**Unit 3 : 3D Object Representation**

Realization In 3-D Graphics:

Many computer graphic application involve the display of 3-D objects and scenes. For examples CAD systems allow the users to manipulate models of machine components, automobile bodies and aircraft parts. Simulation system present a continuously moving picture of a 3-D world to the pilot of ship or aircraft. These applications differ from 2-D applications not only in the added dimension: they also requires concern for realism in the display of objects.

Producing a realistic image of a 3-D scene on a 2D-display present many problem. For example ,depth in the 3rd dimension to be displayed on the scene, modulation of 3D object in a computer so that images can be generated. There are a number of techniques for achieving realism.

The basic problem addressed by visualization technique is sometimes called depth cueing. When   
a 3-D scene is projected onto a 2-D display scene , information about the depth of objects in the   
image tends to be reduced or loosed entirely. Techniques that provide depth cues are designated   
to restore or enhance the communication of depths to the observer. The different technique for   
achieving realism are:

- Parallel projection.

- Perspective projection

- Intensity cues.

- Stereoscopic views.

- Kinetic depth effect.

- Hidden line elimination

- Shadding with hidden surface removed

- 3-D images.

Modelling 3-D scenes:

The techniques used to generate different kinds of 3-D scenes are start from a model of the scene. The model is needed for two purpose.

1. It is used by viewing algorithm together with information about the location of the viewer.

2. It is used to modify and analyze the objects in the scene, activities usually considered part of the application program.

The information in a model of a 3-D scene can be divided into two important classes: geometry and topology. Geometry is concerned with measurements, such as the location of a point or the dimension of the object. Topological information records the structure of a scene: how points are combined to form polygons, how polygons form object and how object from scenes.

3D Object Representation

Graphics scenes can contain many different kinds of objects: trees, flowers, clouds, rocks, water,   
bricks, rubber, paper, marble, steel, glass, plastic, and cloth, just to mention a few. To produce   
realistic displays of scenes, we need to use representations that accurately model object   
characteristics.

 Polygon and quadric surfaces provide precise descriptions for simple Euclidean objects

such as polyhedrons and ellipsoids;

 Spline surfaces and construction techniques are useful for designing aircraft wings, gears,

and other engineering structures with curved surfaces;

 Procedural methods, such as fractal constructions and particle systems, allow us to give

accurate representations for clouds, clumps of grass, and other natural objects;

 Physically based modeling methods using systems of interacting forces can be used to

describe the no rigid behavior of a piece of cloth or a glob of jelly;

 Octree encodings are used to represent internal features of objects, such as those obtained

from medical CT images;

 Isosurface displays, volume renderings, and other visualization techniques are applied to

three-dimensional discrete data sets to obtain visual representations of the data.

Representation schemes for solid objects are often divided into two broad categories:

1. Boundary representations: (B-reps) describe a three-dimensional object as a set of surfaces that separate the object interior from the environment. Typical examples of boundary representations are polygon facets and spline patches

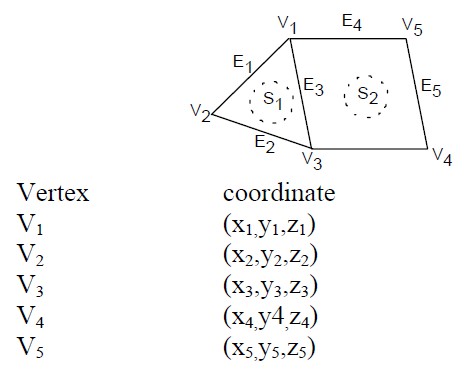
2. Space-partitioning representation: Its representations are used to describe interior properties, by partitioning the spatial region containing an object into a set of small, no overlapping, contiguous solids (usually cubes). A common space-partitioning description for a three-dimensional object is an octree representation.

Boundary representations

Polygon surface:

The most commonly used boundary representation for a 3-D graphic object is a set of surface polygon that enclose the object interior. Many graphic system stored all object description as sets of surface polygon. This simplifies and speeds up the surface rendering and display of a object ,since all the surface are described with linear equation. A polygon representation for a polyhedron precisely defines the surface features of the object. But for other objects, surfaces are tessellated (or tiled) to produce the polygon-mesh approximation. Wire frame display can be done quickly to give general indication of surface structure. Then realistic scenes are produced by interpolating shading patterns across polygon surface to illuminate.

Unit 3 : 3D Object Representation



Polygon Table

A polygon surface is specified with a set of vertex coordinates and associated attribute   
parameters. As information for each polygon is input, the data are placed into tables that are to   
be used in the subsequent processing, display, and manipulation of the objects in a scene.   
Polygon data tables can be organized into two groups: geometric tables and attribute tables.

(a) Geometric table: It contains vertex co-ordinates and parameter to identify the spatial orientation of polygon surface.

(b) Attribute table: It gives attribute information for an object (degree of transparency, surface reflectivity etc)

(a) Geometric table : A convenient organization for storing geometric data is to create three lists: a vertex table, an edge table, and a polygon table.

(i) Vertex table: It stores co-ordinate values for each vertex of the object.

(ii) The edge table stores the Edge information of each edge of polygon facets. It contains pointers back into vertex to identify the edges for each polygon.

Edge Vertex

E1 (v1,v2)

E2 (v2,v3)

E3 (v3,v1)

E4 (v1,v5)

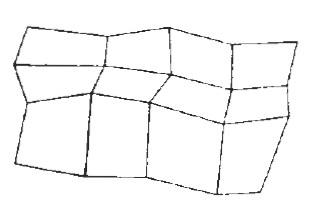
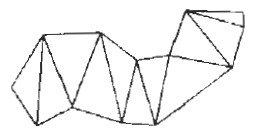
E5 (v5,v4)

E6 (v4,v3)

Surface table:

The polygon surface table stores the surface information for each surface i.e. each surface is   
represented by edge lists of polygon. The surface contains polygonal facets as shown in figure

Unit 3 : 3D Object Representation



Surface Edge

S1 (E1,E2,E3)

S2 (E3,E4,E5,E6)

Rues for creating Geometric table:

1. Every vertex in listed as an end point for at least two edges.

2. Every edge is part of at least one polygon

3. Each polygon has at least one shared edge.

4. Every surface is close.

Polygon Meshes:

A polygon mesh is collection of edges, vertices and polygons connected such that each edge is shared by at most two polygons.

i) An edge connects two vertices and a polygon is a closed sequence of edges.

ii) An edge can be shared by two polygons and a vertex is shared by at least two edges.

When object surface is to be tiled, it is more convenient to specify the surface facets with a mesh   
function. One type of polygon mesh is triangle strip. This function produce n-2 connected   
triangles.

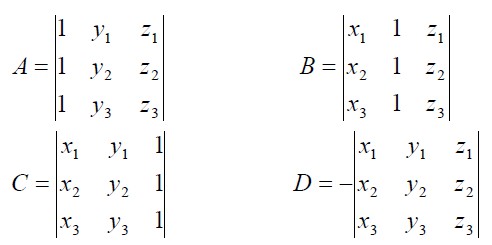
Fig: A triangle strip formed with 11 triangles connecting 13 vertices.

Another similar function is the quadrilateral mesh, which generates a mesh of (n-1) by (m-1) quadrilaterals, given the co-ordinates for an n x m array of vertices.

Fig :A quadrilateral mesh containing 12quadrilaterals construded from a 5 by 4 input vertex array

Plane Equation

Unit 3 : 3D Object Representation



For some of the surface-rendering procedures, we need information about the spatial orientation of the individual surface components of the object. This information is obtained from the vertex coordinate values and the equations that describe the polygon planes.

The equation of the plane is

Ax+By+Cz +D = 0

Solution of plane Co-efficient:

1. Algebraic Approach.

2. Vertex Approach.

We derive the solution for coefficient A,B,C and D using algebraic approach .If (x1,y1,z1) , (x2,y2,z2) and (x3,y3,z3) are three successive vertex of the polygon then

Ax +By+Cz = -D

(A/D)x +(B/D)y+(C/D)z = -1

(A/D)x1 +(B/D)y1+(C/D)z1 = -1 ……….(i)   
(A/D)x2 +(B/D)y2+(C/D)z2 = -1 ……….(ii)   
(A/D)x3 +(B/D)y3+(C/D)z3 = -1 ………(iii)

By cramers rule:

Expanding the determinant we can write that,

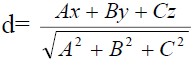
A = (y3z3 - y3z2) - y1(z3-z2) + z1(y3-y2)

B = (z3-z2) - (x2z3 - x3z2)+(x2-x3)

C = (y2 - y3) - y1 (x2 -x3)+ (x2y3 -x3y2)

D = -x2(y2z3-y3z2) +y1(x2z3-x3z2)- z1(x2y3 -x3y2)

Unit 3 : 3D Object Representation



As vertex values and other information are entered into the polygon data structure, values for A, B, C and D are computed for each polygon and store with other polygon data.

The plane equation is used to determine the position of spatial points relative to the plane surface   
of an object. If p(x,y,z) is any point on the plane then the distance between the point and the   
plane is

If d = 0 , Ax+By+Cz+D = 0 i.e point is on the plane.   
If d<0, Ax+By+Cz+D = 0 i.,e point is inside the plane.   
If d>0, Ax+By+Cz+D = 0 i.e point is outside the plane.

The side of the plane that faces the object interior is ‘inside’ face and visible face is ‘outside’   
face. If polygon vertices are specified in counter clockwise direction when viewing the outer side   
of the plane in right handed coordinate system the direction of normal vector will be from inside   
to outside.

Representing Curves Line and Surfaces

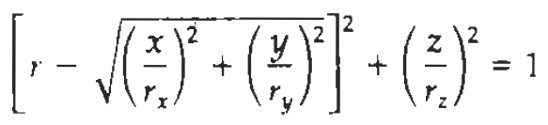
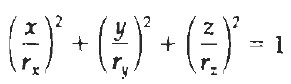
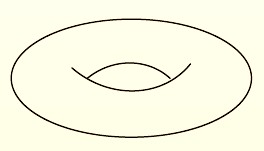
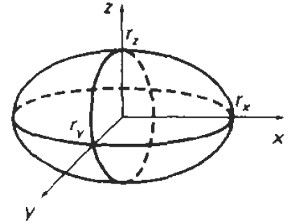
Displays of three dimensional curved lines and surfaces can be generated from an input set of   
mathematical functions defining the objects or from a set of user specified data points. When   
functions are specified, a package can project the defining equations for a curve to the display   
plane and plot pixel positions along the path of the projected function. For surfaces, a functional   
description is often tessellated to produce a polygon-mesh approximation to the surface. Usually,   
this is done with triangular polygon patches to ensure that all vertices of any polygon are in one   
plane. Polygons specified with four or more vertices may not have all vertices in a single plane.

Examples of display surfaces generated from functional descriptions include the quadrics and the   
super quadrics. When a set of discrete coordinate points is used to specify an object shape, a   
functional description is obtained that best fits the designated points according to the constraints   
of the application. Spline representations are examples of this class of curves and surfaces. These   
methods are commonly used to design new object shapes, to digitize drawings, and to describe   
animation paths. Curve-fitting methods are also used to display graphs of data values by fitting   
specified curve functions to the discrete data set, using regression techniques such as the least-  
squares method. Curve and surface equations can be expressed in either a parametric or a   
nonparametric form.

QUADRIC SURFACE

A frequently used class of objects are the quadric surfaces, which are described with second-  
degree equations (quadratics).A quadric is a generalization of a conic section to 3D. They

Unit 3 : 3D Object Representation



include spheres, ellipsoids, tori, paraboloids, and hyperboloids. Quadric surfaces, particularly spheres and ellipsoids, are common elements of graphics scenes, and they are often available in graphics packages as primitives form which more complex objects can be constructed.

There are six basic types of quadric surfaces, which depend on the signs of the parameters. They are the ellipsoid, hyperboloid of one sheet, hyperboloid of two sheets, elliptic cone, elliptic paraboloid, and hyperbolic paraboloid (saddle). All but the hyperbolic paraboloid may be expressed as a surface of revolution.

Sphere

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

x2 + y2 + z2 = r2

Ellipsoid

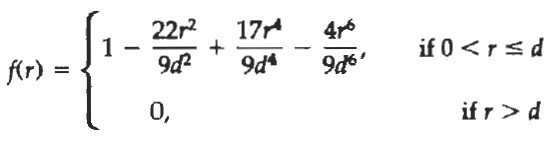
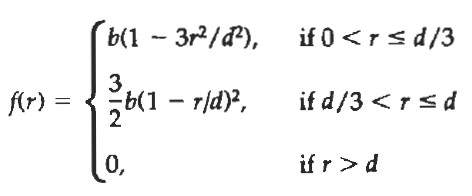
An ellipsoidal surface can be described as an extension of a spherical surface, where the radii in three mutually perpendicular directions can have different values

Torus

A torus is a doughnut-shaped object, as shown in Figure. It can be generated by rotating a circle or other conic about a specified axis. The Cartesian representation for points over the surface of a   
 torus can be written in the form

where r is any given offset value

Unit 3 : 3D Object Representation



BLOBBY OBJECTS

Some objects do not maintain a fixed shape, but change their surface characteristics in certain   
motions or when in proximity to other objects. Examples in this class of objects include   
molecular structures, water droplets and other liquid effects, melting objects, and muscle shapes   
in the human body. These objects can be described as exhibiting "blobbiness" and are often   
simply referred to as blobby objects, since their shapes show a certain degree of fluidity. A   
molecular shape, for example, can be described as spherical in isolation, but this shape changes   
when the molecule approaches another molecule. These characteristics cannot be adequately   
described simply with spherical or elliptical shapes. Similarly, muscle shapes in human arm   
exhibit similar characteristics. In this case, we want to model surface shapes so that the total   
volume remains constant. Several models have been developed for representing blobby objects   
as distribution functions over a region of space. One way to do this is to model objects as   
combinations of Gaussian density functions, or "bumps". A surface function is then defined as:

Where T = Threshold and a and b are used to adjust amount of

blobbness.

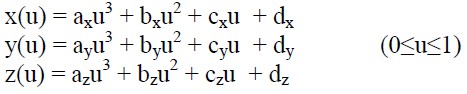
Other methods for generating blobby objects use density functions that fall off to 0 in a finite interval, rather than exponentially. The "metaball" model describes composite objects as combinations of quadratic density functions of the form:

And the "soft object" model uses the function

Parametric Curves

Curves and surfaces can have explicit, implicit, and parametric representations.

Unit 3 : 3D Object Representation



There are multiple ways to represent curves in two dimensions:

i) Explicit: y = f(x), given x, find y.

Example:

The explicit form of a line is y = mx + b.

ii) Implicit: f(x, y) = 0

Example:

The implicit equation for a circle of radius r and center pc = (xc, yc) is   
 (x − xc)2 + (y − yc)2 = r2,

iii) Parametric: P = P0 + u (P1 - P0)

In mathematics, parametric equation is a method of defining a relation using parameters like u in the example. Parametric representations are the most common in computer graphics.

Advantages of parametric forms

 More degrees of freedom   
 Directly transformable   
 Dimension independent   
 No infinite slope problems

 Separates dependent and independent variables  Inherently bounded

 Easy to express in vector and matrix form

 Common form for many curves and surfaces

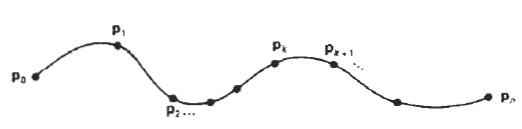
Parametric cubic curves are commonly used in graphics because curves of lower order commonly have too little flexibility, while curves of higher order are usually considered unnecessarily complex and make it easy to introduce undesired wiggles.   
A parametric cubic curve in 3D is defined by:

Here , we will discuss only some curves that are generally used in computer graphics to represent 3D objects like spline curves and Bezier curve ,

Spline Representation

A Spline is a flexible strips used to produce smooth curve through a designated set of points. A   
curve drawn joining this points is spline curve.Mathematically, spline are described as pice-wise   
cubic polynomial functions whose first and second derivatives are continuous across the various   
curve sections. In computer graphics, the term spline curve now refers to any composite curve

Unit 3 : 3D Object Representation



formed with polynomial sections satisfying specified continuity conditions at the boundary of the pieces. A spline surface can be described with two sets of orthogonal spline curves.

There are several different kinds of spline specifications that are used in graphics applications. Each individual specification simply refers to a particular type of polynomial with certain specified boundary conditions. Spline is used in graphics application to design and digitalize drawings for storage in computer and to specify animation path. Typical CAD application for spline includes the design of automobile bodies, aircraft and spacecraft surface etc.

There are three equivalent methods for specifying a particular spline representation:

(1) We can state the set of boundary conditions that are imposed on the spline; or

(2) we can state the matrix that characterizes the spline; or

(3) we can state the set of blending functions (or basis functions) that determine how specified geometric constraints on the curve are combined to calculate positions along the curve path.

CUBIC SPLINE INTERPOLATION METHODS

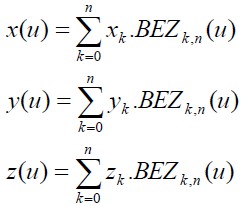
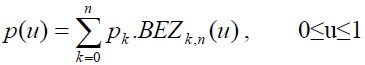
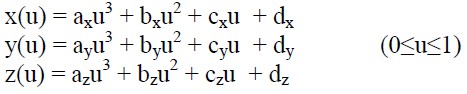
This class of splines is most often used to set up paths for object motions or to provide a representation for an existing object or drawing, but interpolation splines are also used sometimes to design object shapes. Cubic polynomials offer a reasonable compromise between flexibility and speed of computation. Compared to higher-order polynomials, cubic splines require less calculations and memory and they are more stable. Compared to lower-order polynomials, cubic splines are more flexible for modeling arbitrary curve shapes. Given a set of control points, cubic interpolation splines are obtained by fitting the input points with a piecewise cubic polynomial curve that passes through every control point.

Suppose we have n+1 control points specified with co-ordinates. pk = (xk,yk,zk), k = 0,1,2,3…………………..n

A cubic interpolation fit of those points is

We can describe the parametric cubic polynomial that is to be fitted between each pair of control points with the following set of parametric equations.

Unit 3 : 3D Object Representation



Bezier Curves

Bezier curves are used in computer graphics to produce curves which appear reasonably smooth   
at all scales. This spline approximation method was developed by French engineer Pierre Bezier   
for automobile body design. Bezier spline was designed in such a manner that they are very   
useful and convenient for curve and surface design, and are easy to implement Curves are   
trajectories of moving points. We will specify them as functions assigning a location of that   
moving point (in 2D or 3D) to a parameter t, i.e., parametric curves. Curves are useful in   
geometric modeling and they should have a shape which has a clear and intuitive relation to the   
path of the sequence of control points. One family of curves satisfying this requirement are   
Bezier curve.

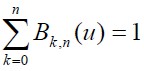
The Bezier curve requires only two end points and other points that control the endpoint tangent   
vector.

Bezier curve is defined by a sequence of N + 1 control points, P0, P1,. . . , Pn. We defined the Bezier curve using the algorithm (invented by DeCasteljeau), based on recursive splitting of the intervals joining the consecutive control points. In general Bezier curve can be fitted to any number of control points. The number of control points to be approximated and their relative position determine the degree of Bezier polynomial. The Bezier curve can be specified with boundary condition, with characterizing matrix or blending functions. But for general blending function specification is most convenient.

The Bezier belending function BEZk,n(u) are the Bernstein polynomial as,

The vector equation (1) represents a set of three parametric equation for individual curve conditions.

Unit 3 : 3D Object Representation



Bezier curve is a polynomial of degree one less than control points i.e. 3 points generate parabola, 4 points a cubic curve and so n.

Properties of Bezier Curve:

1. It always passs through initial and final control points. i.e p(0) = p0 and p(1)=pn.

2. Values of the parametric first derivatives of a Bezier curve at the end points can be calculated from control points as-

p'(0) = -np0 + np1   
p'(1) = -npn-1 + npn

3. The slope at the beginning of the curve is along the line joining the first two points and slope at the end of curve is along the line joining last two points.

4. Parametric second derivative at a Bezier curve at end points are-

p"(0) = n(n-1)[(p2-p1) - (p1-p0)]

p"(1) = n(n-1)[(pn-2-pn-1) - (pn-1-pn)]

Another important property of any Bezier curve is that it lies within the convex hull (convex polygon boundary) of the control points. This follows from the properties of Bezier blending functions: They are all positive and their sum is always 1,

so that any curve position is simply the weighted sum of the control-point positions. The convex-hull property for a Bezier curve ensures that the polynomial will not have erratic oscillations.

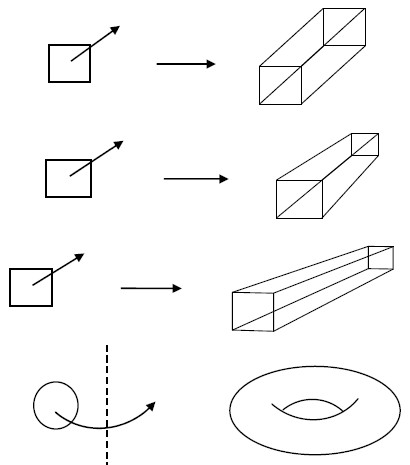
Solid Modeling

Surface representation is the logical evolution using faces (surfaces), edges and vertices. In this sequence of developments, the solid modeling uses topological information in addition to the geometrical information to represent the object unambiguously and completely.

A solid model of an object is a more complete representation than its surface model. It provides   
more topological information in addition to the geometrical information which helps to represent   
the solid unambiguously. Unlike surface representations which contain only geometrical data,   
the solid model uses topological information in addition to the geometrical information to   
represent the object unambiguously and completely. Solid model results in accurate design, helps   
to further the goal of CAD/ CAM, Flexible manufacturing leading to better automation of the   
manufacturing process.

Geometry: The graphical information of dimension, length, angle, area and transformations

Unit 3 : 3D Object Representation



Topology: The invisible information about the connectivity, neighborhood, associatively etc Various methods are described below .

SWEEP REPRESENTATIONS

Sweep representations are useful for constructing three-dimensional objects that possess   
translational, rotational, or other symmetries. We can represent such objects by specifying a two   
dimensional shape and a sweep that moves the shape through a region of space. A set of two-  
dimensional primitives, such as circles and rectangles, can be provided for sweep representations   
as menu options.

Sweep Volume:

Sweeping a 2D area along a trajectory creates a new 3D object

• Translational Sweep: 2D area swept along a linear trajectory normal to the 2D plane

• Tapered Sweep: scale area while sweeping

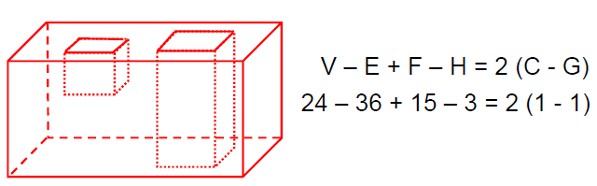
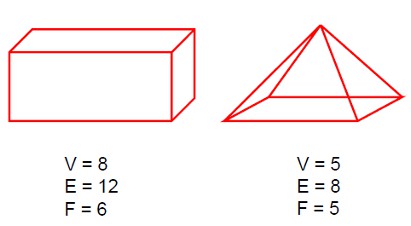
• Slanted Sweep: trajectory is not normal to the 2D plane

• Rotational Sweep: 2D area is rotated about an axis

• General Sweep: object swept along any trajectory and transformed along the sweep

Other methods for obtaining two-dimensional figures include closed spline curve constructions and cross-sectional slices of solid objects.

Unit 3 : 3D Object Representation



Boundary Representations (B-rep)

B-rep describes a solid in terms of its surface boundaries: vertices, edges, and faces. Curved faces can be approximated by polygons or represented by parametric (implicit) surfaces. A closed 2D surface defines a 3D object and at each point on the boundary there is an “in” and an “out” side .Boundary representations can be defined in two ways:

1) Primitive based: A collection of primitives forming the boundary (polygons, for example)

2) Freeform based (splines, parametric surfaces, implicit forms)

Primitive based :

A polyhedron is a solid bounded by a set of polygons. It is constructed from:

- Vertices V

- Edges E

- Faces F

Each edge must connect two vertices and be shared by exactly two faces. At least three edges must meet at each vertex. A simple polyhedron is one that can be deformed into a sphere (contains no holes) and it must satisfy Euler's formula: V-E+F=2

Euler’s formula can be generalized to a polyhedron with holes and multiple components   
 V-E+F-H=2(C-G)

Where:

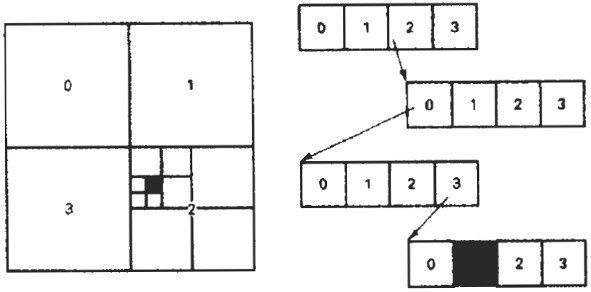
• H is the number of holes in the faces

• C is the number of separate components

• G is the number of pass-through holes (genus if C=1)

• V, E and F are respectively vertices, edges and faces

Unit 3 : 3D Object Representation



Spatial Partitioning Representation

In spatial-partitioning representations, a solid is decomposed into a collection of adjoining, non   
intersecting solids that are more primitive than the original solid. Primitives may vary in type,   
size, position, parameterization, and orientation. Forms of spatial-partitioning representations:

• Cell decomposition

• Spatial-occupancy enumeration

• Octrees

• Binary space-partitioning trees

We will be only explaining Octrees :

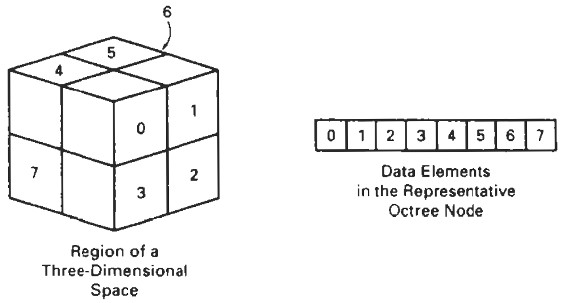
OCTREES

Hierarchical tree structures, called octrees, are used to represent solid objects in some graphics   
systems. Medical imaging and other applications that require displays of object cross sections   
commonly use octree representations. The tree structure is organized so that each node   
corresponds to a region of three-dimensional space. This representation for solids takes   
advantage of spatial coherence to reduce storage requirements for three-dimensional objects. It   
also provides a convenient representation for storing information about object interiors.

The octree encoding procedure for a three-dimensional space is an extension of an encoding scheme for two-dimensional space, called quadtree encoding.Quadtrees are generated by successively dividing a two-dimensional region (usually a square) into quadrants. Each node in the quadtree has four data elements, one for each of the quadrants in the region.

Each node in the quadtree has four data elements, one for each of the quadrants in the region. If   
all pixels within a quadrant have the same color (a homogeneous quadrant), the corresponding   
data element in the node stores that color. In addition, a flag is set in the data element to indicate   
that the quadrant is homogeneous. Otherwise, the quadrant is said to be heterogeneous, and that   
quadrant is itself divided into quadrants (see figure below). The corresponding data element in   
the node now flags the quadrant as heterogeneous and stores the pointer to the next node in the   
quadtree.

Unit 3 : 3D Object Representation



An octree encoding scheme divides regions of three-dimensional space (usually cubes) into   
octants and stores eight data elements in each node of the tree (see figure below). Individual   
elements of a three-dimensional space are called volume elements, or voxels. When all voxels in   
an octant are of the same type, this type value is stored in the corresponding data element of the   
node. Empty regions of space are represented by voxel type "void." Any heterogeneous octant is   
subdivided into octants, and the corresponding data element in the node points to the next node   
in the octree.Voxels in each octant is tested, and octant subdivisions continue until the region of   
space contains only homogeneous octants. Each node in the octree can now have from zero to   
eight immediate descendants.

Algorithms for generating octrees can be structured to accept definitions of objects in any form,   
such as a polygon mesh, curved surface patches, or solid geometry constructions. Using the   
minimum and maximum coordinate values of the object, we can define a box (parallelepiped)   
around the object. This region of three-dimensional space containing the object is then tested,   
octant by octant, to generate the octree representation. Once an octree representation has been   
established for a solid object, various manipulation routines can be applied to the solid. An   
algorithm for performing set operations can be applied to two octree representations for the same   
region of space. The new octree is then formed by either storing the octants where the two   
objects overlap or the region occupied by one object but not the other. Three-dimensional octree   
rotations are accomplished by applying the transformations to the occupied octants. Visible-  
surface identification is carried out by searching the octants from front to back. The first object   
detected is visible, so that information can be transferred to a quadtree representation for display.

BSP TREES

This representation scheme is similar to octree encoding, except we now divide space into two partitions instead of eight at each step. With a binary space-partitioning (BSP) tree, we subdivide a scene into two sections at each step with a plane that can be at any position and orientation. In an octree encoding, the scene is subdivided at each step with three mutually perpendicular planes aligned with the Cartesian coordinate planes.

Unit 3 : 3D Object Representation

For adaptive subdivision of space, BSP trees can provide a more efficient partitioning since we can position and orient the cutting planes to suit the spatial distribution of the objects. This can reduce the depth of the tree representation for a scene, compared to an octree, and thus reduce the time to search the tree. In addition, BSP trees are useful for identifying visible surfaces and for space partitioning in ray-tracing algorithms.

DISPLAYING LIGHT INTENSITIES

Values of intensity calculated by an illumination model must be converted to one of the   
allowable intensity levels for the particular graphics system in use. Some systems are capable of   
displaying several intensity levels, while others are capable of only two levels for each pixel (on   
or off). In the first case, we convert intensities from the lighting model into one of the available   
levels for storage in the frame buffer. For bi-level systems, we can convert intensities into   
halftone patterns.

Assigning Intensity Levels

We first consider how grayscale values on a video monitor can be distributed over the range   
between 0 and 1 so that the distribution corresponds to our perception of equal intensity   
intervals. We perceive relative light intensities the same way that we perceive relative sound   
intensities: on a logarithmic scale. This means that if the ratio of two intensities is the same as   
the ratio of two other intensities, we perceive the difference between each pair of intensities to be   
the same.

As an example, we perceive the difference between intensities 0.20 and 0.22 to be the same as the difference between 0.80 and 0.88. Therefore, to display n + 1 successive intensity levels with equal perceived brightness, the intensity levels on the monitor should be spaced so that the ratio of successive intensities is constant:

I1/ I0 = I2/I1 = ….. = In/In-1 = r

Here, we denote the lowest level that can be displayed on the monitor as I0 and the highest as In. Any intermediate intensity can then be expressed in terms of I0 ,as

Ik = r k I0

We can calculate the value of r, given the values of I0 and n for a particular system,by substituting k = n in the preceding expression. Since In = 1, we have

r= (1/I0)1/n

Thus, the calculation for Ik

Ik= I0(n-k)/n

Unit 3 : 3D Object Representation

Color Models:

A color model is an abstract mathematical model describing the way colors can be represented as tuples of numbers, typically as three or four values or color components. When this model is associated with a precise description of how the components are to be interpreted (viewing conditions, etc.), the resulting set of colors is called color space. This section describes ways in which human color vision can be modeled.

RGB

Media that transmit light (such as television) use additive color mixing with primary colors of red, green, and blue, each of which stimulates one of the three types of the eye's color receptors with as little stimulation as possible of the other two. This is called "RGB" color space. Mixtures of light of these primary colors cover a large part of the human color space and thus produce a large part of human color experiences. This is why color television sets or color computer monitors need only produce mixtures of red, green and blue light. See Additive color. Other primary colors could in principle be used, but with red, green and blue the largest portion of the human color space can be captured. Unfortunately there is no exact consensus as to what loci in the chromaticity diagram the red, green, and blue colors should have, so the same RGB values can give rise to slightly different colors on different screens.

HSV

Recognizing that the geometry of the RGB model is poorly aligned with the color-making   
attributes recognized by human vision, computer graphics researchers developed two alternate   
representations of RGB, HSV and HSL (hue, saturation, value and hue, saturation, lightness), in   
the late 1970s. HSV and HSL improve on the color cube representation of RGB by arranging   
colors of each hue in a radial slice, around a central axis of neutral colors which ranges from   
black at the bottom to white at the top. The fully saturated colors of each hue then lie in a circle,   
a color wheel. HSV models itself on paint mixture, with its saturation and value dimensions   
resembling mixtures of a brightly colored paint with, respectively, white and black. HSL tries to   
resemble more perceptual color models such as NCS or Munsell. It places the fully saturated   
colors in a circle of lightness ½, so that lightness 1 always implies white, and lightness 0 always   
implies black. HSV and HSL are both widely used in computer graphics, particularly as color   
pickers in image editing software. The mathematical transformation from RGB to HSV or HSL   
could be computed in real time, even on computers of the 1970s, and there is an easy-to-  
understand mapping between colors in either of these spaces and their manifestation on a   
physical RGB device.

The HSV (Hue, Saturation, Value) model, also called HSB (Hue, Saturation, Brightness), defines   
a color space commonly used in graphics applications. Hue value ranges from 0 to 360,   
Saturation and Brightness values range from 0 to 100%. The RGB (Red, Green, Blue) is also

Unit 3 : 3D Object Representation



used, primarily in web design. When written, RGB values are commonly specified using three integers between 0 and 255, each representing red, green, and blue intensities.

The RGB model's approach to colors is important because:

* It directly reflects the physical properties of “Truecolor” displays
* As of 2011, most graphics tools support it, even if they prefer another color model.
* It is the only means of specifying a specific color in the CSS2 standard for web pages

In the model, a color is described by specifying the intensity levels of the colors red, green, and   
blue. The typical range of intensity values for each color, 0 - 255, is based on taking a binary   
number with 32 bits and breaking it up into four bytes of 8 bits each. 8 bits can hold a value from

0 to 255. The fourth byte is used to specify the "alpha", or the opacity, of the color. Opacity comes into play when layers with different colors are stacked. If the color in the top layer is less than fully opaque (alpha < 255), the color from underlying layers "shows through". In the RGB model, hues are represented by specifying one color as full intensity (255), a second color with a variable intensity, and the third color with no intensity (0).

HSV

The HSV, or HSB, model describes colors in terms of hue, saturation, and value (brightness). Note that the range of values for each attribute is arbitrarily defined by various tools or standards. Be sure to determine the value ranges before attempting to interpret a value. Hue corresponds directly to the concept of hue in the Color Basics section.

The advantages of using hue are

* The angular relationship between tones around the color circle is easily identified
* Shades, tints and tones can be generated easily without affecting the hue saturation corresponds directly to the concept of tint in the color Basics section, except that full saturation produces no tints, while zero saturation produces white, a shade of gray, or black. Value corresponds directly to the concept of intensity in the color Basics section.
* Pure Colors are produced by specifying a hue with full saturation and value
* Shades are produced by specifying a hue with full saturation and less than full value
* Tints are produced by specifying hue with less than full saturation and full value
* Tones are produced by specifying a hue and both less than full saturation and value
* White is produced by specifying zero saturation and full value, regardless of hue
* Black is produced by specifying zero value, regardless of hue or saturation
* Shades of gray are produced by specifying zero saturation and between zero and full value.