faces of the segment are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

6. Parallelepiped

Length X: length of the parallelepiped according to the X axis
Length Y: length of the parallelepiped according to the Y axis
Length Z: length of the parallelepiped according to the Z axis

7. Straight prism

Straight prism having as base a regular polygon.

diameter of the circumference enclosing the base
height of the prism

8. Regular pyramid

Straight pyramid having as base a regular polygon.

diameter of the circumference enclosing the base
height of the pyramid

9. Simple valve (Figure 6-a)

A simple valve is defined by a cylinder and a cone.

radius of the cylinder
length of the cylinder
radius of the cone

height of the cone

All the lateral faces of the valve are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

10. Complex valve

A complex valve is defined by three orthogonal cones.

radius of the input cone
length of the input cone
radius of the output cone
length of the output cone
radius of the derivation cone
length of the derivation cone

All the lateral faces of the valve are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

11. Ring

exterior diameter of the ring
interior diameter of the ring
height of the ring

All the lateral faces of the ring are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

12. Nozzle

minor diameter of the nozzle
major diameter of the nozzle
minor height of the nozzle
major height of the nozzle

All the lateral faces of the nozzle are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

13. Wedge

radius
height

14. Trestle

length 1
height
length 2
length 3
height H
radius

15. Circular section elbow

radius (section) of the tube
elbow axis radius
angle of the bow

All the lateral faces of the elbow are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

16. Container (Figure 6-a)

diameter
length
height H1
height H2
distance D

All the lateral faces of the container are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

17. Straight T (derivation) (Figure 6-a)

A straight T is defined by two cylinders whose respective main axes and the derivation axis are orthogonal.

length of the main cylinder
radius of the main cylinder
length of the derivation cylinder
radius of the derivation cylinder

All the lateral faces of the derivation are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

18. Slanted T (derivation)

The slanted T is defined by three cylinders: The first one (input cylinder) and the second one (output cylinder) are colineals, whereas the third one (derivation cylinder) is slanted corresponding to the others and according to the user's definition.

length of the input cylinder
radius of the input cylinder
length of the output cylinder
radius of the output cylinder
angle between the main axes (input-output) and of derivation
length of the derivation cylinder
radius of the derivation cylinder

All the lateral faces of the derivation are generated with the flag of cylindrical face, allowing to visualize them as if they were curved surfaces.

19. Rectangular section elbow

length of side 1
length of side 2
elbow axis radius
angle of the bow defining the elbow

20. Rectangular section duct

length of side 1
length of side 2
length of the duct

21. Straight pyramid trunk

Irregular straight pyramid with rectangular base.

length of side 1 of the major base
length of side 2 of the major base
length of side 1 of the minor base
length of side 2 of the minor base
height of the pyramid

22. Slanted pyramid trunk

Irregular slanted pyramid with rectangular base.

length of side 1 of the major base
length of side 2 of the major base
length of side 1 of the minor base
length of side 2 of the minor base
eccentricity according to the X axis
eccentricity according to the Y axis
height of the pyramid

23. Profile in "H"

24. Gear

number of cogs
module
radius of the head
radius of the foot
width

25. Screw

6.2 OBJECT LIBRARIES

Sometimes there exists a series of objects the user uses to incorporate in his design.

Thus, a mechanical designer uses to often employ a series of screws and standardized pieces, an interior decorator uses to employ furniture being more or less fixed, etc.

The system has the possibility to define libraries allowing to classify the objects conveniently according to the user's needs. Following the example of the interior decorator, he would use libraries classifying the different elements of his design in: libraries for tables, chairs, cupboards, decorative elements, etc.

Within a design, every library object being read keeps a reference to its library for all times. This offers a great advantage with regard to the maintenance of the designs having already been completed and stored on the disk: If a library object has to be modified (changing a dimension or the form or size, etc.) it is sufficient to read, modify and substitute it in the library. Every design referencing this object will automatically undergo the same modfication.

In order to manage the libraries and the contained objects, the system provides the realization of the following operations:

- generation of a library
- change of a default library
- listing of a library's content
- reading of a library object
- inclusion of an object within a library
- elimination of a library object
- substitution of a library object

7 ACKNOWLEDGEMENTS

The authors would like to express their gratitude to Susanne Wurster for her nice work in translating the raport into english. They would like to also mention all those persons who have worked on the design and implementation of the system, and in particularr Marga Busqui, Lluis Solano, Cesc Batlle, Alvar Vinacua, Isabel Navazo, Dolors Ayala, Jordi Pascual, Albert Trill, Josep Giralt, Javier Adrian, Josep Fondecaba and Javier Carrasco.

8 REFERENCES

[1] I. Navazo, P. Brunet, D. Ayala: A Geometric Modeler based on Exact Octree Representation of Polyhedra; Computer Graphics Forum, Vol. 5, Num. 2, 1985, pages 91-104.

[2] I. Navazo, J. Fontdecaba: Boolean Operations Using Extended Octrees; Report DMI01-86, Dept. de Metodes Informatics, Polytechnical Univ. of Catalonia, Barcelona, 1986, in Catalan.

[3] P. Brunet, I. Navazo: Solid Representation and Operation Using Extended Octrees; ACM Transactions on Graphics, Vol. 9, 1990, to be published.

[4] P. Brunet: Un Sistema de Modelado Geométrico de Sólidos basado en el Modelo de Fronteras y Árboles Octales; Report LSI-89-4.

[5] Y. Lee, A. Requicha: Algorithms for Computing the Volume and Other Integral Proprieties of Solid Objects; Tech. Memo. No. 35a, Production Automation Project, University of Rochester, Feb. 1980 (also appeared in Comm. ACM 1982).

[6] P. Brunet, J. Pascual, R. Vila: L'Algorisme de Visualització per Partició Binaria de l'Espai; Report LSI-89-3.

[7] E. Torres: Optimization of the Binary Space Partition Algorithm (BSP) for the Visualization of Dynamic Scenes; Proceedings of EG '90 Conference, 1990.

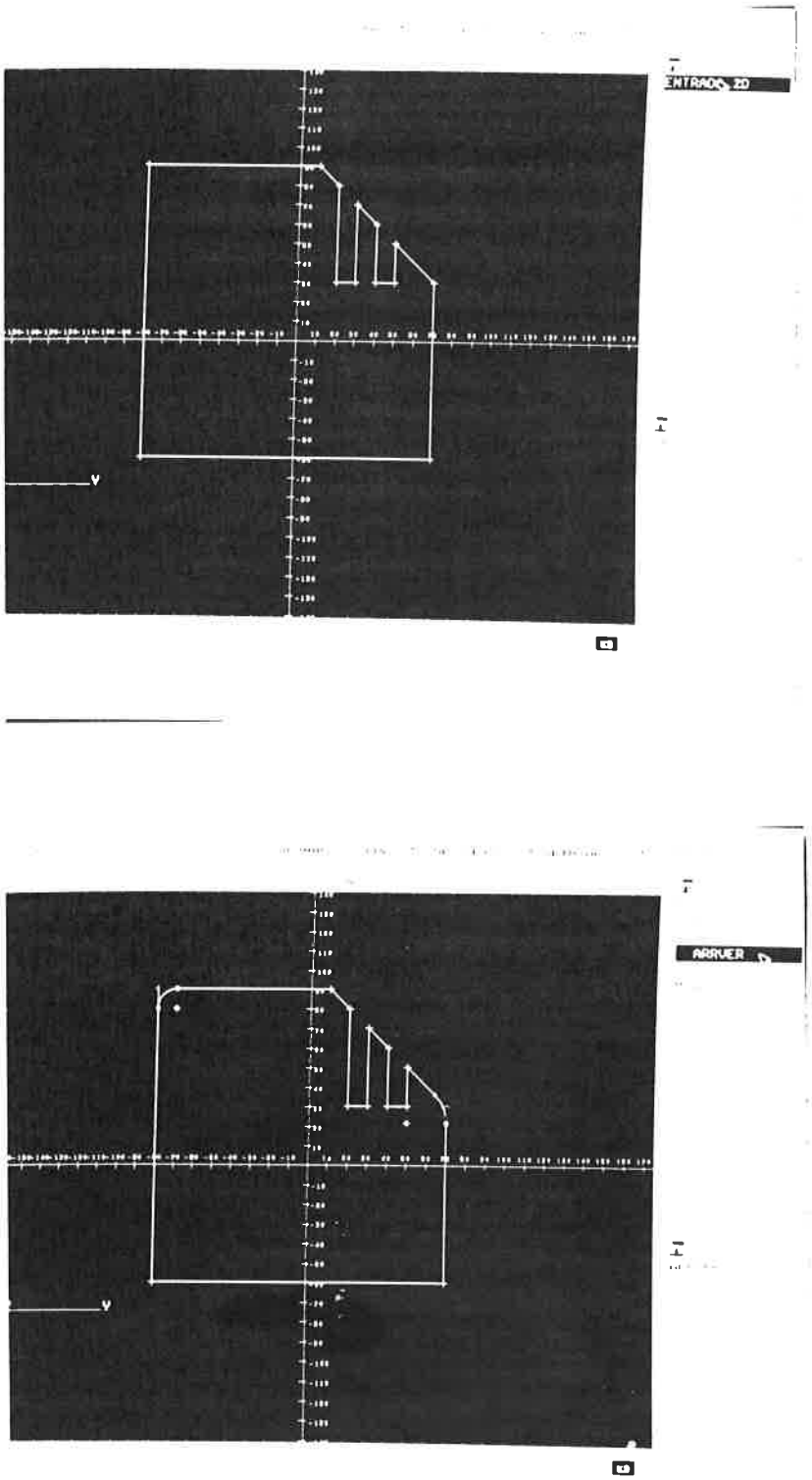


Figure 1. Initial drawing of a planar face. Automatic rounding of two corners (below).

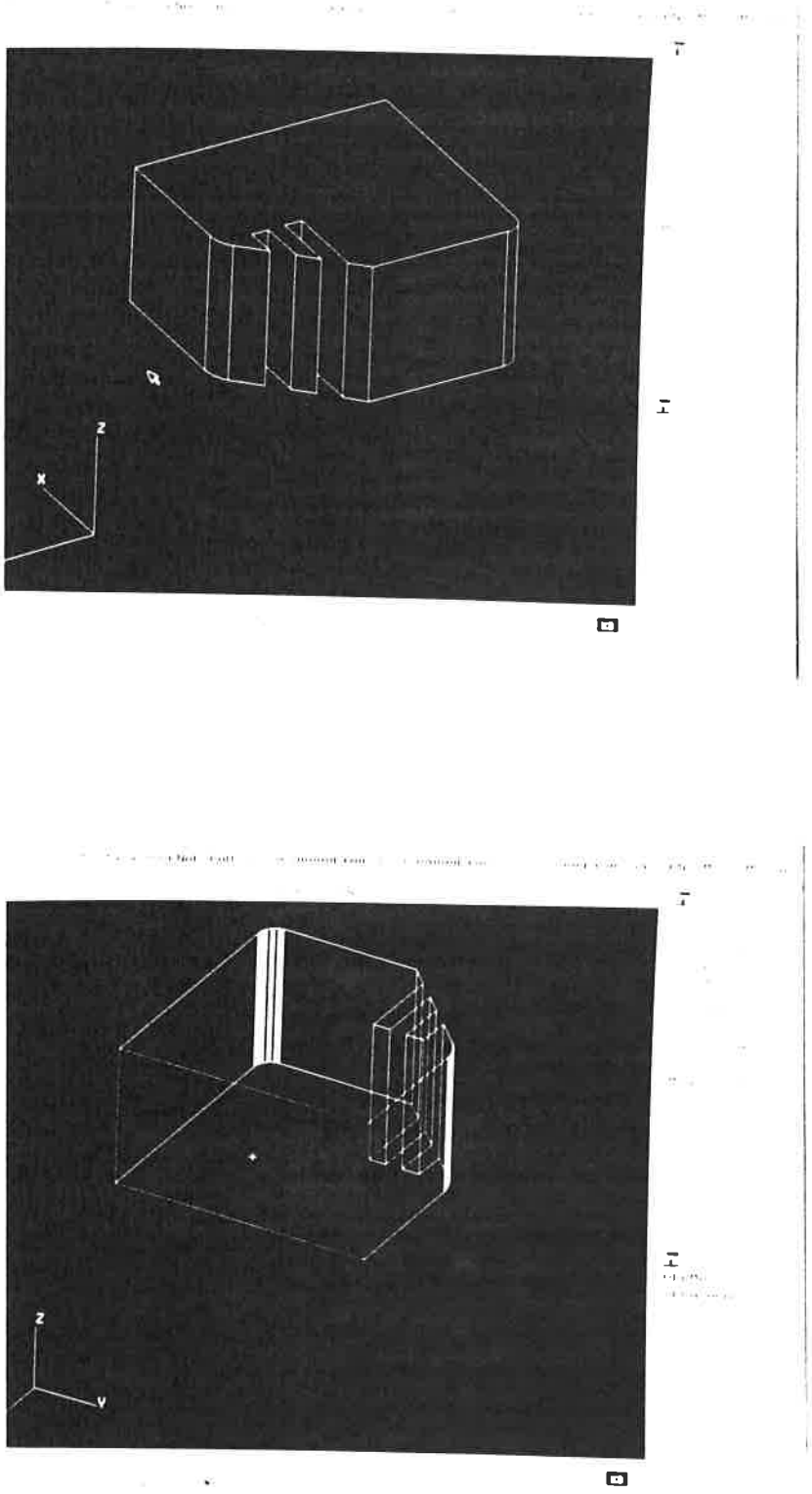


Figure 2. Object generated by a sweep operation from the initial face. Above, axonometric view with hidden lines removed. Below, wireframe representation and selection of a face for further operations.

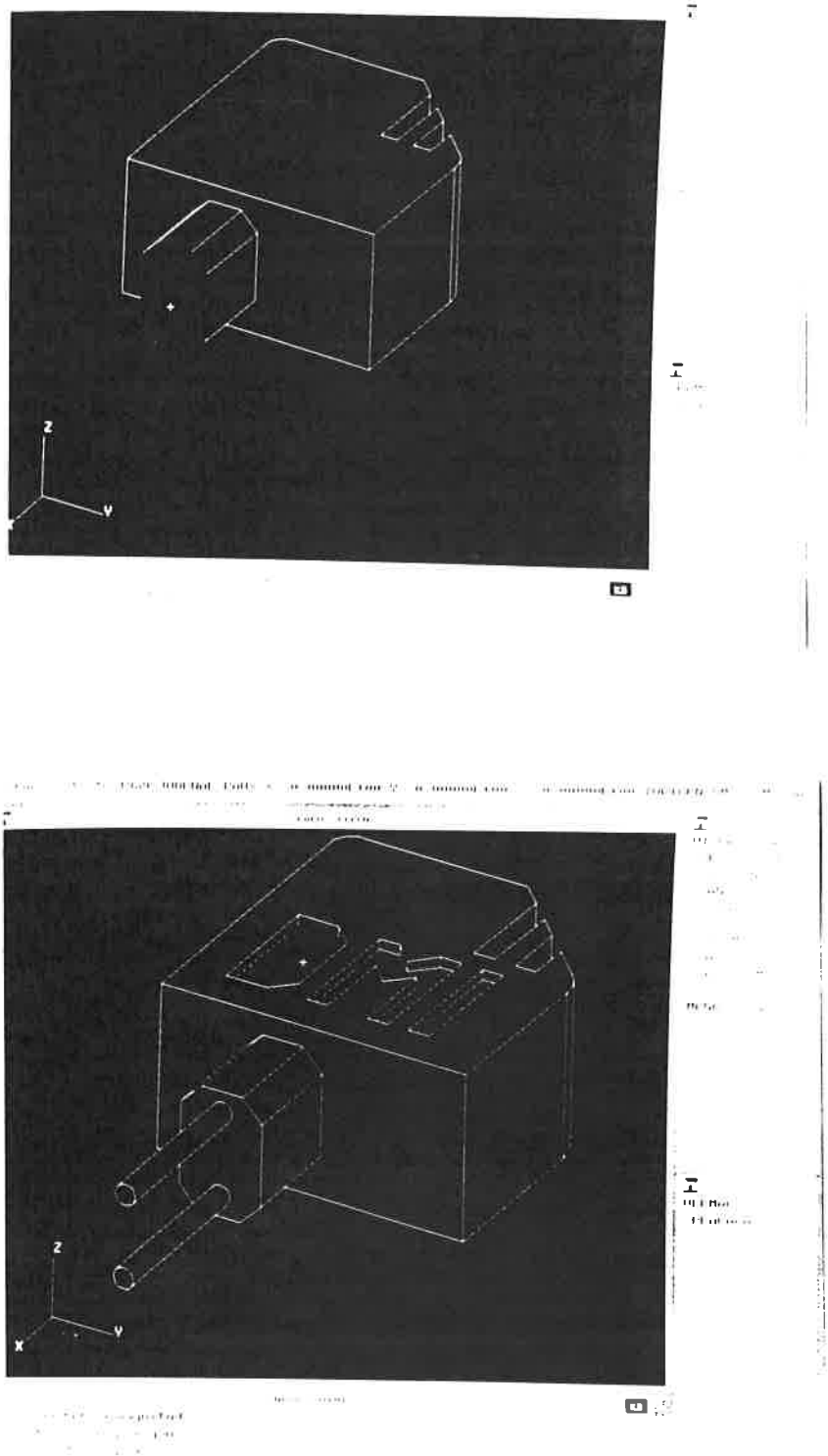


Figure 3. The object in figures 1 and 2 is being completed by succesive sweep operations.

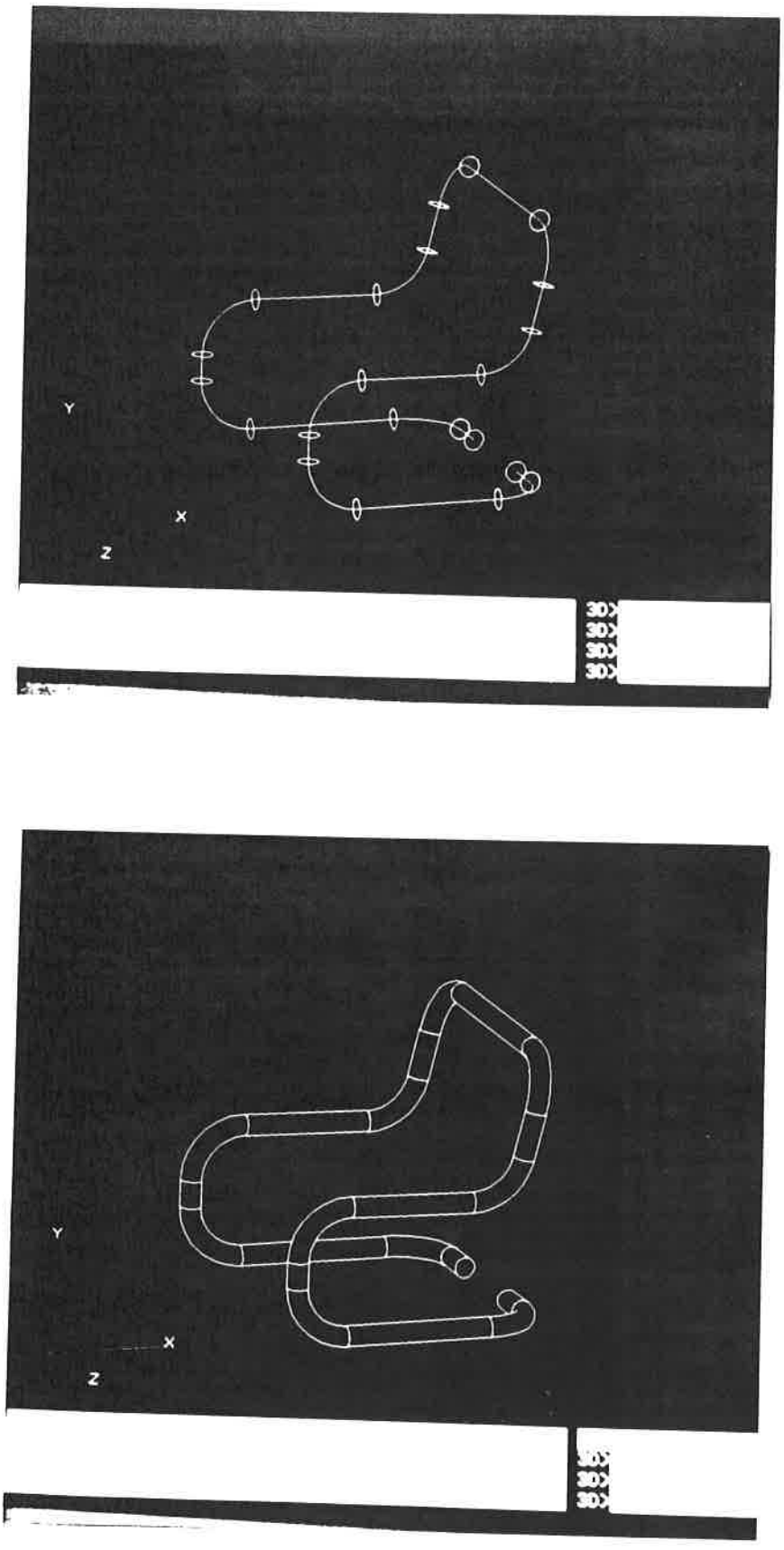


Figure 4. Generation of pipes in the DMI system. Wireframe representation of the axis, and realistic representation with hidden lines removed.

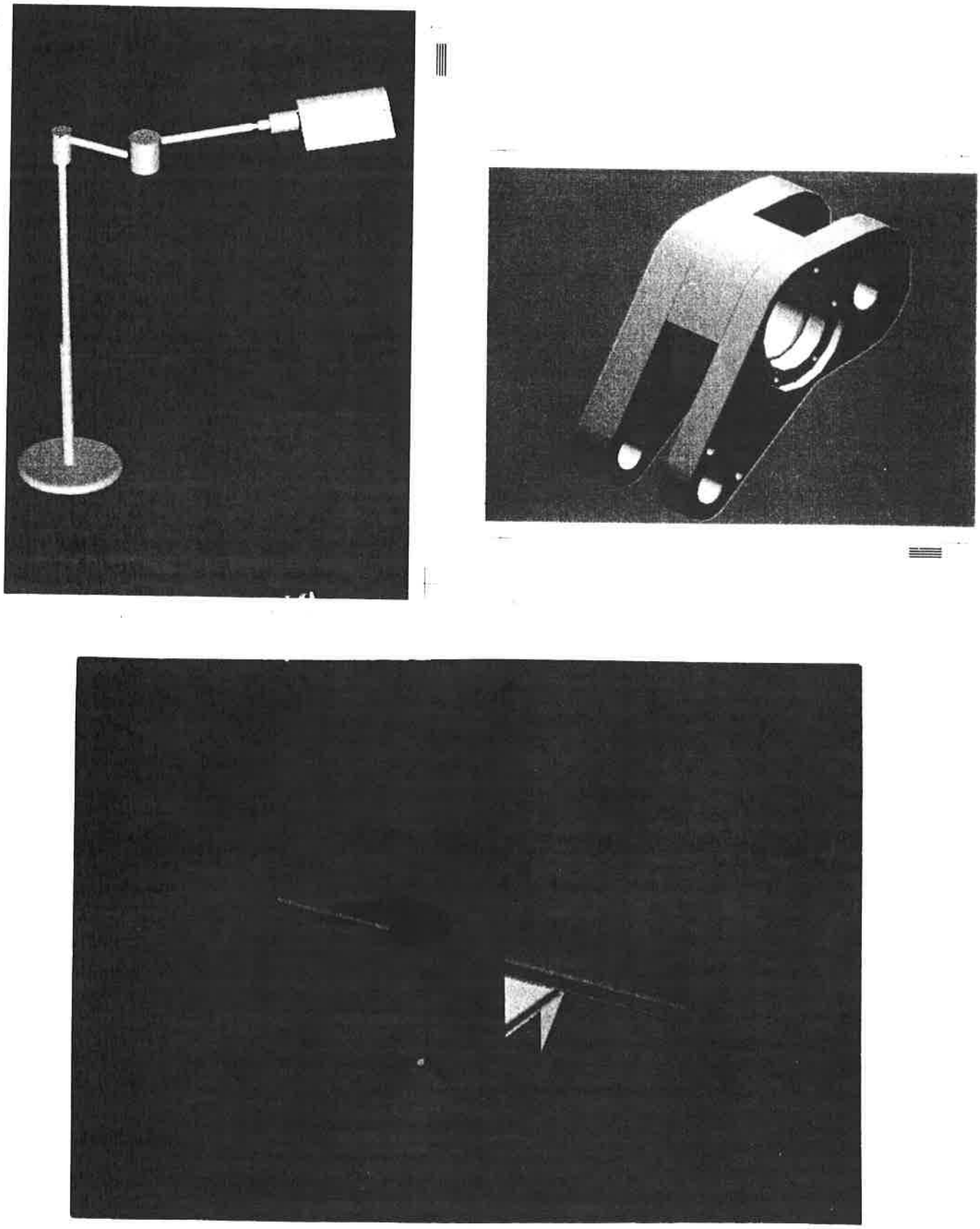
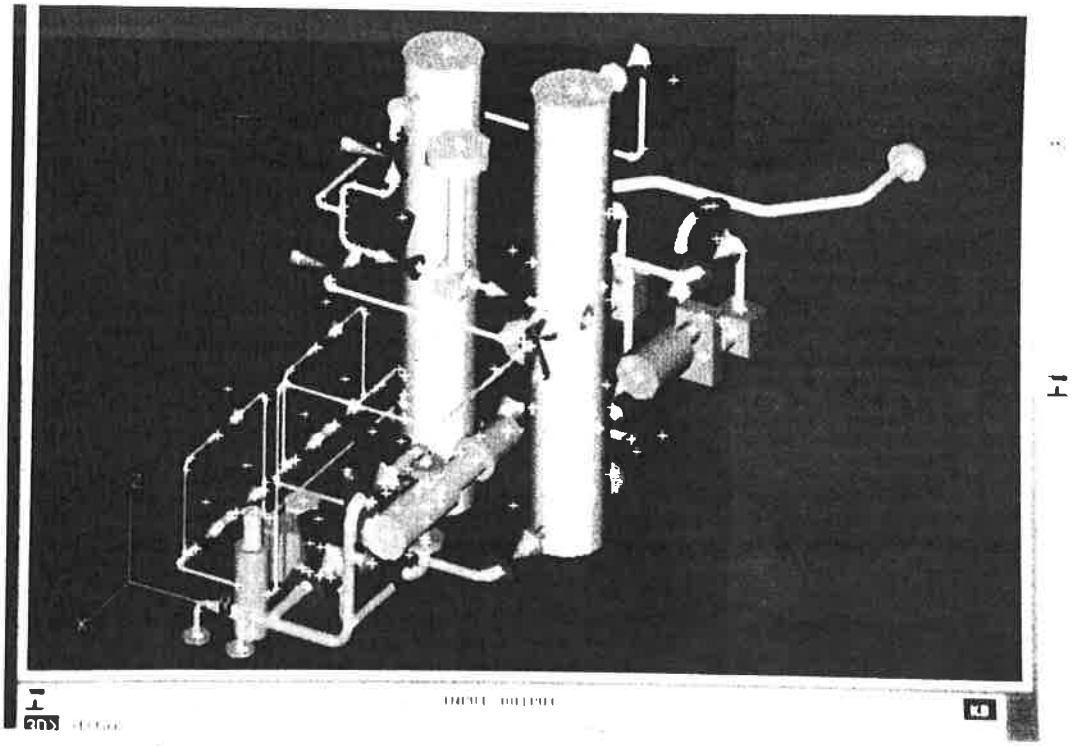


Figure 5. Z-Buffer rendering of several objects.



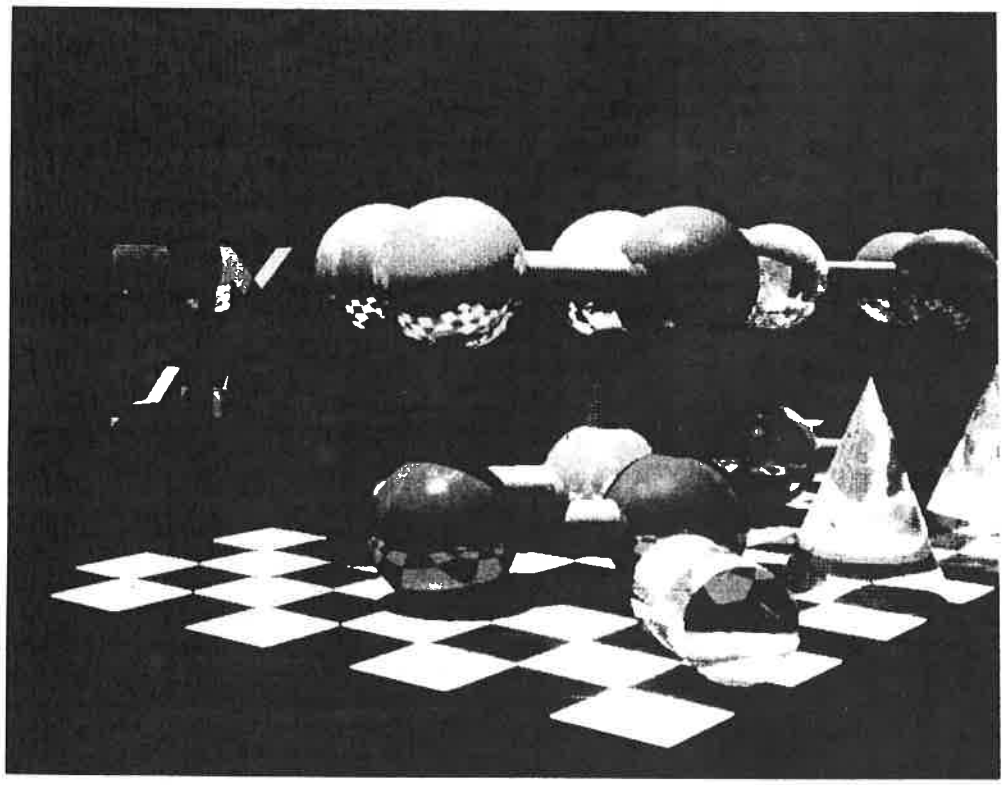
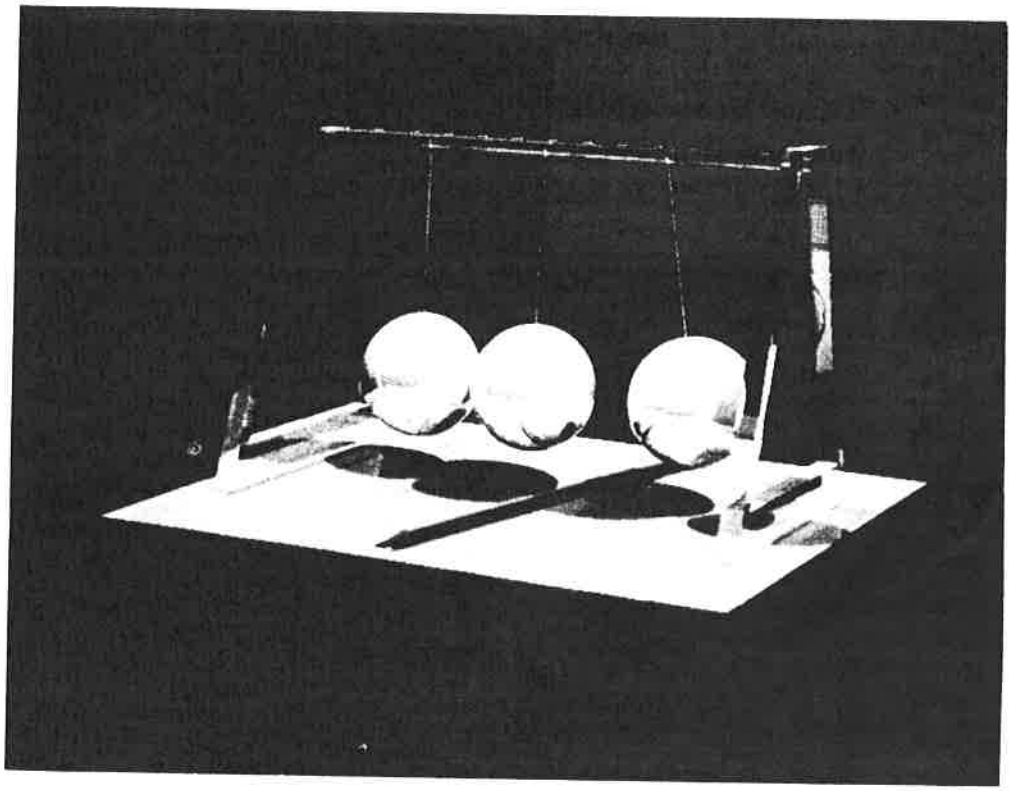


Figure 6. Above, Z-Buffer rendering of a piping design. Axonometric view showing connection points between elements. Below, Ray tracing of a scene in a 256 colour display.



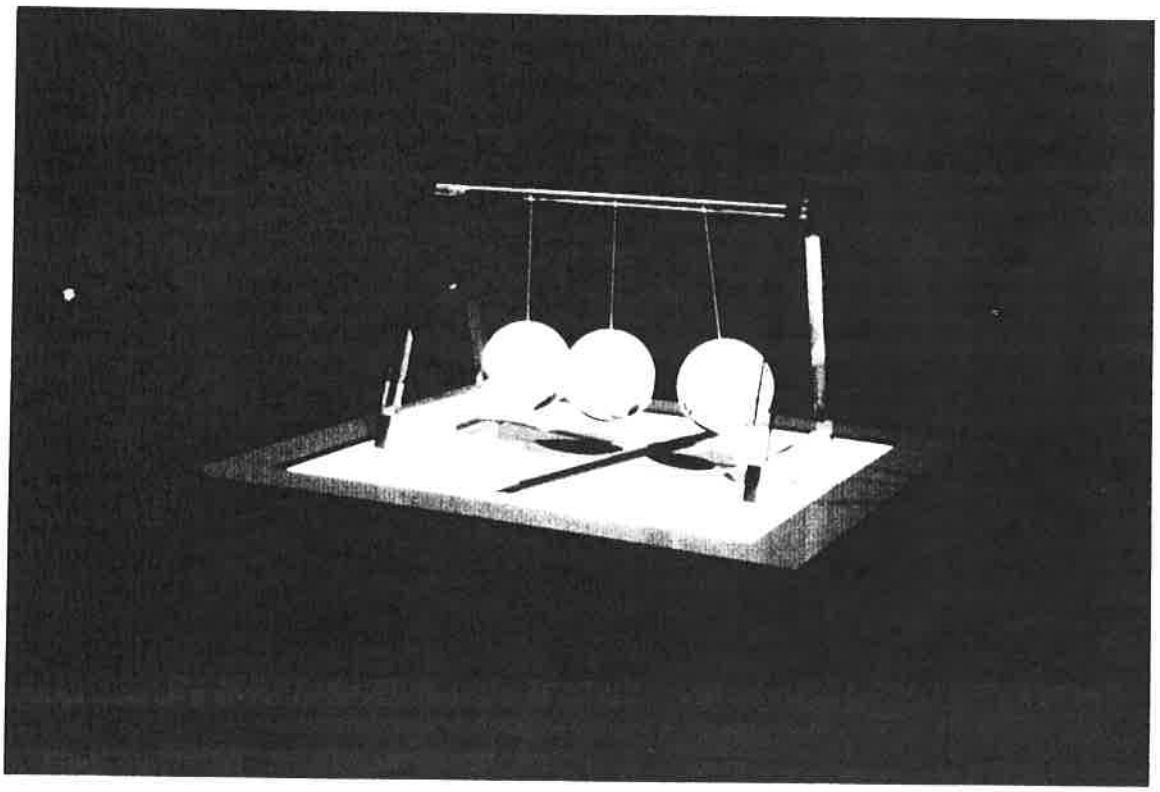


Figure 7. Ray tracing rendering. Above, without antialiasing. Below, with antialiasing.