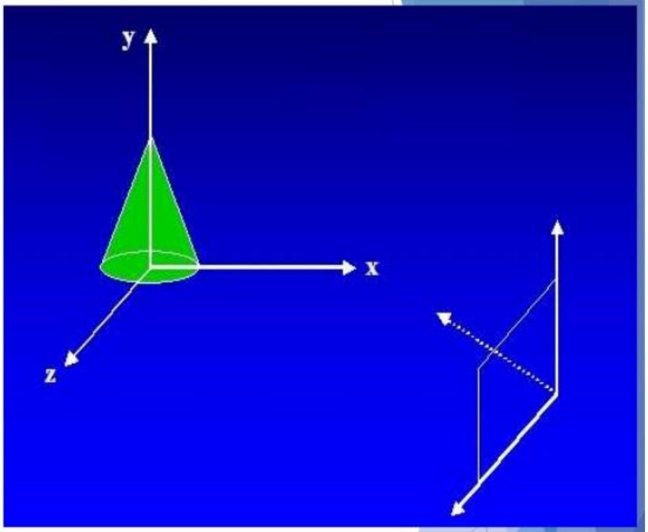
3D Display Methods

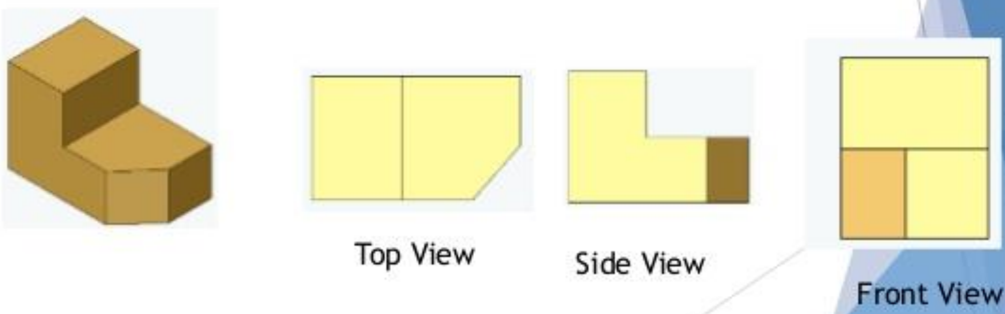
- 3D graphics deals with generating and displaying three dimensional objects in a two-dimensional space(eg: display screen).
- In addition to color and brightness, a 3-D pixels adds a depth property that indicates where the point lies on the imaginary z-axis.
- To generate realistic picture we have to first setup a coordinate reference for camera. This co-ordinate reference defines the position and orientation for the plane of the camera.

- This plane used to display a view of the object
- Object description has to transfer to the camera reference co-ordinates and projected onto the selected display plane.



Parallel Projection

- Project points on the object surface along parallel lines onto the display plane.
- Parallel lines are still parallel after projection.
- Used in engineering and architectural drawings.
- Views maintain relative proportions of the object.



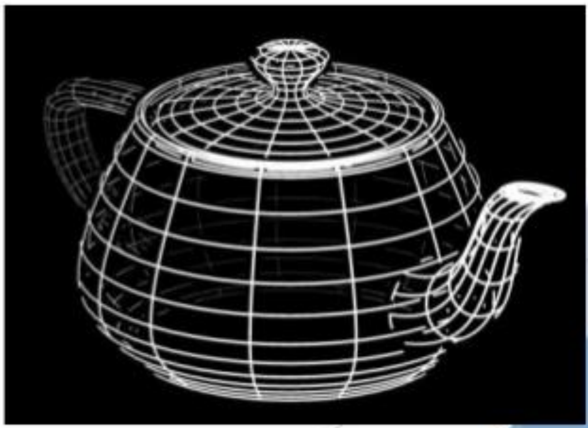
Perspective Projection

- Project points to the display plane along converging paths.
- This is the way that our eyes and a camera lens form images and so the displays are more realistic.
- Parallel lines appear to converge to a distant point in the background.
- Distant objects appear smaller than objects closer to the viewing position.

Depth Cueing

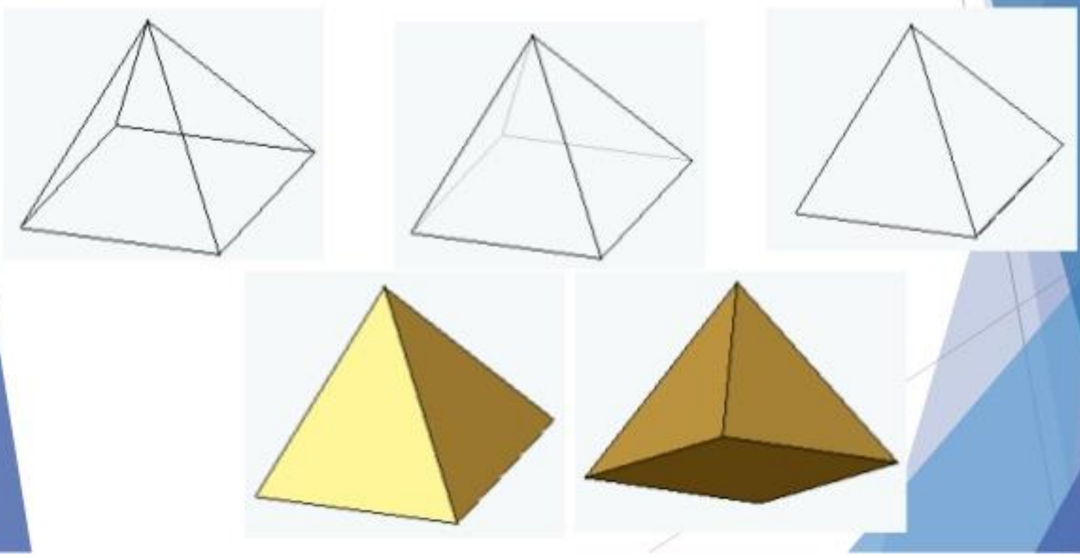
- To easily identify the front and back of display objects.
- Depth information can be included using various methods.
- A simple method to vary the intensity of objects according to their distance from the viewing position.
- Eg: lines closest to the viewing position are displayed with the higher intensities and lines farther away are displayed with lower intensities.

▶ **Application :**modeling the effect of the atmosphere on the pixel intensity of objects. More distant objects appear dimmer to us than nearer objects due to light scattering by dust particles, smoke etc.



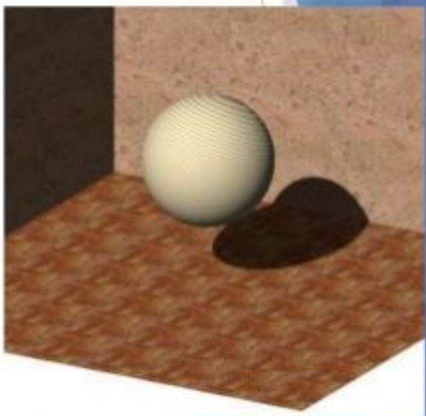
Visible line and surface identification

- Highlight the visible lines or display them in different color
- Display nonvisible lines as dashed lines
- Remove the nonvisible lines



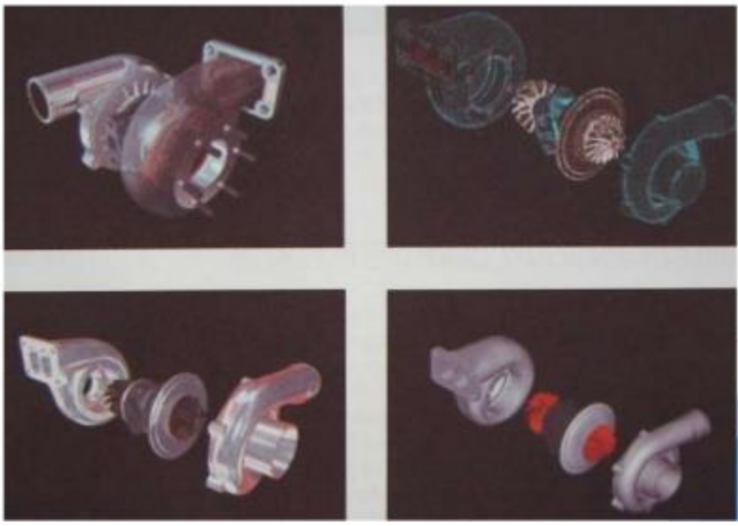
Surface rendering

- Set the surface intensity of objects according to
  - Lighting conditions in the scene
  - Assigned surface characteristics
- Lighting specifications include the intensity and positions of light sources and the general background illumination required for a scene.
- Surface properties include degree of transparency and how rough or smooth of the surfaces



Exploded and Cutaway Views

- To maintain a hierarchical structures to include internal details.
- These views show the internal structure and relationships of the object parts



Stereoscopic Views

- To display objects using stereoscopic views Stereoscopic devices present 2 views of scene
  - One for left eye.
  - Other for right eye.
- These two views displayed on alternate refresh cycle of a raster monitor
- Then viewed through glasses that alternately darken first one lens then the other in synchronized with the monitor refresh cycle.

3D Object Representation

- Graphics scenes contain many different kinds of objects and material surfaces
  - Trees, flowers, clouds, rocks, water, bricks, wood paneling, rubber, paper, steel, glass, plastic and cloth
- **Polygon and Quadric surfaces**: For simple Euclidean objects eg: polyhedron and ellipsoid
- **Spline surfaces and construction**: For curved surfaces eg: aircraft wings , gears
- **Procedural methods – Fractals**: For natural objects eg: cloud, grass
- **Octree Encoding**: For internal features of objects eg:CT image

Representation schemes categories into 2

Boundary representation(B –reps)

- A set of surfaces that separate the object interior from the environment
- Eg) Polygon facets, spline patches

Space-partitioning representation

- Used to describe interior properties.
- Partitioning the spatial region into a set of small, non overlapping, contiguous solids (usually cubes)
- Eg) octree representation

Polygon Surfaces

- Most commonly used boundary representation.

Polygon table

- Specify a polygon surfaces using vertex coordinates and attribute parameter.

Polygon data table organized into 2 group.

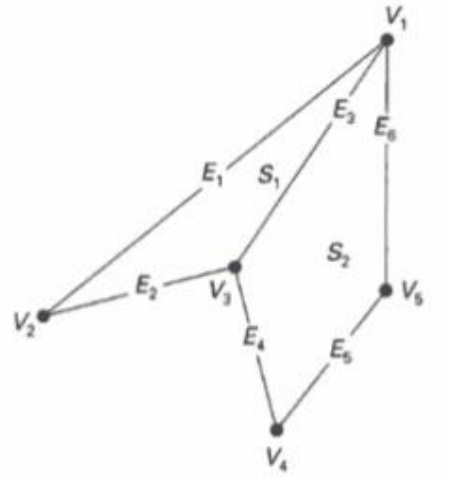
1. **Geometric data table**: vertex coordinate and parameter to identify the spatial orientation.

3 lists

Vertex table –coordinate values of each vertex.

Edge table - pointer back to vertex table to identify the vertices for polygon edge.

Polygon table- pointer back to edge table to identify the edges for each polygon



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

2. **Attribute table**: 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.
- Each polygon has at least one shared edge.

Plane Equation

- The equation for a plane surface expressed at the form

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

- We can obtain the values of A,B,C,D by solving 3 plane equations using the coordinate values for 3 noncollinear points in the plane(x_1,y_1, z_1), (x_2,y_2, z_2), (x_3,y_3, z_3).

$$(A/D)x_k+(B/D)y_k+(C/D)z_k=-1, \quad k=1,2,3$$

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

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

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

$Ax + By + Cz + D \neq 0$ implies that the point (x,y,z) not on a surface

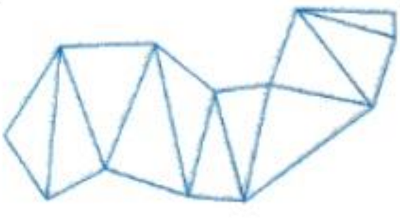
$Ax + By + Cz + D < 0$ implies that the point (x,y,z) is inside the surface.

$Ax + By + Cz + D > 0$ implies that the point (x,y,z) is outside the surface.

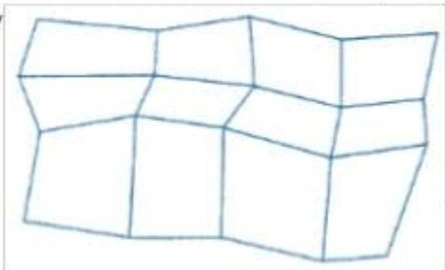
Polygon Meshes

- Object surfaces are tiled to specify the surface facets with mesh function.

  Triangle strip - Produce **n-2** connected triangles ,for **n** vertices



  Quadrilateral mesh - Produce **(n-1)×(m-1)** quadrilateral for **n×m** array



Quadric Surfaces

- Described with second degree equations

Quadric surfaces include:

- Spheres
- Ellipsoids
- Torus

Sphere

- A spherical surface with radius r centred on the origin is defined as the set of points (x, y, z) that satisfy the equation

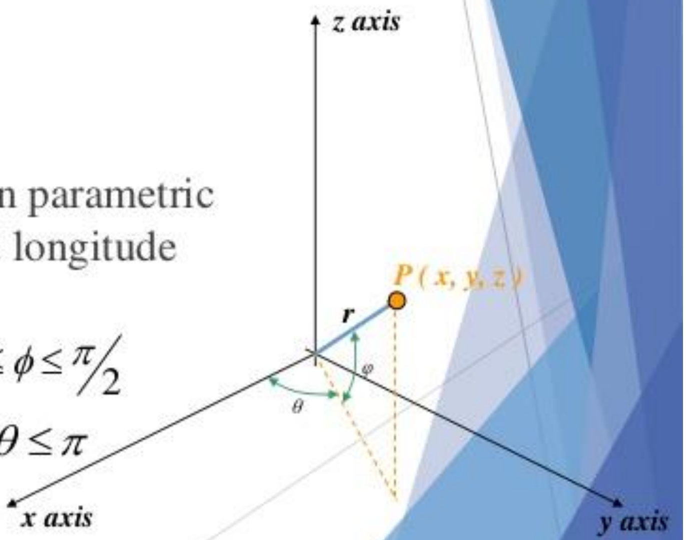
$$x^2 + y^2 + z^2 = r^2$$

- This can also be done in parametric form using latitude and longitude angles

$$x = r\cos\phi\cos\theta \quad -\pi/2 \le \phi \le \pi/2$$

$$y = r\cos\phi\sin\theta \quad -\pi \le \theta \le \pi$$

$$z = r\sin\phi$$



Ellipsoid

- An extension of a spherical surface

Where the radii in three mutually perpendicular directions ,have different values.

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

- parametric form using latitude and longitude angles

$$\begin{aligned} x &= r_x \cos\phi\cos\theta \qquad -\pi/2 \le \phi \le \pi/2 \\ y &= r_y \cos\phi\sin\theta \qquad -\pi \le \theta \le \pi \\ z &= r_z \sin\phi \end{aligned}$$

Torus

- Doughnut –shaped object.

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

- parametric form using latitude and longitude angles

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

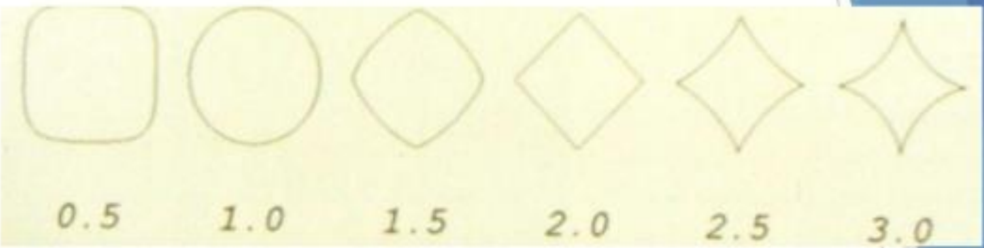
SuperQuadrics

- A generalization of quadric surfaces, formed by including additional parameters into quadric equations
  - Increased flexibility for adjusting object shapes.

Superellipse

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

- When s=1 ,get an ordinary ellipse



- Parametric representation.

$$x = r_x \cos^s \theta \qquad -\pi \le \theta \le \pi$$
$$y = r_y \sin^s \theta$$

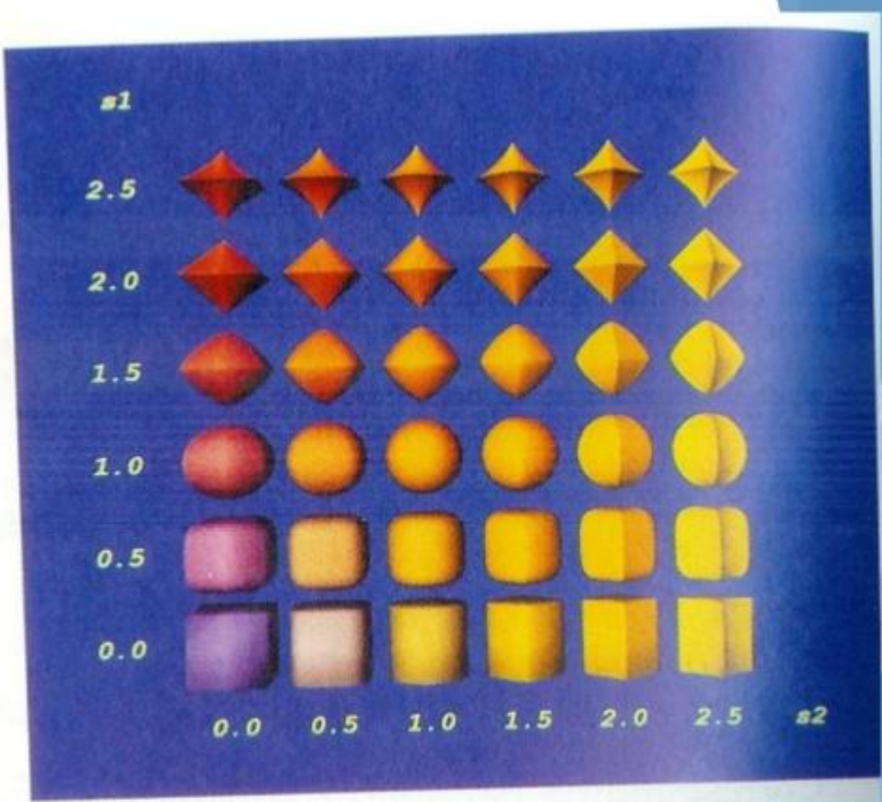
Superellipsoid

$$\left[\left(\frac{x}{r_x}\right)^{2/S_1}+\left(\frac{y}{r_y}\right)^{2/S_2}\right]^{S_2/S_1}+\left(\frac{z}{r_z}\right)^{2/S_1}=1$$

- For s1=s2=1, get an ordinary ellipsoid

- Parametric representation.

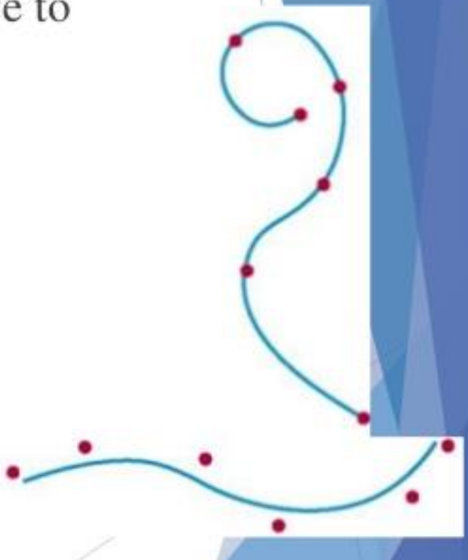
$$x = r_x \cos^{s_1}\phi \cos^{s_2}\theta \qquad -\pi/2 \le \phi \le \pi/2$$
$$y = r_y \cos^{s_1}\phi \sin^{s_2}\theta \qquad -\pi \le \theta \le \pi$$
$$z = r_z \sin^{s_1}\phi$$



Spline Representation

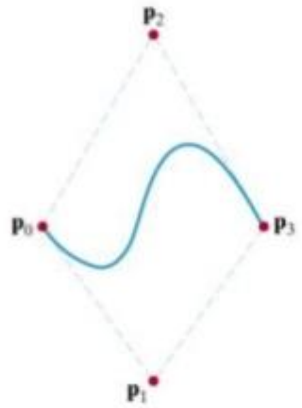
- Spline is a flexible strip used to produce a smooth curve through a designed set of points.
- Spline mathematically describe with a piecewise cubic polynomial function whose first and second derivative are continuous across the various curve section.

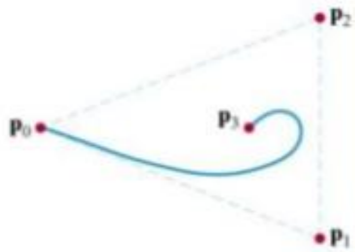
- A spline curve is specified using a set of coordinate position called **control points ,** which indicates the general shape of the curve.
- There are two ways to fit a curve to these points:
  - **Interpolation -** the curve passes through all of the control points.
  - **Approximation -** the curve does not pass through all of the control points, that are fitted to the general control-point path



- The spline curve is defined, modified and manipulated with operation on the control points.

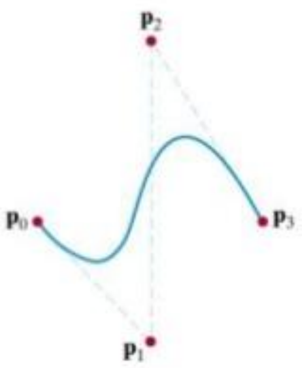
- The boundary formed by the set of control points for a spline is known as a **convex hull**

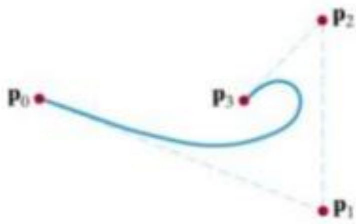




- A polyline connecting the control points is known as a **control graph.**

- Usually displayed to help designers keep track of their splines.





Parametric Continuity Condition

- For the smooth transition from one section of a piecewise parametric curve to the next, impose continuity condition at the connection points.

- Each section of a spline is described with parametric coordinate functions

$$x = x(u)$$
$$y = y(u) \qquad u_1 \le u \le u_2$$
$$z = z(u)$$

Zero –order Parametric continuity (C^0)

- Simply means that the curves meet. That is x,y and z evaluated at u2 for the first curve section are equal to the values of x,y and z evaluated at u1 for the next curve section.

First –order Parametric continuity (C^1)

- The first parametric derivatives(tangent lines) of the coordinate functions for two successive curve sections are equal at their joining point.

Second –order Parametric continuity (C^2)

- Both first and second parametric derivatives of the two curve sections are the same at the intersection.

Geometric Continuity Condition

- In Geometric Continuity ,only require parametric derivatives of the two sections to be proportional to each other at their common boundary

Zero –order Geometric continuity (G^0)

- Same as Zero –order parametric continuity. That is the two curves sections must have the same coordinate position at the boundary point.

First –order Geometric continuity (G^1)

- The first parametric derivatives(tangent lines) of the coordinate functions for two successive curve sections are proportional at their joining point.

Second –order Geometric continuity (G^2)

- Both first and second parametric derivatives of the two curve sections are proportional at their boundary.

Spline Specification

- Three methods for specifying a spline representation.

1. We can state the set of boundary conditions that are imposed on the spline.
2. We can state the matrix that characterizes the spline.
3. We can state the set of blending functions.

- Parametric cubic polynomial representation for the x coordinate of a spline section

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

- Boundary condition set on the endpoint coordinates x(0) and x(1) and on first parametric derivatives at the endpoints x'(0) and x'(1).

- From the boundary condition, obtain the matrix that characterizes the spline curve.

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

$$= U \cdot C$$

$$C = M_{spline} \cdot M_{geom}$$

$$x(u) = U \cdot M_{spline} \cdot M_{geom}$$

Finally the polynomial representation

$$x(u) = \sum_{k=0}^{3} g_k \cdot BF_k(u)$$

Bezier Curve and Surfaces

- This spline approximation method developed by the French engineer Pierre Bezier for use in the design of Renault automobile bodies.
- Easy to implement.
- Available in CAD system, graphic package, drawing and painting packages.

Bezier Curve

- A Bezier curve can be fitted to any number of control points.
- Given $n+1$ control points position

$$p_k=(x_k,\ y_k,\ z_k) \qquad 0 \le k \le n$$

- The coordinate positions are blended to produce the position vector $P(u)$ which describes the path of the Bezier polynomial function between p_0 and p_n

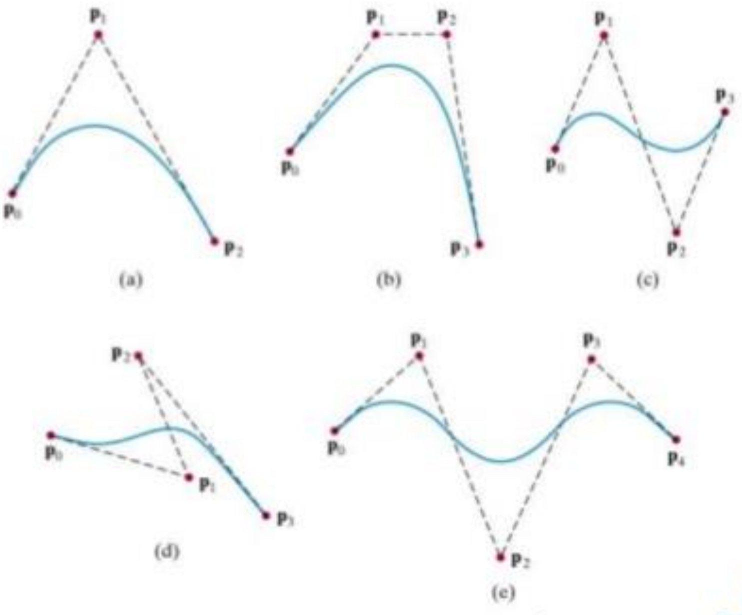
$$P(u) = \sum_{k=0}^{n} p_k BEZ_{k,n}(u), \qquad 0 \le u \le 1$$

- The Bezier blending functions $BEZ_{k,n}(u)$ are the *Bernstein polynomials*

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

Properties Of Bezier Curves

- Bezier Curve is a polynomial of degree one less than the number of control points



- Bezier Curves always passes through the first and last control points.

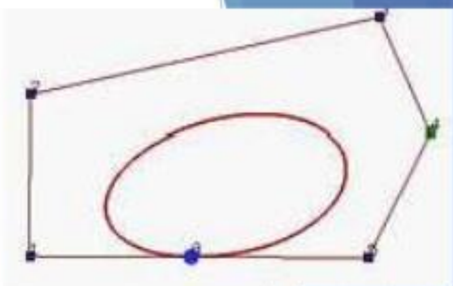
$$P(0) = p_0$$
$$P(1) = p_n$$

- Bezier curves are tangent to their first and last edges of control garph.
- The curve lies within the convex hull as the Bezier blending functions are all positive and sum to 1

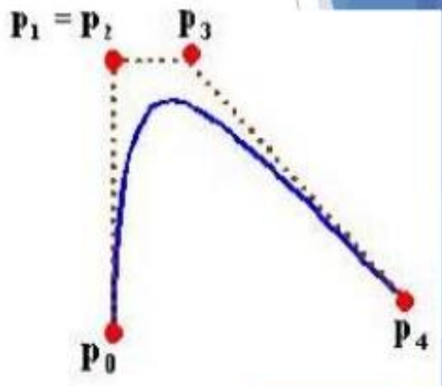
$$\sum_{k=0}^{n} BEZ_{k,n}(u) = 1$$

Design Techniques

- Closed Bezier curves are generated by specifying the first and last control points at same position.



- Specifying multiple control points at a single coordinate position gives more weight to that position.



Cubic Bezier Curve

- Cubic Bezier curves are generated with **4 control points.**
- Cubic Bezier curves gives reasonable design flexibility while avoiding the increased calculations needed with higher order polynomials.

The blending functions when $n = 3$

$$BEZ_{0,3} = (1-u)^3$$

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

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

$$BEZ_{3,3} = u^3$$

- At u=0, $BEZ_{0,3}$=1, and at u=1, $BEZ_{3,3}$=1. thus, the curve will always pass through control points P_0 and P_3.
- The functions $BEZ_{1,3}$ and $BEZ_{2,3}$, influence the shape of the curve at intermediate values of parameter u.
- The resulting curve tends toward points P1 and P3.

Bezier Surface

- Two sets of orthogonal Bezier curves are used to design surface.

$$P(u,v)=\sum_{j=0}^{m}\sum_{k=0}^{n} p_{j,k} BEZ_{j,m}(v) BEZ_{k,n}(u)$$

$P_{j,k}$ specify the location of the control points.

B-Spline Curves and Surfaces

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

2. B-splines allow local control over the shape of a spline curve (or surface)

- The point on the curve that corresponds to a knot is referred to as a *knot vector*.

- The knot vector divide a B-spline curve into curve subinterval, each of which is defined on a knot span.

- Given $n + 1$ control points $P_0, P_1, ..., P_n$
- Knot vector $U = \{ u_0, u_1, ..., u_{n+d} \}$
- The B-spline curve defined by these control points and knot vector

$$P(u) = \sum_{k=0}^{n} p_k B_{k,d}(u), \quad u_{\min} \le u_{\max}, \quad 2 \le d \le n+1$$

P_k is kth control point

Blending function $B_{k,d}$ of degree d-1

- Blending functions defined with Cox-deBoox recursive form

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

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

To change the shape of a B-spline curve, modify one or more of these control parameters:

1. The positions of control points
2. The positions of knots
3. The degree of the curve

Uniform B-Spline

- The spacing between knot values is constant.

Non-uniform B-spline

- Unequal spacing between the knot values.

Open uniform B-Spline

- This B-Spline is across between Uniform B-Spline and non-uniform B-Spline.
- The knot spacing is uniform expect at the ends where knot values are repeated d times

B-Spline Surfaces

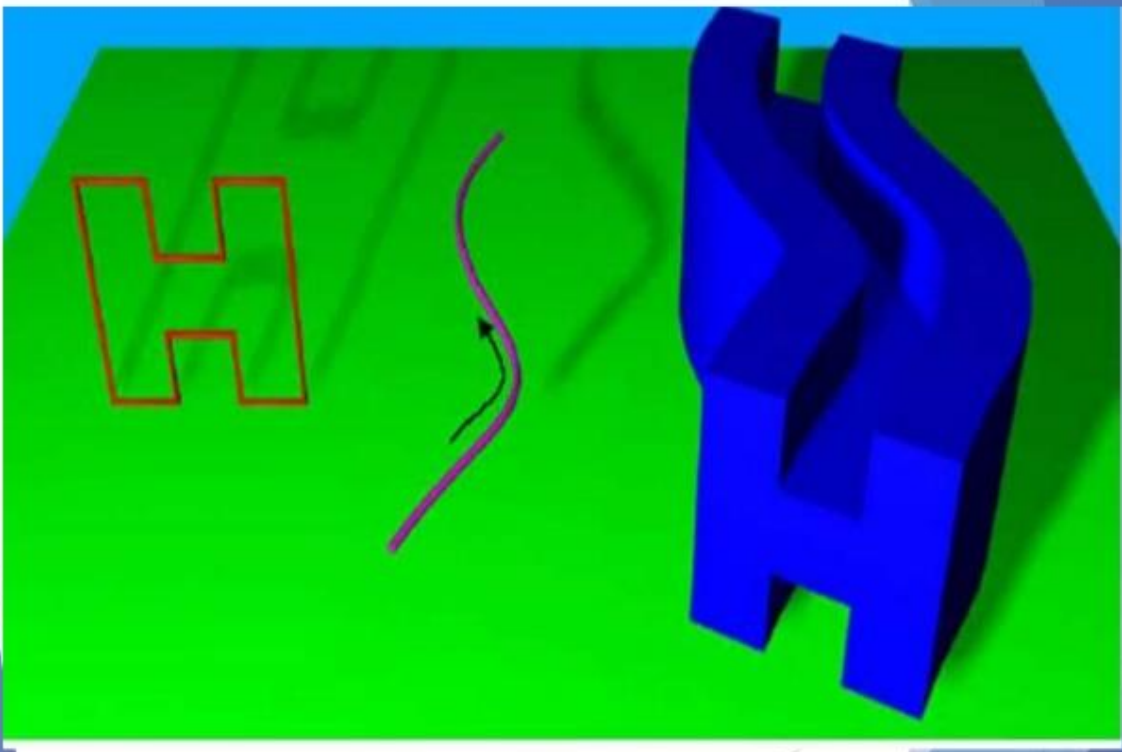
Similar to Bezier surface

$$P(u,v) = \sum_{k_1=0}^{n_1} \sum_{k_2=0}^{n_2} p_{k_1,k_2} B_{k_1,d_1}(u) B_{k_2,d_2}(v)$$

Sweep Representations

- Sweep representations are useful for constructing 3 dimensional objects that possess translational, rotational or other symmetries.

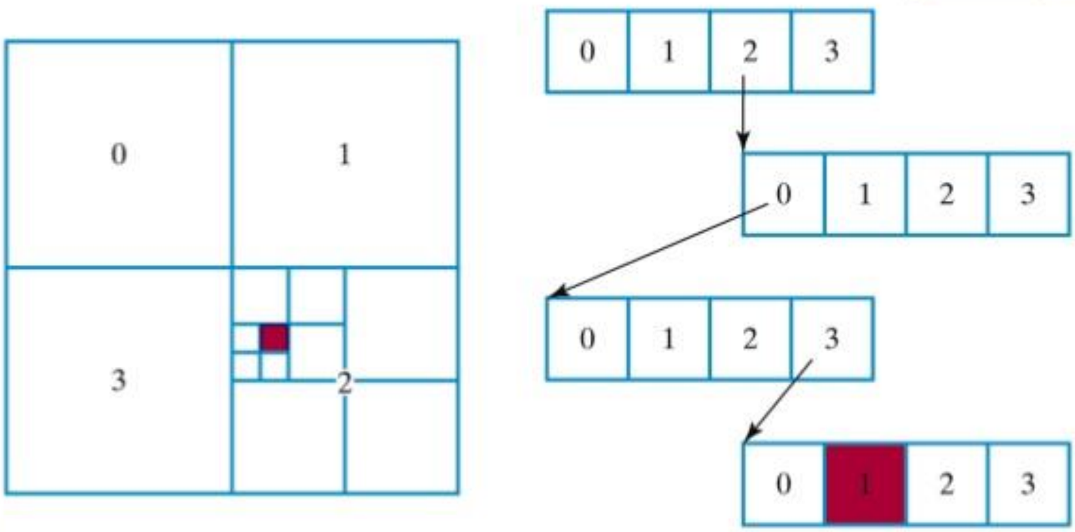
- Objects are specified as a 2 dimensional shape and a sweep that moves that shape through a region of space



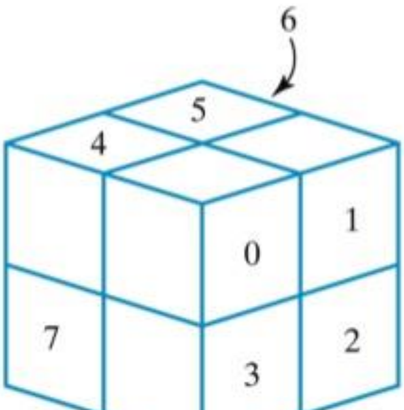
Octrees & Quadtrees

- Octrees are based on a two-dimensional representation scheme called **quadtree** encoding.
- Quadtree encoding divides a square region of space into four equal areas until *homogeneous regions* are found.
- These regions can then be arranged in a tree

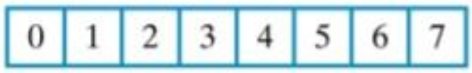
- Quadtree Example



- An octree takes the same approach as quadtrees, but divides a cube region of 3D space into octants.
- Each region within an octree is referred to as a **volume element** or **voxel.**
- Division is continued until homogeneous regions are discovered



Region of a
Three-Dimensional
Space



Data Elements
in the Representative
Octree Node