Tribhuvan University
Institute of Science and Technology
Bachelor of Science in
Computer Science and Information Technology

3RD Semester

CSC209: Computer Graphics

-RAJESH KUMAR BAJGAIN

U5: 3D Objects Representation 7Hrs

5.1 Representing Surfaces: Boundary and Space partitioning

- 5.1.1 Polygon Surface: Polygon tables , Surface normal and Spatial orientation of surfaces, Plane equations, Polygon meshes
- 5.1.2 Wireframe Representation
- 5.1.3 Blobby Objects

5.2 Representing Curves: Parametric Cubic Curves, Spline Representation, Cubic spline interpolation, Hermite Curves, Bezier and B-spline Curve and surface

5.3 Quadric Surface: Sphere and Ellipsoid

3D Object Representation

Different kinds of objects like trees, flowers, clouds, rocks, water etc. cannot be describe with only one method

There is no single method that we can use to describe objects that will include all the characteristics of these different materials.

Polygon and quadratic surface provide precise description for simple Euclidean objects such as polyhedron and ellipsoids.

3D Object Representation

Spline surfaces are useful for designing aircraft wings, gears and engineering objects.

Procedural methods and particle system allows us to give accurate representation of clouds, clumps of grass and other natural objects.

Octree encodings are used to represent internal features of the objects, like medical CT images.

3D Object Representation

3D object representation is divided into two categories:

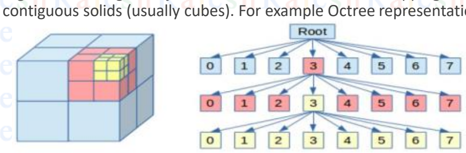
1. Boundary representations (B-reps):

- Describe a three-dimensional object as a set of surfaces that separate the object interior from the environment.
- B-reps describe the objects exterior as a set of surfaces that encloses the objects interior. Examples: Polygon surfaces and spline patches.
- B-reps for single polyhedron satisfy Euler's formula: V-E+F=2

3D Object Representation

2. Space Partitioning Representation:

- Used to describe interior properties, by partitioning the spatial region containing an object into a set of small, non overlapping, contiguous solids (usually cubes). For example Octree representation.



Boundary Representation:
1. Polygon Surface

- Most common representation for 3D graphics object.
- A 3D object is represented by a set of surfaces that enclose the object interior.
- It simplifies and speeds up the surface rendering and display of the object since all surfaces are described with linear equations
- For this reason, polygon descriptions are often referred to as "**standard graphics objects**"
- Polygon surface can be represented by:

a) Polygon Table

- A polygon surface is specified with a set of vertex co-ordinates and associated attributes. Polygon data tables can be organized into two groups: geometrical and attribute tables.
- **Geometric tables**: It contain vertex coordinates and parameters to identify the spatial orientation of polygon surfaces
- **Attribute table**: It gives attribute information for an object (Degree of transparency, surface reflectivity, texture characteristics, etc.).

a) Polygon Table

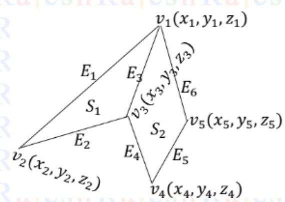
- Geometric data consists of three tables: a vertex table, an edge table, and a surface table.

• **Vertex table**: It stores co-ordinate values for each vertex of the object.

• **Edge Table**: The edge table contains pointers back into the vertex table to identify the vertices for each polygon edge.

• **Surface table**: And the polygon table contains pointers back into the edge table to identify the edges for each polygon surfaces.

a) Polygon Table



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

$E_1: v_1, v_2$
$E_2: v_2, v_3$
$E_3: v_3, v_1$
$E_4: v_3, v_4$
$E_5: v_4, v_5$
$E_6: v_5, v_1$

Surface table

$S_1: E_1, E_2, E_3$
$S_2: E_3, E_4, E_5, E_6$

Vertices

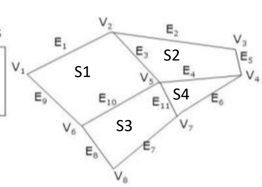
$V_1:(x_1,y_1,z_1)$
$V_2:(x_2,y_2,z_2)$
$V_3:(x_3,y_3,z_3)$
$V_4:(x_4,y_4,z_4)$
$V_5:(x_5,y_5,z_5)$
$V_6:(x_6,y_6,z_6)$
$V_7:(x_7,y_7,z_7)$
$V_8:(x_8,y_8,z_8)$

Edges

$E_1: V_1,V_2$
$E_2: V_2,V_3$
$E_3: V_2,V_5$
$E_4: V_4,V_5$
$E_5: V_3,V_4$
$E_6: V_4,V_7$
$E_7: V_7,V_8$
$E_8: V_6,V_8$
$E_9: V_1,V_6$
$E_{10}: V_5,V_6$
$E_{11}: V_5,V_7$

Polygons

$S_1: E_1,E_3,E_{10},E_9$
$S_2: E_2,E_5,E_4,E_3$
$S_3: E_{10},E_{11},E_7,E_8$
$S_4: E_4,E_6,E_{11}$



Forward pointers:
i.e. to access
adjacent surfaces
edges

$V_1: E_1,E_9$
$V_2: E_1,E_2,E_3$
$V_3: E_2,E_5$
$V_4: E_4,E_5,E_6$
$V_5: E_3,E_4E_{10},E_{11}$
$V_6: E_8,E_9,E_{10}$
$V_7: E_6,E_7,E_{11}$
$V_8: E_7,E_8$

$E_1: S_1$
$E_2: S_2$
$E_3: S_1,S_2$
$E_4: S_2,S_4$
$E_5: S_2$ $E_6: S_4$
$E_7: S_3$ $E_8: S_3$
$E_9: S_1$ $E_{10}: S_1,S_3$
$E_{11}: S_3,S_4$

a) Polygon Table

Attribute tables:

Attribute information for an object includes parameters specifying the degree of transparency of the object and its surface reflectivity and texture characteristics.

The above three table also include the polygon attribute according to their pointer information.

2. Plane equation

It this method polygon surface is represented by the equation of plane in the coordinate system.

The 3D object is represented through the set of equations for the plane surface, represented as:

$$Ax + By + Cz + D = 0$$

Where x, y, z is any point on the plane and A, B, C & D are coefficient of plane equation and represents the spatial orientation of the polygon surface in space coordinate system.

Hence, the value of coefficient must be known to represent the 3D object.

Plane equation

The value of A, B, C & D can be obtained by solving a set of three plane equation using coordinate of three non-collinear point on plane.

Let us assume that three vertices of plane are $(x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)$. Then,

$$Ax_1 + By_1 + Cz_1 + D = 0$$
$$Ax_2 + By_2 + Cz_2 + D = 0$$
$$Ax_3 + By_3 + Cz_3 + D = 0$$

By Cramer's rule

Plane equation

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

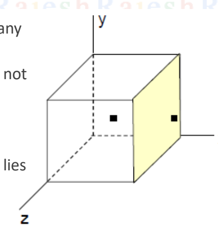
Expanding the determinant we can write that,

$$A = y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2)$$
$$B = z_1(x_2 - x_3) + z_2(x_3 - x_1) + z_3(x_1 - x_2)$$
$$C = x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2)$$
$$D = -x_1(y_2 z_3 - y_3 z_2) - x_2(y_3 z_1 - y_1 z_3) - x_3(y_1 z_2 - y_2 z_1)$$

Plane equation

Inside outside tests of the surface for any points (x, y, z)

- If Ax + By + Cz + D ≠ 0, then (x, y, z) is not on the plane.
- If Ax + By + Cz + D < 0, then (x, y, z) is inside the plane i. e. invisible side
- If Ax + By + Cz + D > 0, then (x, y, z) is lies outside the surface.

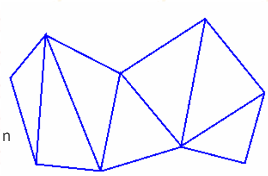


Polygon meshes

A polygon mesh is a collection of vertices, edges and faces that defines the shape of a polyhedral object in 3D computer graphics

Triangular Mesh :

It produces n – 2 connected triangles, given the coordinates for n vertices.

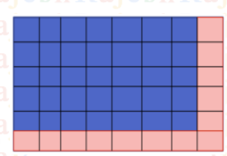


Polygon meshes

Quadrilateral Mesh

Another similar functions the quadrilateral mesh that generates a mesh of (n-1)(m-1) quadrilaterals, given the coordinates for an n by m array of vertices

If the surface of 3D object is planner then it is comfortable to represent surface with meshes



6 by 8 vertices array , 35 element quadrilateral mesh

Polygon meshes

- An edge connects two vertices and a polygon is a closed sequence of edges.
- An edge is shared by at most two polygons and a vertex is shared by at least two edges.
- This method can be used to represent a broad class of solids/surfaces in graphics.
- The polygon mesh can be represented by three ways-
  - Explicit representation
  - Pointers to a vertex list
  - Pointers to an edge list

Polygon meshes

Explicit representation

Each polygon is represented by a list of vertex co-ordinates.

$PP = ((x_1, y_1, z_1), (x_2, y_2, z_2), \ldots \ldots \ldots, (x_n, y_n, z_n))$

Pointers to a vertex list

Each vertex is stored just once, in vertex list

$VV = (v_1, v_2, \ldots \ldots \ldots, v_n)$

E.g. A polygon made up of vertices 3, 5, 7, 10 in vertex list be represented as $P_1 = \{3,5,7,10\}$

Polygon meshes

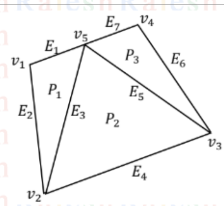
Pointers to a vertex list

Representing polygon mesh with each polygon as vertex list.

$PP_1 = \{v_1, v_2, v_5\}$

$PP_2 = \{v_2, v_3, v_5\}$

$PP_3 = \{v_3, v_4, v_5\}$



Polygon meshes

Pointers to an edge list

we have vertex list V, represent the polygon as a list of pointers to an edge list.

Each edge in edge list points to the two vertices in the vertex list.

Also to one or two polygon, the edge belongs.

Hence, we describe polygon as $P = (E_1, E_2, \ldots \ldots \ldots, E_n)$ and an edge as $E = (v_1, v_2, P_1, P_2)$

Here if edge belongs to only one polygon, either then $P1$ or $P2$ is null.

Polygon meshes

Pointers to an edge list

For the mesh given above,

$V = \{v1, v2, v3, v4, v5\} = \{(x1, y1, z1), \ldots \ldots \ldots, (x5, y5, z5)\}$

$E1 = (v1, v5, P1, N)$ $E6 = (v3, v4, P3, N)$

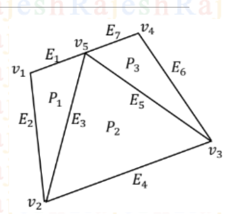
$E2 = (v1, v2, P1, N)$ $E7 = (v4, v5, P3, N)$

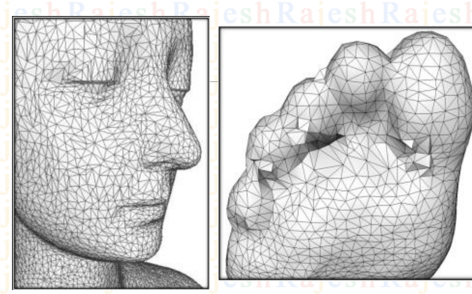
$E3 = (v2, v5, P1, P2)$ $P1 = (E1, E2, E3)$

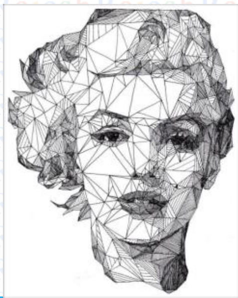
$E4 = (v2, v3, P2, N)$ $P2 = (E3, E4, E5)$

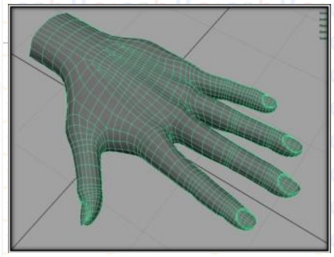
$E5 = (v3, v5, P1, P3)$ $P3 = (E5, E6, E7)$

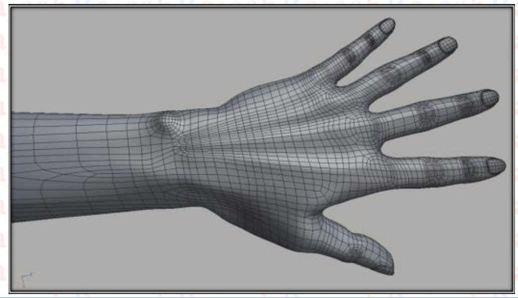
Here, N represents Null.

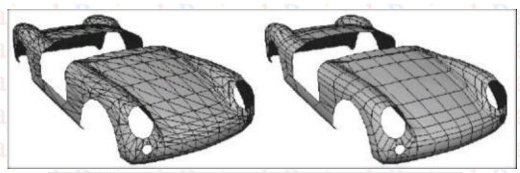


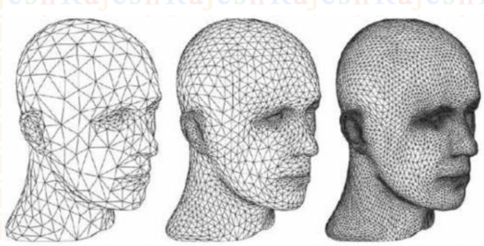






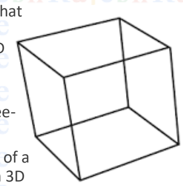






Wireframe

- A wireframe is a three-dimensional model that only includes vertices and lines. It does not contain surfaces, textures, or lighting like a 3D mesh.
- Instead, a wireframe model is a 3D image comprised of only "wires" that represent three-dimensional shapes.
- A wire-frame model is a visual presentation of a 3- dimensional (3D) or physical object used in 3D computer graphics.



Wireframe

- often used as the starting point in 3D modeling since they create a "frame" for 3D structures.

For example, a 3D graphic designer can create a model from scratch by simply defining points (vertices) and connecting them with lines (paths).

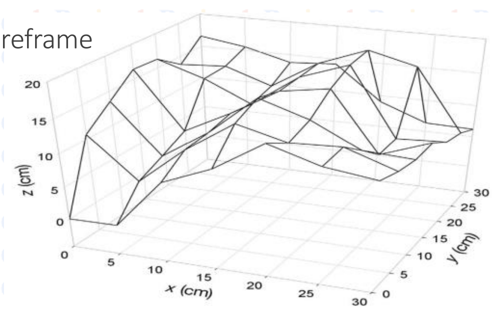
- Once the shape is created, surfaces or textures can be added to make the model appear more realistic.

Wireframe

- The lines within a wireframe connect to create polygons, such as triangles and rectangles, that together represent three dimensional shapes.
- The result may be as simple as a cube or as complex as a three-dimensional scene with people and objects.

The number of polygons within a wireframe is typically a good indicator of how detailed the 3D model is.

Wireframe



Blobby Objects

By a blobby object means a non-rigid object; i.e. things; like cloth, rubber, liquids, water droplets, etc.

- These objects tend to exhibit a degree of fluidity.
- Several models have been developed to handle these kind of objects.
- One technique is to use a combination of **Gaussian density functions** (Gaussian bumps).

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

Blobby Objects

Where, $r_k^2 = x_k^2 + y_k^2 + z_k^2$

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

T= some specified threshold

a, b= parameters used to adjust the amount of blobbiness for individual object.

- Another technique called the **meta-ball technique** is to describe the object as being made of density functions much like balls.
- The advantage here is that the density function falls of in a finite interval.

Blobby Objects

Advantages

- Can represent organic, blobby or liquid line structures.
- Suitable for modeling natural phenomena like water, human body.
- Surface properties can be easily derived from mathematical equations.

Disadvantages

- Requires expensive computation
- Requires special rendering engine
- Not supported by most graphics hardware

Surface Normal and Spatial Orientation of surfaces

A normal is the technical term used in CG (and Geometry) to describe the orientation of a surface of a geometric object at a point on that surface.

Technically, the surface normal to a surface at a point P can be seen as the vector perpendicular to a plane tangent to the surface at P.

At the same time, the direction of this vector determines the orientation of the surface.

In the case of polygons, this direction is usually determined by the right-hand-rule.

Surface Normal and Spatial Orientation of surfaces

In CG, manipulation of normal vector are often used as a way to simulate geometrical details on otherwise planer surfaces.

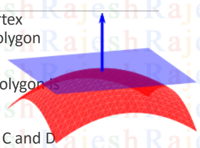
In this case, a function will determine small aberrations of the true direction of the normal vector on every point of the surface, in order special create highlight or shadow effects.

Normal plays an important role in shading where they are used to compute the brightness of the objects.

If, e.g., the vector is slightly shifted in accordance to a sinus function, then the surface will appear in a rendered image, as if it were made of corrugated material (except for the edges).

Surface Normal and Spatial Orientation of surfaces

<u>**Spatial Orientation**</u> of a polygon face is the vertex coordinates values and the equations of the polygon surfaces.



The general equation of a plane containing a polygon is

$Ax + By + Cz + D = 0$

Where (x, y, z) is any point on the plane; A, B, C and D are plane parameters giving the spatial properties of the plane and A, B, C and D can be calculated by three non-collinear points in the plane, selected in a strictly counterclockwise order, viewing the surface along a front-to-back direction.

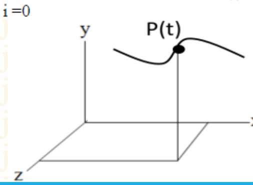
Representing Curves: Parametric Cubic Curves

A parametric cubic curve is defined as $P(t) = \sum_{i=0}^{3} a_i t^i \qquad 0 <= t <= 1$ ------ (i)

Where, P(t) is a point on the curve

a= algebraic coefficients

t= tangent Vector



Parametric Cubic Curves

Expanding equation (i) yield

$P(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0$ ------------------------------ (ii)

This equation is separated into three components of P (t)

$x(t) = a_{3x} t^3 + a_{2x} t^2 + a_{1x} t + a_{0x}$

$y(t) = a_{3y} t^3 + a_{2y} t^2 + a_{1y} t + a_{0y}$

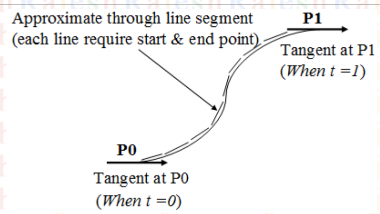
$z(t) = a_{3z} t^3 + a_{2z} t^2 + a_{1z} t + a_{0z}$ ---------------------(iii)

- To be able to solve (iii) the **twelve unknown** coefficients **a_{ij}** (algebraic coefficients) must be specified

Parametric Cubic Curves

- From the known end point coordinates of each segment, six of the twelve needed equations are obtained.
- The other six are found by using tangent vectors at the two ends of each segment
- The direction of the tangent vectors establishes the slopes(direction cosines) of the curve at the end point

Parametric Cubic Curves



Parametric Cubic Curves

- This procedure for defining a cubic curve using end points and tangent vector is one form of Hermite interpolation
- Each cubic curve segment is parameterized from 0 to 1 so that known end points correspond to the limit values of the parametric variable t, that is P(0) and P(1)
- Substituting t = 0 and t = 1 the relationship between two end point vectors and the algebraic coefficients are found

$P(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0$

$P(0) = a_0$ $P(1) = a_3 + a_2 + a_1 + a_0$ ---------------- (IV)

Parametric Cubic Curves

- To find the tangent vectors equation (ii) must be differentiated with respect to t

$P(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0$

$P'(t) = 3 a_3 t^2 + 2 a_2 t + a_1$

- The tangent vectors at the two end points are found by substituting t = 0 and t = 1 in this equation

$P'(0) = a_1$ $P'(1) = 3 a_3 + 2 a_2 + a_1$ ------------------ (V)

Parametric Cubic Curve

- The algebraic coefficients 'a_i' in equation (ii) can now be written explicitly in terms of boundary conditions – endpoints and tangent vectors are

$a_0 = P(0)$ $a_1 = P'(0)$

$a_2 = -3 P(0) - 3 P(1) - 2 P'(0) - P'(1)$

$a_3 = 2 P(0) - 2 P(1) + P'(0) + P'(1)$

(Note: - The value of a_2 & a_3 can be determined by solving the equation IV & V)

Parametric Cubic Curve

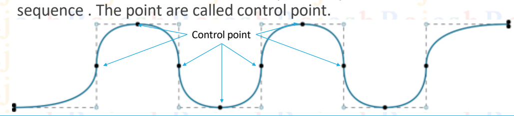
- Substituting these values of 'a_i' in equation (ii) and rearranging the terms yields

$P(t) = (2t^3 - 3t^2 + 1) P(0) + (-2t^3 + 3t^2) P(1) + (t^3 - 2t^2 + t) P'(0) + (t^3 - t^2) P'(1)$

- The values of P(0), P(1), P'(0), P'(1) are called geometric coefficients and represent the known vector quantities in the above equation
- The polynomial coefficients of these vector quantities are commonly known as blending functions By varying parameter t in these blending function from 0 to 1 several points on curve segments can be found

Representing Curves: Spline Representation

- A spline curve is a mathematical representation for which it is easy to build an interface that will allow a user to design and control the shape of complex curves and shapes
- The general approach is that the user enters a sequence of points and a curve is constructed whose shape closely follows this sequence . The point are called control point.



Spline Representation

- A curve that actually passes through each control point is called interpolating curve
- A curve that passes near to the control point but not necessarily through them is called an approximating curve.

Spline Representation

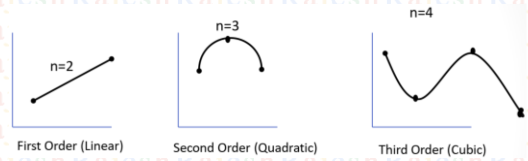


Fig : Interpolating

Spline Specifications:

- Spline curve also used for design of automobile bodies, spacecraft, specification of animation path, home appliance etc.

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

a) Boundary condition

b) Characterizing matrix

c) Blending Function

Spline Specifications:

Boundary condition:

We can state the set of boundary conditions that are imposed on the spline.

$x(u) = a_x u^3 + b_x u^2 + c_x u + d_x \qquad 0 \le u \le 1$

Boundary condition for this curve can be set for x(0), x(1), x'(0) & x'(1). These four conditions are sufficient to determine the values of four coefficient a_x, b_x, c_x & d_x.

Spline Specifications:

Characterizing matrix:

We can state the matrix that characterizes the spline.

From the boundary condition, the characterizing matrix for spline is:

$$x(u) = [u^3\ u^2\ u\ 1]\begin{bmatrix} a_x \\ b_x \\ c_x \\ d_x \end{bmatrix}$$

$$= U.C$$

Spline Specifications:

Blending Function:

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.

$xx(u) = \sum_{k=0}^{3} g_k \cdot BFk(u)$

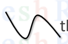
gg_k = Geometric constrain parameter

$BBF_k(u)$=Polynomial blending function

Cubic spline Interpolation

- The three degree polynomial known as cubic polynomial is typically used for constructing smooth curve in computer graphics because of following reasons:

a. It is lowest degree polynomial that can support an inflection (a point at which curve crosses its tangent i.e. curve changes from concave to convex).

b. The curves are smooth like  this and not jumpy like this 

Cubic spline Interpolation

- It is used to set up path for object motions or to provide a representation for an existing object or drawings.

- Compared to higher-order polynomials, cubic splines requires less calculation and memory and they are more stable.

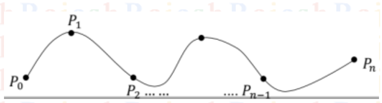
Compared to lower-order polynomials, cubic splines are more flexible for modeling arbitrary curve shapes.

- Cubic interpolation spline is obtained by fitting the input points with a piecewise cubic polynomial curve that passes through every control points.

Cubic spline Interpolation

Suppose we have n+1 control points having co-ordinates

$P_k = (x_k, y_k, z_k)$ $K = 0, 1, 2, 3, \ldots \ldots \ldots \ldots, n$



A parametric cubic polynomial that is to be fitted between each pair of control points have following equations:

Cubic spline Interpolation

$$x(u) = a_x u^3 + b_x u^2 + c_x u + d_x$$
$$y(u) = a_y u^3 + b_y u^2 + c_y u + d_y \qquad (0 \le u \le 1)$$
$$z(u) = a_z u^3 + b_z u^2 + c_z u + d_z$$

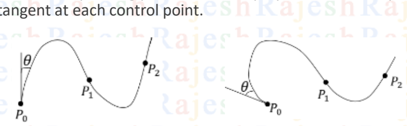
We need to determine the values of the four coefficients a, b, c, and d in the polynomial representation for each of the n curve section.

We do this by setting enough boundary conditions at the "joints" between curve sections we can obtain numerical values for all the coefficients.

- Cubic splines are more flexible for modeling arbitrary curve shapes.

Hermite Interpolation (Hermite curve)

It is an interpolating piecewise cubic polynomial with a specified tangent at each control point.



If we change the control point at P_0, then the curve will also change, so that angle θ between P_0 and tangent at P_0 will remain constant.

Hermite curve

- It has local control over the curve i.e. each curve section depend on its end point only.

- The vector equivalence of Hermite curve is

$$P(u) = au^3 + bu^2 + cu + d \;\ldots\ldots\ldots\ldots\ldots\ldots \text{(i)}$$

Where, x component of $P(u)$ is

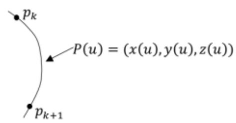
$$x(u) = a_x u^3 + b_x u^2 + c_x u + d_x$$

Similarly, y and z component

$$y(u) = a_y u^3 + b_y u^2 + c_y u + d_y$$
$$z(u) = a_z u^3 + b_z u^2 + c_z u + d_z$$

Let $P(u)$ denotes the parametric cubic point function for the curve section between control point p_k & p_{k+1}.

Hermite curve



At p_k, u=0

$\therefore P(0) = p_k$

At p_{k+1}, u=1

$\therefore P(1) = p_{k+1}$

Also let Dp_k & Dp_{k+1} denote the slope at p_k & p_{k+1}.

$\therefore P'(0) = Dp_k$

$PP'(1) = Dp_{k+1}$

Hermite curve

Hence boundary condition for Hermite curve

$PP(0) = p_k$

$PP(1) = p_{k+1}$ .... (ii)

$PP'(0) = Dp_k$

$PP'(1) = Dp_{k+1}$

Hermite curve

Matrix equivalent of eq. (i) $P(u) = au^3 + bu^2 + cu + d$ is

$$PP(u) = [u^3\ u^2\ u\ 1]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \text{ .... (iii)}$$

Similarly derivative of point function can be represented as,

$$PP'(u) = [3u^2\ 2u\ 1\ 0]\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

Hermite curve

In matrix form, the Hermite boundary condition from eq. (ii) can be represented as

$$\begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \text{ .... (iv)}$$

Solving eq. (iv) for polynomial coefficient, we have

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1}\begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix} = \begin{bmatrix} -2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix}$$

Hermite curve

$$= M_H \cdot \begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix}$$

Where, M_H is the Hermite matrix.

Hence eq. (iii) can be represented as

$$PP(u) = [u^3\ u^2\ u\ 1] \cdot M_H \cdot \begin{bmatrix} p_k \\ p_{k+1} \\ Dp_k \\ Dp_{k+1} \end{bmatrix} \text{ .... (v)}$$

Hermite curve

Expanding (v)

$PP(u) = p_k(2u^3 - 3u^2 + 1) + p_{k+1}(-2u^3 + 3u^2) + Dp_k(u^3 - 2u^2 + u) + Dp_{k+1}(u^3 - u^2)$

In terms of Hermite blending function, 'H', the Hermite curve can be represented as:

$PP(u) = p_k H_0(u) + p_{k+1} H_1(u) + Dp_k H_2(u) + Dp_{k+1} H_3(u)$

Bezier Curve

Bezier curve is developed by the French engineer Pierre Bezier for the design of Renault automobile bodies.

- It is an approximating spline widely used in various CAD system.

- Bezier curve is generated under the control of points known as control points.

Bezier Curve

General Bezier curve for (n+1) control point, denoted as $p_k = (x_k, y_k, z_k)$ with 'k' varying from 0 to n is given by

$PP(u) = \sum_{k=0}^{n} p_k BEZ_{k,n}(u)$, $0 \le u \le 1$ ……… (i)

Where, $P(u)$ is a point on Bezier Curve.

pp_k is a control point.

$BEZ_{k,n}(u)$ is a Bezier blending function also known as Bernstein Polynomial.

Bezier blending function is defined as

$BBEZ_{k,n}(u) = C(n, k)u^k(1-u)^{n-k}$

Bezier Curve

Bezier blending function is defined as

$BBEZ_{k,n}(u) = C(n, k)u^k(1-u)^{n-k}$

Where, $C(n, k) = \frac{n!}{k!(n-k)!}$

Individual x, y, z coordinates an a Bezier curve is given by,

$x(u) = \sum_{k=0}^{n} x_k BEZ_{k,n}(u)$

$y(u) = \sum_{k=0}^{n} y_k BEZ_{k,n}(u)$

$z(u) = \sum_{k=0}^{n} z_k BEZ_{k,n}(u)$

Bezier Curve

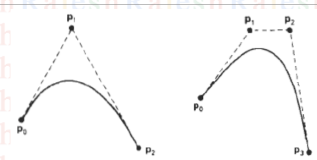


Fig: Bezier curve generated by three and four control point.

Properties of Bezier curve:

a) The basis functions are real.

b) The Bezier curve always passes through first and last control point i.e. $p(0) = p_0$ & $p(1) = p_n$.

c) The degree of polynomial representing Bezier curve is one less than the number of control points.

d) The Bezier curve always follows convex hull formed by control points.

Properties of Bezier curve:

e) The Bezier curve always lies inside the polygon formed by control points.

f) Bezier blending functions are positive and sum is equal to 1.

$\sum_{k=0}^{n} BEZ_{k,n}(u) = 1$

g) The direction of the tangent vector at the end points is same like vector determined by first and last segment.

Cubic Bezier Curve

- It is a Bezier curve generated by four control points.

- General equation for cubic Bezier curve is

$PP(u) = \sum_{k=0}^{3} p_k BEZ_{k,3}(u), \quad 0 \le u \le 1 \quad \ldots\ldots\ldots (i)$

$PP(u) = p_0 BEZ_{0,3}(u) + p_1 BEZ_{1,3}(u) + p_2 BEZ_{2,3}(u) + p_3 BEZ_{3,3}(u)$

Where, $BEZ_{0,3}(u) = C(3,0)u^0(1-u)^{3-0} = \frac{3!}{0!\,(3-0)!} \times (1-u)^3 = (1-u)^3$

Similarly,

$BBEZ_{1,3}(u) = 3u(1-u)^2 \qquad BEZ_{2,3}(u) = 3u^2(1-u) \qquad BEZ_{3,3}(u) = u^3$

$\therefore P(u) = p_0(1-u)^3 + p_1 3u(1-u)^2 + p_2 3u^2(1-u) + p_3 u^3$

Cubic Bezier Curve

In matrix form,

$$PP(u) = [u^3\ u^2\ u\ 1] \cdot M_{BEZ} \cdot \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

Where,

$$MM_{BEZ} = Bezier\ matrix = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Bezier surfaces

- Generalizations of Bezier curves to higher dimensions are called Bezier surfaces.

- The parametric vector function for the Bezier surface is formed as the Cartesian product of Bezier blending function:

$PP(u, v) = \sum_{j=0}^{m} \sum_{k=0}^{n} p_{j,\ kBEZj,\ m}(v) BEZk_{,\ n}(u)$

With $p_{j,k}$ specifying the location of the (m+1) by (n+1) control points.

- Bezier surfaces have the same properties as Bezier curves, and they provide a convenient method for interactive design applications.

Construct Bezier curve for control points (4, 2), (8, 8) and (16, 4).

Construct the Bezier curve of order 3 with 4 vertices of the control polygon $\boldsymbol{p_0(0, 0)}$, $\boldsymbol{p_1(1, 2)}$, $\boldsymbol{p_2(3, 2)}$& $\boldsymbol{p_3(2, 0)}$. Generate at least 5 points on the curve.

B-spline Curve

B-spline curve is a set of piecewise polynomial segments that passes close to a set of control points.

It has two advantage over Bezier curve:

a) The degree of B-spline polynomial can be set independently of the number of control points.

b) It allows local control over the shape of a spline curve.

General equation of B-spline curve is given by

$PP(u) = \sum_{k=0}^{n} p_k B_{k,\ d}(u), \qquad 0 \le u \le$ n+d, 2 ≤ d ≤ n+1

B-spline Curve

$PP(u) = \sum_{k=0}^{n} p_k B_{k,d}(u)$, $0 \le u \le n+d,\ 2 \le d \le n+1$

Where, p_k is a set of (n+1) control points.

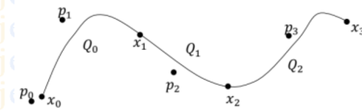
$BB_{k,d}(u)$ is the B-spline blending function.

Blending function for B-spline curves are defined by

$$B_{k,1}(u) = \begin{cases} 1, & if\ uk \le u \le u_{k+1} \\ 0, & Otherwise \end{cases}$$

$$Bk,d(u) = \frac{u-uk}{u_{k+d-1}-uk} B_{k,d-1}(u) + \frac{u_{k+d}-u}{u_{k+d}-u_{k+1}} B_{k+1,d-1}(u)$$

B-spline Curve



$pp_0, p_1, p_2, p_3 \rightarrow$ Control point

$xx_0, x_1, x_2, x_3 \rightarrow$ Knot values

$QQ_0, Q_1, Q_2 \rightarrow$ Curve segment

- The knots produce a vector that defines the domain of the curve.

Properties of B-Spline curve

a) The polynomial curve has $d - 1$ degree.

b) For (n+1) control points, the curve is described with (n+1) blending function.

c) Each blending function $B_{k,d}$ is defined over 'd' sub-interval of the total range of 'u', starting at knot value u_k.

d) The sum of B-spline basis functions for any parameter value is 1

$\sum_{k=0}^{n} B_{k,d}(u) = 1$

Properties of B-Spline curve

e) Each basis function is positive or zero for all parameter value.

f) The range of parameter 'u' is divided into (n+d) sub interval by (n+d+1) values specified in knot vector.

g) Each section of spline curve is influenced by 'd' control point.

h) Any one control point can affect the shape of at most 'd' curve sections.

Properties of B-Spline curve

i) The maximum order of curve is equal to the number of vertices of defining polygon.

j) The curve generally follows the shape of defining polygon.

k) The degree of B-spline polynomial is independent on the number of vertices of defining polygon.

B-spline

On the basis of the knot points and interval length of segment there are two types of spline

→**Periodic B-spline:** Knot points are equi-space to each other and splines are generated through the set of the equi-interval segments then such splines are called periodic Bsplines.

→**Aperiodic B-spline:** If knot points are not equi-space to each other and splines are not generated through the set of the equi interval segments then such splines are called aperiodic B-splines.

Knot vector

There are three general classifications for knot vectors:

- uniform,
- open uniform and
- non-uniform.

Uniform, periodic B-splines:

When the spacing between knot values is constant, the resulting curve is a uniform B-spline. For e.g. {0, 1, 2, 3, 4, 5}

- Uniform B-splines have periodic blending function.

That is, for given value of ‘n’ and ‘d’, all blending functions have the same shape

- Periodic splines are particularly useful for generating certain closed curves.

Open uniform B-splines:

For open B-splines, the knot spacing is uniform expect at the ends where knot values are repeated ‘d’ times. For e.g.

{0, 0, 1, 2, 3, 3} for d=2, and n=3

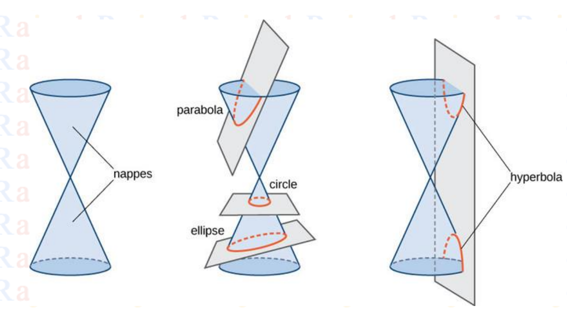
Non-uniform B-splines:

For non-uniform B-splines, we can choose multiple internal knot values and unequal spacing between the knot values. For e.g.

{0, 1, 2, 3, 3, 4}

{0, 0.2, 0.6, 0.9, 1.0}

- Non-uniform B-splines provide increased flexibility in controlling a curve shape.



Quadric surfaces

If a surface is the graph in three-space of an equation of second degree, it is called a quadric surface. Cross section of quadric surface are conics.

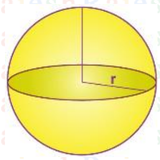
Quadric Surface is one of the frequently used 3D objects surface representation.

The quadric surface can be represented by a second degree polynomial. This includes:

Quadric surfaces

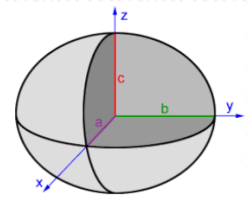
1. Sphere: For the set of surfaces points (x,y,z) the spherical surface is represented as:

$x^2 + y^2 + z^2 = r^2$, with radius r and centered at co-ordinate origin.



Quadric surfaces

2. Ellipsoid: $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$, where (x,y,z) is the surface points and a,b,c are the radii on X,Y,Z directions respectively



Quadric surfaces

3. Elliptic parboiled: $\frac{x^2}{a^2} + \frac{y^2}{b^2} = z$

4. Hyperbolic parboiled: $\frac{x^2}{a^2} - \frac{y^2}{b^2} = z$

5. Elliptic cone: $\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{z^2}{c^2} = 0$

6. Hyperboloid of one sheet: $\frac{x^2}{a^2} + \frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$

7. Hyperboloid of two sheet: $\frac{x^2}{a^2} - \frac{y^2}{b^2} - \frac{z^2}{c^2} = 1$

Note: Check these equations using GeoGebra 3D Calculator:- www.geogebra.org/3d

END OF UNIT