Computer Graphics
&
Multimedia Techniques

Unit-4

Three Dimensional Graphics

3D display methods:

- To obtain A display of a three-dimensional scene that has been modeled in world coordinates. we must first set up a coordinate reference for the "camera". This coordinate reference defines the position and orientation for the plane of the camera film, which is the plane we want to use to display a view of the objects in the scene. Object descriptions are then transferred to the camera reference coordinates and projected onto the selected display plane.
- Projection Types:
  - **Parallel Projection and Perspective Projection**

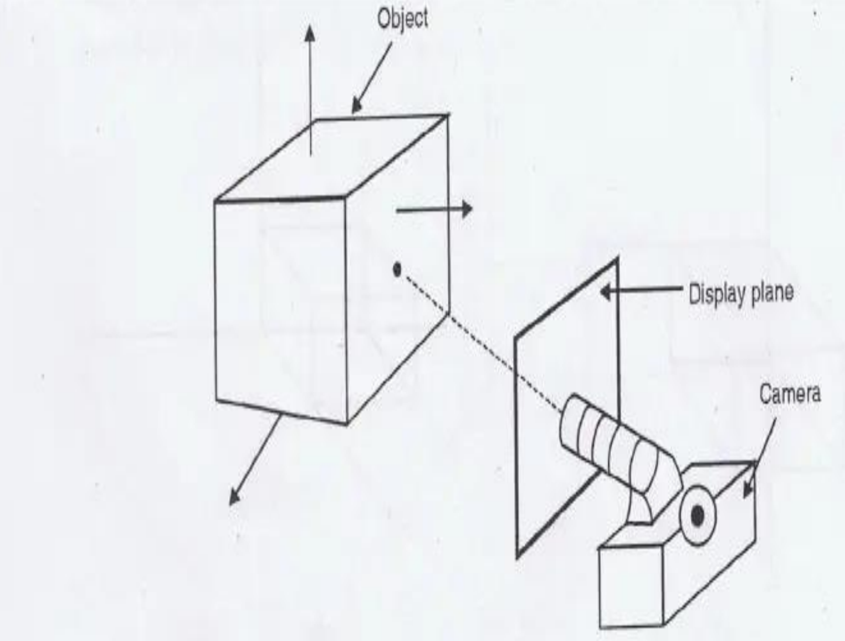
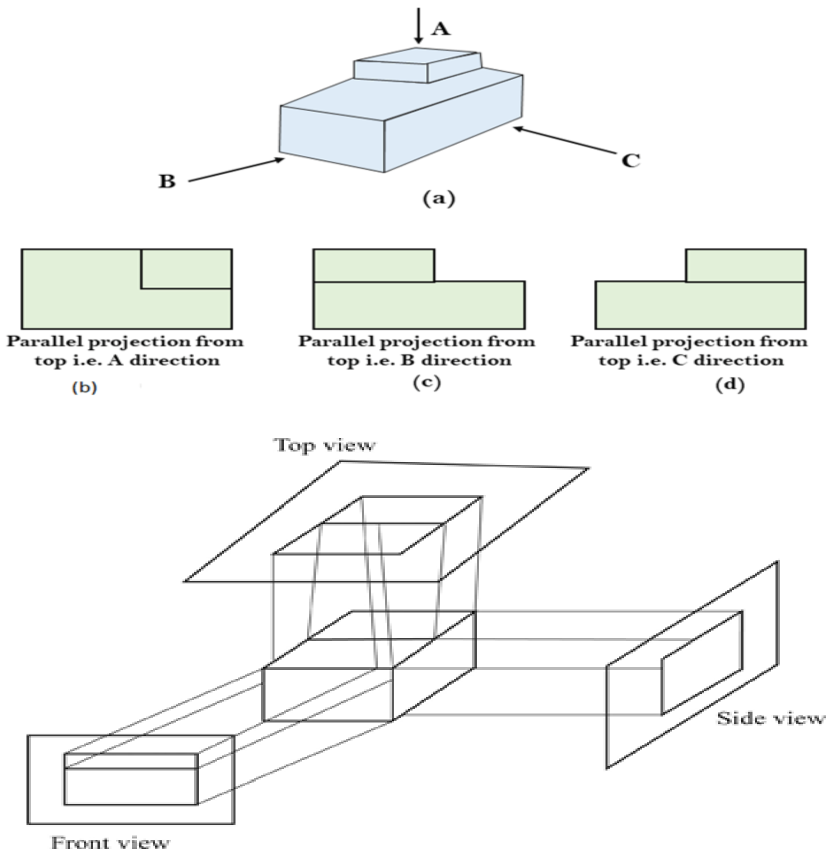


Fig. 4.3.1 : Setting a co-ordinate reference.

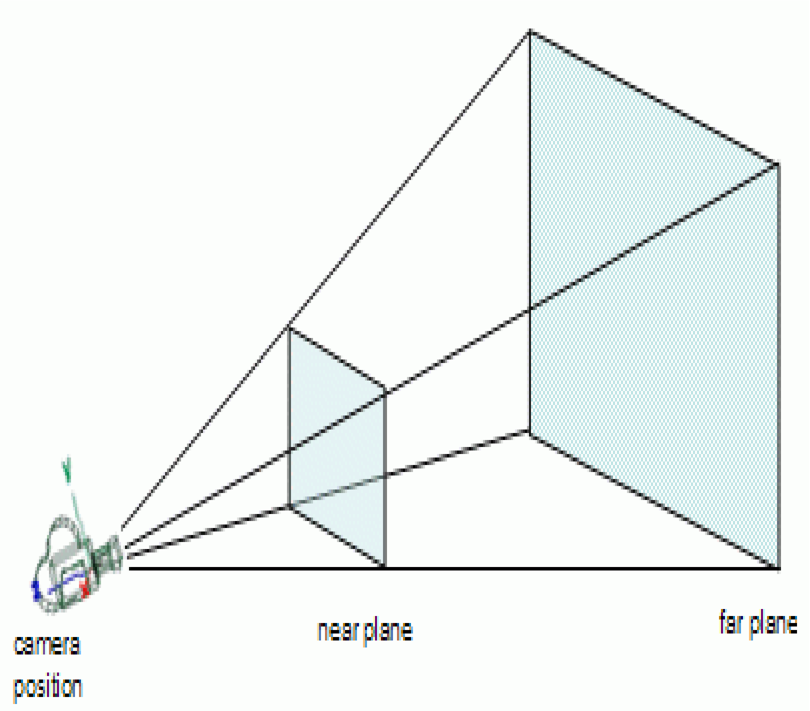
Parallel Projection:

- One method for generating a view of a solid object is to project points on the object surface along parallel lines onto the display plane. By selecting different viewing positions, we can project visible points on the object onto the display plane to obtain different two-dimensional views of the object as shown in figure. In a parallel projection, parallel lines in the world-coordinate scene projected into parallel lines on the two-dimensional display plane. This technique is used in engineering and architectural drawings to represent an object with a set of views that maintain relative proportions of the object.



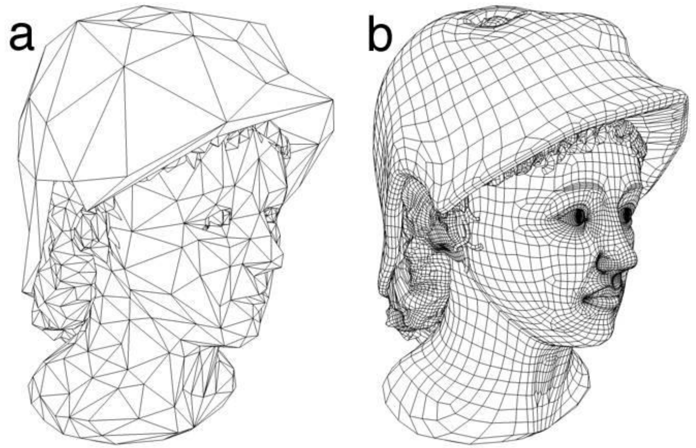
Perspective Projection:

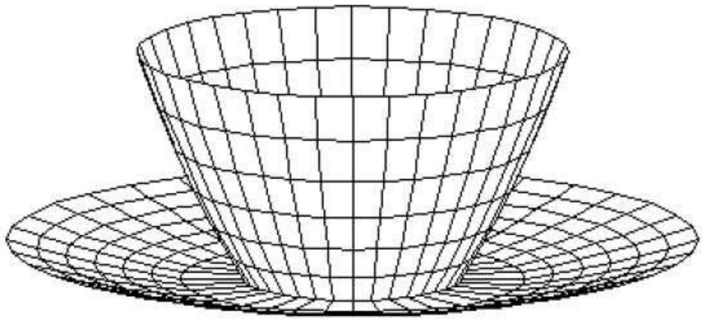
- Another method for generating a view of a three-dimensional scene is to project points to the display plane along converging paths. This causes objects farther from the viewing position to be displayed smaller than objects of the same size that are nearer to the viewing position. In a perspective projection, parallel lines in a scene that are not parallel to the display plane are projected into converging lines. Scenes displayed using perspective projections appear more realistic, since this is the way that our eyes and a camera lens form images as shown in figure.



Polygon Surfaces:

- Polygon surfaces are boundary representations for a 3D graphics object is a set of polygons that enclose the object interior.
- Set of adjacent polygons representing the object exteriors.
- All operations linear, so fast.
- Non-polyhedron shapes can be approximated by polygon meshes.
- Smoothness is provided either by increasing the number of polygons or interpolated shading methods.





Polygon Tables:

- The polygon surface is specified with a set of vertex coordinates and associated attribute parameters.
- For each polygon input, the data are placed into tables that are to be used in the subsequent processing.
- Polygon data tables can be organized into two groups: **Geometric tables** and **attribute tables**.
- **Geometric Tables :** It contain vertex coordinates and parameters to identify the spatial orientation of the polygon surfaces.
- **Attribute tables :** It contain attribute information for an object such as parameters specifying the degree of transparency of the object and its surface reflectivity and texture characteristics.

Polygon Tables:

- A convenient organization for storing geometric data is to create three lists:

1. **The Vertex Table:** Coordinate values for each vertex in the object are stored in this table.
2. **The Edge Table:** It contains pointers back into the vertex table to identify the vertices for each polygon edge.
3. **The Polygon Table:** It contains pointers back into the edge table to identify the edges for each polygon.

Vertex table

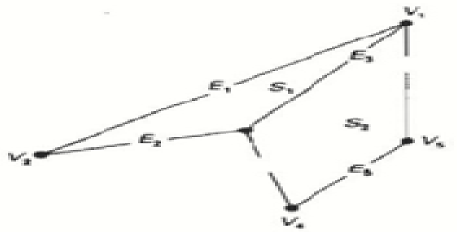
V1 : X1, Y1, Z1
V2 : X2, Y2, Z2
V3 : X3, Y3, Z3
V4 : X4, Y4, Z4
V5 : X5, Y5, Z5

Edge Table

E1 : V1, V2
E2 : V2, V3
E3 : V3, V1
E4 : V3, V4
E5 : V4, V5
E6 : V5, V1

Polygon surface table

S1 : E1, E2, E3
S2 : E3, E4, E5, E6



Polygon Tables:

- Listing the geometric data in three tables provides a convenient to the individual component (vertices, edges and polygons) of each object.
- The object can be displayed efficiently by using data from the edge table to draw the component lines.
- Extra information can be added to the data tables for faster information extraction. For instance, edge table can be expanded to include forward points into the polygon table so that common edges between polygons can be identified more rapidly.
- This is useful for the rendering procedure that must vary surface shading smoothly across the edges from one polygon to the next. Similarly, the vertex table can be expanded so that vertices are cross-referenced to corresponding edges.

$E_1 : V_1, V_2, S_1$

$E_2 : V_2, V_3, S_1$

$E_3 : V_3, V_1, S_1, S_2$

$E_4 : V_3, V_4, S_2$

$E_5 : V_4, V_5, S_2$

$E_6 : V_5, V_1, S_2$

Plane Equations:

- The equation for a plane surface is

$$Ax + By + Cz + D = 0 \text{ ----(1)}$$

- Where (x, y, z) is any point on the plane, and the coefficients A,B,C and D are constants describing the spatial properties of the plane.
- We can obtain the values of A,B,C and D by solving a set of three plane equations using the coordinate values for three non collinear points in the plane.
- For that, we can select three successive polygon vertices (x1, y1, z1), (x2, y2, z2) and (x3, y3, z3) and solve the following set of simultaneous linear plane equations for the ratios A/D, B/D and C/D.

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

Plane Equations:

The solution for this set of equations can be obtained in determinant form, using Cramer's rule as

$$A = \begin{bmatrix} 1 & y1 & z1 \\ 1 & y2 & z2 \\ 1 & y3 & z3 \end{bmatrix} \quad B = \begin{bmatrix} x1 & 1 & z1 \\ x2 & 1 & z2 \\ x3 & 1 & z3 \end{bmatrix} \quad C = \begin{bmatrix} x1 & y1 & 1 \\ x2 & y2 & 1 \\ x3 & y3 & 1 \end{bmatrix} \quad D = -\begin{bmatrix} x1 & y1 & z1 \\ x2 & y2 & z2 \\ x3 & y3 & z3 \end{bmatrix} \qquad (3)$$

Expanding the determinants, we can write the calculations for the plane coefficients in the form:

$$A = y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_3)$$

$$B = z_1 (x_2 - x_3) + z_2 (x_3 - x_1) + z_3(x_1 - x_3)$$

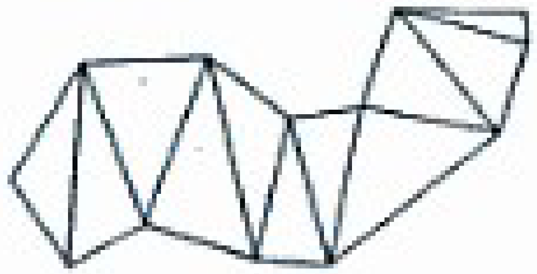
$$C = x_1(y_2 - y_3) + x_2 (y_3 - y_1) + x_3 (y_1 - y_3)$$

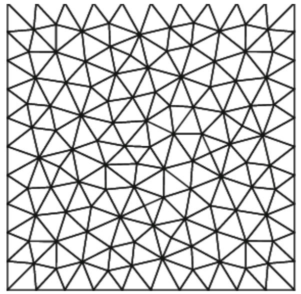
$$D = - x_1(y_2 z_3 - y_3 z_2) - x_2(y_3 z_1 - y_1 z_3) - x_3(y_1 z_2 - y_2 z_1) \qquad (4)$$

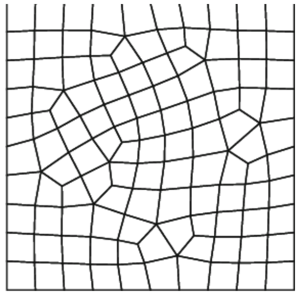
- As vertex values and other information are entered into the polygon data structure, values for A, B, C and D are computed for each polygon and stored with the other polygon data.
- Plane equations are used also to identify the position of spatial points relative to the plane surfaces of an object. For any point (x, y, z) hot on a plane with parameters A,B,C,D, we have $Ax + By + Cz \neq 0$

Polygon Meshes:

- A single plane surface can be specified with a function such as fill Area. But when object surfaces are to be tiled, it is more convenient to specify the surface facets with a mesh function.
- One type of polygon mesh is the triangle strip A triangle strip formed with 11 triangles connecting 13 vertices. This function produces n-2 connected triangles given the coordinates for n vertices.
- Another similar function in the quadrilateral mesh, which generates a mesh of (n-1) by (m-1) quadrilaterals, given the coordinates for an n by m array of vertices. Figure shows 20 vertices forming a mesh of 12 quadrilaterals.





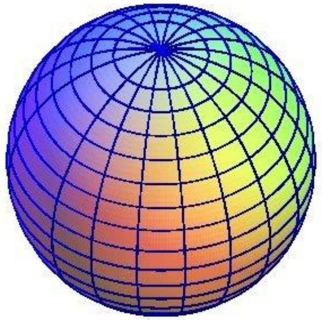


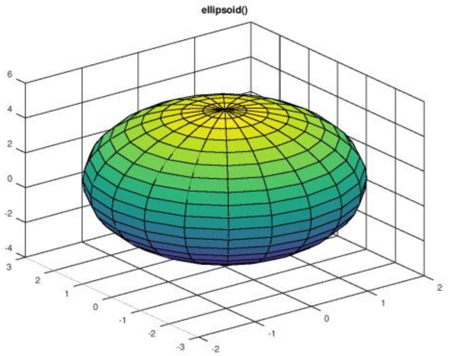
Curved Line and Surfaces:

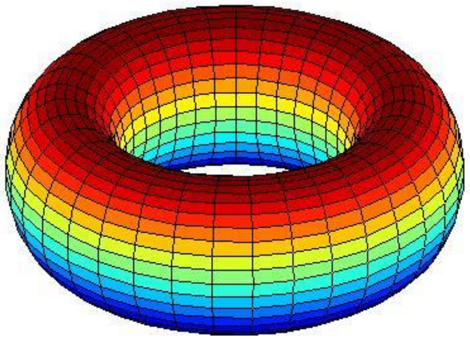
- Displays of three dimensional curved lines and surface can be generated from an input set of mathematical functions defining the objects or from a set of user specified data points.
- When function are specified ,a package can project the defining equations for a curve to the display plane and plot pixel positions along the path of the projected function.
- For surfaces, a functional description in decorated to produce a polygon-mesh approximation to the surface.

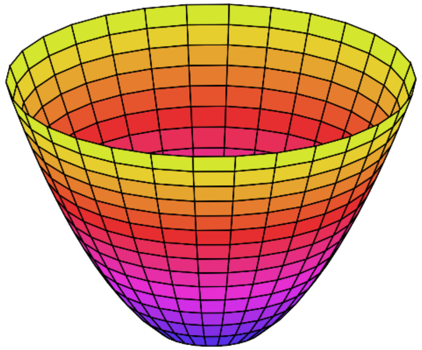
Quadric Surfaces:

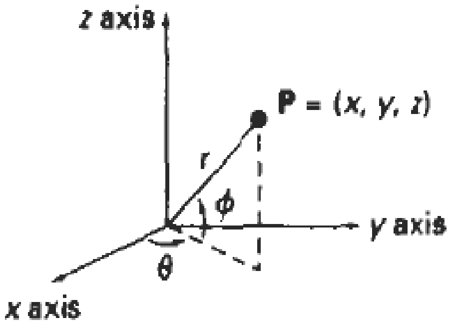
- The quadric surfaces are described with second degree equations (quadratics). They include **spheres, ellipsoids,** torus, parabolids, and hyperboloids.

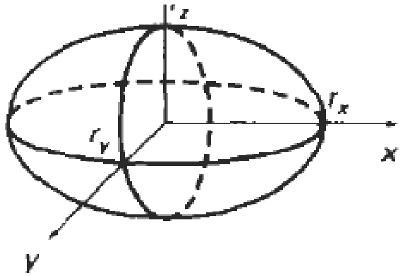


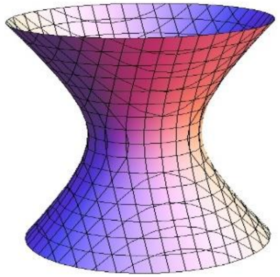




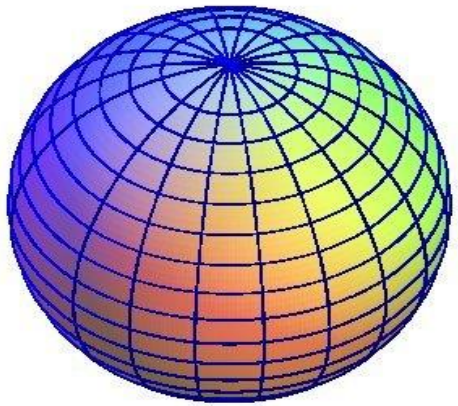








Sphere:

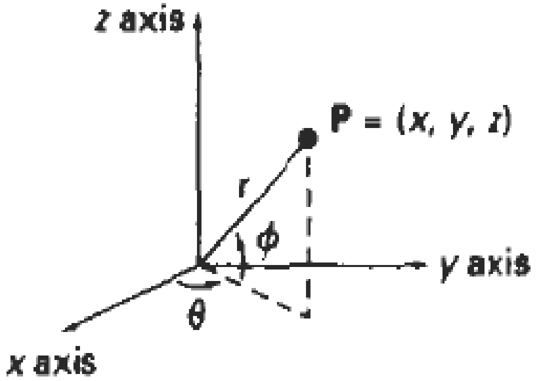


- In Cartesian coordinates, a spherical surface with radius r cantered on the coordinates origin is defined as the set of points (x, y, z) that satisfy the equation.

$$x2 + y2 + z2 = r2 \qquad (1)$$

- The spherical surface can be represented in parametric form by using latitude and longitude angles, the parameter representation in eqn.(2) provides a symmetric range for the angular parameter θ and ϕ.

$$\begin{aligned} x &= r\cos\varphi\cos\theta, && -\pi/2 <= \varphi <= \pi/2 \\ y &= r\cos\varphi\sin\theta, && -\pi <= \varphi <= \pi \qquad (2) \\ z &= r\sin\varphi \end{aligned}$$



Ellipsoids:

Ellipsoid surface is an extension of a spherical surface where the radius in three mutually perpendicular directions can have different values the Cartesian representation for points over the surface of an ellipsoid centered on the origin is

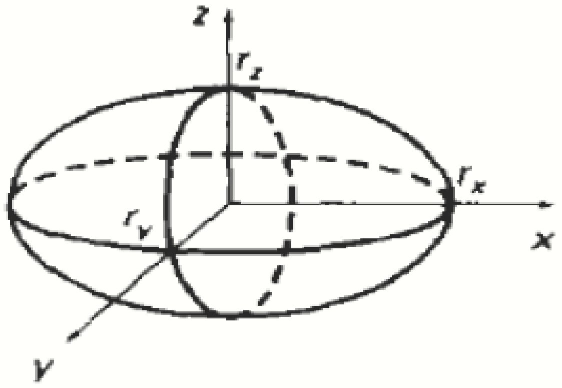
$$\frac{x^2}{r_x} + \frac{y^2}{r_y} + \frac{z^2}{r_z} = 1$$

The parametric representation for the ellipsoid in terms of the latitude angle φ and the longitude angle θ is

$x = r_x \cos\varphi \cos\theta, \qquad -\pi/2 <= \varphi <= \pi/2$

$y = r_y \cos\varphi \sin\theta, \qquad -\pi <= \varphi <= \pi$

$z = r_z \sin\varphi$



Spline Representations:

A Spline is a flexible strip used to produce a smooth curve through a designated set of points.

Several small weights are distributed along the length of the strip to hold it in position on the drafting table as the curve is drawn.

The Spline curve refers to any sections curve 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.

Splines are used in graphics applications to design curve and surface shapes, to digitize drawings for computer storage, and to specify animation paths for the objects or the camera in the scene.

CAD applications for splines include the design of automobiles bodies, aircraft and spacecraft surfaces, and ship hulls.

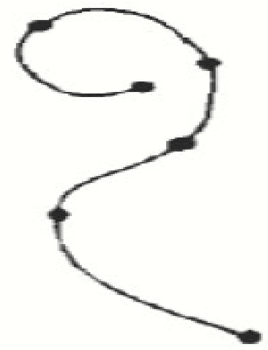
Spline Representation:

Interpolation and Approximation Splines

- Spline curve can be specified by a set of coordinate positions called control points which indicates the general shape of the curve.
- These control points are fitted with piecewise continuous parametric polynomial functions in one of the two ways.

Interpolation

- When polynomial sections are fitted so that the curve passes through each control point the resulting curve is said to interpolate the set of control points.
- A set of six control points interpolated with piecewise continuous polynomial sections



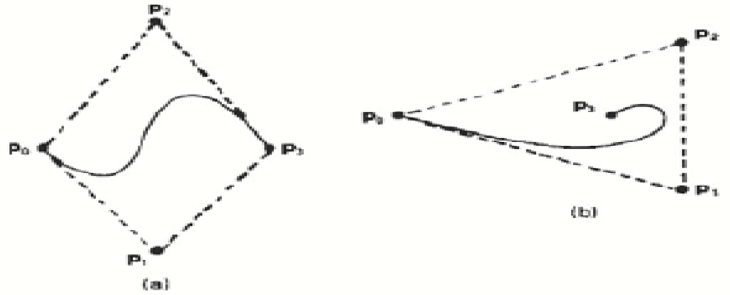
Spline Representation:

<u>Approximation</u>

When the polynomials are fitted to the general control point path without necessarily passing through any control points, the resulting curve is said to approximate the set of control points.

- A set of six control points approximated with piecewise continuous polynomial sections

- Interpolation curves are used to digitize drawings or to specify animation paths.
- Approximation curves are used as design tools to structure object surfaces.
- A spline curve is designed , modified and manipulated with operations on the control points. The curve can be translated, rotated or scaled with transformation applied to the control points.
- The convex polygon boundary that encloses a set of control points is called the convex hull.
- The shape of the convex hull is to imagine a rubber band stretched around the position of the control points so that each control point is either on the perimeter of the hull or inside it.
- Convex hull shapes (dashed lines) for two sets of control points



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.
- 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 coordinates.

$$\mathbf{p}_k = (x_k, y_k, z_k), \qquad k = 0, 1, 2, \ldots, n$$

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

$$y(u) = a_y u^3 + b_y u^2 + c_y u + d_y, \qquad (0 \le u \le 1)$$

$$z(u) = a_z u^3 + b_z u^2 + c_z u + d_z$$

Hermite Interplolation:

Hemite spline is an interpolating piecewise cubic polynomial with a specified tangent at each control point. Unlike the natural cubic splines, Hermite spline can be adjusted locally because each curve section is only dependent on its endpoint constraints.

Cardinal Spline:

As with Hermite splines, cardinal splines are interpolating piecewise cubics with specified endpoint tangents at the boundary of each curve section. The difference is that we do not have to give the values for the endpoint tangents. For a cardinal spline, the value for the slope at a control point is calculated from the coordinates of the two adjacent control points.

Bezier Curve:

- Bezier curve is discovered by the French engineer **Pierre Bézier**. These curves can be generated under the control of other points. Approximate tangents by using control points are used to generate curve. The Bezier curve can be represented mathematically as –

$$\sum_{k=0}^{n} P_i B_i^n(t)$$

Where p_i is the set of points and $B_i^n(t)$ represents the Bernstein polynomials which are given by –

$$B_i^n(t) = \binom{n}{i} (1-t)^{n-i} t^i$$

Where **n** is the polynomial degree, **i** is the index, and **t** is the variable.

The simplest Bézier curve is the straight line from the point P_0 to P_1 . A quadratic Bezier curve is determined by three control points. A cubic Bezier curve is determined by four control points.

Simple Bezier Curve **Quadratic Bazier Curve** **Cubic Bazier Curve**

Bezier Curve Properties:

- They generally follow the shape of the control polygon, which consists of the segments joining the control points.
- They always pass through the first and last control points.
- They are contained in the convex hull of their defining control points.
- The degree of the polynomial defining the curve segment is one less that the number of defining polygon point. Therefore, for 4 control points, the degree of the polynomial is 3, i.e. cubic polynomial.
- A Bezier curve generally follows the shape of the defining polygon.
- The direction of the tangent vector at the end points is same as that of the vector determined by first and last segments.
- The convex hull property for a Bezier curve ensures that the polynomial smoothly follows the control points.
- No straight line intersects a Bezier curve more times than it intersects its control polygon.
- They are invariant under an affine transformation.
- Bezier curves exhibit global control means moving a control point alters the shape of the whole curve.
- A given Bezier curve can be subdivided at a point t=t0 into two Bezier segments which join together at the point corresponding to the parameter value t=t0.

Bezier Curve: Ex.1

Draw Bezier Curve with control Points 1

P1(1,2) P2(3,4) P3(6,−6) P4(10,8) step size of 0.2 for [x(t) y(t)]

Sol. so $t = 0, 0.2, 0.4, 0.6, 0.8, 1$

step size is 0.2

Parametric Eq. of Bezier Curve

eq. (1)
$$\begin{cases} x = x_4 t^3 + 3x_3 t^2(1-t) + 3x_2 t(1-t)^2 + x_1(1-t)^3 \\ y = y_4 t^3 + 3y_3 t^2(1-t) + 3y_2 t(1-t)^2 + y_1(1-t)^3 \end{cases}$$

$x_1 = 1$ $\quad y_1 = 2$

$x_2 = 3$ $\quad y_2 = 4$

$x_3 = 6$ $\quad y_3 = -6$

$x_4 = 10$ $\quad y_4 = 8$

Bezier Curve: Ex.1

Putting the values of x and y in eq. (1)

$$x = 10t^3 + 18t^2(1-t) + 9t(1-t)^2 + 1(1-t)^3$$

$$y = 8t^3 - 18t^2(1-t) + 12t(1-t)^2 + 2(1-t)^3$$

1.) $t = 0$
$x = 1$
$y = 2$

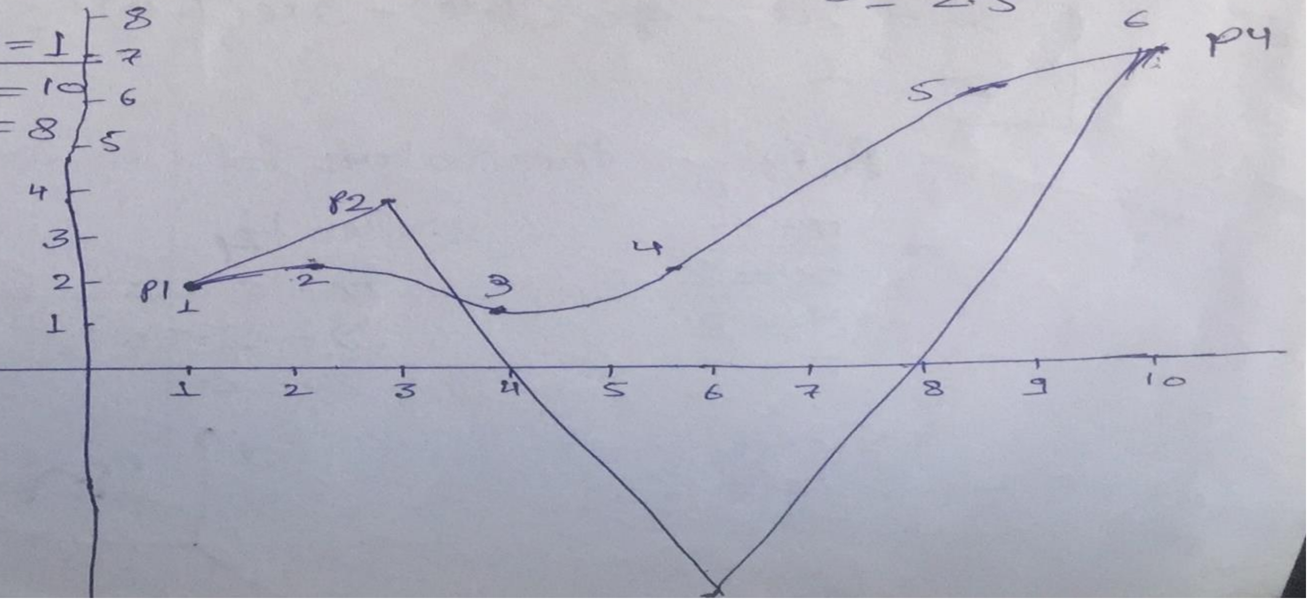
2.) $t = 0.2$
$x = 2.32$
$y = 2.1$

3.) $t = 0.4$
$x = 3.88$
$y = 1.5$

4.) $t = 0.6$
$x = 5.6$
$y = 2.5$

5.) $t = 0.8$
$x = 7.72$
$y = 7.3$

6. $t = 1$
$x = 10$
$y = 8$



Bezier curve:Ex.2

Cubic Bezier Curve four Control Points

$P1 \rightarrow (\overset{x_1}{2}, \overset{y_1}{2}, \overset{z_1}{0})$

$P2 \rightarrow (\overset{x_2}{2}, \overset{y_2}{3}, \overset{z_2}{0})$

$P3 \rightarrow (\overset{x_3}{3}, \overset{y_3}{3}, \overset{z_3}{0})$

$P4 \rightarrow (\overset{x_4}{3}, \overset{y_4}{2}, \overset{z_4}{0})$

Parametric Eqn for Bezier curve

$$x = x_4 u^3 + 3x_3 u^2(1-u) + 3x_2 u(1-u)^2 + x_1(1-u)^3$$

$$y = y_4 u^3 + 3y_3 u^2(1-u) + 3y_2 u(1-u)^2 + y_1(1-u)^3$$

$$z = z_4 u^3 + 3z_3 u^2(1-u) + 3z_2 u(1-u)^2 + z_1(1-u)^3$$

Replace value of x_1, x_2, x_3, x_4 and same for the y_1, y_2, y_3, y_4, z_1, z_2, z_3, z_4

Bezier curve: Ex.2

$$\text{I}\left[\begin{array}{l} x = 3u^3 + 9u^2(1-u) + 6u(1-u)^2 + 2(1-u)^3 \\ y = 2u^3 + 9u^2(1-u) + 9u(1-u)^2 + 2(1-u)^3 \\ z = 0 \end{array}\right.$$

Putting the value of u in eq (1)

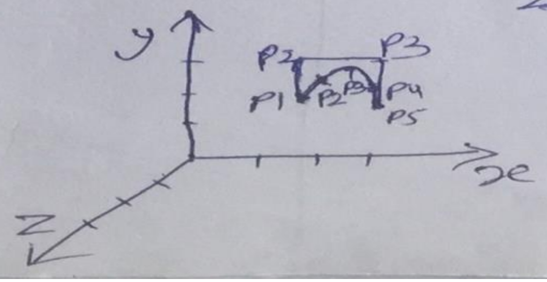
1.) $u = 0$
$x = 2$
$y = 2$
$z = 0$

2.) $u = 1/4$
$x = 2.156$
$y = 2.56$
$z = 0$

3.) $u = \frac{1}{2}$
$x = 2.5$
$y = 2.75$
$z = 0$

4.) $u = \frac{3}{4}$
$x = 2.84$
$y = 2.56$
$z = 0$

5.) $u = 1$
$x = 3$
$y = 2$
$z = 0$



Bezier curve: Ex.3

3

$P1 \rightarrow (2,2,2)$

$P2 \rightarrow (2,3,2)$

$P3 \rightarrow (3,3,3)$

$P4 \rightarrow (3,2,3)$

$$x = x_4 u^3 + 3x_3 u^2(1-u) + 3x_2 u(1-u)^2 + x_1(1-u)^3$$

$$y = y_4 u^3 + 3y_3 u^2(1-u) + 3y_2 u(1-u)^2 + y_1(1-u)^3$$

$$z = z_4 u^3 + 3z_3 u^2(1-u) + 3z_2 u(1-u)^2 + z_1(1-u)^3$$

Replace x, y, z

eq (1)

$$x = 3u^3 + 9u^2(1-u) + 6u(1-u)^2 + 2(1-u)^3$$

$$y = 2u^3 + 9u^2(1-u) + 9u(1-u)^2 + 2(1-u)^3$$

$$z = 3u^3 + 9u^2(1-u) + 6u(1-u)^2 + 2(1-u)^3$$

Putting the value of u in eq-1

Bezier curve: Ex.3

1:) $u = 0$

$x = 2$
$y = 2$
$z = 2$

2:) $u = 1/4$

$x = 2.156$
$y = 2.56$
$z = 2.156$

3:) $u = \frac{1}{2}$

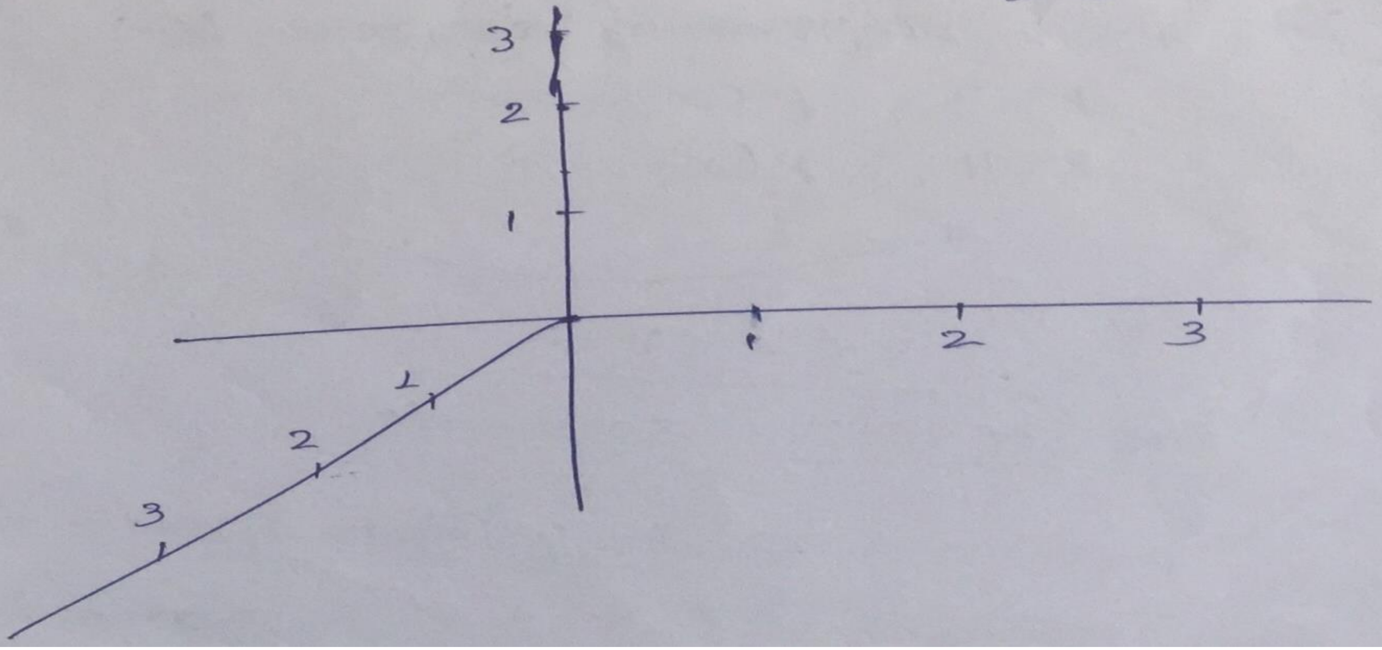
$x = 2.5$
$y = 2.75$
$z = 2.5$

4:) $u = \frac{3}{4}$

$x = 2.84$
$y = 2.56$
$z = 2.84$

5:) $u = 1$

$x = 3$
$y = 2$
$z = 3$



B-Spline Curve:

The Bezier-curve produced by the Bernstein basis function has limited flexibility.

- First, the number of specified polygon vertices fixes the order of the resulting polynomial which defines the curve.
- The second limiting characteristic is that the value of the blending function is nonzero for all parameter values over the entire curve.

The B-spline basis contains the Bernstein basis as the special case. The B-spline basis is non-global.

A B-spline curve is defined as a linear combination of control points Pi and B-spline basis function

$N_{i,}$ k t given by

$$C(t) = \sum_{i=0}^{n} P_i N_{i,k}(t), \qquad n \geq k-1, \qquad t \,\epsilon\, [tk-1, tn+1]$$

B-Spline Curve:

$$C(t) = \sum_{i=0}^{n} P_i N_{i,k}(t), \qquad n \geq k-1, \qquad t \,\epsilon\, [tk-1, tn+1]$$

Where,

- { p_i : i=0, 1, 2….n} are the control points
- k is the order of the polynomial segments of the B-spline curve. Order k means that the curve is made up of piecewise polynomial segments of degree k - 1,
- the $N_{i,k}(t)$ are the “normalized B-spline blending functions”. They are described by the order k and by a non-decreasing sequence of real numbers normally called the “knot sequence”.

$$t_i : i = 0, \ldots n + K$$

B-Spline Curve:

The N_i, k functions are described as follows −

$$N_{i,1}(t) = \begin{cases} 1, & if\, u \,\epsilon\, [t_i, t_{i+1}) \\ 0, & Otherwise \end{cases}$$

and if k > 1,

$$N_{i,k}(t) = \frac{t - t_i}{t_{i+k-1}} N_{i,k-1}(t) + \frac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t)$$

and

$$t \,\epsilon\, [t_{k-1}, t_{n+1})$$

B-Spline Curve:

Properties of B-spline Curve

B-spline curves have the following properties −

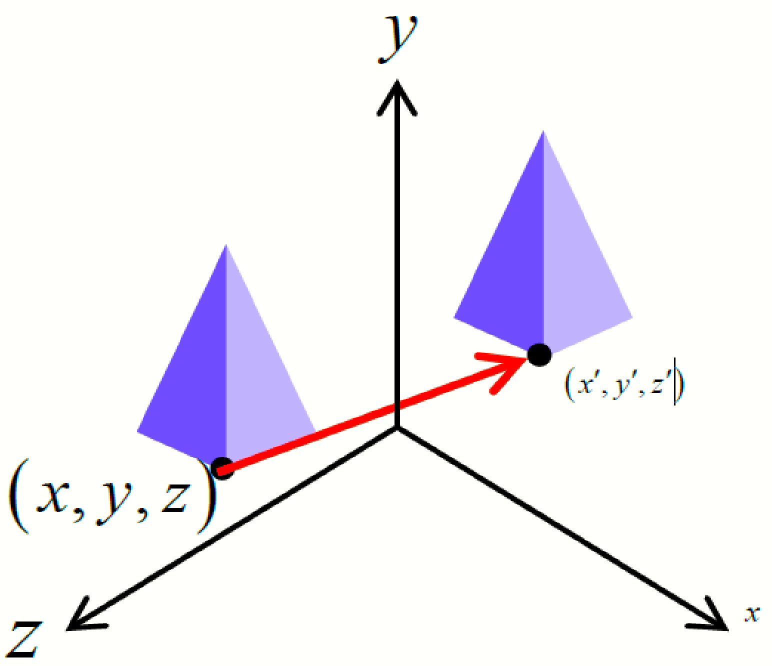
- The sum of the B-spline basis functions for any parameter value is 1.
- Each basis function is positive or zero for all parameter values.
- Each basis function has precisely one maximum value, except for k=1.
- The maximum order of the curve is equal to the number of vertices of defining polygon.
- The degree of B-spline polynomial is independent on the number of vertices of defining polygon.
- B-spline allows the local control over the curve surface because each vertex affects the shape of a curve only over a range of parameter values where its associated basis function is nonzero.
- The curve exhibits the variation diminishing property.
- The curve generally follows the shape of defining polygon.
- Any affine transformation can be applied to the curve by applying it to the vertices of defining polygon.
- The curve line within the convex hull of its defining polygon.

3D Transformation: Translation

- A translation moves all points in an object along the same straight line path to new positions.
- The path is represented by a vector, called the translation or shift vector.

We can write the components as

Translation



$$x' = x + t_x$$
$$y' = y + t_y$$
$$z' = z + t_z$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3D Transformation: Rotation

- For 3D object rotation transformation we need to pick an axis to rotate about and the amount of angular rotation.
- 3D rotation can be specified around any line in space.
- The most common choices are the X-axis, the Y-axis, and the Z-axis.

Z-axis rotation is identical to the 2D case:

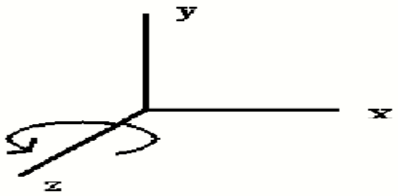
$x' = x\cos\theta - y\sin\theta$

$y' = x\sin\theta + y\cos\theta$

$z' = z$

The parameter θ is specifies the rotation angle. In homogeneous coordinate form ,the 3D z-axis rotation equations are expressed as

$$[x', y', z', 1] = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



We can write more compactly as

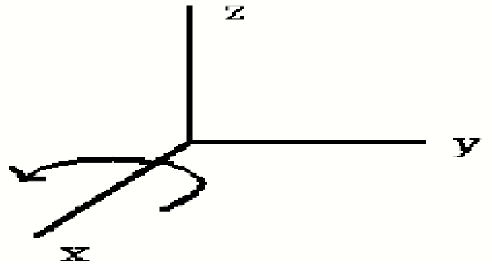
$P' = R_z(\theta).P$

3D Transformation: Rotation

Rotation about X- axis

X-axis rotation looks like Z-axis rotation if replace: X axis with Y axis ,Y axis with Z axis Z axis with X axis.So we do the same replacement in the equations:

Same argument as for rotation about *z* axis For rotation about *x* axis, *x* is unchanged



$$x' = x$$
$$y' = y\cos\theta - z\sin\theta$$
$$z' = y\sin\theta + z\cos\theta$$

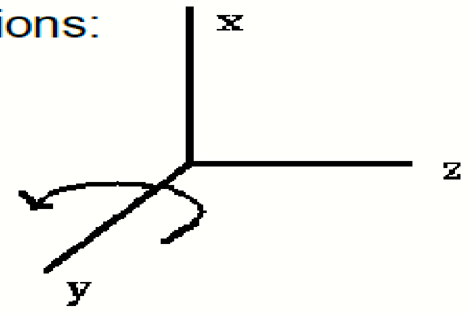
$$[x', y', z', 1] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$P' = R_x(\theta).P$$

3D Transformation: Rotation

Rotation about y axis

- Y-axis rotation looks like Z-axis rotation if replace: X axis with Z axis ,Y axis with X axis ,Z axis with Y axis. So we do the same replacement in equations:
- Same argument as for rotation about *z* axis
- For rotation about *y* axis, *y* is unchanged



$x' = x\cos\theta + z\sin\theta$

$y' = y$

$z' = z\cos\theta - x\sin\theta$

$$[x', y', z', 1] = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

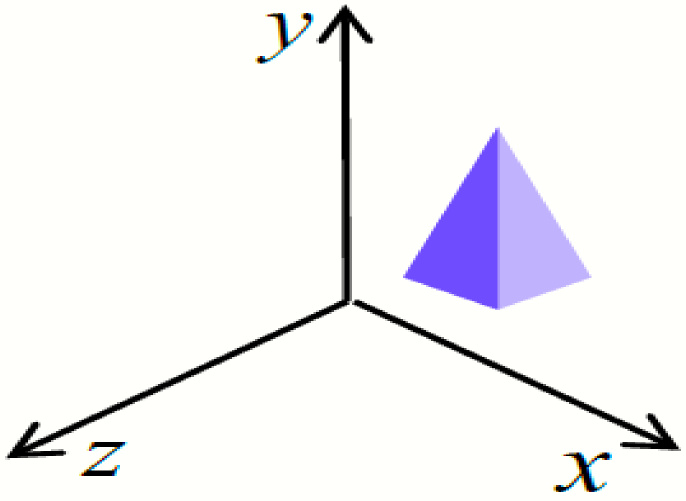
$P' = R_y(\theta).P$

3D Transformation: Scaling

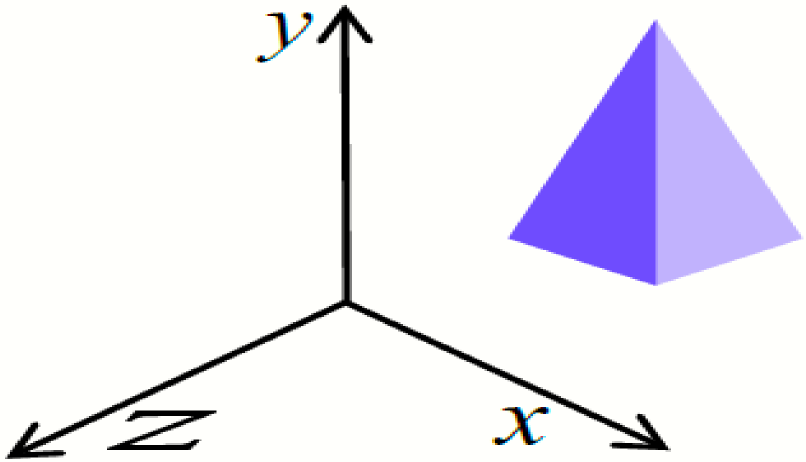
Scaling changes the size of an object and involves the scale factors.

The scaling parameters S_x , S_y and S_z are assigned any positive values.

Scales the object about the origin, Here changes the size of the object along x,y and z-coordinate is same



The object before scaling



Object after scaling

3D Transformation: Scaling

$$x' = S_x . x$$

$$y' = S_y . y$$

$$z' = S_z . z$$

$$\mathbf{P'} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{S} \cdot \mathbf{P}$$

3D Transformation: Reflection

- A Three-dimensional can be performed relative to a selected reflection axes or with respect to selected reflected plane.
- Three-dimensional reflection matrices are set up similar to those for two dimensional.
- Reflection relative to given axis are equivalent to 180° rotation about that axis.
- The reflection planes are either xy, xz or yz

- The matrix expression for the reflection transformation of a position P = (*x*, *y*, *z*) relative to *xy* plane can be written as:

$$RF_z = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Transformation matrices for inverting *x* and *y* values are defined similarly, as reflections relative to *yz* plane and *xz* plane, respectively

3D Transformation: Shear

The matrix expression for the shearing transformation of a position P = (*x*, *y*, *z*), to produce *z*-axis shear, can be written as:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Parameters *a* and *b* can be assigned any real values. The effect of this transformation is to alter *x*- and *y*- coordinate values by an amount that is proportional to the *z* value, while leaving the *z* coordinate unchanged.
- Shearing transformations for the *x* axis and *y* axis are defined similarly.

Composite Transformation:

We can form arbitrary affine transformation matrices by multiplying together rotation, translation, and scaling matrices

Because the same transformation is applied to many vertices, the cost of forming a matrix M=ABCD is not significant compared to the cost of computing Mp for many vertices p

The difficult part is how to form a desired transformation from the specifications in the application

Consider the composite transformation matrix **M=ABC**

When we calculate **Mp,** matrix C is the first applied, then B, then A

Mathematically, the following are equivalent

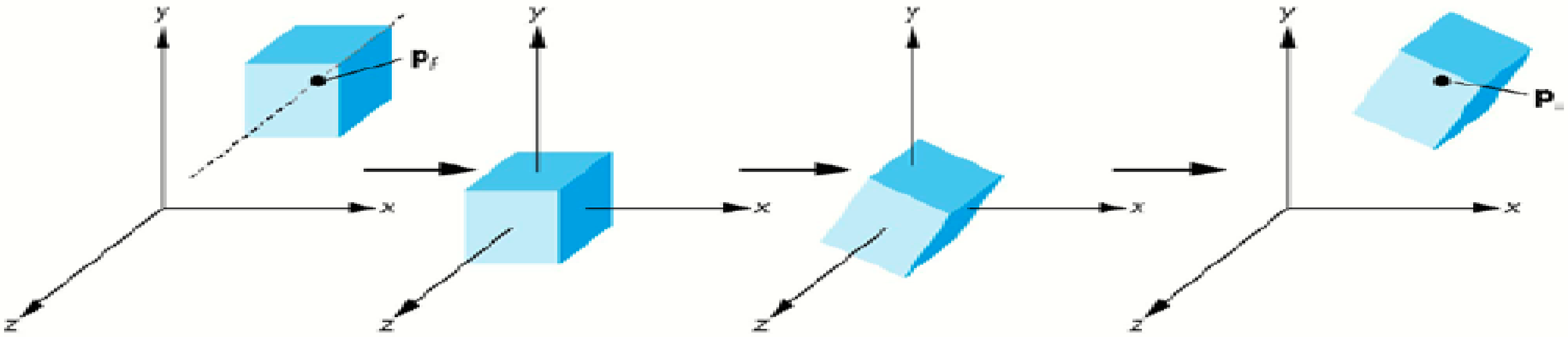
p' = ABCp = A(B(Cp))

Hence composition order really matters.

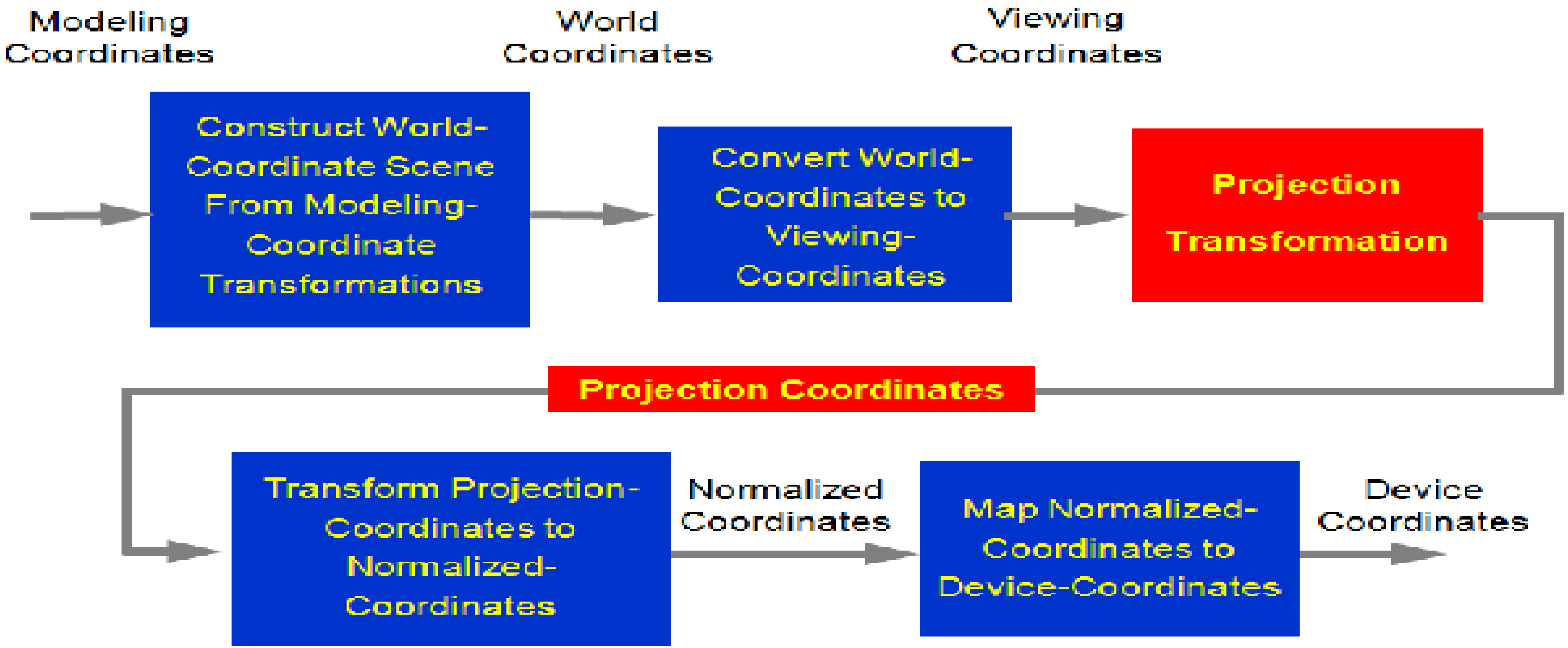
Composite Transformation:

Rotation About Point P

- Move fixed point P to origin
- Rotate by desired angle
- Move fixed point P back
- $M = T(p_f)\ R(q)\ T(-p_f)$



Viewing Pipeline:



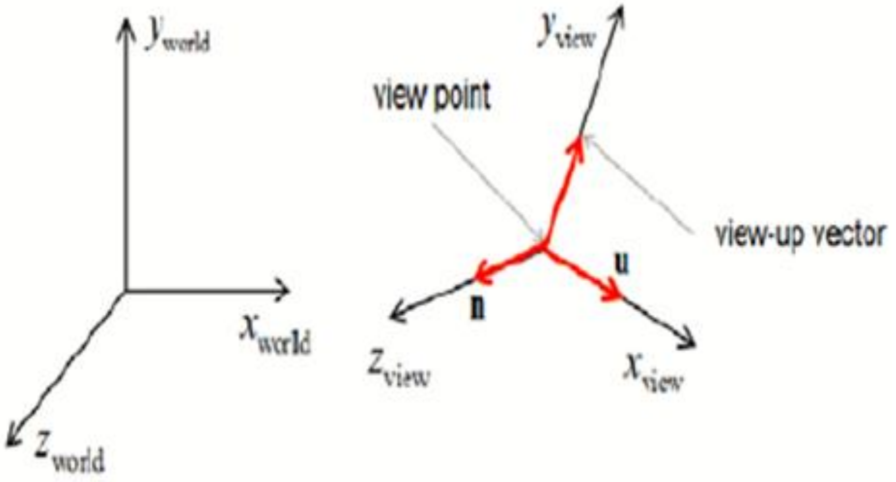
Viewing Pipeline:

Processing Steps

- Once the scene has been model, world coordinates position is converted to viewing coordinates.
- The viewing coordinates system is used in graphics packages as a reference for specifying the observer viewing position and the position of the projection plane.
- Projection operations are performed to convert the viewing coordinate description of the scene to coordinate positions on the projection plane, which will then be mapped to the output device.
- Objects outside the viewing limits are clipped from further consideration, and the remaining objects are processed through visible surface identification and surface rendering procedures to produce the display within the device viewport.

Viewing Coordinates :

- A view of an object in 3D is similar to photographing the object.
- Whatever appears in the viewfinder is projected into the flat film surface.
- The type and the size of the camera lens determineswhich parts of the scene appear in the final picture.



- These ideas incorporated into 3D graphics packages so that views of ascene can be generated, given the spatial position, orientation, and aperture size of the camera

Viewing Coordinates :

Specifying the View Plane

- We choose a particular view for a scene by first establishes the viewing-coordinate system also called view reference coordinate as shown in above diagram.
- A view plane or projection plane is then set up perpendicular to the viewing z_v axis.
- World-coordinate position in the scene are transferred to viewing coordinates, then viewing coordinates are transferred into device coordinate.

$\mathbf{n}$: viewing directionon

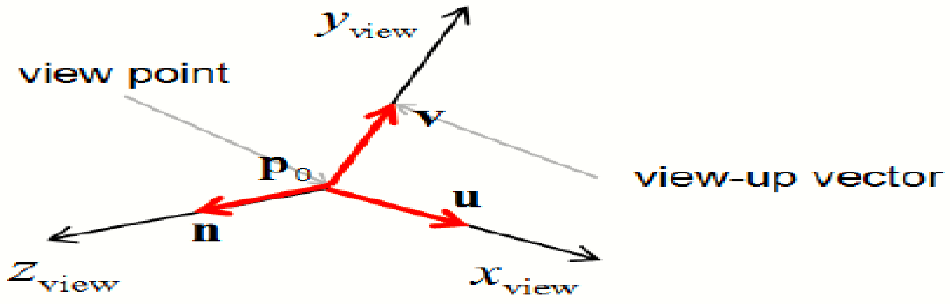
$\mathbf{u}-\mathbf{v}$: viewing plane

$\mathbf{u}=(u_x, u_y, u_z)$

$\mathbf{v}=(v_x, v_y, v_z)$

$\mathbf{n}=(n_x, n_y, n_z)$

$\mathbf{p}_0=(x_0, y_0, z_0)$

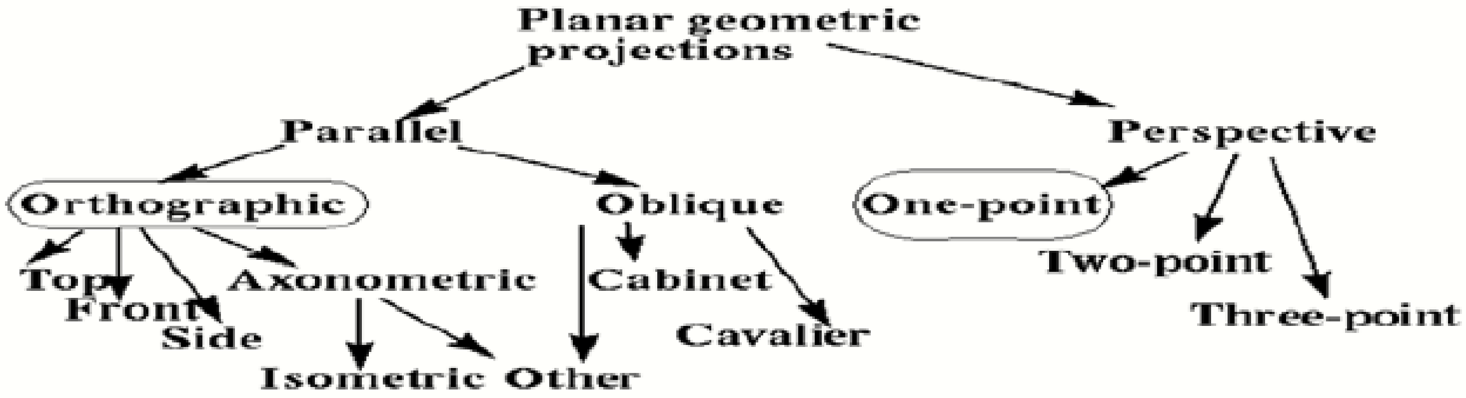


world to viewing transformation

$$\mathbf{M}_{WC,VC}=\underbrace{\begin{bmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ n_x & n_y & n_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{rotation}} \cdot \underbrace{\begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{translation}}$$

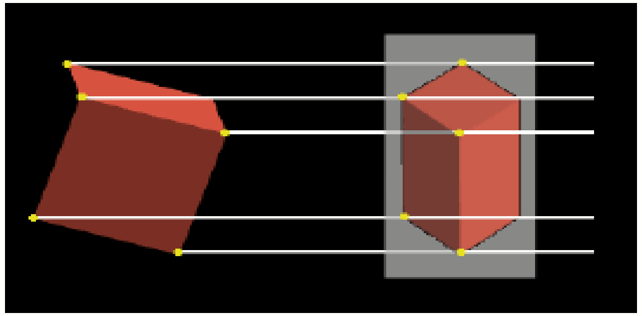
Projection :

- In computer graphics, you can use the virtual camera to generate a picture on the window of the viewing plane, either by using parallel projection or perspective projection. Types of Projection is represented as diagram



Projection : Parallel

- If COP (center of projection) is located at infinity all the projectors are parallel and the result is parallel projection.
- In parallel projection, the polygons are projected on to the viewing plane using parallel lines called projectors. In this case the direction of projection (DOP) needs to be defined.
- Parallel projection offers no sense of depth. As a line of fixed height moves away from the camera it is drawn as the same size.

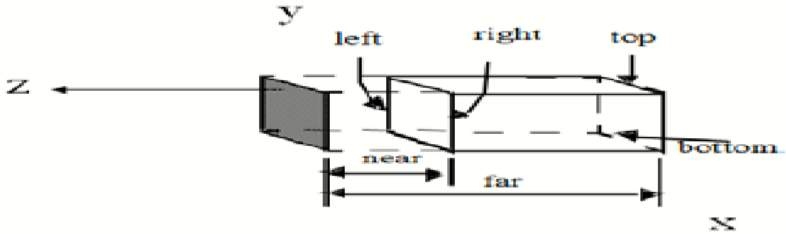


- centre of projection infinitely far from view plane
- projectors will be parallel to each other
- need to define the direction of projection (vector)
- 2 sub-types
  - orthographic - direction of projection is normal to view plane
  - oblique - direction of projection not normal to view plane
- better for drafting / CAD applications

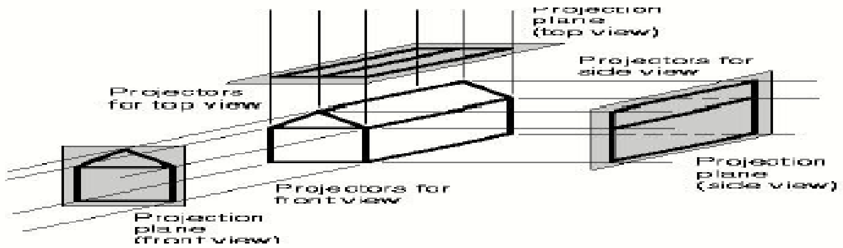
Projection : Parallel

Orthographic Parallel Projections -The direction of projection normal to the projection plane

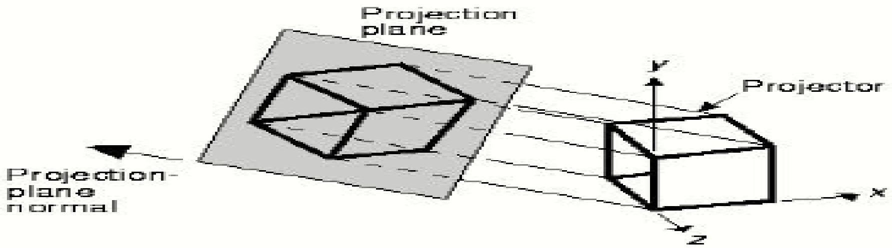
Oblique Parallel Projection-otherwise



Orthographic Parallel Projections

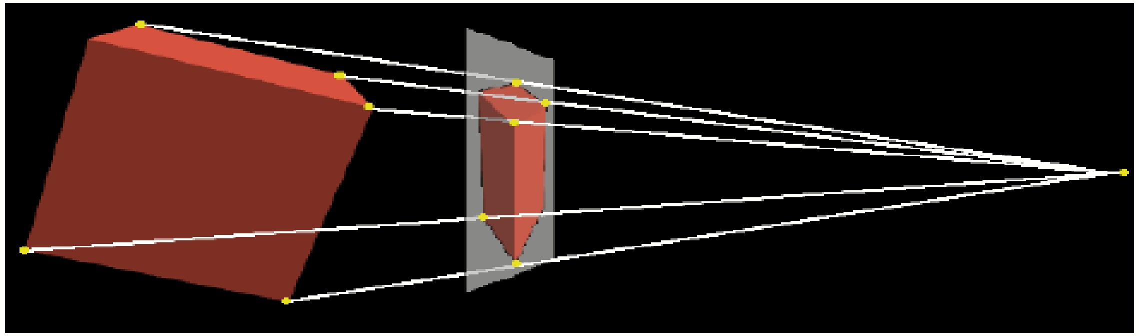


- Projection plane perpendicular to a principle axis
- Most common types
  - Front-elevation
  - Top-elevation (plane-elevation)
  - Side-elevation
- Used in engineering drawing (such as machine parts)
- Hard to deduce 3D nature
- Axonometric orthographic projections
  - Projection plane not normal to a principle axis
  - Several faces of an object can be shown at once
  - Parallelism reserved. distances can be measured



Projection : Perspective

- If COP(centre of projection) is located at finite point in 3 space ,the result is prospective projection.
- In perspective projection, projectors are projected towards a point referred to as the centre of projection or the projection reference point (PRP). When a scene is rendered using perspective projection, a line of a fixed height will be drawn shorter as it moves away from the camera. Therefore this technique allows the viewer to perceive depth in a two dimensional image.



- Vanishing point: the perspective projections of any set of parallel lines that are not parallel to the projection plane converge to a vanishing point (a point at infinity) .
- If a set of lines is parallel to a coordinate axis, the vanishing point is called the principle vanishing point.
- Perspective projections are categorized by their number of principal vanishing points; current graphics projection only deals with the one point projection.


Thanks!