7. Three-Dimensional Viewing

Viewing in 3D involves the following considerations:

- We can view an object from any spatial position, eg.

  In front of an object,
  Behind the object,
  In the middle of a group of objects,
  Inside an object, etc.

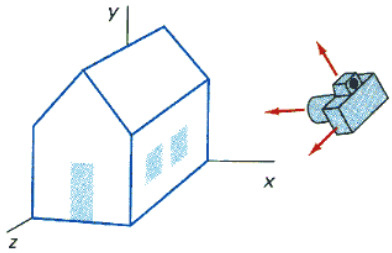
- 3D descriptions of objects must be projected onto the flat viewing surface of the output device.

- The clipping boundaries enclose a volume of space.

7.1 Viewing Pipeline

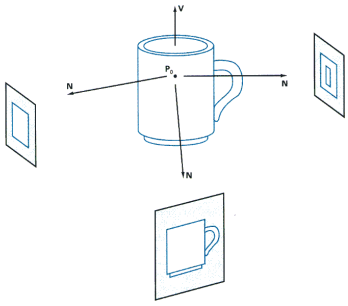
Modelling Coordinates
⇓
[Modelling Transformations]
⇓
World Coordinates
⇓
[Viewing Transformation]
⇓
Viewing Coordinates
⇓
[Projection Transformation]
⇓
Projection Coordinates
⇓
[Workstation Transformation]
⇓
Device Coordinates

Explanation:



Modelling Transformation and Viewing Transformation can be done by 3D transformations.

The viewing-coordinate system is used in graphics packages as a reference for specifying the observer viewing position and the position of the projection plane.



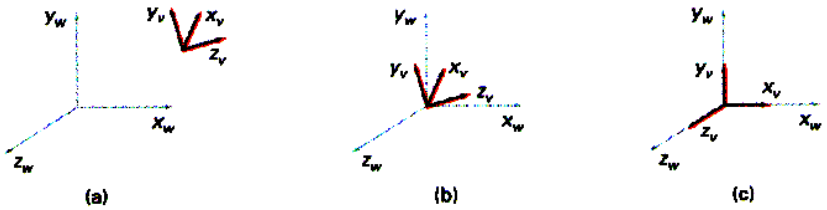
Projection operations convert the viewing-coordinate description to coordinate positions on the projection plane. (Involve clipping, visual-surface identification, and surface-rendering)

Workstation transformation maps the coordinate positions on the projection plane to the output device.

7.2 Viewing Transformation

Conversion of objection descriptions from world to viewing coordinates is equivalent to a transformation that superimposes the viewing reference frame onto the world frame using the basic geometric translate-rotate operations:

1. Translate the view reference point to the origin of the world-coordinate system.
2. Apply rotations to align the x_v, y_v, and z_v axes (viewing coordinate system) with the world x_w, y_w, z_w axes, respectively.

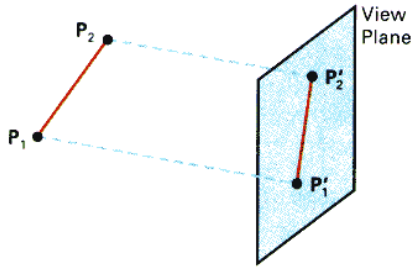


7.3 Projections

There are 2 basic projection methods:

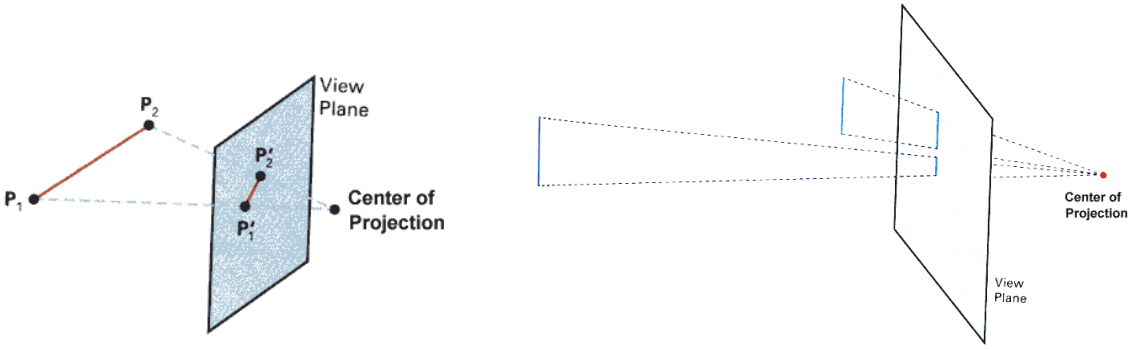
1. Parallel Projection transforms object positions to the view plane along parallel lines.

A parallel projection preserves relative proportions of objects. Accurate views of the various sides of an object are obtained with a parallel projection. But not a realistic representation.



2. Perspective Projection transforms object positions to the view plane while converging to a center point of projection.

Perspective projection produces realistic views but does not preserve relative proportions. Projections of distant objects are smaller than the projections of objects of the same size that are closer to the projection plane.



Parallel Projection

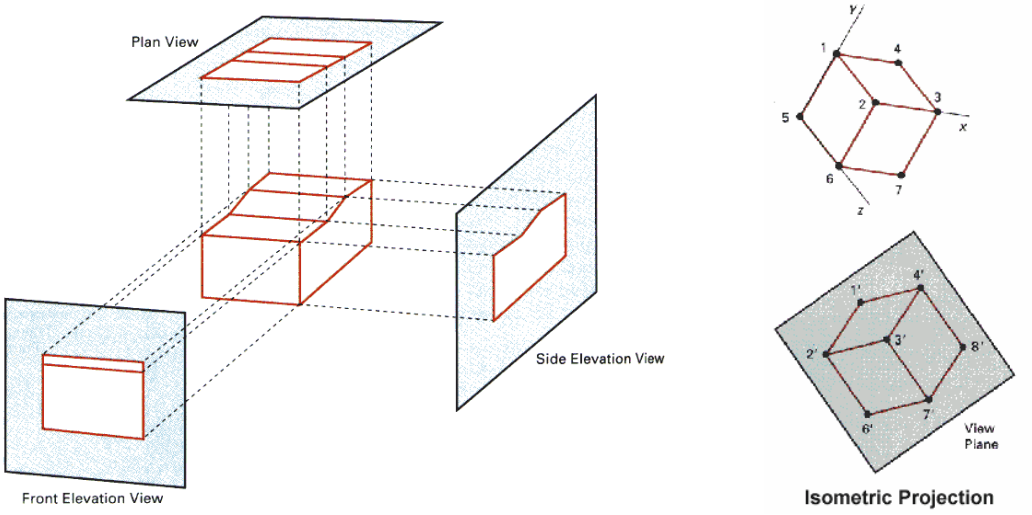
Classification:

Orthographic Parallel Projection and Oblique Projection:

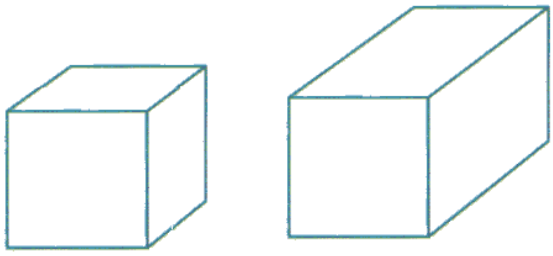
Orthographic parallel projections are done by projecting points along parallel lines that are perpendicular to the projection plane.

Oblique projections are obtained by projecting along parallel lines that are NOT perpendicular to the projection plane.

Some special Orthographic Parallel Projections involve Plan View (Top projection), Side Elevations, and Isometric Projection:



The following results can be obtained from oblique projections of a cube:



Interested students may find detailed discussion on the steps to do these projections in the text book.

Perspective Projection

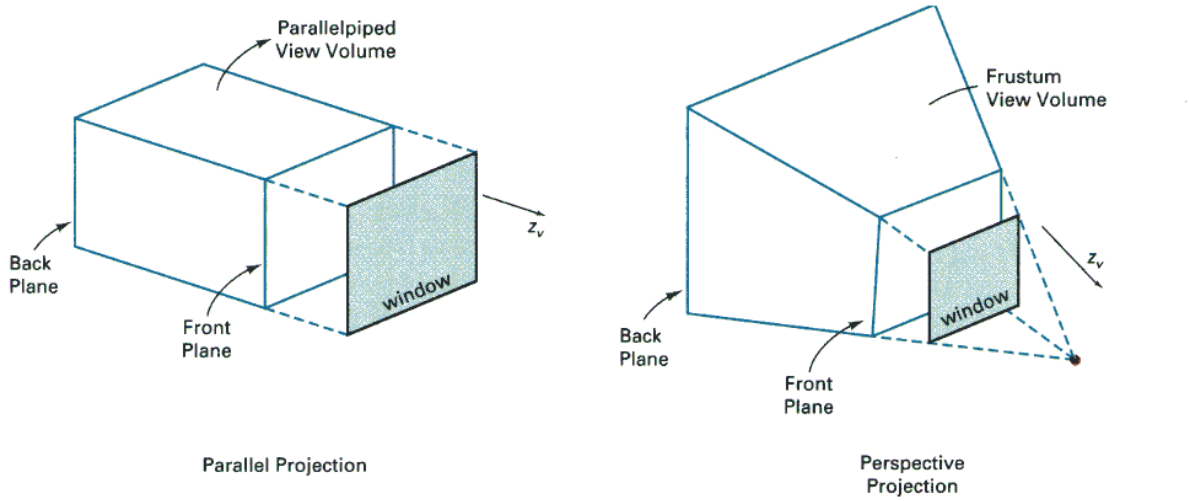
The details of perspective projection are not covered in this course. Interested students can study the topic according to the text book.

7.4 View Volumes

View window - A rectangular area in the view plane which controls how much of the scene is viewed. The edges of the view window are parallel to the x_v and y_v viewing axes.

View volume - formed by the view window and the type of projection to be used. Only those objects within the view volume will appear in the generated display. So we can exclude objects that are beyond the view volume when we render the objects in the scene.

A finite view volume is obtained by bounding with front plane and back plane (or the near plane and the far plane). Hence a view volume is bounded by 6 planes => rectangular parallelepiped or a frustum, for parallel projection and perspective projection respectively.



Some facts:

Perspective effects depend on the positioning of the center point of projection. If it is close to the view plane, perspective effects are emphasised, ie. closer objects will appear larger than more distant objects of the same size.
The projected size of an object is also affected by the relative position of the object and the view plane.

'Viewing' a static view:

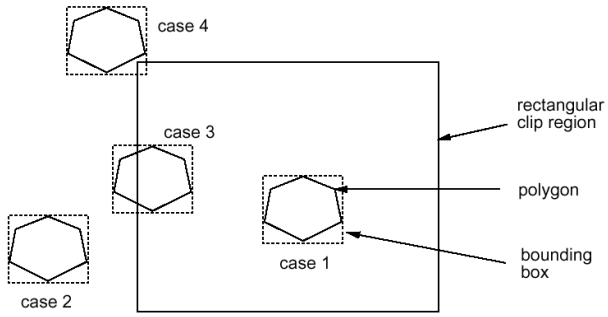
The view plane is usually placed at the viewing-coordinate origin and the center of projection is positioned to obtain the amount of perspective desired.

'Viewing' an animation sequence:

Usually the center of projection point is placed at the viewing-coordinate origin and the view plane is placed in front of the scene. The size of the view window is adjusted to obtain the amount of scene desired. We move through the scene by moving the viewing reference frame (ie. the viewing coordinate system).

7.5 Clipping

The purpose of 3D clipping is to identify and save all surface segments within the view volume for display on the output device. All parts of objects that are outside the view volume are discarded. Thus the computing time is saved.

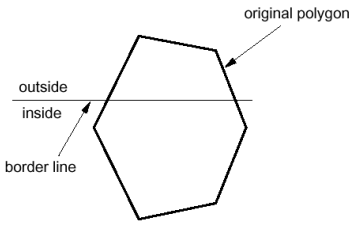


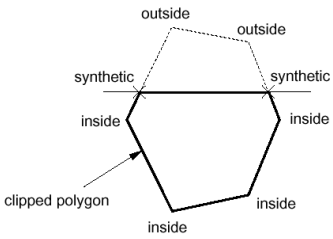
3D clipping is based on 2D clipping. To understand the basic concept we consider the following algorithm:

Polygon Clipping

Assuming the clip region is a rectangular area,

1. The rectangular clip region can be represented by x_{min}, x_{max}, y_{min} and y_{max}.
2. Find the bounding box for the polygon: ie. the smallest rectangle enclosing the entire polygon.
3. Compare the bounding box with the clip region (by comparing their x_{min}, x_{max}, y_{min} and y_{max}).
4. If the bounding box for the polygon is completely outside the clip region (case 2), the polygon is outside the clip region and no clipping is needed.
5. If the bounding box for the polygon is completely inside the clip region (case 1), the polygon is inside the clip region and no clipping is needed.
6. Otherwise, the bounding box for the polygon overlaps with the clip region (cases 3 and 4) and the polygon is likely to be partly inside and partly outside of the clip region. In that case, we clip the polygon against each of the 4 border lines of the clip region in sequence as follows:
    1. Using the first vertex as the current vertex. If the point is in the inside of the border line, mark it as 'inside'. If it is outside, mark it as 'outside'.
    2. Check the next vertex. Again mark it 'inside' or 'outside' accordingly.
    3. Compare the current and the next vertices. If one is marked 'inside' and the other 'outside', the edge joining the 2 vertices crosses the border line.
    4. In this case, we need to calculate where the edge intersects the border (ie. intersection between 2 lines).
    5. The intersection point becomes a new vertex and we mark it as 'synthetic'.
    6. Now we set the next vertex as the current vertex and the following vertex as the next vertex, and we repeat the same operations until all the edges of the polygon have been considered.
    7. After the whole polygon has been clipped by a border, we throw away all the vertices marked 'outside' while keeping those marked as 'inside' or 'synthetic' to create a new polygon.
    8. We repeat the clipping process with the new polygon against the next border line of the clip region.



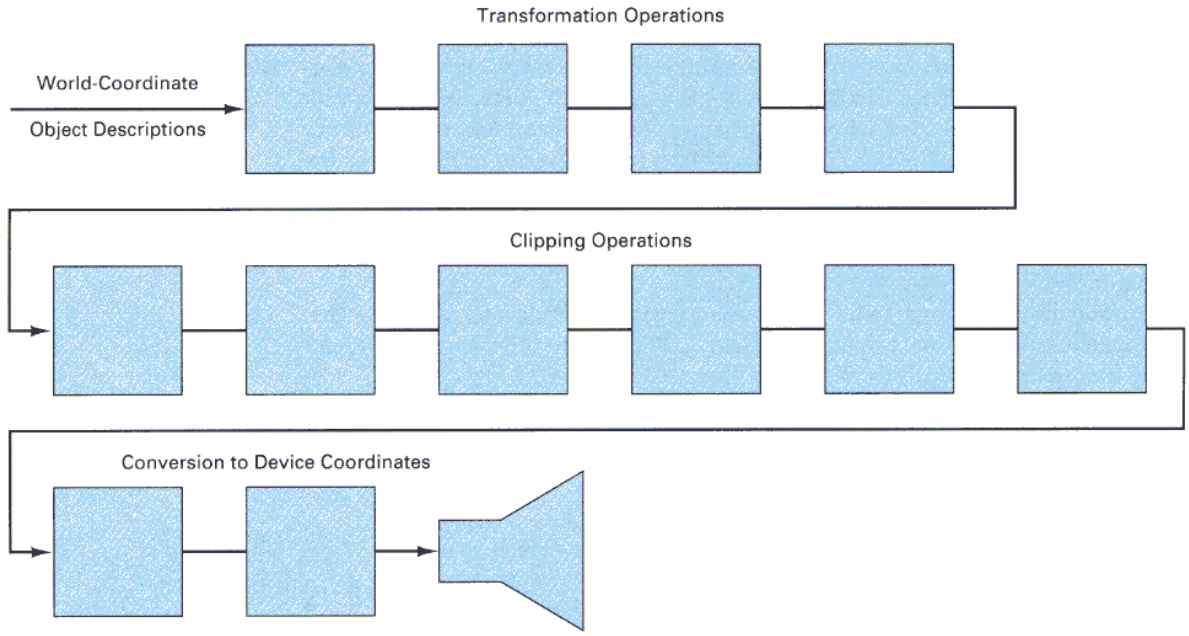


7. This clipping operation results in a polygon which is totally inside the clip region.

7.6 Hardware Implementations

Most graphics processes are now implemented in graphics chip sets. Hardware systems are now designed to transform, clip, and project objects to the output device for either 3D or 2D applications.

In a typical arrangement, each of the individual chips in a chip set is responsible for geometric transformations, projection transformation, clipping, visible-surface identification, surface-shading procedure, octree representation processing, or ray-tracing etc., in a pipe-line way.



A hardware implementation of three-dimensional viewing operations using 12 chips for the coordinate transformations and clipping operations.

8. 3D Object Representations

Methods:

- Polygon and Quadric surfaces: For simple Euclidean objects
- Spline surfaces and construction: For curved surfaces
- Procedural methods: Eg. Fractals, Particle systems
- Physically based modeling methods
- Octree Encoding
- Isosurface displays, Volume rendering, etc.

Classification:
Boundary Representations (B-reps) eg. Polygon facets and spline patches
Space-partitioning representations eg. Octree Representation

Objects may also associate with other properties such as mass, volume, so as to determine their response to stress and temperature etc.

8.1 Polygon Surfaces

This method simplifies and speeds up the surface rendering and display of objects.

For other 3D objection representations, they are often converted into polygon surfaces before rendering.

Polygon Mesh
- Using a set of connected polygonally bounded planar surfaces to represent an object, which may have curved surfaces or curved edges.
- The wireframe display of such object can be displayed quickly to give general indication of the surface structure.
- Realistic renderings can be produced by interpolating shading patterns across the polygon surfaces to eliminate or reduce the presence of polygon edge boundaries.

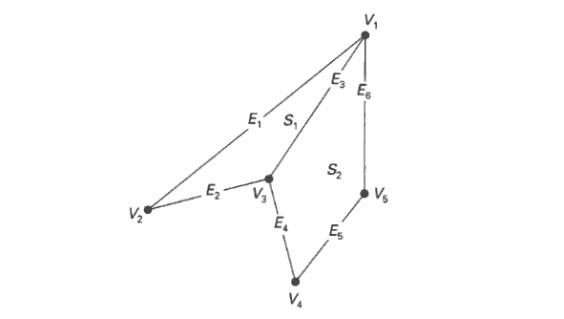
Polygon Tables

This is the specification of polygon surfaces using vertex coordinates and other attributes:

1. Geometric data table: vertices, edges, and polygon surfaces.
2. Attribute table: eg. Degree of transparency and surface reflectivity etc.

Some consistency checks of the geometric data table:

- Every vertex is listed as an endpoint for at least 2 edges
- Every edge is part of at least one polygon
- Every polygon is closed

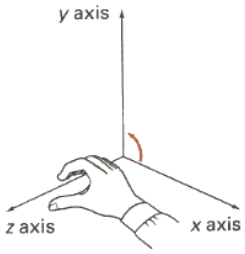


VERTEX TABLE	
V_1:	x_1, y_1, z_1
V_2:	x_2, y_2, z_2
V_3:	x_3, y_3, z_3
V_4:	x_4, y_4, z_4
V_5:	x_5, y_5, z_5

EDGE TABLE	
E_1:	V_1, V_2
E_2:	V_2, V_3
E_3:	V_3, V_1
E_4:	V_3, V_4
E_5:	V_4, V_5
E_6:	V_5, V_1

POLYGON-SURFACE TABLE	
S_1:	E_1, E_2, E_3
S_2:	E_3, E_4, E_5, E_6

Plane equation and visible points



Consider a cube, each of the 6 planes has 2 sides: inside face and outside face.

For each plane (in a right-handed coordinate system), if we look at its surface and take 3 points in counter-clockwise direction: (x_1,y_1), (x_2,y_2), and (x_3,y_3), we can compute 4 values: A,B,C,D as

$$A = \begin{vmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{vmatrix} \quad B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix} \quad C = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} \quad D = -\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix}$$

Then, the plane equation at the form: Ax+By+Cz+D=0 has the property that:

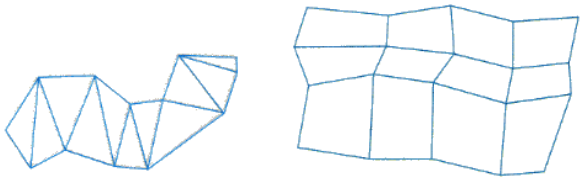
If we substitute any arbitrary point (x,y) into this equation, then,

Ax + By + Cz + D < 0 implies that the point (x,y) is inside the surface, and

Ax + By + Cz + D < 1 implies that the point (x,y) is outside the surface.

Polygon Meshes

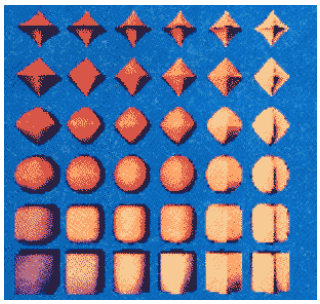
Common types of polygon meshes are triangle strip and quadrilateral mesh.



Fast hardware-implemented polygon renderers are capable of displaying up to 1,000,000 or more shaded triangles per second, including the application of surface texture and special lighting effects.

8.2 Curved Surfaces

1. Regular curved surfaces can be generated as
- Quadric Surfaces, eg. Sphere, Ellipsoid, or
- Superquadrics, eg. Superellipsoids



These surfaces can be represented by some simple parametric equations, eg, for ellipsoid:

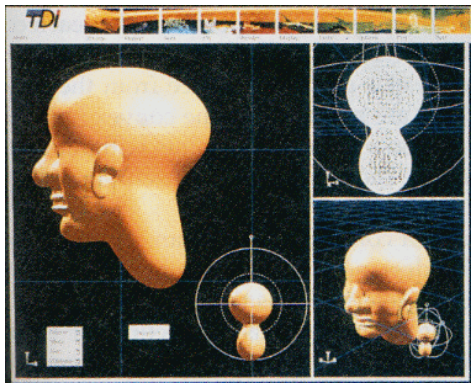
$x = r_x \cos^{s1}\phi \cos^{s2}\theta, \quad -\pi/2 <= \phi <= \pi/2$

$y = r_y \cos^{s1}\phi \sin^{s2}\theta, \quad -\pi <= \theta <= \pi$

$z = r_z \sin^{s1}\phi$

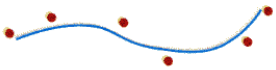
Where s1, r_x, r_y, and r_x are constants. By varying the values of ϕ and θ, points on the surface can be computed.

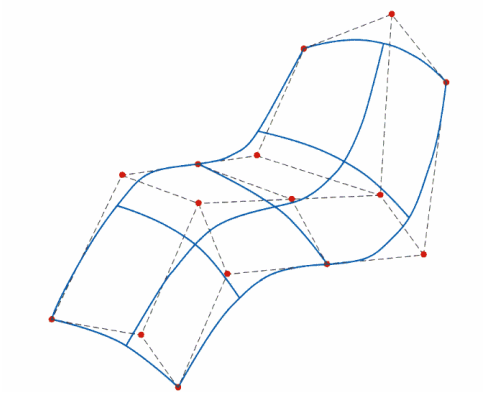
2. Irregular surfaces can also be generated using some special formulating approach, to form a kind of **blobby objects** -- The shapes showing a certain degree of fluidity.



3. Spline Representations

Spline means a flexible strip used to produce a smooth curve through a designated set of points. Several small weights are distributed along the length of the strip to hold it in position on the drafting table as the curve is drawn.
We can mathematically describe such a curve with a piecewise cubic polynomial function => spline curves. Then a spline surface can be described with 2 sets of orthogonal spline curves.

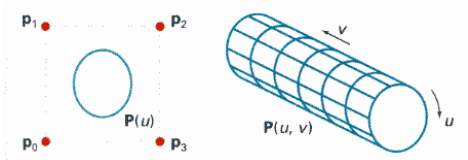




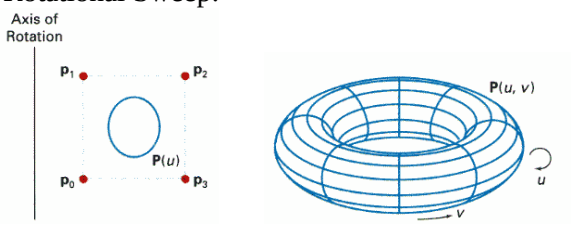
8.3 Sweep Representations

Sweep representations mean sweeping a 2D surface in 3D space to create an object. However, the objects created by this method are usually converted into polygon meshes and/or parametric surfaces before storing.

A Translational Sweep:



A Rotational Sweep:



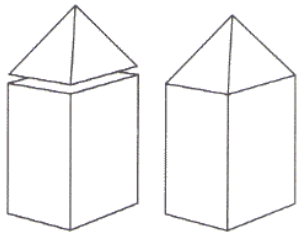
Other variations:
- We can specify special path for the sweep as some curve function.
- We can vary the shape or size of the cross section along the sweep path.
- We can also vary the orientation of the cross section relative to the sweep path.

8.4 Constructive Solid-Geometry Methods

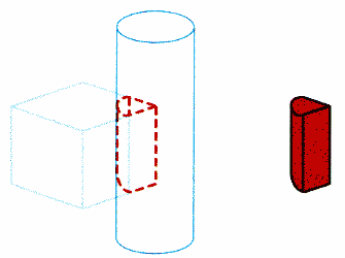
The Constructive Solid-Geometry Method (CSG) combine the volumes occupied by overlapping 3D objects using set operations:

- Union
- Intersection
- Difference

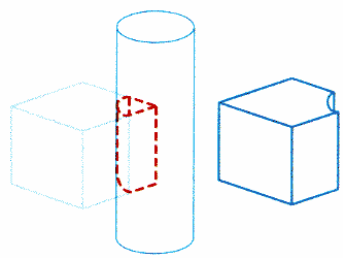
Object created by a union operation:



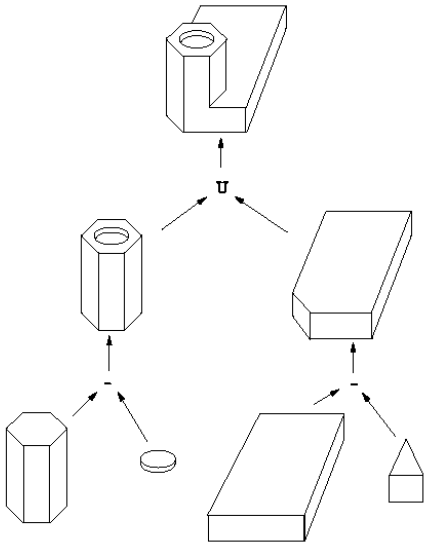
Object created by a intersection operation:



Object created by a difference operation:



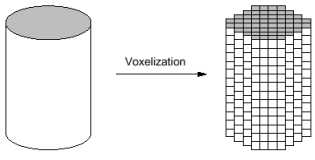
A CSG object can be represented with a binary tree:



8.5 Voxel Representation

In voxel representation, an object is decomposed into identical cells arranged in a fixed regular grid. These cells are called voxels (volume elements), in analogy to pixels.

Eg. A cylinder can be represented as follows by voxels. A '1' may represent inside the cylinder wile a '0' may represent outside of the cylinder. Alternatively we may use 8 bits to represent the transparency value. If a voxel has a value of '0', it is a fully-transparent cell. If a voxel has a value of '255', it is a non-transparent cell.

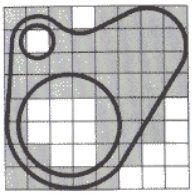


8.6 Octrees

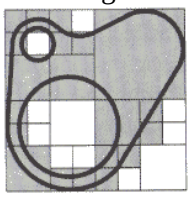
Octrees are hierarchical tree structures that describe each region of 3D space as nodes. When compared with the basic voxel representation, octrees reduce storage requirements for 3D objects. It also provides a convenient representation for storing information about object interiors.

Octree encoding procedure is an extension of the quadtree encoding of 2D images:

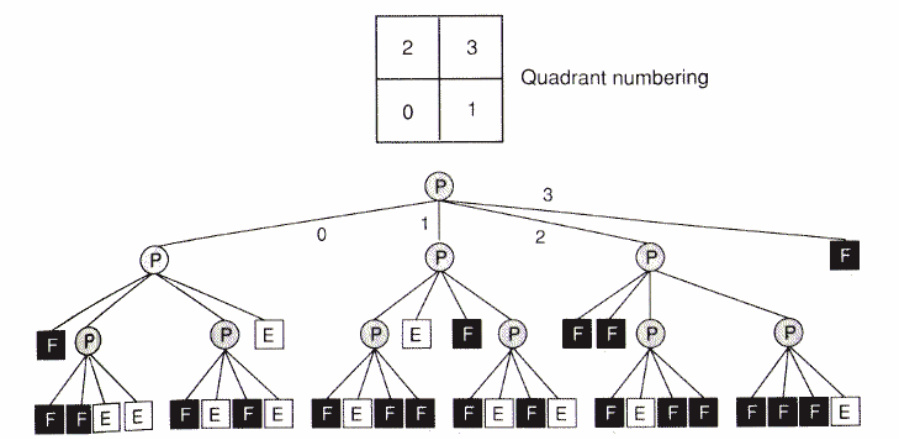
Bitmap representation of a 2D object:



Quadtree encoding of a 2D object:

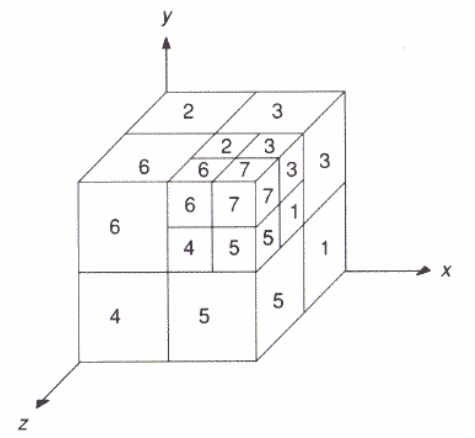


Quadtree data structure for the above object: F = full, P = partially full, E = empty.



The code of this image in quadree representation is: P PPPF FPPE PEFP FFPP FFEE FEFE FEFF FEFE FEFF FFFE.

For octree representation of 3D data, the number method is as follows:



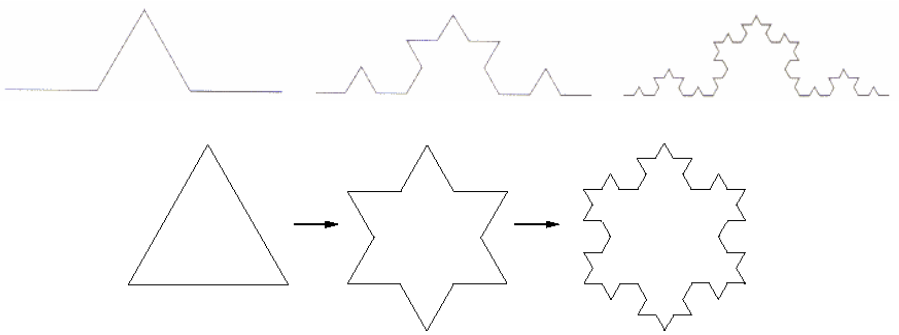
8.7 Fractals

Fractal objects refer to those objects which are self-similar at all resolution.

Most of the natural objects such as trees, mountains and coastlines are considered as fractal objects because no matter how far or how close one looks at them, they always appear to be somewhat similar.

Fractal objects can also be generated recursively by applying the same transformation function to an object, eg. Scale down + rotate + translate.

For example, a Fractal Snowflake:

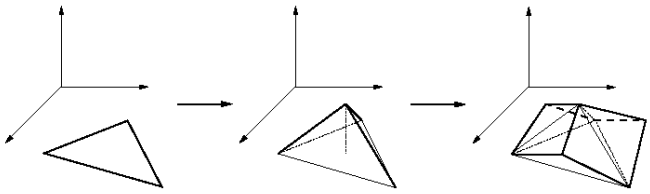


The name "Fractal" comes from its property: fractional dimension. Unlike Euclidean dimension, the dimensions of fractal objects may not be integers, and we call these dimensions as 'fractal dimension'.

Euclidean dimensions and Fractal dimensions:

- A line segment is 1D. If we divide a line into N equal parts, the parts each look like the original line scaled down by a factor of $N = N^{1/1}$.
- A square is 2D. If we divide it into N parts, each part looks like the original scaled down by a factor of $N^{1/2}$.
- For the fractal snowflake, when it is divided into 4 pieces, each resulting piece looks like the original scaled down by a factor of 3, so it has the dimension ***d*** such that $4^{1/d}=3$. That is, d = 1.26.

We may also create 3D objects in a similar way by adding a third dimension. The following example replaces each triangle of the object with a pyramid, by inserting a single point, at each step (except for the bottom face). This will result in a mountain like object, with a regular appearance.



To give a more natural appearance to the created object, we usually allow some limited random variations at each level of recursion: