COMPUTER AIDED DESIGN
(BME-42)

Unit-IV: 3D Graphics

(7 Lectures)

- **Introduction, Wireframe modeling, Surface modeling,**
- **Polygon surfaces-polygon meshes**, polygon equations,
- Quadric and Superquadric surfaces, Blobby objects,
- Solid modeling-
  - Boolean set operations,
  - regularized set operations,
  - Primitive instancing,
  - Sweep representation-translational, rotational and hybrid sweeps,
  - Boundary representation-topology, geometry, boundary models
  - Constructive solid geometry-unbounded and bounded primitives

Lecture 27

Topics Covered

3D Graphics

Wireframe Modeling

Surface Modeling

Polygon Surfaces

Polygon Meshes

Explicit Representation, Pointers to a Vertex List

Prepared By

Prof. S. K. Srivastava

MED, MMMUT, Gorakhpur (UP)
sksme@mmmut.ac.in

Introduction

- In graphics scenes, we need different types of objects such as trees, clouds, flowers, buildings, mountains, roads, river, rocks, glass, steel, bricks, plastics, and so on.
- Similarly, engineering applications require the components like nut, bolt, cylinder, piston, gear, structural components, etc.
- These objects are very smooth, and most of the computer graphics involves the modeling of these objects, comprising the real world.
- The geometry of CAD, high quality fonts, plots and sketches produce smooth curves and surfaces.
- Animation path of camera or object is also smooth. The color of objects through intensity variations further improves the smoothness of the surfaces.

3D GRAPHICS

In computer graphics, the need to represent these 3D objects arises in two cases:

I. Modeling of the existing objects, e.g. a car body, a face, a mountain, a building or tree, for which mathematical description of the object is not available.

Often, we approximate these objects by joining the pieces of two-dimensional (e.g., curves, plane, etc.) and three-dimensional (cube, sphere, cylinder, etc.) primitives as per the requirement.

II. Modeling of the new objects, starting from *scratch*, when no preexisting physical objects is available.

The user creates the object through the modeling process so that it matches the object exactly, as far as possible.

The user creates the object interactively, describe it mathematically, or sometimes develop the approximate description of object through a computer program.

3D GRAPHICS

- Most often, the polygon and quadric surface descriptions are used for the simple Euclidean objects such as *polyhedrons* and *ellipsoids*.
- The geometry (coordinate of vertices, rigid motion and geometric transformations, metric information such as angle, diameter, etc.) of objects is created by performing Euclidean calculations.
- Apart from the polygon and quadric surfaces, *spline surfaces* are also very useful for designing the engineering components such as gears, aircraft wings, turbine blades, aerodynamic bodies and objects involving the curved surfaces.

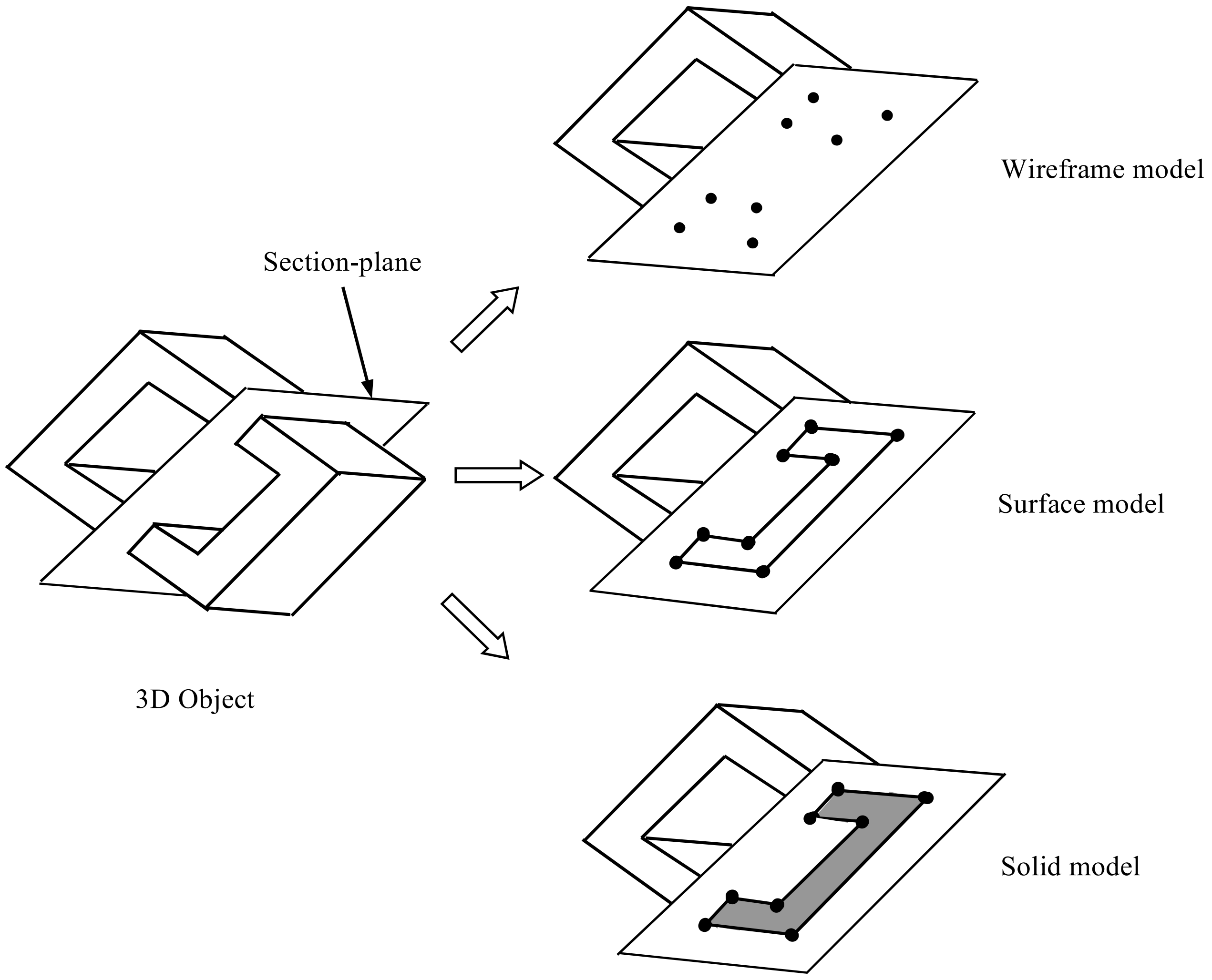
There are three distinct methods of geometric modeling:

1. **Wireframe modeling**
2. **Surface modeling**
3. **Solid modeling**

3D GRAPHICS

- Figure shows the difference between the modeling representations of an object by considering transverse cross-section plan.
- For the wireframe model, the box is represented as a collection of vertices and edges; and the section of box is represented as the collection of its disconnected edges by the points.
- For the surface model, the box section is represented as a collection of vertices (corner points), edge lines and faces (surfaces) and its cross-section is represented as a set of points and lines (i.e., edges and faces).
- For solid models, the box is represented as a collection of corner points, edges, faces (surfaces) and interior volumes; and its section is represented as set of points, lines and interior plane segments.

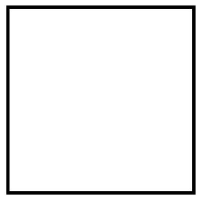
3D GRAPHICS



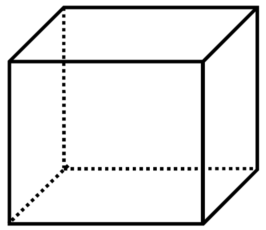
Wireframe, surface and solid representations of an object

3D GRAPHICS

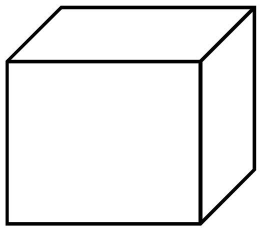
Figure shows the 2D, $2\frac{1}{2}$D and 3D versions of image in computer graphics



2D image



$2\frac{1}{2}$ D wireframe image



3D solid image

Image dimension

WIREFRAME MODELING

- A wireframe model is the simplest geometric model that is easy to create on computer screen and simple to use.
- Wireframe model provides sufficient information about the solid objects.
- They require very little computer time and memory; therefore, can be done on an inexpensive CAD systems based on the personal computers.
- The word *wireframe* is related to the fact that object may be imagined to consists of a wire that is bent to follow the object edges to generate the model.
- In this model, the edges of an object are shown as lines connecting the vertices; therefore, it is also referred to an edge representation of object.
- Wireframe modeling may be considered as natural extension of traditional drafting. It forms basis for the surface modeling.
- A wireframe model has no surface or solid attached to it, i.e., the object is transparent and one can view the back side from the front.

WIREFRAME MODELING…

- Typically, a wireframe model uses the basic two- dimensional entities such as points, lines, arcs, circles, conics (ellipses, parabolas and hyperbolas), fillets, chamfers and synthetic curves (Hermite, Bézier, B-spline, β (beta)-spline).

Limitations

- Wireframe models are ambiguous and sometimes cannot provide designer the required information.
- The wireframe model will appear clear (without any ambiguity) if the hidden lines are removed.
- Wireframe modeling does not distinguish between the inside and outside of the surfaces of an object; therefore, seems inadequate for generating cutter paths to drive NC machines for manufacture of the objects.
- It is difficult to decide as to which side of the surface is solid. It is due to the ambiguity in the surface definition of wireframe model in the database.

WIREFRAME MODELING…

- Since the wireframe object has no faces attached to it; therefore, section property and mass calculations of the object is impossible.
- It is not possible to visualize graphically the object because wireframe modeling does not define a unique solid object.

The wireframe modeling requires both *topological* and *geometrical* information whereas solid modeling requires only the geometrical information of the object.

Applications

- Wireframe modeling automatically generates the standard views and auxiliary views of the object.
- Surfaces of objects can be created on wireframe modeling.
- It can be used to check the visual interference between the objects.
- It is used to find intersections of objects in the space.
- Wireframe modeling enhances the visualization of 3D objects.

SURFACE MODELING

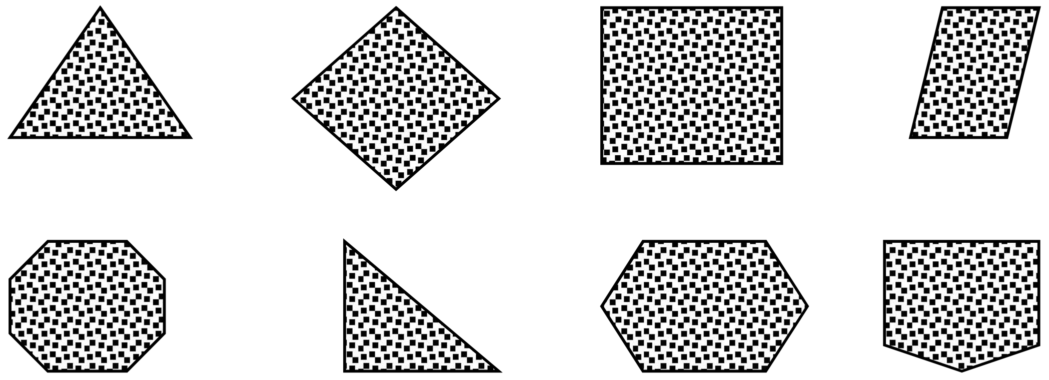
- A surface model can be built by defining the surfaces on the wireframe model.
- This describes a 3D object as a set of surfaces that separates the object interior from the environment, e.g., polygon facets and spline patches.
- Thus, a three-dimensional object, described with the help of two-dimensional boundaries (surfaces), represents only an envelope of the part geometry.
- Surface models define part geometry more precisely, as compared to the wireframe models, and used directly to generate the path profile for NC machines.
- Complex shaped objects are generated by combining planes, ruled surfaces of revolution, sweep surfaces and fillet surfaces.
- An object generated through the surface modeling displays on the screen in such a way that it looks like a solid object.

SURFACE MODELING…

- There are three most popular surface representation schemes, when the *shapes of the objects remain same*, are presented
  - **Polygon mesh surfaces**
  - **Quadric surfaces, and**
  - **Superquadric surfaces**
- Next, the modeling of object which *changes their shapes under the external disturbances* is known as **Blooby objects**

POLYGON SURFACES

- A polygon can be represented as a group of connected edges forming a closed object as shown in Figure
- Many graphics systems store object descriptions as a set of polygon surfaces.
- This most common three-dimensional object representation scheme encloses the object interior.
- The method is most suitable for displaying the surface when interior details are not important.
- The simple objects such as boxes, building exteriors, cabinets, etc. can be represented by the polygon meshes; thus, represent a volume bounded by the planar surfaces.



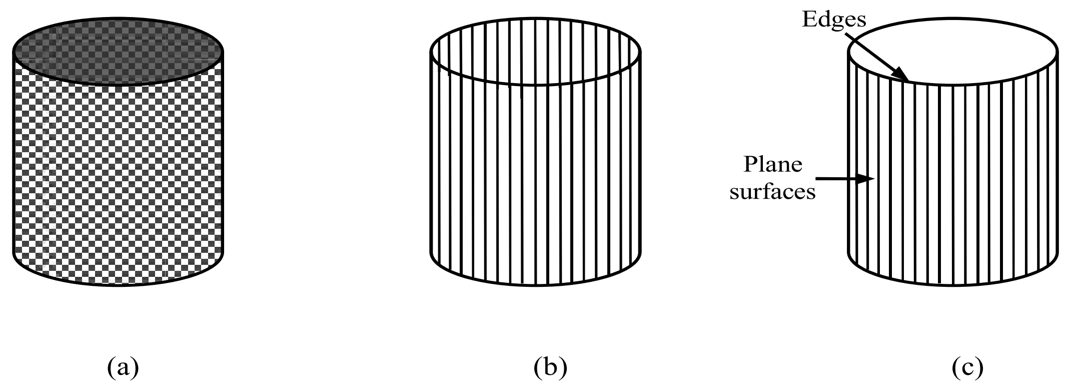
Series of polygon surfaces

POLYGON SURFACES…

- Polygon surfaces can also be used to represent the complex objects with curved surfaces (e.g. surfaces of revolution or sculpted surfaces based on Bézier and B-spline curves) as shown in Figures
- However, this representation will be approximate because of the sides of the polygons.
- A more accurate representation of curved surfaces can be obtained with large number of small polygons using better piecewise linear approximation; however, it increases the memory (space) requirements as well as execution time for the algorithms.
- Logically, if we have infinite number of polygons then the object may be exact.
- In computer graphics, the polygon descriptions of objects are often referred to as *standard graphics objects*.
- For polyhedron objects, the polygon representation accurately defines the surface features of the object.

POLYGON SURFACES…

- But, for the objects with curved surfaces, e.g., cylindrical surface (Fig.), surfaces are tilted to produce the polygon mesh approximation.
- This wireframe outline of the object is common in design and solid modeling applications; and it is used to indicate the surface structure of a solid object.
- The removal of back (hidden) lines improves the appearance of objects.
- Further improvement in realism can be achieved using the shading patters across the polygon surfaces, by eliminating the polygon boundaries (edges).



(a) A 3D curved object (b) polygonal representation (c) polygonal representation with hidden lines removed

POLYGON SURFACES…

Polygon Meshes

- A polygon mesh is a set of connected polygonally bounded planar surfaces. It is a collection of *edges, vertices*, and *polygons* joined in such a way that each edge is shared by at most two adjacent polygon surfaces.
- The vertices of shared polygons form a closed sequence of edges.
- There are several ways to represent the polygon mesh. Each method basically evaluates the following information related to the three-dimensional objects:
  - Number of edges incident on a vertex
  - Number of two vertices of an edge
  - Number of polygons sharing an edge
  - Total number of all edges defining a polygon
  - Case of displaying a polygon mesh
  - Errors in representation of missing edge, vertex, or polygon
  - Amount of computer storage required

POLYGON SURFACES…

Polygon Meshes…

In general, more explicitly the relations among the vertices, edges and polygons are represented, the faster the operations for image generation but it requires more space to store the data. There are four ways to represent the polygon mesh of three-dimensional objects, namely:

I. Explicit representation
II. Pointers to a vertex list
III. Pointers to an edge list
IV. Method of polyhedrons

Explicit representation

- A polygon is generated by joining the vertices in a sequence.
- Each polygon is represented by a list of vertex coordinates in the same order in which they occur around the defining polygon as

$$P = ((x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3), \ldots\ldots (x_n, y_n, z_n))$$

POLYGON MESHES

Explicit representation…

- Each coordinates of vertex are connected by the edges and, last and first coordinates are connected by the edges.
- This type of representation is space efficient for a single polygon; however, for objects consisting of polygon mesh, the extra space is required because the coordinates of shared vertices are duplicated.
- Similarly, when adjacent polygons of an object are outlined, each vertex and each edge of the polygons are transformed which results into drawing the shared edges twice, causing the problems on pen plotters and vector displays due to the overwriting.
- Moreover, the situation becomes complex on raster displays if edges are drawn in opposite directions requiring extra number of pixels.

POLYGON MESHES

Explicit representation…

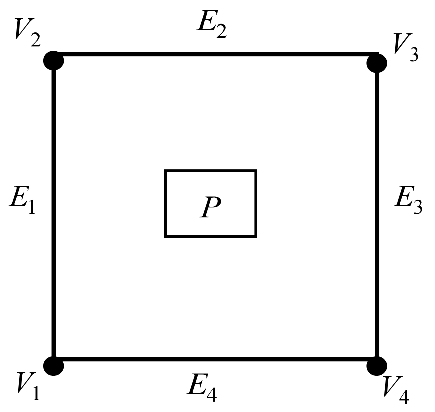
Figure shows a single polygon and a polygon mesh.

For example, vertex V_1 will be stored in four ways corresponding to the polygons P_1, P_2, P_3 and P_4; and edges E_8 & E_5 are shared edges between the polygons P_1 & P_2 and P_3 & P_4, respectively.

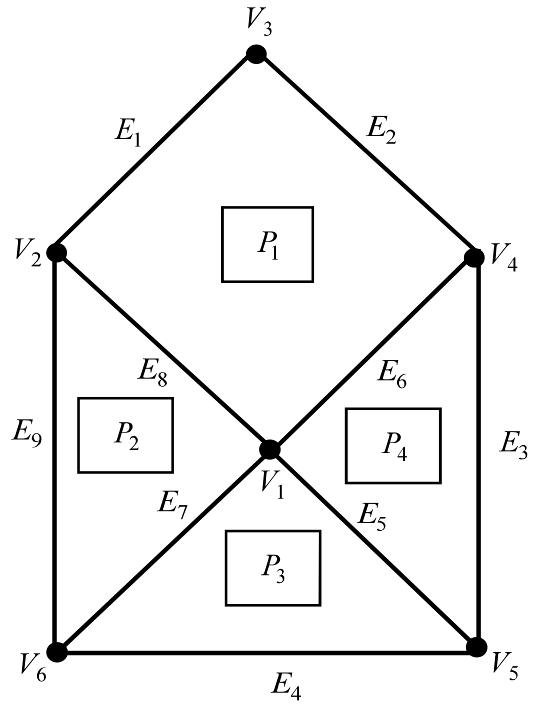
Polygon mesh

(a) single polygon

(b) multiple polygons



(a)



(b)

In spite of its weaknesses such as more space requirement and increased execution time, the explicit representation is still useful in CAD applications because of its simplicity.

POLYGON MESHES

Pointers to a Vertex List

- Polygons defined by the pointers to a vertex list define each vertex in the polygon mesh database just once in the vertex list.

- For each polygon, pointers (or indices) are formed that consist of serial number of the vertices, as occur in the vertex array

For example, the pointer for polygon P_1 in the polygon mesh made up of vertices V_1, V_4, V_3, V_2, V_1, as shown in Fig., may read as $P_1 = (1, 4, 3, 2, 1)$.

Thus, the polygon P_1 is drawn by joining the vertices 1, 4, 3, 2, 1 in sequence.

In another example of polygon mesh, the vertices V_1, V_2, V_3, V_4 of the polygons P_1 and P_2 are defined by an array of vertex and polygon as

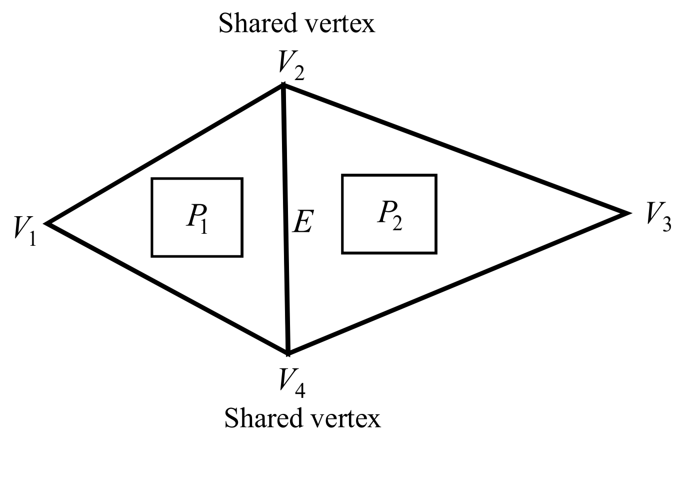
$$V = (V_1, V_2, V_3, V_4) = ((x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3), (x_4, y_4, z_4))$$

$$P_1 = (1, 2, 4)$$

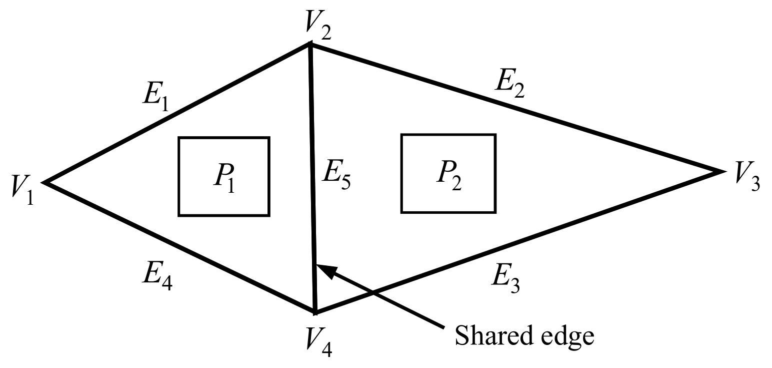
$$P_2 = (4, 2, 3)$$

POLYGON MESHES

Pointers to a Vertex List



(a)



(b)

Polygon mesh defined by (a) vertex list (b) edge list

Thus, the polygons P_1 and P_2 are drawn by joining the vertices 1, 2, 4 and 4, 2, 3 in sequence. The method of pointers to a vertex list has several advantages over the explicit representation of polygons given as

(i). Each vertex is stored once in the vertex list thereby reducing the space requirement.

(ii). This makes the modification of a particular vertex easy in the vertex list.

However, the method still suffers with the problem of drawing the shared (coincident) edges, e.g., shared edge E is drawn twice as shown in Fig. (a).

COMPUTER AIDED DESIGN (BME-42)

Unit-IV: 3D Graphics

(7 Lectures)

- Introduction, Wireframe modeling, Surface modeling,
- Polygon surfaces-**polygon meshes**, **polygon equations**,
- **Quadric and Superquadric surfaces, Blobby objects**,
- Solid modeling-
  - Boolean set operations,
  - regularized set operations,
  - Primitive instancing,
  - Sweep representation-translational, rotational and hybrid sweeps,
  - Boundary representation-topology, geometry, boundary models
  - Constructive solid geometry-unbounded and bounded primitives

Lecture 28

Topics Covered

Polygon Meshes

Pointers to an Edge List, Method of Polyhedrons

Quadric Surfaces

Sphere, Ellipsoid, Torus

Superquadric Surfaces

Superellipse, Superellipsoid

Blobby Objects

Prepared By

Prof. S. K. SRIVASTAVA

MED, MMMUT, Gorakhpur (UP)
sksme@mmmut.ac.in

POLYGON MESHES

Pointers to an Edge List

- Polygons defined by the pointers to an edge list define each vertex and each edge in the polygon mesh database *just once* in the vertex list and edge list, respectively.
- In vertex list, each vertex is defined once in the vertex list.
- In edge list, each edge points to the two vertices in the vertex list defining the edge, and also to the one or two polygons to which the edge belongs to the object.
- Thus, vertices, polygons and edges of an object are defined as

$$V = (V_1, V_2, V_3, \ldots\ldots, V_n)$$

$$P = (E_1, E_2, E_3, \ldots\ldots, E_n)$$

$$E = (V_1, V_2, P_1, P_2)$$

When an edge belongs to only one polygon (say E_1, E_2 in Fig.) then either P_1 or P_2 is null and represented by λ. For example, the vertex, polygon and edge lists for the object are defined as

POLYGON MESHES…

Pointers to an Edge List

$$V = (V_1, V_2, V_3, V_4) = ((x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3), (x_4, y_4, z_4))$$

$$E_1 = (V_1, V_2, P_1, \lambda)$$

$$E_2 = (V_2, V_3, P_2, \lambda)$$

$$E_3 = (V_3, V_4, P_2, \lambda)$$

$$E_4 = (V_4, V_1, P_1, \lambda)$$

$$E_5 = (V_4, V_2, P_1, P_2)$$

$$P_1 = (E_1, E_5, E_4)$$

$$P_2 = (E_2, E_3, E_5)$$

- The counting of the vertices and edges take place in *clockwise* direction.
- For 3D objects, some edges are shared by more than two polygons; therefore, the edge descriptions may be extended to include the arbitrary number of polygons as given below:

$$E = (V_1, V_2, P_1, P_2, P_3, \ldots., P_n)$$

POLYGON MESHES…

Method of Polyhedrons

- Method of polyhedrons is applicable when polyhedrons are employed to completely enclose the object.
- In polyhedrons, two adjacent polygons share common edges.
- At least three edges meet at each vertex of the polyhedron.
- *Euler– Poincaré* (Euler law) proved that polyhedral that are *homeomorphic to a sphere*, i.e., their faces are non-self-intersecting and belong to a closed orientable surfaces (viz., surfaces can be distinguished using the direction of surface normal to point to the inside or outside of the solid model under consideration) are *topologically* valid if

$$F - E + V - L = 2(B - G)$$

where $F =$ Number of faces

$E =$ Number of edges

$V =$ Number of vertices

$L =$ Number of inner loops

$B =$ Number of open bodies

$G =$ Number of genus (through holes)

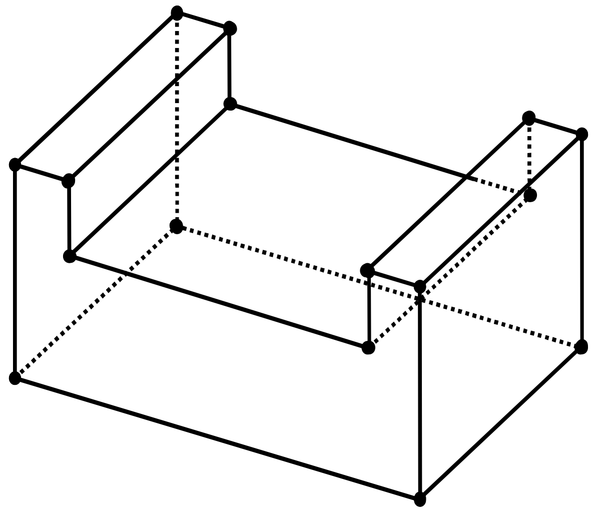
POLYGON MESHES…

Method of Polyhedrons…

For *simple polyhedrons*, three-dimensional objects consisting of polygons without inner loops and through holes, Euler law is expressed as

$$V + F - E = 2$$

For simple polyhedron object, Fig. has the following parameters:



Faces,	$F = 10$
Edges,	$E = 24$
Vertices,	$V = 16$
Body,	$B = 1$
Loop,	$L = 0$
Genus,	$G = 0$

Number of faces, $F = 10$

Number of edges, $E = 24$

Number of vertices, $V = 16$

Euler law is satisfied; therefore, polyhedron solid is topologically valid.

POLYGON MESHES…

Method of Polyhedrons…

Surface modeling is easy to implement compared to the solid modeling because of the following reasons:

- *It is computationally efficient*
- *Easy to generate finite element meshes*
- *It is useful for rendering and generating the machine tool paths*
- *Makes the simulation easy for different objects in plant layouts*
- *Useful for simulation of machinery designs and robotic applications*

QUADRIC SURFACES

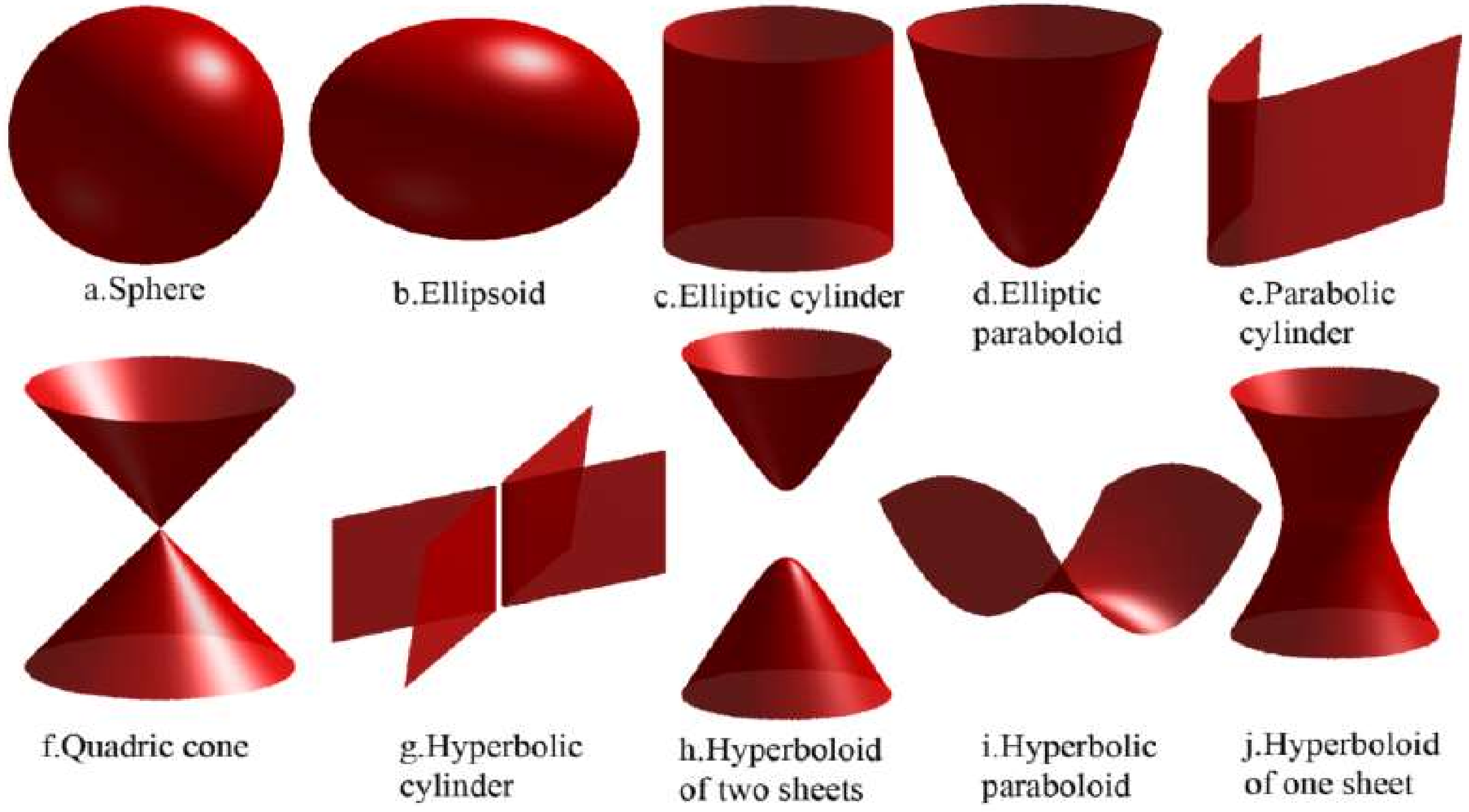
- *Second-degree* equations (quadratics) are used for describing the quadric surfaces.
- These surfaces are in the form of sphere, ellipsoid, paraboloid, cone, cylinder, hyperboloid and torus.
- Mathematically, a general quadric surface may be expressed as

$$f(x, y, z) = c_0x^2 + c_1y^2 + c_2z^2 + c_3xy + c_4yz + c_5zx + c_6x + c_7y + c_8z + c_9$$

- Quadric surfaces, particularly spheres and ellipsoids, are common elements for the surface modeling and often available as primitives in graphics packages.
- Spheres, cones and cylinders belong to the general family of parametric surfaces called as *quadrics*, which are 3D analogous of 2D conics.
- A general 2D conic equation is expressed as

$$f(x, y) = c_0x^2 + c_1xy + c_2y^2 + c_3x + c_4y + c_5$$

QUADRIC SURFACES



QUADRIC SURFACES…

- The useful 3D analogous of conics are *ellipsoid* (sphere is a special case), the infinite cylinder, infinite cone, etc.
- Hyperboloids and paraboloids are also defined by these equations; however, their use is limited, e.g., satellite dishes (paraboloids), headlamp reflectors (paraboloids), power station cooling towers (hyperboloids), etc.

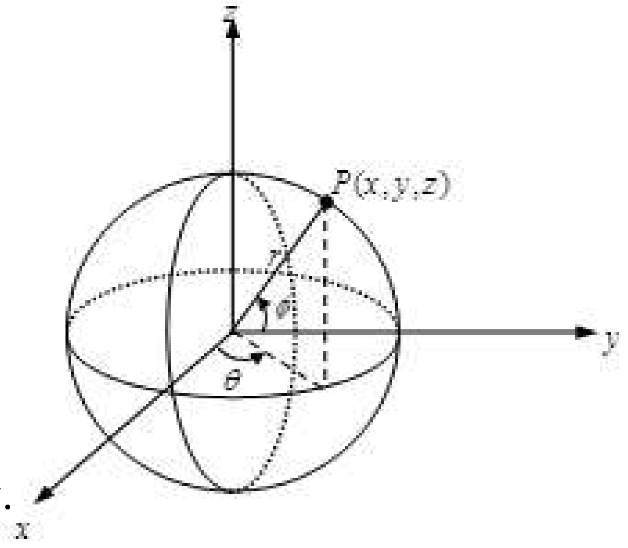
Sphere

In Cartesian coordinates, spherical surface with radius r centered on the coordinate origin is defined as the set of points (x, y, z) satisfying the equation

$$x^2 + y^2 + z^2 = r^2$$

The parametric representation of spherical surface is

$$\left.\begin{aligned} x &= r\cos\phi.\cos\theta\,, \\ y &= r\cos\phi.\sin\theta\,, \\ z &= r\sin\phi \end{aligned}\right\} \quad \begin{aligned} &(-\pi/2) \le \phi \le (\pi/2) \\ &-\pi \le \theta \le \pi \\ & \end{aligned}$$



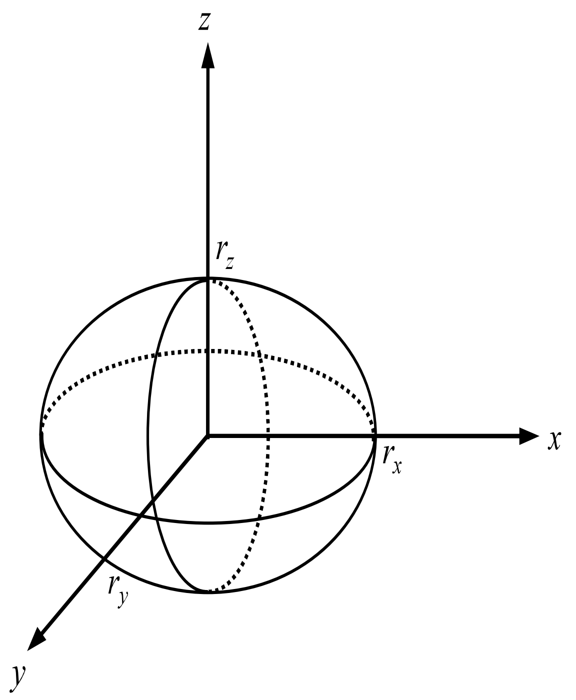
where θ and ϕ are the latitude and longitude angles, respectively.

QUADRIC SURFACES…

Ellipsoid

The ellipsoid surface is an extension of a spherical surface with unequal radii r_x, r_y and r_z along the coordinate axes x, y and z, respectively. In Cartesian coordinates, the points (x, y, z) over the ellipsoid surface, centered on the coordinate origin, is given by

$$\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2 + \left(\frac{z}{r_z}\right)^2 = 1$$



Moreover, the parametric representation of *ellipsoidal surface* is expressed as

$$\left.\begin{aligned} x &= r_x \cos\phi . \cos\theta , \\ y &= r_y \cos\phi . \sin\theta , \\ z &= r_z \sin\phi \end{aligned}\right\} \quad \begin{aligned} &(-\pi/2) \le \phi \le (\pi/2) \\ &-\pi \le \theta \le \pi \\ & \end{aligned}$$

where θ and ϕ are the latitude and longitude angles, respectively.

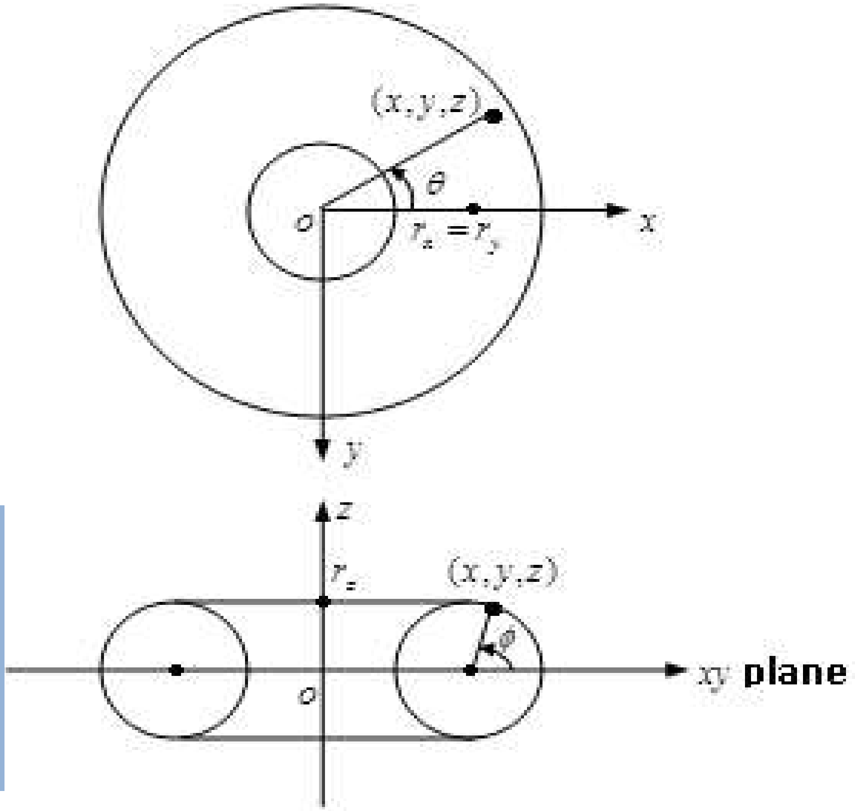
QUADRIC SURFACES…

Torus

It is a doughnut shaped object having circular cross-section. It can be generated by rotating a circle or other conics around a specified axis. In Cartesian coordinates, the point (x, y, z) over the torus surface centered on the coordinate origin is defined as

$$\left[r - \sqrt{\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2}\right]^2 + \left(\frac{z}{r_z}\right)^2 = 1$$

where r is any given offset value.



A parametric representation of torus surface is expressed as

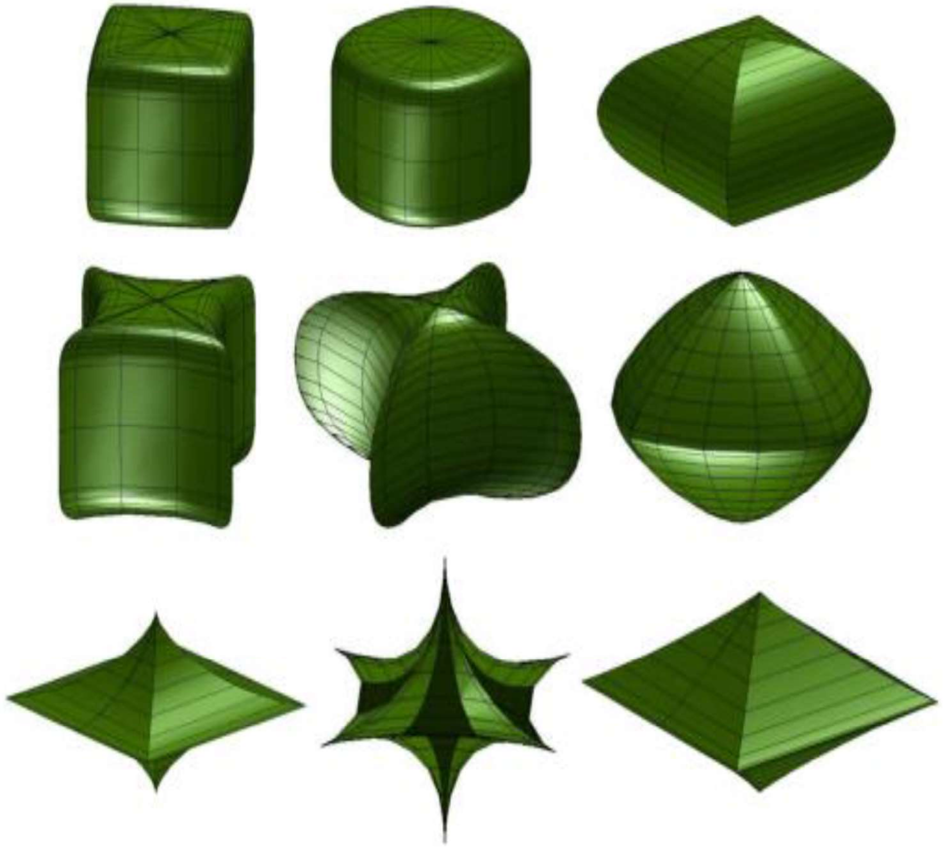
$$\left.\begin{aligned} x &= r_x(r+\cos\phi).\cos\theta\,, \\ y &= r_y(r+\cos\phi).\sin\theta\,, \\ z &= r_z \sin\phi \end{aligned}\right\} \qquad \begin{aligned} -\pi \le \phi \le \pi \\ -\pi \le \theta \le \pi \end{aligned}$$

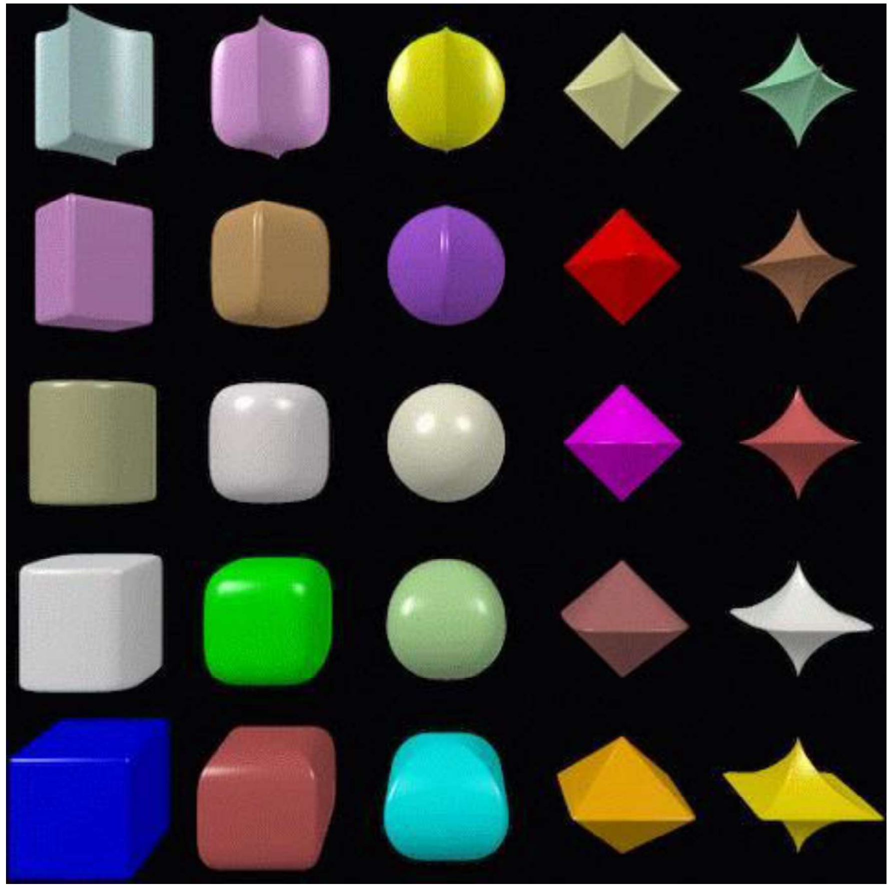
where θ and ϕ are the latitude and longitude angles, respectively.

SUPERQUADRIC SURFACES

- Superquadric surfaces provide simple way of introducing the concept of geometric modeling.
- They are truly solid models since the inside and outside surfaces are differentiable from each other.
- In fact, these objects are the *generalization of quadric representations*.
- Superquadrics are formed by incorporating additional parameters into the quadric equations, which provide additional modifications in the object shapes.
- The number of additional parameters is equal to the dimensions of the object, e.g., one parameter for the curve and two parameters for the surfaces.
- Superquadric surfaces produce primitives which are useful in Constructive Solid Geometry method of solid modeling.
- *Superellipse* and *superellipsoid* are the typical superquadric surfaces.

SUPERQUADRIC SURFACES





SUPERQUADRIC SURFACES…

Superellipse

In Cartesian representation, superellipse equation is obtained from the ellipse equation by allowing the variable exponents on x and y in the form

$$\left(\frac{x}{r_x}\right)^{2/s} + \left(\frac{y}{r_y}\right)^{2/s} = 1$$

where s is a variable, which provides different shapes to the superellipse.

For $s = 1$, we get ellipse in standard form. In parametric form, superellipse can be represented as

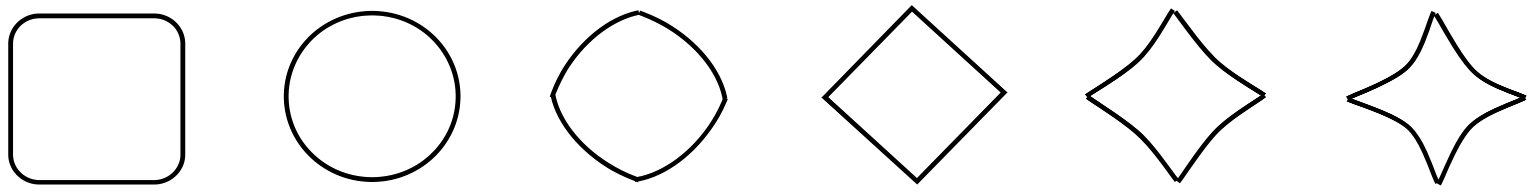
$$x = r_x . \cos^s \theta, \qquad -\pi \le \theta \le \pi$$

$$y = r_y . \sin^s \theta, \qquad -\pi \le \theta \le \pi$$

Fig. shows supercircle shapes for different values of the parameter s and a circle result for $s = 1.0$.

SUPERQUADRIC SURFACES…

Superellipse…



Superellipses for different values of parameter s with $r_x = r_y$

Superellipsoid

In Cartesian representation, the general equation of superellipsoid is obtained from the ellipsoid eqn., by allowing the variable exponents on x , y and z terms in the form

$$\left[\left(\frac{x}{r_x}\right)^{2/s_2} + \left(\frac{y}{r_y}\right)^{2/s_2}\right]^{s_2/s_1} + \left(\frac{z}{r_z}\right)^{2/s_1} = 1$$

SUPERQUADRIC SURFACES…

Superellipsoid…

For $s_1 = s_2 = 1$, a typical equation of superellipsoid is obtained in the form

$$\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2 + \left(\frac{z}{r_z}\right)^2 = 1$$

This is an ordinary equation of the ellipsoid.

In parametric form, a point on the superellipsoidal surface is expressed as

$$\left.\begin{aligned} x &= r_x \cos^{s_1}\phi . \cos^{s_2}\theta \\ y &= r_y \cos^{s_1}\phi . \sin^{s_2}\theta \\ z &= r_z \sin^{s_1}\phi \end{aligned}\right\} \begin{aligned} &(-\pi/2) \le \phi \le (\pi/2) \\ &-\pi \le \theta \le \pi \end{aligned}$$

where θ and ϕ are the latitude and longitude angles, respectively.

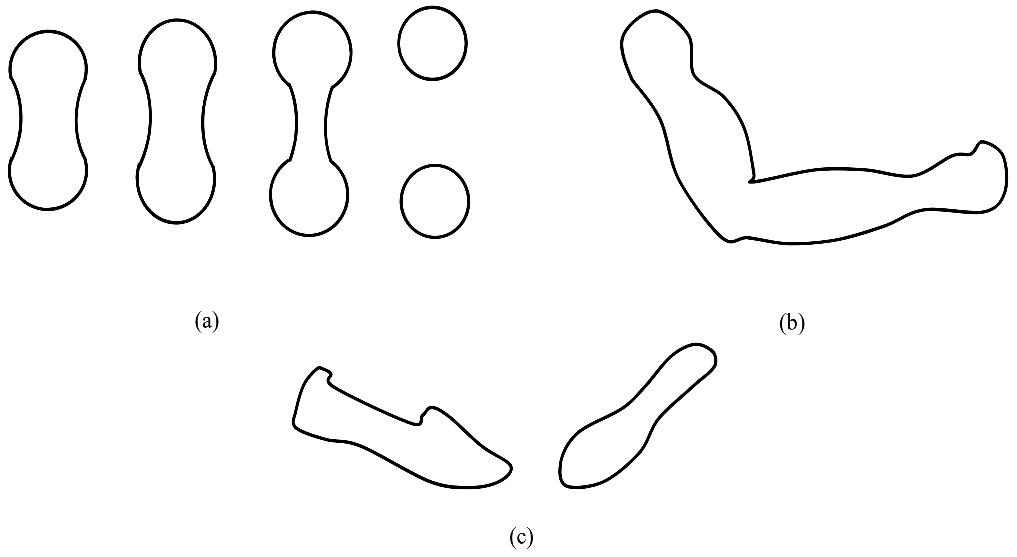
Supersphere shapes can be generated for different values of the parameters s_1 and s_2.

For $s_1 = s_2 = 1$, we get equation of sphere in standard form.

The quadric and superquadric surfaces are combined for creating the more complex shapes such as the furniture, threaded bolts, gears and other complex engineering components.

BLOOBY OBJECTS

- The quadrics and superquadrics primitives based objects maintain their shapes during the action or under certain motion or when kept in the proximity to the other objects.
- In number of situations, object boundary shape changes under the external disturbances such as molecular structure, melting objects, water droplets and human muscles shape, etc.
- These objects exhibit *blobbiness* and are often known as *blobby objects*, since their shapes behave like a fluid.



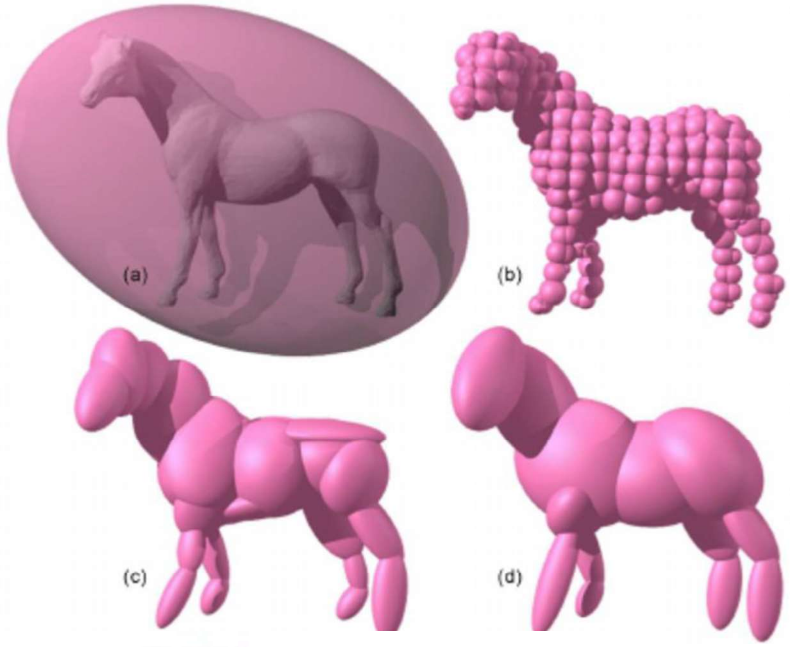
Blobby objects

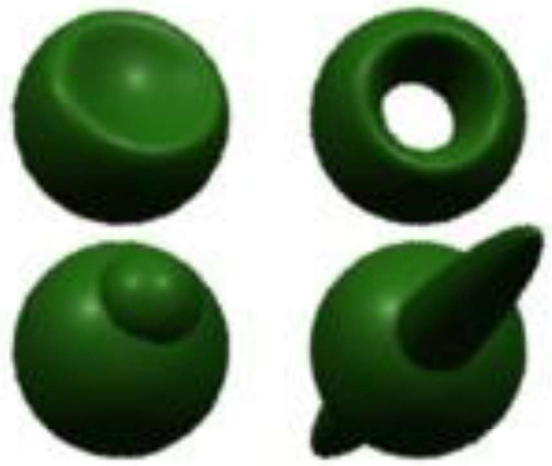
(a) Molecular bonding (two molecules move away from each other)

(b) Muscle shapes in human arms

(c) Shoe and shoe sole

BLOOBY OBJECTS





BLOOBY OBJECTS

- The modeling of blobby objects is carried out with the assumption that the *volume of material remains constant during the shape change*.
- Several models have been proposed for representing the blobby objects as distribution functions over a region of space.

Model I

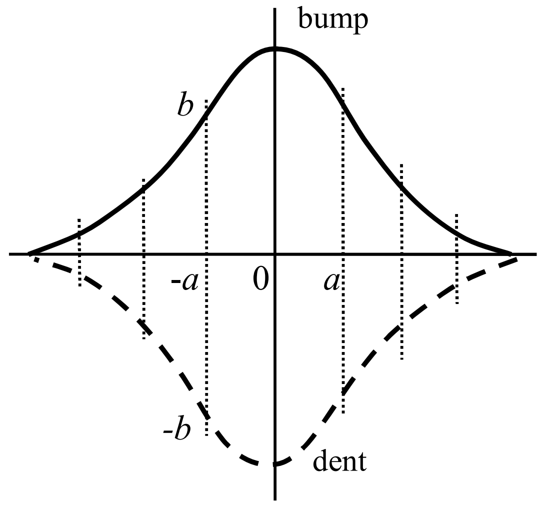
One of the simplest models for the blobby objects is the Gaussian density functions distributions or *bumps/ dents* as shown in Fig. A point (x, y, z) on the blobby surface, defined by the surface density function, as

$$f(x, y, z) = \sum_k b_k e^{-a_k r_k^2} - T = 0$$

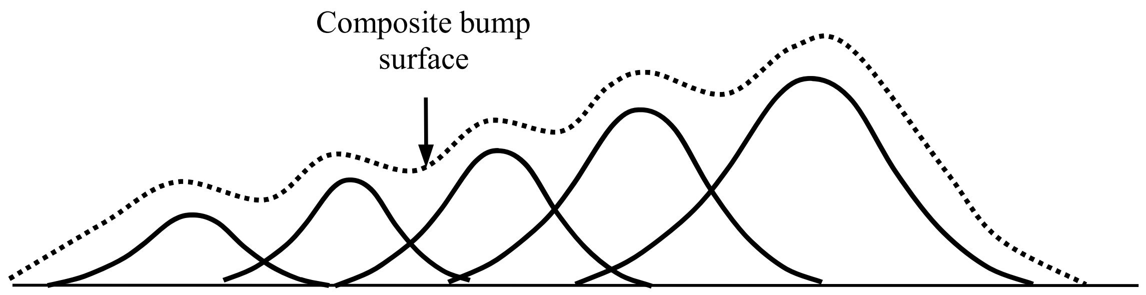
where $r_k^2 = \sqrt{x_k^2 + y_k^2 + z_k^2}$, parameter T is some specified threshold value and the parameters a and b decides the amount of blobbiness of the individual objects. Negative value of the parameter b produces *dents* instead of *bumps*. Fig. shows the composite shape of blobby surface formed with five Gaussian density functions (bumps).

BLOOBY OBJECTS

Model I



(a)



(b)

(a) A 3D Gaussian bump/dent centered at origin, with height *b* and standard deviation *a*

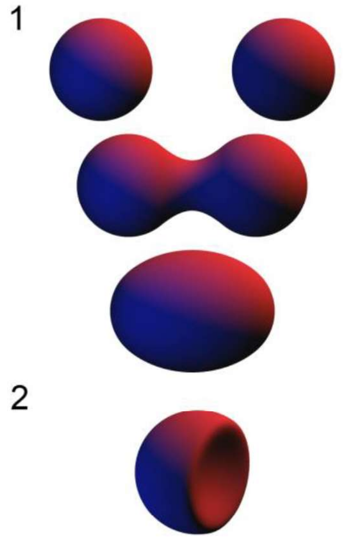
(b) A composite bump surface formed with five Gaussian bumps

BLOOBY OBJECTS

Model II

- The next model for generating the blobby objects employ the density functions that fall off to zero in a finite interval of time, than the exponential decay, known as *metaball* model.
- It describes composite objects as a combination of the quadratic density functions as

$$f(r) = \begin{cases} b\left(1 - \frac{3r^2}{d^2}\right), & \text{if } 0 < r \le \frac{d}{3} \\ \frac{3}{2}b\left(1 - \frac{r}{d}\right)^2, & \text{if } \frac{d}{3} < r \le d \\ 0, & \text{if } r > d \end{cases}$$



The modeling of human head and neck may be carried out using the metaball model.

BLOOBY OBJECTS

Model III

Soft object model uses the function

$$f(r) = \begin{cases} 1 - \dfrac{22r^2}{9d^2} + \dfrac{17r^4}{9d^4} - \dfrac{4r^6}{9d^6}, & \text{if } 0 < r \le d \\ 0, & \text{if } r > d \end{cases}$$

- The modeling of shoes and shoe soles may be carried out using the soft object model.
- Some of the design and painting software now provide blobby objects modeling, which is not possible adequately with the help of polygons, quadrics, superquadrics or spline functions alone.

COMPUTER AIDED DESIGN (BME-42)

Unit-IV: 3D Graphics

(7 Lectures)

- Introduction, Wireframe modeling, Surface modeling,
- Polygon surfaces-**polygon meshes**, **polygon equations**,
- Quadric and Superquadric surfaces, Blobby objects,
- Solid modeling-
  - Boolean set operations,
  - regularized set operations,
  - Primitive instancing,
  - Sweep representation-translational, rotational and hybrid sweeps,
  - Boundary representation-topology, geometry, boundary models
  - Constructive solid geometry-unbounded and bounded primitives

Lecture 29

Topics Covered

Limitations of Surface Modeling

Solid Modeling

Boolean Set Operations

Ordinary Boolean Operations

Regularized Boolean Operations

Prepared By

Prof. S. K. Srivastava

MED, MMMUT, Gorakhpur (UP)
sksme@mmmut.ac.in

LIMITATIONS OF SURFACE MODELING

- Both wireframe models and surface models are incapable of handling spatial addressability and topological validity of the object.
- A surface model may be used to represent a solid because surface models provide a precise definition of the surfaces and can handle complex shapes of the object
- They are slow in surface rendering process, computationally intensive and not suitable for the CAD/CAM automation and integration in future.

The applications of surface modeling scheme is limited due to the following reasons:

a) The surface modeling is unable to distinguish between the exterior and interior of a three- dimensional object; therefore, cannot be used for evaluating the internal properties of objects such as volume, weight, etc.

b) It cannot be used to create sections or to remove the hidden lines completely.

LIMITATIONS OF SURFACE MODELING…

c) Surface models, like wireframe models, are incapable of handling spatial addressability, i.e., they are incapable of verifying whether the two objects occupy same space or not.

d) Surface modeling process is slow and computationally expensive.

e) Surface modeling, similar to the wireframe modeling, stores only the geometrical information of the object.

Solid modelers store both, *geometrical* as well as *topological*, information of the object than the wireframe and surface modelers (*geometrical* information); hence, they are highly suitable for representing the three-dimensional objects.

SOLID MODELING

- The highest level of sophistication in geometric modeling is 3D (or solid) modeling.
- Solid modeling approach is an improvement over the wireframe models, both in terms of realism to the user and generating object definition in the computer.
- In this modeling approach, the models are displayed as solid objects, with very little risk of misinterpretation without any confusion.
- The model of an object is created to represent the size and shape of a component.
- The addition of colors and shading results into the strikingly realistic picture.
- Solid models are better in the sense that they allow the solid nature of an object to be defined in computer and thus help to calculate the mass (internal) properties of the object.
- The frame buffers for the solid modeling, store enough information related to the object, capable of performing *animation* of the object.

SOLID MODELING…

- In solid modelers, a 3D object is modeled in such a way that more and more external and internal information about the object can be included.
- The sectional views of the object help to understand the internal details.
- The external and internal information is used for determining the other important properties of the solids such as volume, mass, surface area, length, mass moment of inertia, etc.
- The finite element analysis (as pre or post processing) can be carried out for the solid models to study its behavior under the different types of mechanical and thermal loadings.
- Solid modeling provides a precise computer based representation of components geometry for engineering applications such as architecture, CAM, animation, rapid prototyping, molecular and aerospace/hydrospace simulation, mass property calculation, etc.

SOLID MODELING…

- The business graphics such as *bar* chart or *pie* chart can be developed through the solid modeling packages.
- The use of solid modeling in CAD/CAM is increasing rapidly due to the reduced computational costs, fast computing hardware, improved user interfaces, increased capabilities of solid modeling using advanced versions of software.
- Solid modeling approach requires a great deal of computational power, both in terms of the speed and in terms of the memory.
- All solid modelers provide facilities for creating, modifying three-dimensional objects.
- A solid model not only requires the surface and boundary geometry definitions, but also requires topological information such as interior, connectivity, holes, etc.
- Wireframe and surface models cannot describe these properties adequately.

SOLID MODELING…

- The representation schemes for the solid modeling may be divided into **five** general classes:

  I. **Boolean set operations**

  II. **Primitive instancing**

  III. **Sweep representations**

  IV. **Boundary representations (B-rep)**

  V. **Constructive Solid Geometry (CSG)**

BOOLEAN SET OPERATIONS

- Two-dimensional and three-dimensional primitives can be combined to construct solid models by Boolean set of operations such as Union, Intersection, and Difference.
- In this approach, first the solid modeler package must position the primitives at the proper position, then applying the required Boolean operator to achieve the desired shape.
- There is a wide variety of primitives available in commercial packages.
- However, the four most common primitives are the *block*, *cylinder*, *cone* and *sphere*, based on the four natural quadrics: *planes*, *cylinders*, *cones* and *spheres*, respectively.
- They are considered as natural because they represent most of the commonly used surfaces of the solids, occurring in mechanical engineering applications, such as rolling, milling, shaping and drilling operations.

BOOLEAN SET OPERATIONS…

- Boolean operations employ the following three-dimensional primitives

I. ***Block:*** This is a box of known width w, height h and depth d in the local coordinate system $x_L, y_{L,} z_L$ as shown in Fig. a, with o' as origin.

II. ***Cylinder:*** This is a right circular cylinder of known radius r (or diameter) and height h in the local coordinates system $x_L, y_{L,} z_L$ as shown in Fig. b, with o' as origin.

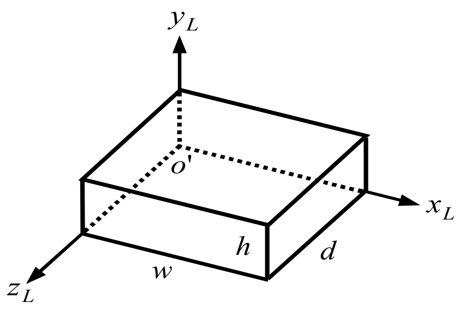
III. ***Cone:*** This is a right circular cone or frustum of right circular cone of known base radius r (and top radius for truncated cone) and height h in the local coordinates system $x_L, y_{L,} z_L$ as shown in Fig. c, with o' as origin.

IV. ***Sphere:*** This is a sphere of known radius r (or diameter) in the local coordinate system $x_L, y_{L,} z_L$ as shown in Fig. d, with o' as origin.

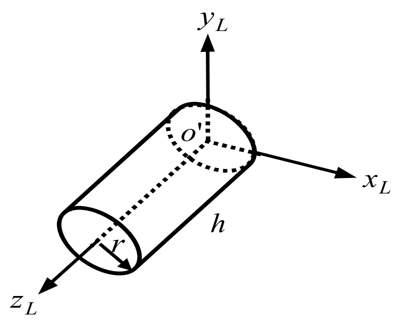
V. ***Wedge:*** This is a right angle wedge of known height h, width w and depth d in the local coordinate system $x_L, y_{L,} z_L$ as shown in Fig. e, with o' as origin.

VI. ***Torus:*** This is obtained by rotating a circle of known radius r_1 about an axis in its plane (e.g., about z_L axis) with a radius of centerline r (from z_L axis) in the local coordinate system $x_L, y_{L,} z_L$ as shown in Fig. f, with o' as origin.

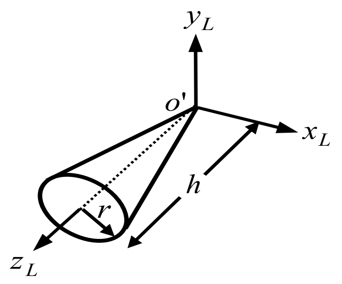
BOOLEAN SET OPERATIONS…



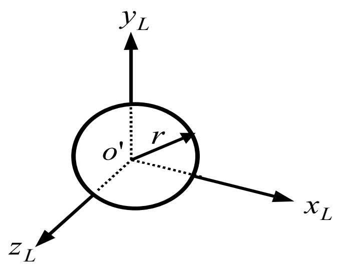
(a) Block



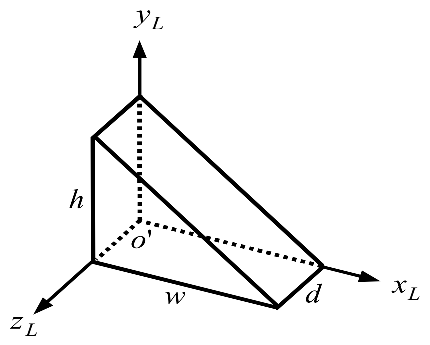
(b) Cylinder



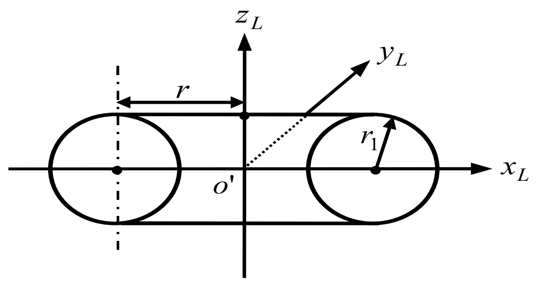
(c) Cone



(d) Sphere



(e) Wedge



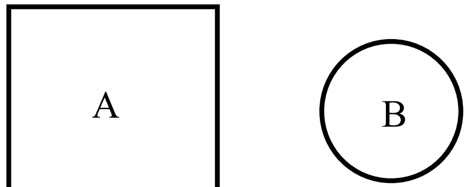
(f) Torus

BOOLEAN SET OPERATIONS…

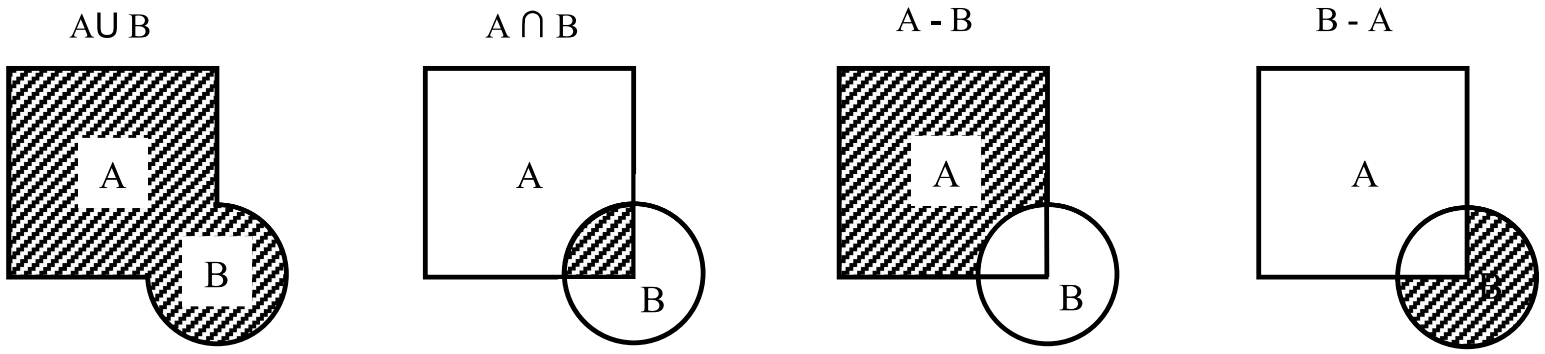
- These primitives have default location and default geometry (e.g. default values of 1) in solid modeling packages.
- The local coordinates system may also differ from package to package.
- Therefore, the primitives are usually translated and/or rotated to achieve the desired shapes and positions of the solid before applying the Boolean set operations, viz., Union (U or +), Intersection (∩ or I) and Difference (-).
- The union operator adds or combines two primitives, intersection represents the common volume of two primitives, and difference operator subtracts one primitive from the other giving the difference in their volumes.

BOOLEAN SET OPERATIONS…

Fig. shows the Two-dimensional Boolean operations, applied on the primitives.



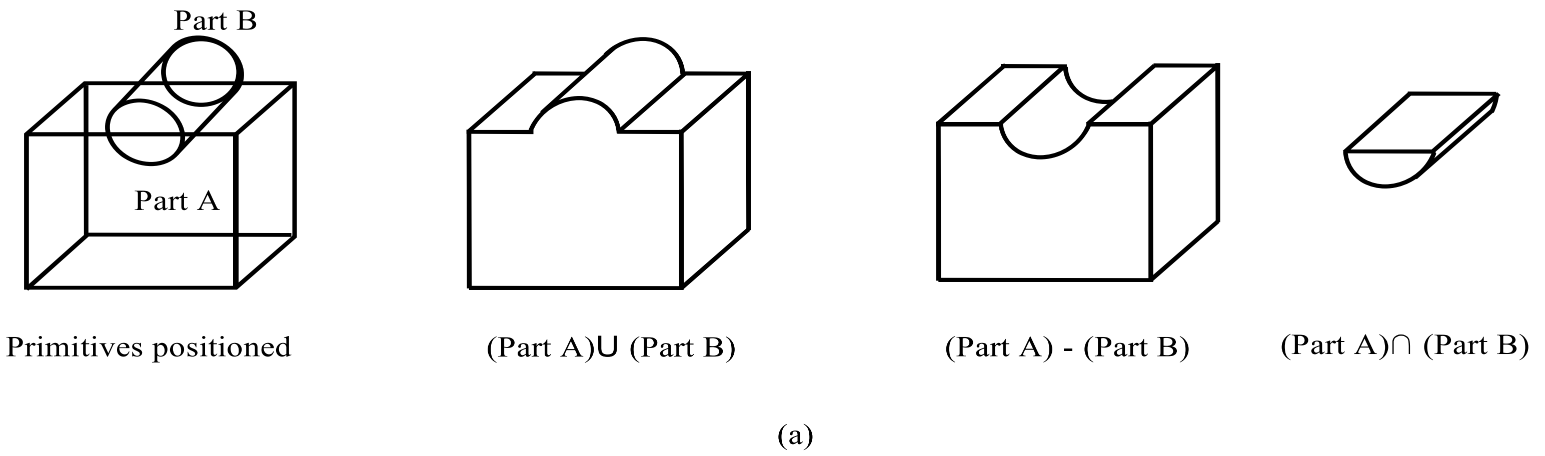
Primitives

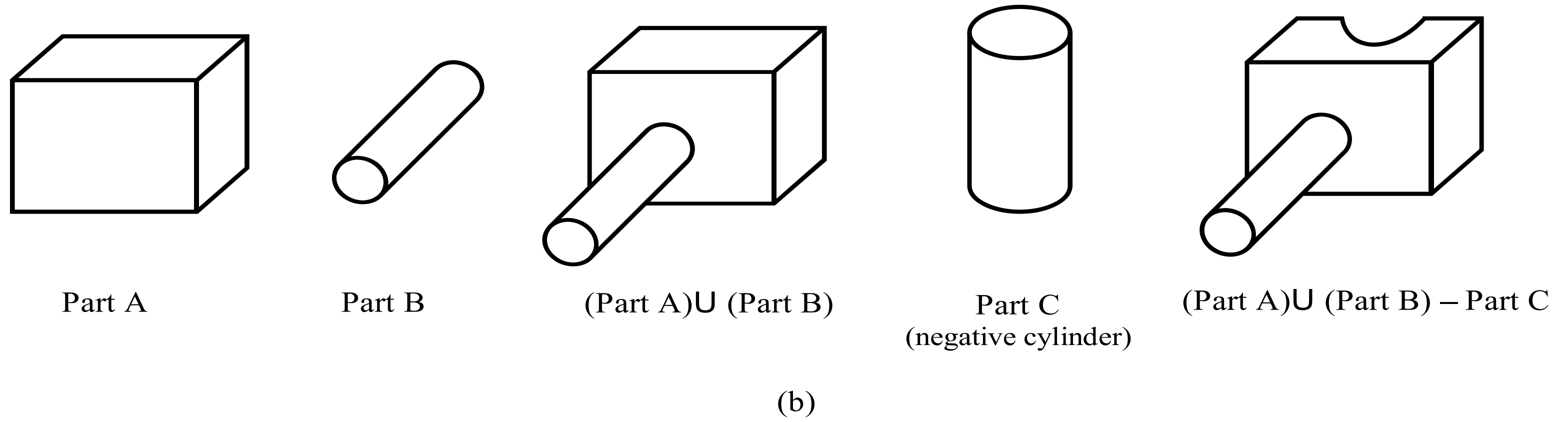


Two-dimensional Boolean operations

BOOLEAN SET OPERATIONS…

- Fig. a shows the way by which three-dimensional primitives (cube and cylinder can be added, subtracted and intersected.





Three dimensional Boolean Operations

BOOLEAN SET OPERATIONS…

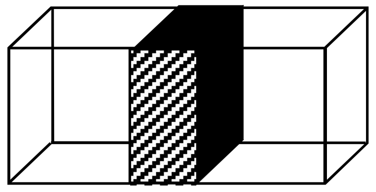
- In Fig. b, the three-dimensional primitives can be added together at some position using the union operation to create the solid.
- A hole can be generated in the resulting solid by intersecting it with a negative cylinder. Boolean operation approach is highly suitable for creating the simple objects.
- Rather than applying ordinary Boolean set operations, we use regularized Boolean set operations for generating the solid models.

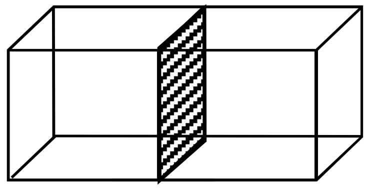
Ordinary Boolean Operations

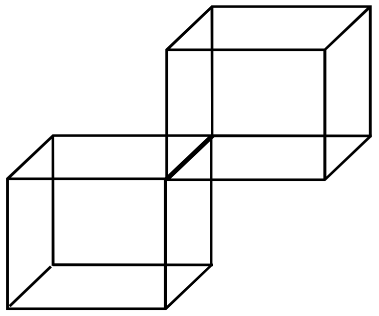
- Applying an ordinary Boolean set operation to the two solid primitives, however, does not always yield a solid object.
- This can be understood by an example. The ordinary Boolean set intersections of two cubes yield a solid (Fig. a), a plane (Fig. b), a line (Fig. c), a point (Fig. d) and the null object (Fig. e).

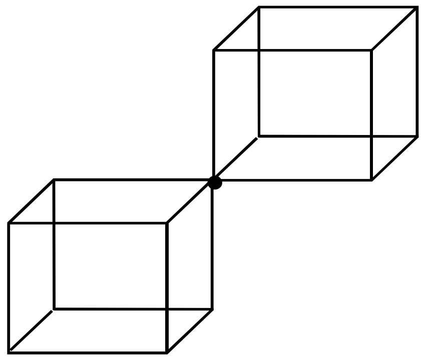
BOOLEAN SET OPERATIONS…

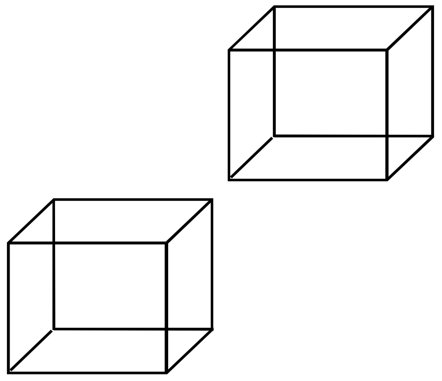
Ordinary Boolean Operations…


(a) solid


(b) plane


(c) line


(d) point


(e) null object

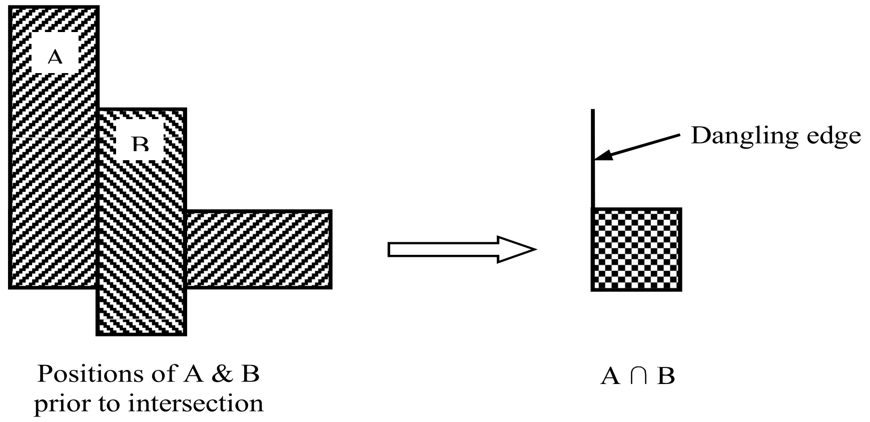
Ordinary Boolean Operations on Two Cubes

BOOLEAN SET OPERATIONS…

Ordinary Boolean Operations…

- The objects resulting from ordinary Boolean operations (U, ∩, -) may lack geometric closure, may be difficult to validate; therefore, may be inadequate for the solid modeling applications.
- The strict applications of ordinary Boolean operators like union, intersection or difference to primitives for creating a solid model would lead to a nonsense (anomalous) object with pendent (*dangling*) edge as shown in Fig.
- Therefore, these operators are replaced by the regularized Boolean operators, which results into valid or no-nonsense solid models.

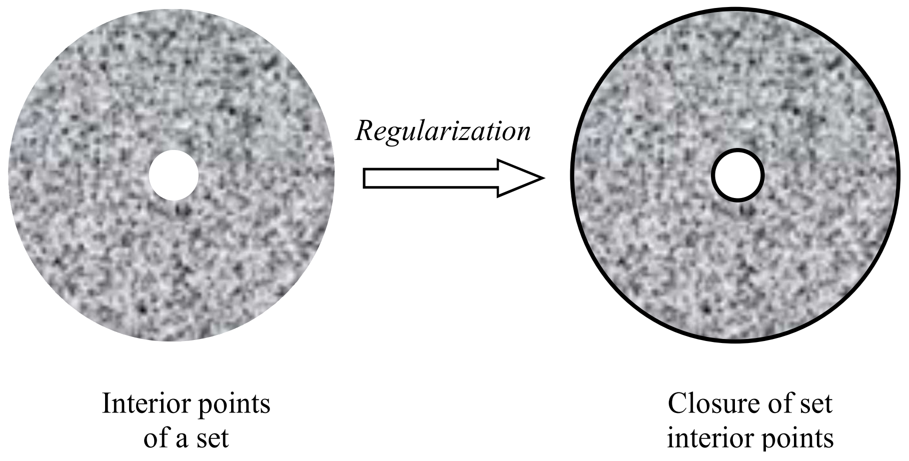
Ordinary Boolean intersection for Shapes A and B



BOOLEAN SET OPERATIONS…

Regularized Boolean Operations…

- The regularized Boolean set operations omit dangling edges, faces or vertices if occur. A regularization of a group of points (set) is defined as the closure of the set's interior points.
- Fig. shows the closure of the object.
- It should be noted that a regular set does not have those boundary points, which are not adjacent to some interior points; hence, it does not have any dangling points, edges (lines) and faces (surfaces).



Regularization of Set Interior Points

BOOLEAN SET OPERATIONS…

Regularized Boolean Operations…

The regularized Boolean set operator are defined as

A (operator)* B = Closure (interior (A (operator) B))

where operators are either Union ($\cup$) or Intersection ($\cap$) or Difference (-).

For intersection operator (say), we write

$$A \cap^* B = \text{Closure}(\text{interior}(A \cap B))$$

- The regularized Boolean set operator produces only regular set when applied onto an ordinary set.
- When regularized Boolean intersection is applied on the cubes, then Figs. a & e yield same results as their ordinary Boolean intersection, but is empty in Figs. b, c & d.

BOOLEAN SET OPERATIONS…

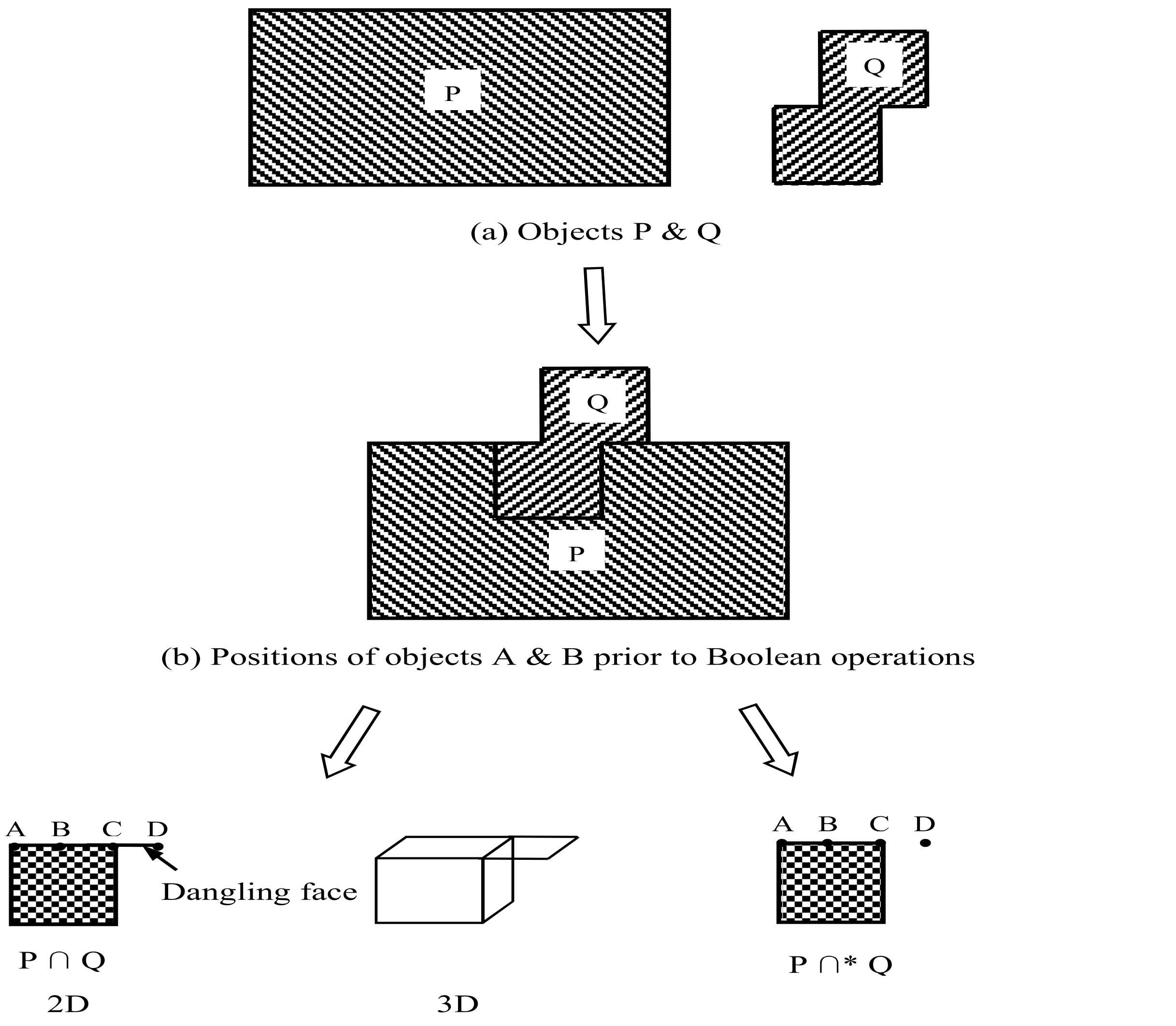
Comparison of Ordinary and Regularized Boolean Operations

- The ordinary Boolean intersection of two objects P and Q, shown in Fig., contains the intersection of the interior and boundary points of each object with the interior and boundary of the other.
- They result into a dangling face CD, shown as a line CD, in the cross-section.
- It includes a shared boundary (AB) in the resulting boundary if both objects lie on the same side of it, and excludes the shared boundary (CD) if the objects lie on the opposite sides.
- However, boundary interior intersection (BC) is always included to maintain the closure.
- The regularized Boolean set operators are used as a user interface to develop the complex objects from simple ones, in most of the three-dimensional object representation schemes.
- They also explicitly used in the Constructive Solid Geometry (CSG) scheme of solid modeling.

BOOLEAN SET OPERATIONS…

Boolean Intersection of

Objects P and Q



P
Q
(a) Objects P & Q
Q
P
(b) Positions of objects A & B prior to Boolean operations
A B C D
Dangling face
P ∩ Q
2D
3D
(c) Ordinary Boolean intersection
A B C D
P ∩* Q
(d) Regularized Boolean intersection