Chapter 4
Graphics in 3 Dimensions

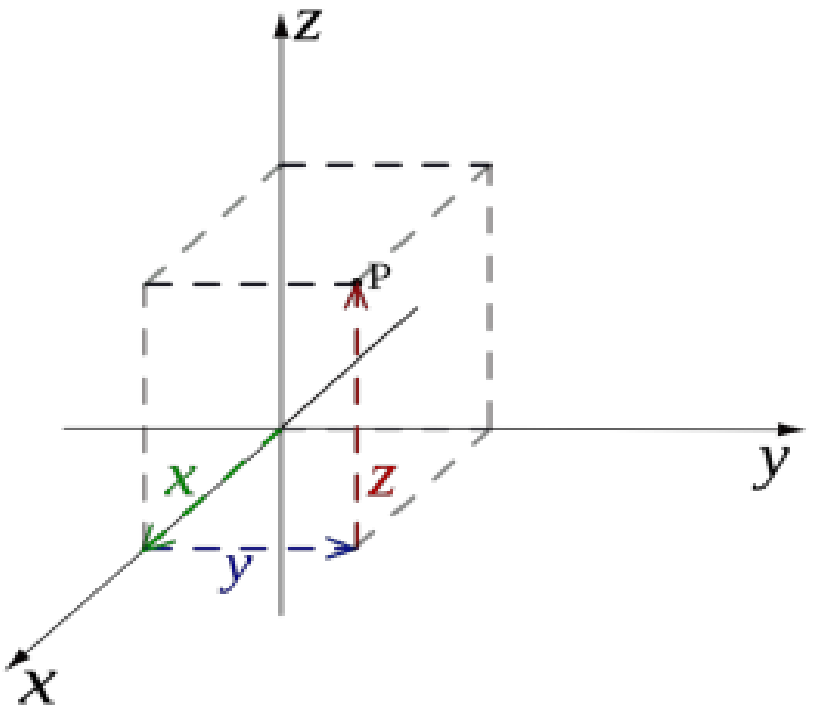
Credit hours: 6 hrs

Contents

4.1 Three Dimensional Coordinate Systems

Three-dimensional Space

- Three-dimensional space is a geometric 3-parameters model of the physical universe in which all known matter exists.
- These three dimensions can be labeled by a combination of length, breadth, and depth.
- Any three directions can be chosen, provided that they do not all lie in the same plane.
- **3 Dimensional Object**
- An object that has height, width and depth, like any object in the real world is a 3 dimensional object.
- **Types of objects:** Trees, terrains, clouds, rocks, glass, hair, furniture, human body, flowers, rubber et



3D Graphics

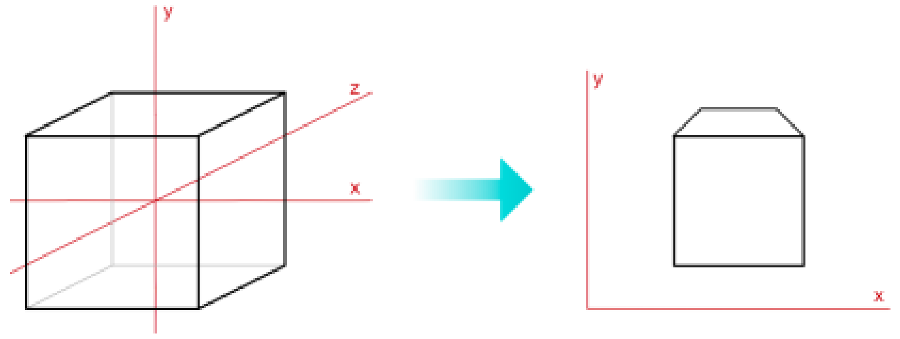
- 3D computer graphics or three-dimensional computer graphics are graphics that **use a three-dimensional representation** of geometric data that is stored in the computer for the purposes of performing calculations and rendering 2D images.
- 2D is "flat", using the horizontal and vertical (X and Y) dimensions, the image has only two dimensions.
- 3D **adds the depth (Z)** dimension.
- This third dimension allows for rotation and visualization from multiple perspectives.
- It is essentially the difference between a photo and a sculpture.
- We can perform different transformation by specifying the three dimensional transformation vector, however the 3D transformation is more complex than 2D transformation.

Issues in 3D that makes it more complex than 2D

- When we model and display a three-dimensional scene, there are many more considerations we must take into account besides just including coordinate values as 2D, some of them are:
  - Relatively more co-ordinate points are necessary.
  - Object boundaries can be constructed with various combinations of plane and curved surfaces.
  - Consideration of projection (dimension change with distance) and transparency.
  - Many consideration on visible surface detection and remove the hidden surfaces.

3D display method

- Three-dimensional viewing coordinates must be transformed onto two dimensional device coordinates to view the scene.
- 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.
- We can then apply lighting and surface-rendering techniques to shade the visible surfaces.



4.2 Three Dimensional Transformations

4.2.1 Translation, Rotation, Scaling, Reflection

3-Dimensional Transformations

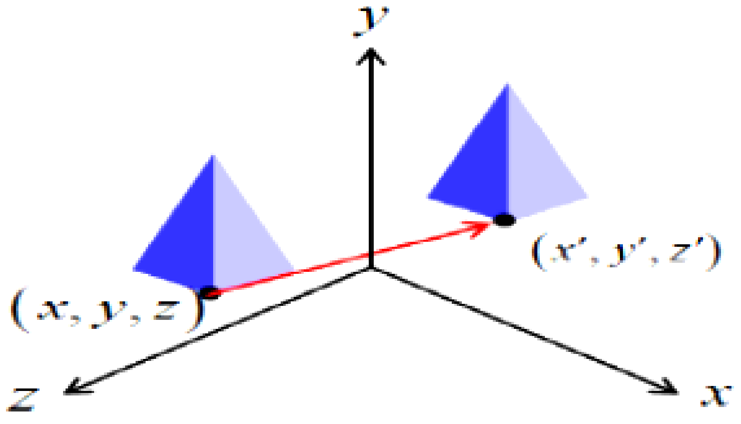
- Methods for geometric transformations and object modeling in three dimensions **are extended from two-dimensional methods by including considerations for the z coordinate.**
- We now translate an object by specifying a three-dimensional translation vector, which determines how much the object is to be moved in each of the three coordinate directions.
- 2D transformations can be represented by 3 x 3 matrices using homogeneous coordinates, so 3D transformations can be represented by 4 x 4 matrices, providing we use homogeneous coordinate representations of points in 2 spaces as well.
- Thus instead of representing a point as (x, y, z), we represent it as (x, y, z, H), where two these quadruples represent the same point it one is a non-zero multiple of the other the quadruple (0, 0, 0, 0) is not allowed.
- A standard representation of a point (x,y, z, H) with H not zero is given by (x/H, y/H, z/H, 1).
- Transforming the point to this form is called homogenizing

- Just as 2D-transfromtion can be represented by 3x3 matrices using homogeneous co-ordinate can be represented by 4x4 matrices, provided we use homogenous co-ordinate representation of points in 3D space as well.
  1. Translation
  2. Rotation
  3. Scaling
  4. Reflection
  5. Shear

a) Translation in 3-D

- Translation in 3D is similar to translation in the 2D except that there is one more direction parallel to the z-axis. If, tx, ty, and tz are used to represent the translation vectors. Then the translation of the position P(x, y, z) into the point P' (x', y', z') is done by
  - x' = x + tx
  - y' = y + ty
  - z' = z + tz

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



$$P' = T.P$$

- In matrix notation using homogeneous coordinate this is performed by the matrix multiplication,

b) Scaling in 3-D

- Equations for 3-D scaling are:

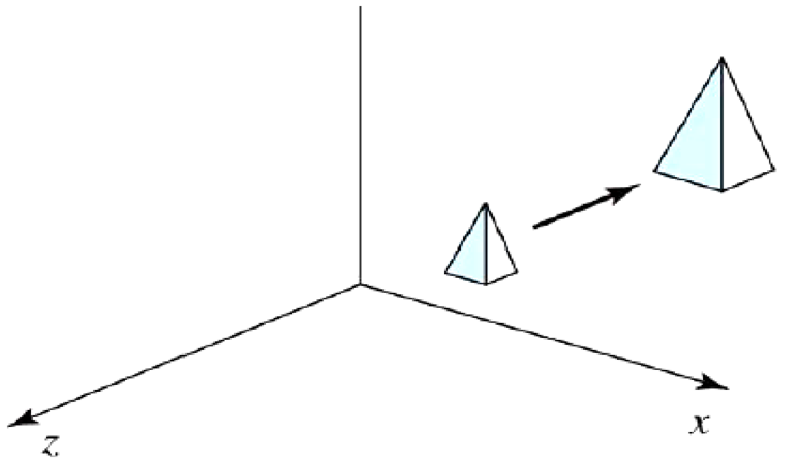
Scaling about origin

X' = X . Sx
Y' = Y . Sy
Z' = Z . Sz

- Matrix expression for scaling transformation of a position P = (x, y, z) relative to the coordinate origin can be written as :

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

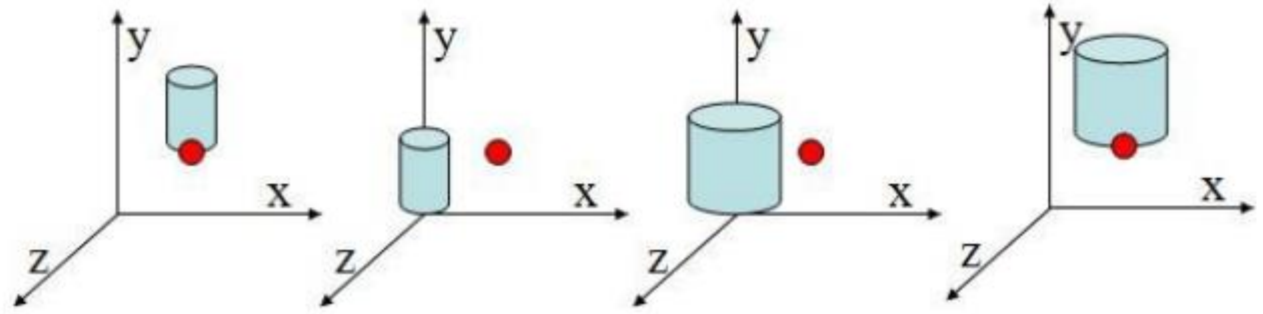
$$\mathbf{P'} = \mathbf{S} \cdot \mathbf{P}$$



- **Scaling with respect to a selected fixed point (x_f , y_f , z_f)**
  1. **Translate** fixed point to the origin
  2. **Scale** object relative to the coordinate origin
  3. **Translate** fixed point **back** to its original position
- i.e $T_{(x,\ y,\ z)} \cdot S_{(x,\ y,\ z)} \cdot T_{(-x,\ -y,\ -z)}$

$$= \begin{bmatrix} 1 & 0 & 0 & x_f \\ 0 & 1 & 0 & y_f \\ 0 & 0 & 1 & z_f \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_f \\ 0 & 1 & 0 & -y_f \\ 0 & 0 & 1 & -z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} s_x & 0 & 0 & (1-s_x)x_f \\ 0 & s_y & 0 & (1-s_y)y_f \\ 0 & 0 & s_z & (1-s_z)z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

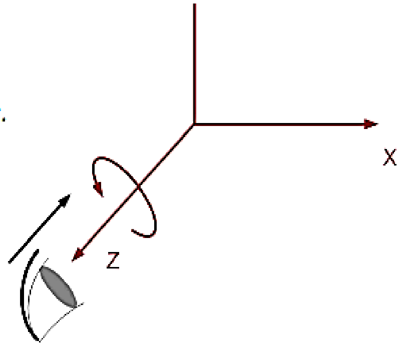


c) Rotation in 3-D

i) **Rotation About z-axis:**

Z-component does not change.

X' = X cosθ – Y sinθ
Y' = X sinθ + Y cosθ
Z' = Z



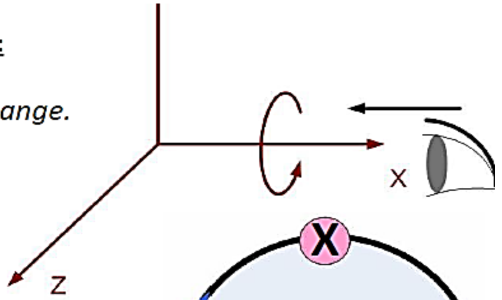
Matrix representation for rotation around z-axis,

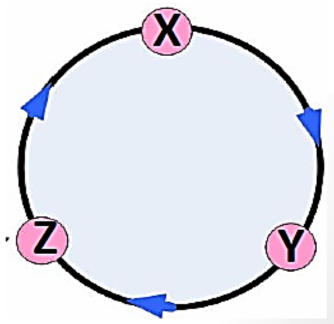
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

ii) **Rotation About x-axis:**

X-component does not change.

Y' = Y cosθ - Z sinθ
Z' = Y sinθ + Z cosθ
X' = X





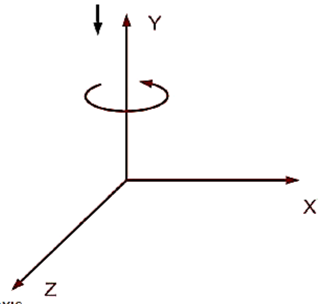
Matrix representation for rotation around X-axis,

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

iii) **Rotation About Y-axis:**

Y-component does not change.

Z' = Z cosθ – X sinθ
X' = Z sinθ + X cosθ
Y' = Y



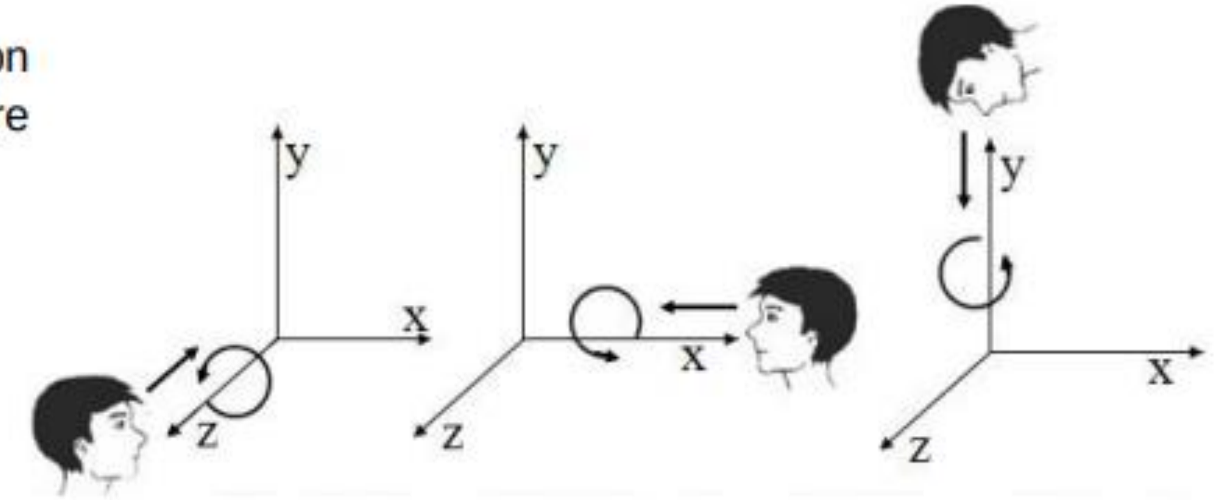
Matrix representation for rotation around Y-axis,

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

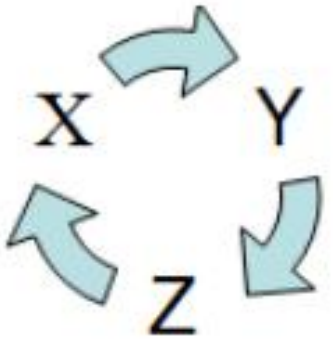
Rotation

To generate a rotation transformation for an object in 3D space, we require the following:

- Angle of rotation.
- Pivot point
- Direction of rotation
- Axis of rotation



Cyclic permutation of the coordinate parameters x, y and z are used to get transformation equations for rotations about the coordinates:



Taking origin as the center of rotation, when a point P(x, y, z) is rotated through an angle about any one of the axes to get the transformed point P'(x', y', z'), we have the following equation for each.

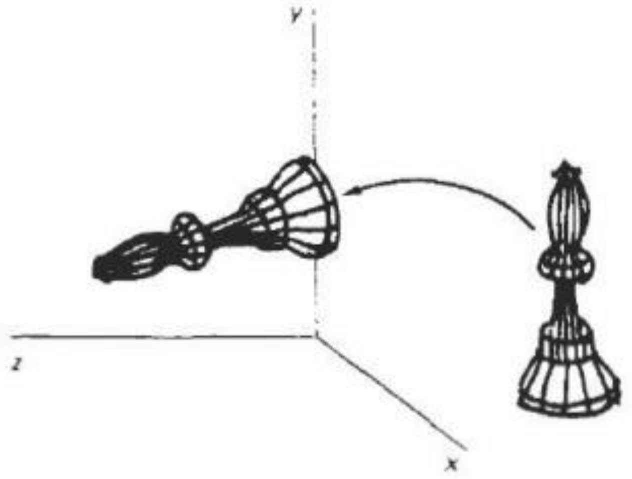
a) **3D x-axis rotation ($P' = R_{x(\theta)} . P$)**

$x' = x$

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

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

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$



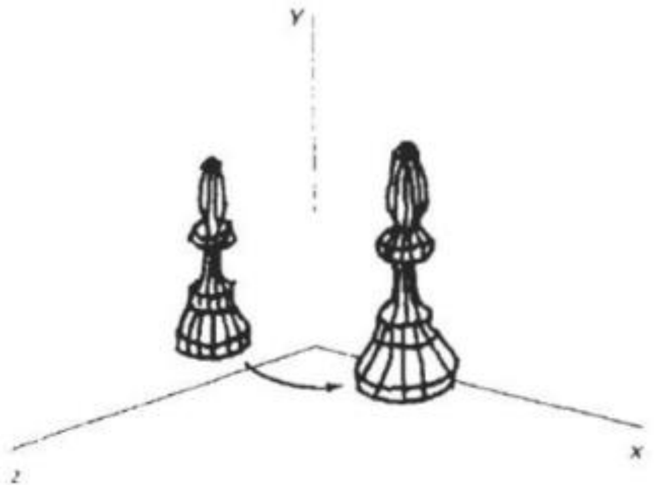
b) **3D y-axis rotation ($P' = R_{y(\theta)} . P$)**

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

$y' = y$

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

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$



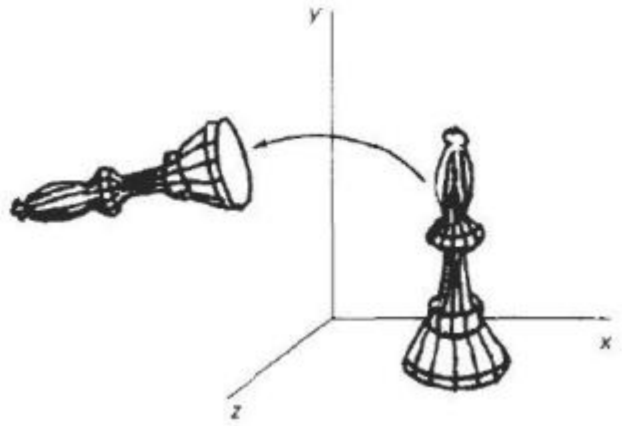
c) **3D z-axis rotation ($P' = R_{z(\theta)} . P$)**

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

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

$z' = z$

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

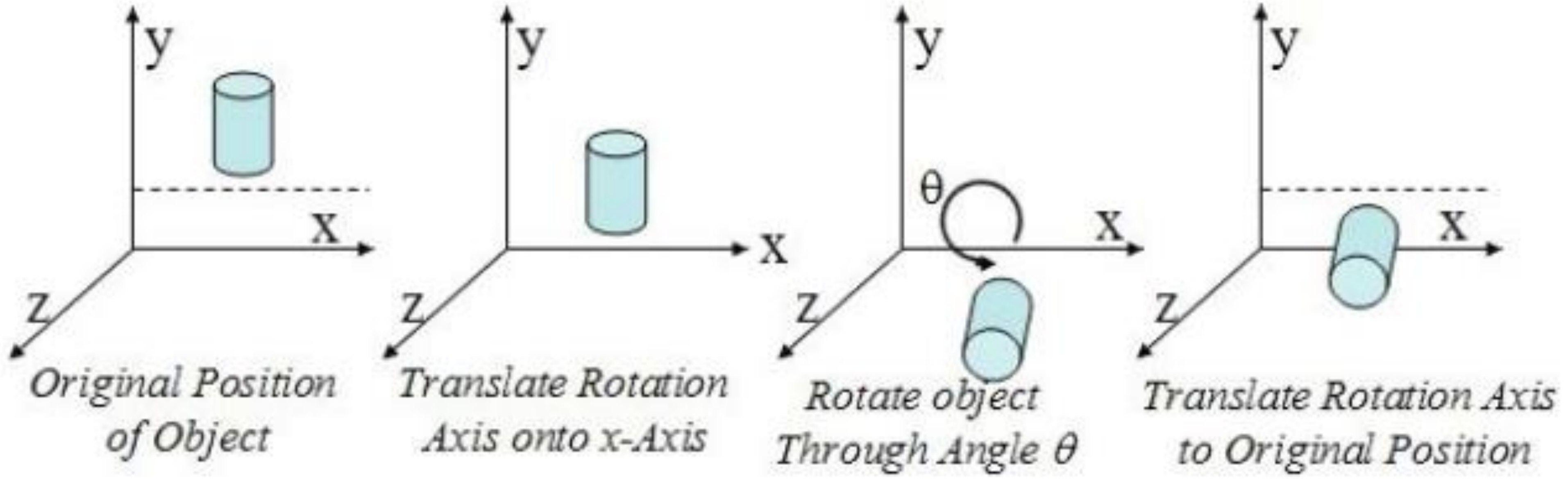


d) **Rotation about an axis parallel to one of the coordinate axes**

Steps:

- Translate object so that rotation axis coincides with the parallel coordinate axis.
- Perform specified rotation about that axis
- Translate object back to its original Position

$$\boxed{P' = [\,T^{-1} \,.\, R_{(\theta)} \,.\, T\,] \,.\, P}$$


y
x
z
Original Position of Object
y
x
z
Translate Rotation Axis onto x-Axis
y
θ
x
z
Rotate object Through Angle θ
y
x
z
Translate Rotation Axis to Original Position

d) Shearing in 3-D

- Shearing transformations are used to modify object shapes.

a) **Z axis Shearing**

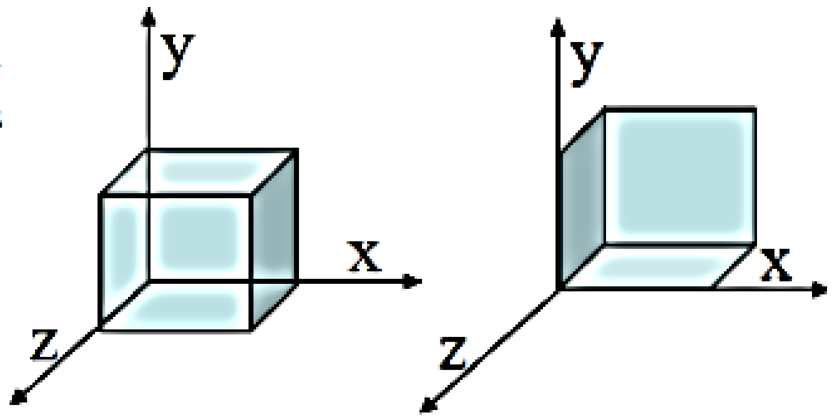
This transformation alters x and y co-ordinate values by amount that is proportional to the z value while leaving z co-ordinate unchanged.

$x' = x + S_{hx}.Z$

$Y' = Y + S_{hy} . Z$

$z' = z$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & S_{hx} & 0 \\ 0 & 1 & S_{hy} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



Parameters S_{hx} and S_{hy} can be assigned any real values

b) **X axis Shearing**

This transformation alters Y and Z coordinate values by an amount that is proportional to the X value while leaving the X value unchanged i.e.

$x' = x$

$y' = y + S_{hy}x$

$z' = z + s_{hz}X$

$$SH_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ S_{hy} & 1 & 0 & 0 \\ S_{hz} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

c) **Y axis shearing**

This transformation alters Y and Z coordinate values by an amount that is proportional to the X value while leaving the X value unchanged i.e.

$x' = x + S_{hx}.y$

$y' = y$

$z' = Z + S_{hz}.\,Y$

$$SH_y = \begin{bmatrix} 1 & S_{hx} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & S_{hz} & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

e) Reflection in 3-D

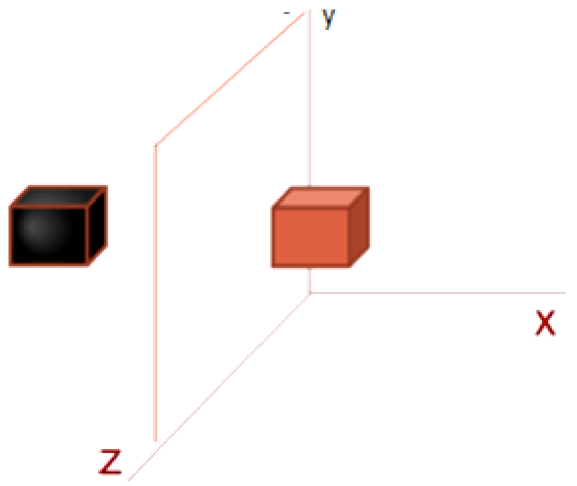
i) Reflection about yz plane

X' = -X

Y' = Y

$\boxed{Z' = Z}$

$$T_{yz} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



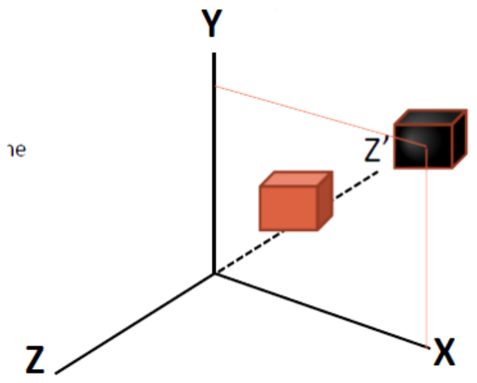
ii) Reflection about XY plane

X' = X

Y' = Y

$\boxed{Z' = -Z}$

$$T_{xy} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



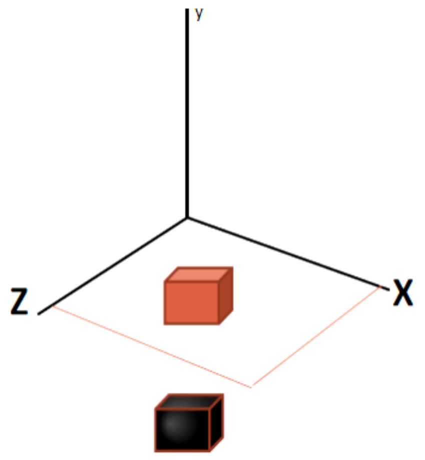
iii) Reflection about XZ plane

X' = X

Y' = -Y

$\boxed{Z' = Z}$

$$T_{xz} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



- In 3D-reflection, the reflection takes place about a plane. The matrix in case of pure reflections, along basic planes, viz. X-Y plane, Y-Z plane and Z-X plane are given below:

a) **X-Y plane**

This transformation changes the sign of the z coordinates, leaving the x and y coordinate values unchanged.

$$RFz = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

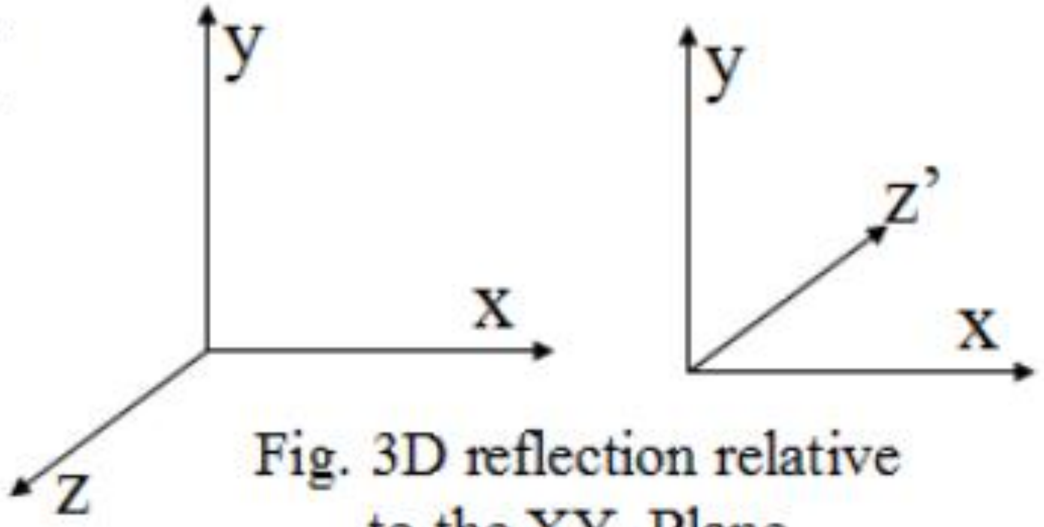


Fig. 3D reflection relative to the XY- Plane

b) **Y-Z plane**

This transformation changes the sign of the x coordinates, leaving the y and z coordinate values unchanged.

$$RFx = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

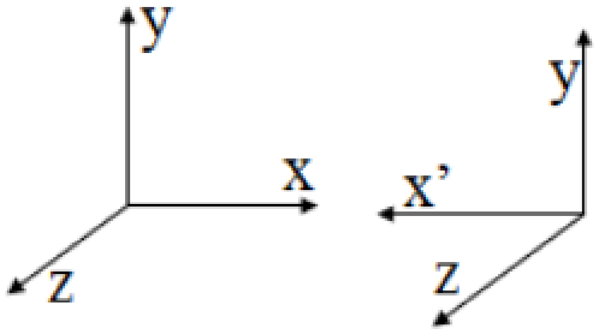


Fig. 3D reflection relative to the YZ- Plane

c) **Z-X plane**

This transformation changes the sign of the y coordinates, leaving the x and z coordinate values unchanged.

$$RFy = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

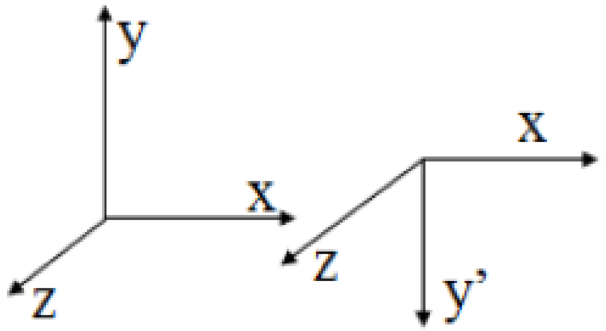


Fig. 3D reflection relative to the ZX- Plane

d) Reflection about any axis parallel to one of the coordinate axes

Steps:

1. Translate object so that rotation axis coincides with the parallel coordinate axis.
2. Perform specified reflection about that axis
3. Translate object back to its original Position

- The transformation is given by:

$$P' = [\, T^{-1} . R . T \,] . P$$

Assignment:

- Perform rotation of a line 10,10,10),(20,20,15) about Y-axis in clockwise direction by 90 degree.
- Reflect the object (0,0,0), (2,3,0) and (5,0,4) about the plane y=4.
- Find the new coordinates of a unit cube 90 degree rotated about an axis defined by its end points A(2,1,0) and B(3,3,1).
- ***Refer book PDF: Insights on Computer Graphics***

4.3 Three Dimensional Representations

4.3.1 Polygon Surfaces

4.3.2 Cubic Spline and Bezier Curve

4.3.3 Non-Planer Surface: Bezier Surface

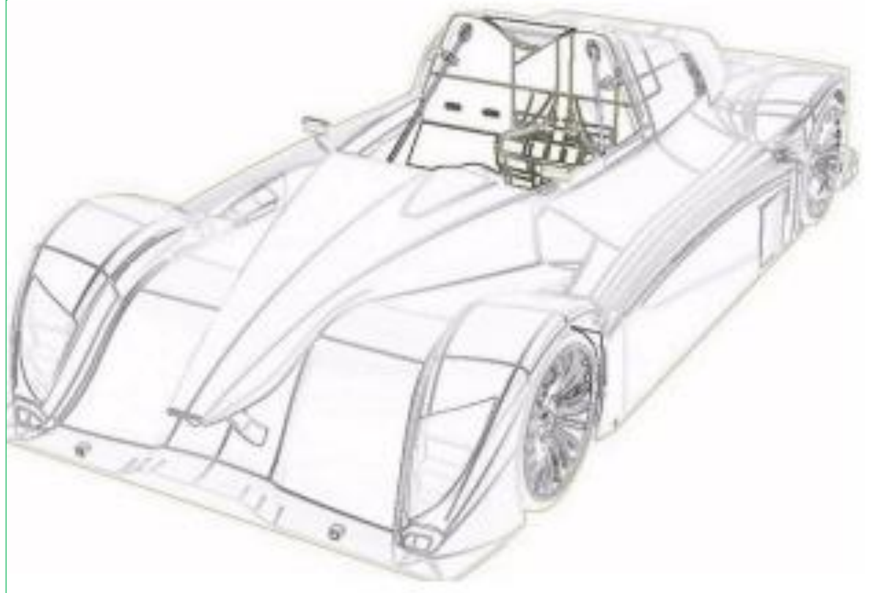
4.3.4 Fractal Geometry Method

Three Dimensional Representations

- Graphics scenes can contain many different kinds of objects like trees, flowers, clouds, rocks, water etc.
- These cannot be describe with only one methods but obviously require large precisions such as polygon & quadratic surfaces, spline surfaces, procedural methods, volume rendering, visualization techniques etc.
- Representation schemes for solid objects are often divided into two broad categories:
    1. Boundary Representations (B-reps)
    2. Space-partitioning representations

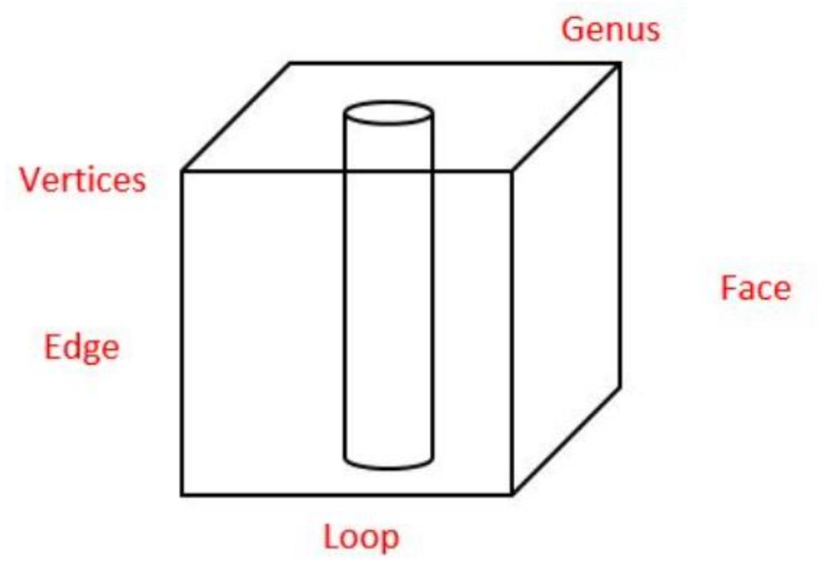
a) Boundary Representation (B-reps)

- BREP is a method for representing shapes (a set of surfaces that separate the object interior from the environment) using the limits.
- A solid is represented as a collection of connected surface elements, the boundary between solid and non-solid.
- For examples: polygon surfaces and spline patches.
- Boundary representation of models are composed of two parts: topology and geometry (surfaces, curves and points).
- The main topological items are: faces, edges and vertices.
  - A face is a bounded portion of a surface;
  - an edge is a bounded piece of a curve and
  - a vertex lies at a point.
- Other elements are the shell (a set of connected faces), the loop (a circuit of edges bounding a face) and loop-edge links (also known as winged edge links or half-edges) which are used to create the edge circuits.
- The edges are like the edges of a table, bounding a surface portion.



a) Boundary Representation (B-reps)

B-rep or boundary representation is a rendering technique for solid modeling. It is a popular approach to create a solid model of a physical object. Brep is that a three-dimensional object model is enclosed by surfaces or faces and has its own interior and exterior. It describes the shape as a collection of surfaces which separate its interior from the external environment. It is suitable for complex designs, Polygon facets are one of the examples of boundary representation.



These are the following primitives of B-rep:-

1) Vertices: It is a point where two or more edges meet with another.

2) Edges: It is a line or curve enclosed between two vertices.

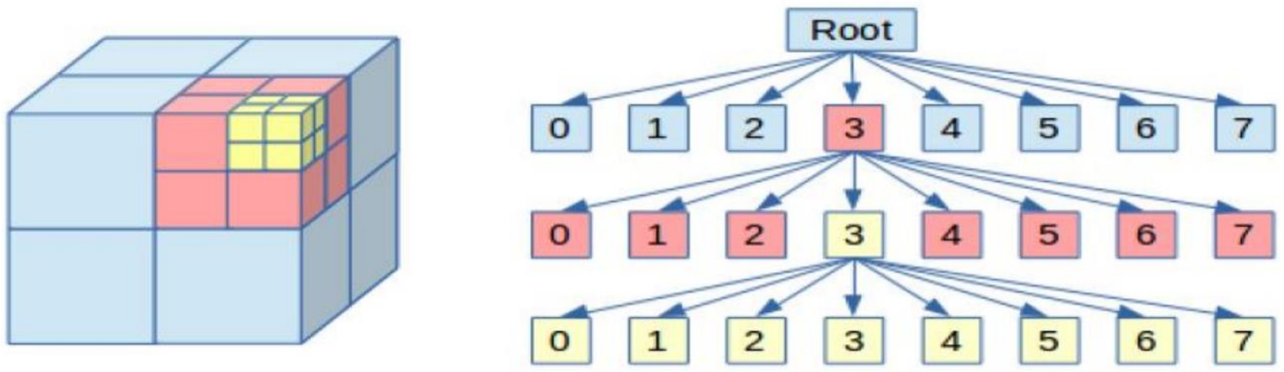
3) Faces: It is a surface or plane of the solid.

4) Loop: It is a hole in a face.

5) Genus: it is through a hole in a solid.

b) Space-partitioning Representation

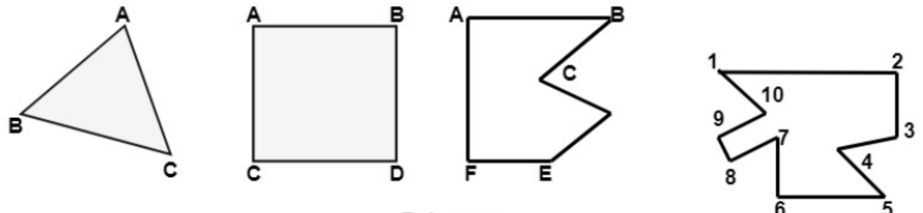
- Space-partitioning representations **are used to describe interior properties**, by partitioning the spatial region containing an object into a set of small, non-overlapping, contiguous solids (usually cubes).
- Space-partitioning systems are often hierarchical, meaning that a space (or a region of space) is divided into several regions, and then the same space-partitioning system is recursively applied to each of the regions thus created.
- The regions can be organized into a tree, called a space-partitioning tree,
- For example: **Octree representation**



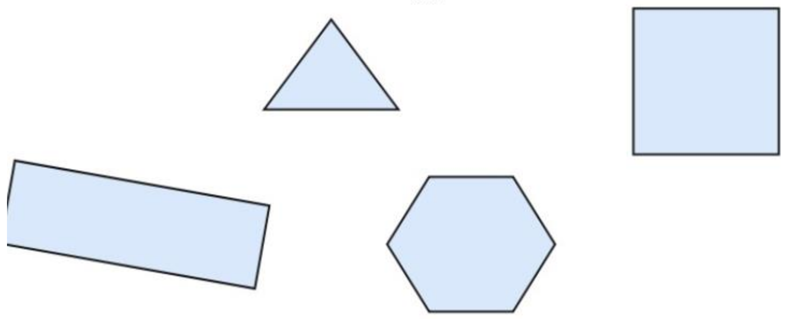
- The visual realism in 3D object can be maintained by maintaining the transparency, projection, lighting & shading effect on each portion. The representation of 3D object include following three stages:
    1. Represent the objects with multiple polygons i.e. wired-frame representation.
    2. Fill all the polygons either with flat filling or smooth filling.
    3. Give the lightning or shading effect and the coloring effect.

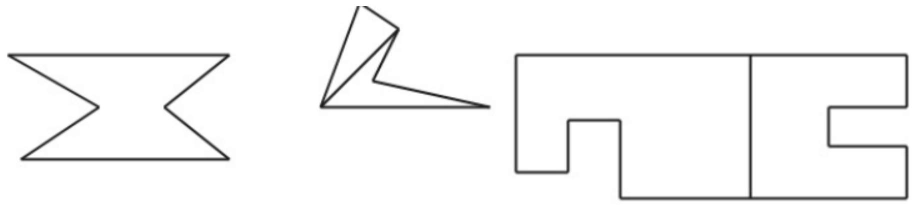
4.3.1 Polygon Surfaces

- **Polygon:** Polygon is a **representation of the surface**. It is primitive which is closed in nature. It is formed using a collection of lines. It is also called as many-sided figure. The lines combined to form polygon are called sides or edges. The lines are obtained by combining two vertices.
- Example of Polygon: Triangle, Rectangle, Hexagon, Pentagon
- A polygon is called **convex** of line joining any two interior points of the polygon lies inside the polygon.
- A non-convex polygon is said to be **concave**. A concave polygon has one interior angle greater than 180°. So that it can be clipped into similar polygons



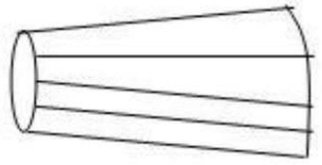
Polygons

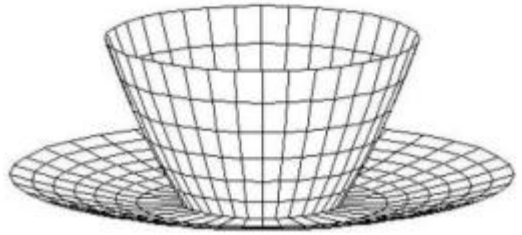

Convex Polygon


Concave Polygon

4.3.1 Polygon Surfaces

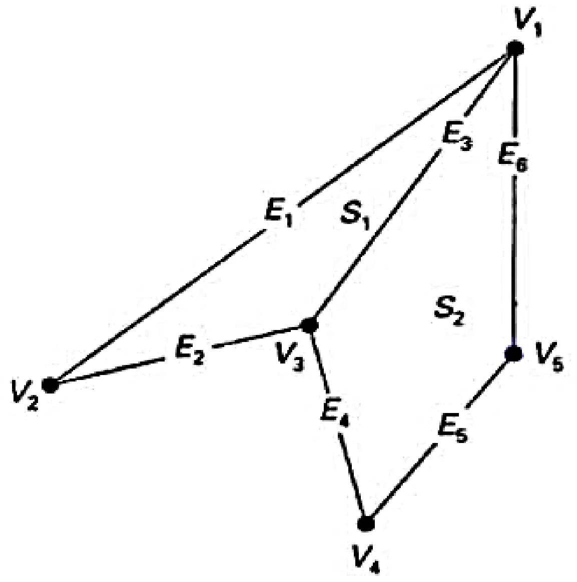
- Polygons are **used in computer graphics to compose images** that are three-dimensional in appearance. Usually (but not always) triangular, polygons arise when an object's surface is modeled, vertices are selected, and the object is rendered.
- The most commonly used boundary representation for a three-dimensional graphics object is a set of surface polygons that enclose the object interior.
- Many graphics systems store all object descriptions as sets of surface polygons.
- This simplifies and speeds up the surface rendering and display of objects, since all surfaces are described with linear equations. For this reason, polygon descriptions are often referred to as "standard graphics objects".
- The wire frame representations are common in design & solid modeling application because they can be displayed quickly to of a give a general indication of the surface structure.
- The polygon surfaces are common in design and solid-modeling applications, since their **wireframe display** can be done quickly to give general indication of surface structure. Then realistic scenes are produced by interpolating shading patterns across polygon surface to illuminate.
- **Generally polygon surfaces are specified using;**
    - **Polygon Table ,**
    - **Plane Equations,**
    - **Polygon Meshes.**


A 3D object represented by polygons



i) Polygon tables

- To specify a polygon surface, **a set of vertex coordinates and associated attribute parameters are placed into tables** & that are used in the subsequent processing, display, error checking and manipulation of the objects in a scene.
- Polygon tables can be organized into two groups:
  1. **Geometric data tables:** Contain vertex coordinates and parameters to identify the spatial orientation of the polygon surfaces.
     - Geometric data consists of three tables:
       - a) **Vertex table:** It **stores co-ordinate values** for each vertex of the object.
       - b) **Edge Table:** The edge table **contains pointers back into the vertex table to identify the vertices** for each polygon edge.
       - c) **Surface table:** And the polygon table contains pointers back into the edge table to **identify the edges for each polygon surface**.



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_6, 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_6
E_6:	V_6, 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 tables:** Provide information for an object and includes parameters specifying the degree of transparency of the object and its surface reflectivity and texture **characteristics**.

ii) Plane equations

- Plane equation method is the **method for representing polygon surface for 3D object**. The information about spatial orientation of object is described by its individual surface which is obtained by vertex coordinate values & equation of each plane gives a description of each surface.
- The equation for plane surface can be expressed in the form of:

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

  where (x,y,z) is any point on plane, & A, B, C, D are constants describing spatial properties of the plane.
- The values of A, B, C, D can be obtained by solving set of 3 plane equations using coordinate values of three non-collinear points on plane.

- Let (x1 ,y1 ,z1), (x2 ,y2 ,z2) , (x3 ,y3 ,z3) are three such points on points on the plane. Then:

  Ax1 + By1 + Cz1 + D = 0 …………… (1)

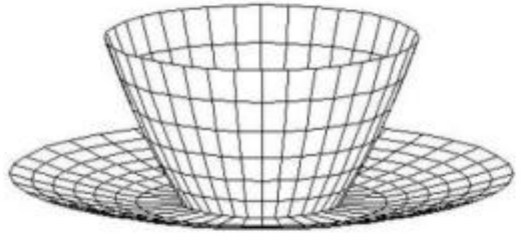
  Ax2 + By2 + Cz2 + D = 0 …………… (2)

  Ax3 + By3 + Cz3 + D = 0 …………… (3)

- The solution of these equations can be obtained in the determinant form using Cramer's rule as:

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

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



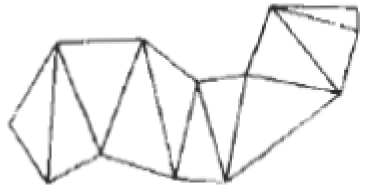
For any point (x,y,z)

if $Ax+By+Cz+D \neq 0$ then (x,y,z) is not on the plane

if $Ax+By+Cz+D < 0,$ the point (x,y,z) is inside plane

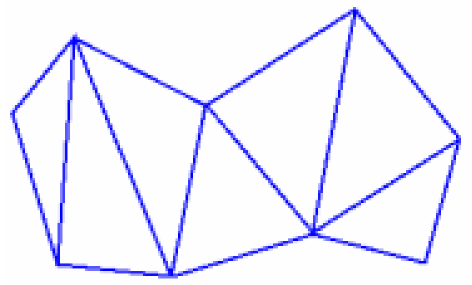
i.e. invisible side

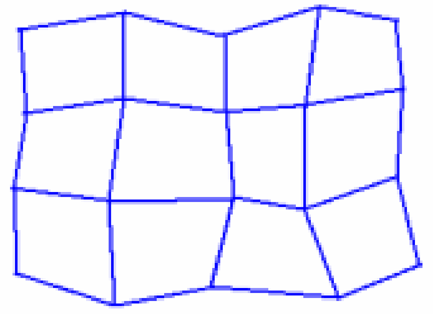
if $Ax+By+Cz+D > 0$ the point lies outside the surface.

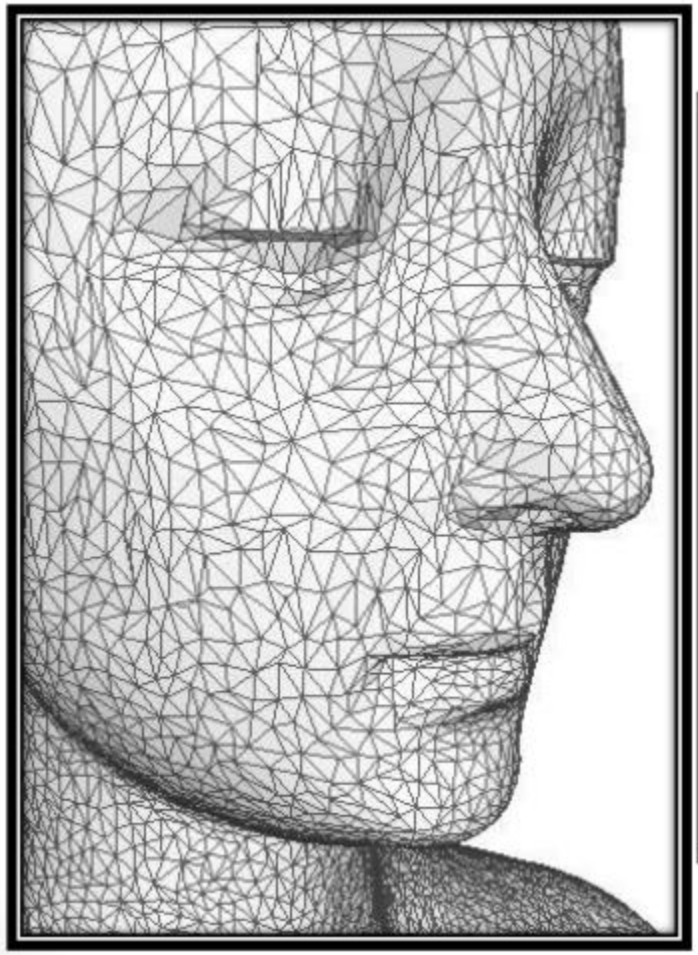
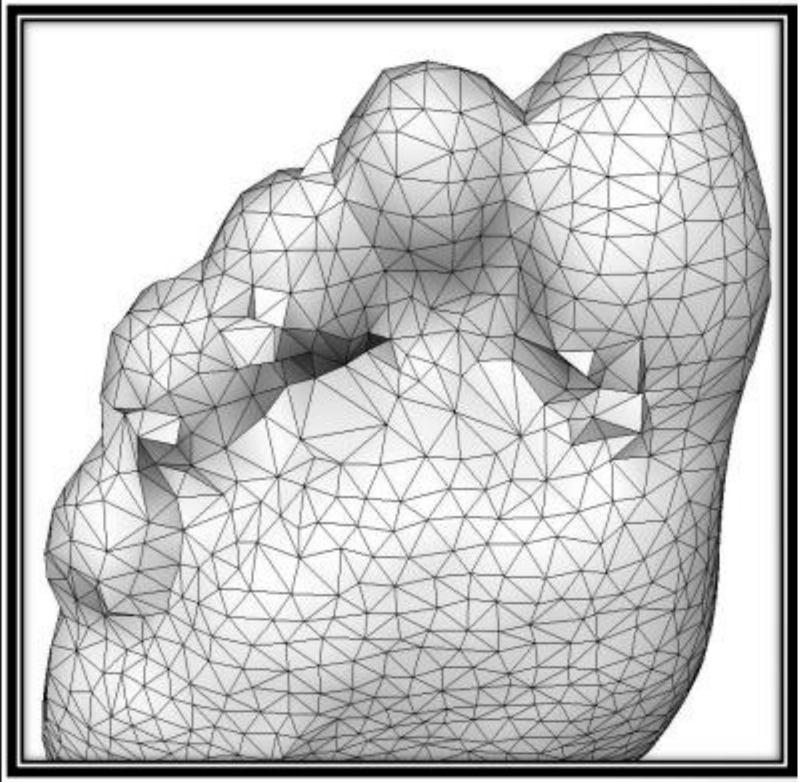
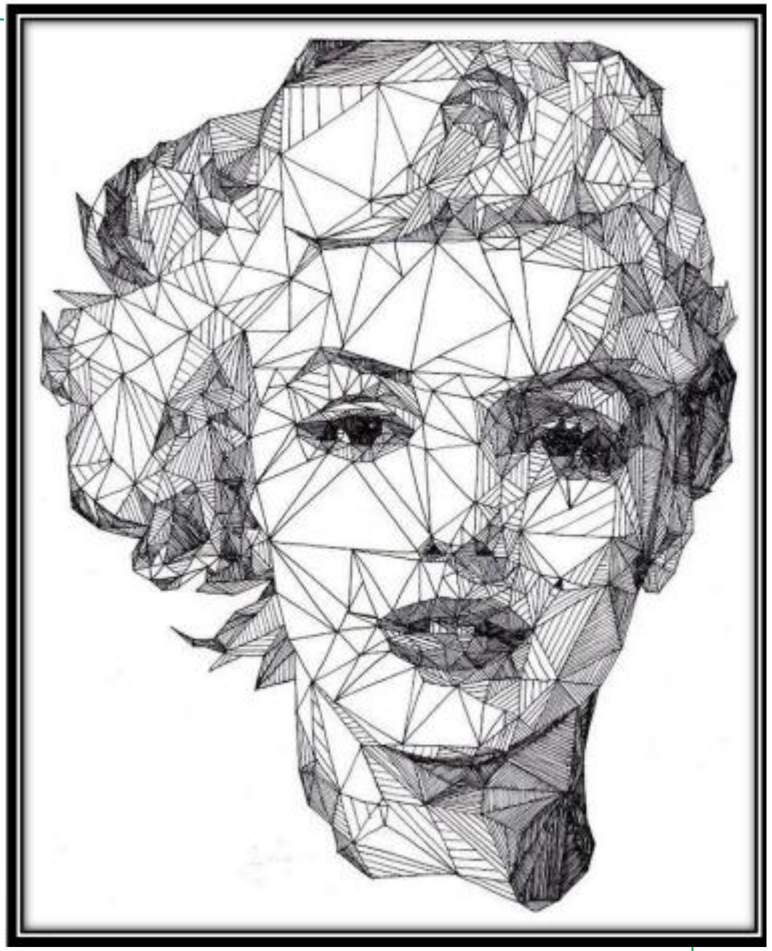


iii) Polygon meshes

- A polygon mesh is **a collection of edges, vertices and polygons** connected such that each edge is shared by at most two polygons & hence bounding the planer surface.
- One type of polygon mesh is the **triangle strip** that produces n–2 connected triangles, given the coordinates for n vertices.
- 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.
- Advantages:
  - It can be used to model almost any object.
  - They are easy to represent as a collection of vertices. They are easy to transform.
  - They are easy to draw on computer screen.
- Disadvantages
  - Curved surfaces can only be approximately described.
  - It is difficult to simulate some type of objects like hair or liquid.



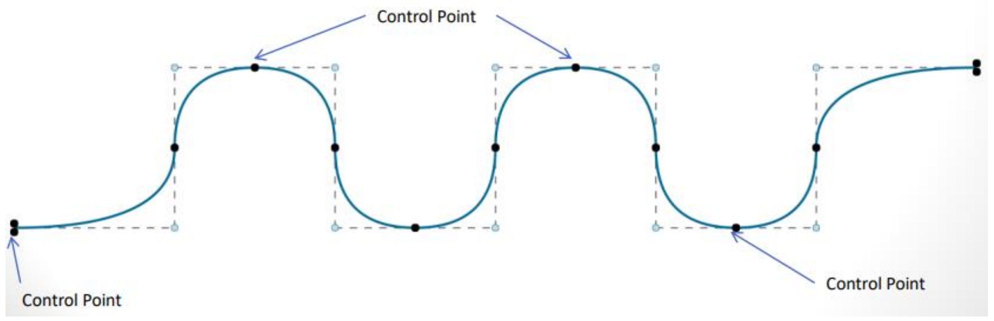


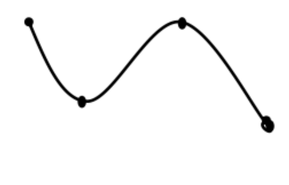




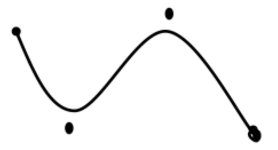
4.3.2 Cubic Spline and Bezier Curve

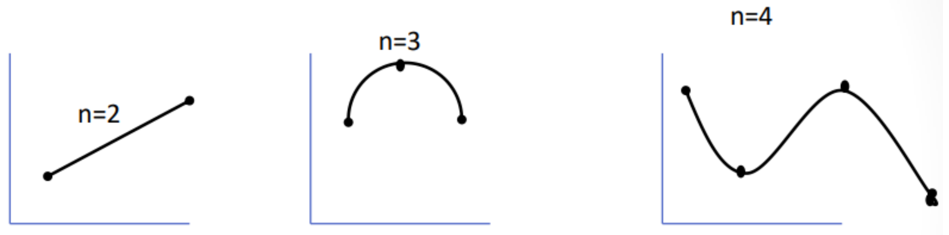
Spline Representation:

- 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.**
- A curve 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.**




Interpolating Curves

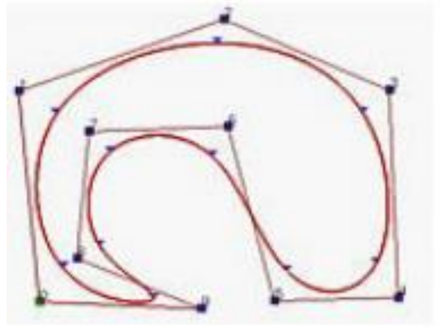

Approximation Curves

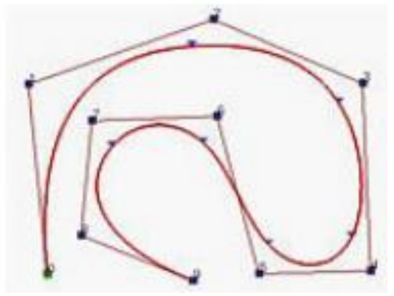


First Order (Linear) Second Order (Quadratic) Third Order (Cubic)

Fig : Interpolating

- **Splines are the smooth curves passing through a set of given points**. By varying the number and position of the lead weights, the shape of the spline is changed so that it passes through some designated set of points.
- In computer graphics, continuous curve that are formed with polynomial section with certain boundary conditions are called spline curve. Splines are of two types:
  - **a)** **Closed Splines:** the generated curve will touch the first and last legs of the control
  - **b)** **Open Splines** : the generated curve will not touch the first and last legs of the control
- A **cubic spline is a spline constructed of piecewise third-order polynomials** which pass through a set m of control points.
- **Splines are used to:**
  - To design curve and surface shapes in graphics applications,
  - To digitize drawings for computer storage
  - To specify animation paths for the objects or image.
  - Typical CAD applications for splines include the design of automobile bodies, aircraft and spacecraft surfaces, and ship hulls.





- We specify a spline curve by giving a set of coordinate positions, called **control points,** which indicates the general shape of the curve These, control points are then fitted with piecewise continuous parametric polynomial functions in one of two ways.
  - **Interpolation curve:** The polynomial sections **are fitted by passing the curve through each control points.** Interpolation curves are commonly used to digitize drawings or to specify animation paths.
  - **Approximation curve:** The polynomials **are fitted to the general control-point path without necessarily passing through any control point.** Approximation curves are primarily used as design tools to structure object surfaces.

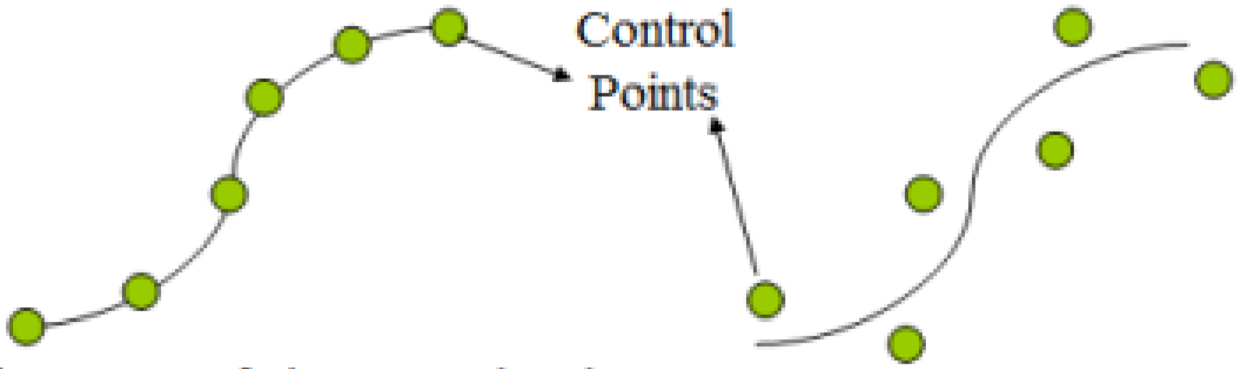


Fig. A set of six control points ***interpolated*** with piecewise continuous polynomial sections

Fig. A set of six control points ***approximated*** with piecewise continuous polynomial sections

Beizer Curve

- A Bezier curve is a mathematically defined curve used in two-dimensional graphic applications.
- The curve is defined by four points: the initial position and the terminating position (which are called "**anchors**") and two separate middle points (which are called "**handles**").
- The shape of a Bezier curve **can be altered by moving the handles.**
- The mathematical method for drawing curves was created by Pierre Bézier in the late 1960's for the manufacturing of automobiles.

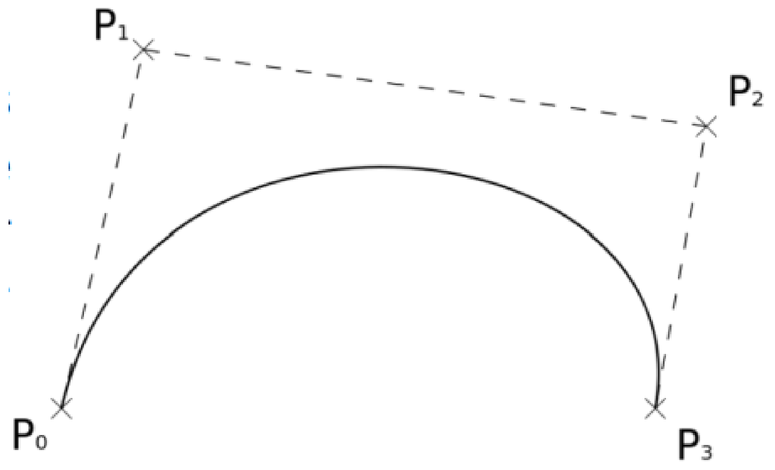
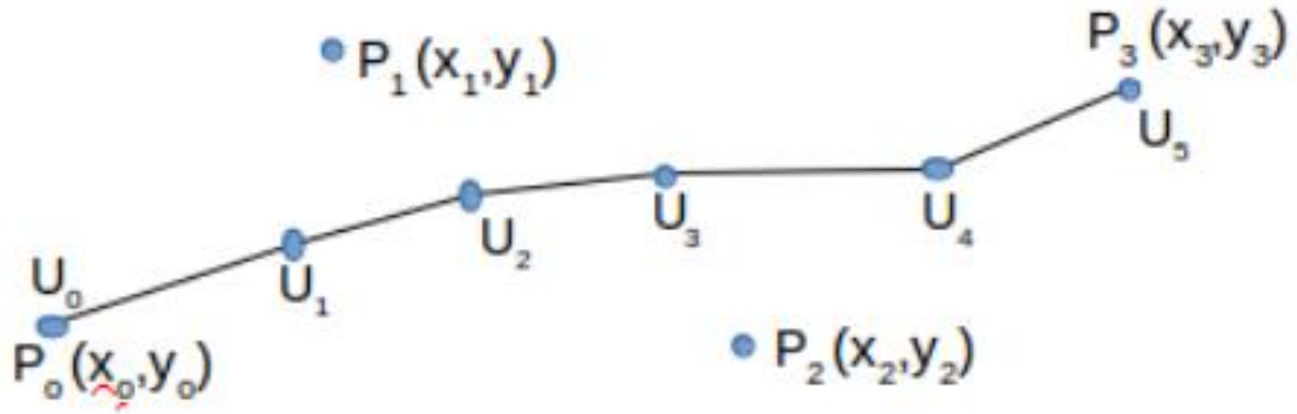
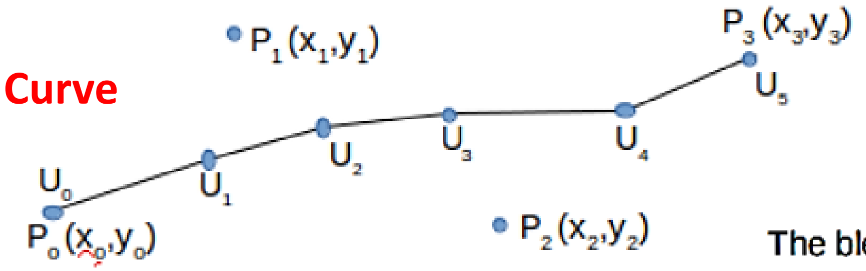


Fig: Bezier curve with four control points

- In general, a Bezier curve can be fitted to any number of control points.
- The number of control points to be approximated and their relative position determine the degree of the Bezier polynomial.
- As with the interpolation splines, a Bezier curve can be specified with boundary conditions, with a characterizing matrix, or with blending functions.
- The Bezier curve has **two important properties:**
  a) It always **passes through the first and last control points.**
  b) It **lies within the convex hull** (convex polynomial boundary) of the control points.



Parametric equation for Bezier Curve (with 4 control points):



Let, P be the control points and N_{seg} is number of segments in a curve segment.

i.e $u = \dfrac{i}{N_{seg}}$; where $i = 0$ to N_{seg} i.e $0 \le u \le 1$

we have,

$$x(u) = \sum_{j=0}^{n} x_j BeZ_{(j,n)}(u) \quad \text{where n= no. of control points.}$$

$$= x_0BeZ_{0,3}(u) + x_1BeZ_{1,3}(u) + x_2BeZ_{2,3}(u)+ x_3BeZ_{3,3}(u)$$

$$y(u) = \sum_{j=0}^{n} y_j BeZ_{(j,n)}(u)$$

$$= y_0BeZ_{0,3}(u) + y_1 BeZ_{1,3}(u) + y_2BeZ_{2,3}(u)+ y_3BeZ_{3,3}(u)$$

We have the parametric equation for Bezier curve is as:

$$Q(u) = P_0 BeZ_{0,3}(u) +P_1 BeZ_{1,3}(u) +P_2BeZ_{2,3}(u) +P_3 BeZ_{3,3}(u) \quad .. (i)$$

The blending function $BeZ_{j,n}(u)$ is defined as a Bernstein polynomial

$$BeZ_{j,n}(u) = \frac{n!}{j!(n-j)!} u^j (1-u)^{n-j}$$

So for n=4 control points,

$$BeZ_{0,3}(u) = \frac{3!}{0!(3-0)!} u^0 (1-u)^{3-0} = (1-u)^3$$

$$BeZ_{1,3}(u) = \frac{3!}{1!(3-1)!} u^1 (1-u)^{3-1} = 3u(1-u)^2$$

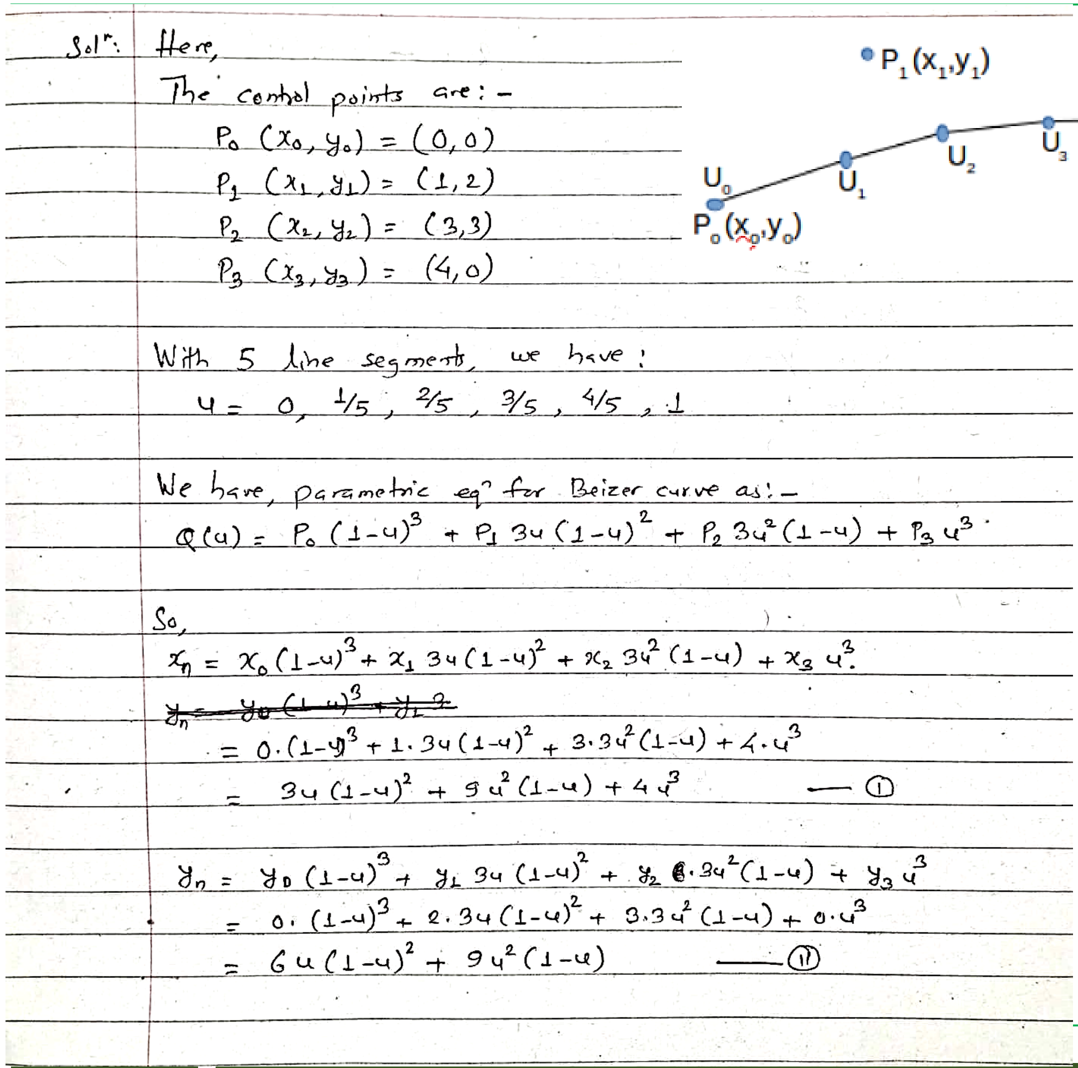
$$BeZ_{2,3}(u) = \frac{3!}{2!(3-2)!} u^2 (1-u)^{3-2} = 3u^2(1-u)$$

$$BeZ_{3,3}(u) = \frac{3!}{3!(3-3)!} u^3 (1-u)^{3-3} = u^3$$

Substituting these values in equation (i), the parametric equation for Bezier curve is:

$$Q(u) = P_0 (1-u)^3 +P_1 3u(1-u)^2 +P_2 3u^2(1-u) +P_3 u^3$$

Calculate the co-ordinates of Bezier curve described by 4 control points $P_0(0,0)$, $P_1(1,2)$, $P_2(3,3)$, $P_3(4,0)$ and approximate with 5 lines segments.



Sol^n: Here,

The control points are:-

$P_0\ (x_0, y_0) = (0,0)$

$P_1\ (x_1, y_1) = (1,2)$

$P_2\ (x_2, y_2) = (3,3)$

$P_3\ (x_3, y_3) = (4,0)$

With 5 line segments, we have:

$u = 0,\ 1/5,\ 2/5,\ 3/5,\ 4/5,\ 1$

We have, parametric eq^n for Beizer curve as:-

$Q(u) = P_0(1-u)^3 + P_1\, 3u(1-u)^2 + P_2\, 3u^2(1-u) + P_3 u^3.$

So,

$x_n = x_0(1-u)^3 + x_1\, 3u(1-u)^2 + x_2\, 3u^2(1-u) + x_3 u^3.$

~~$y_n = y_0(1-u)^3 + y_1 2$~~

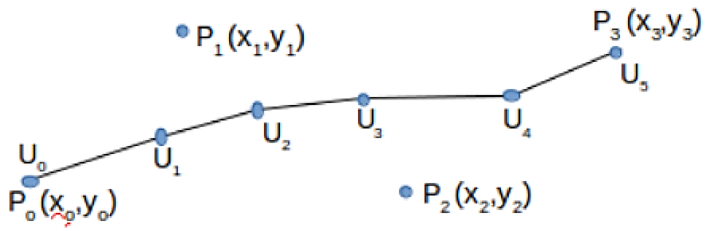
$= 0\cdot(1-u)^3 + 1\cdot 3u(1-u)^2 + 3\cdot 3u^2(1-u) + 4\cdot u^3$

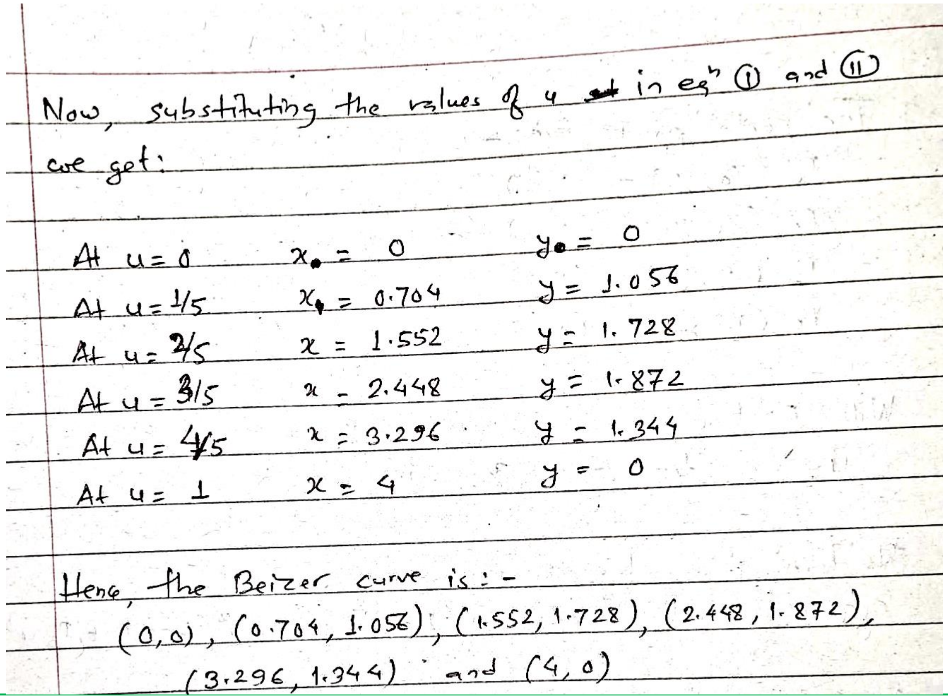
$= 3u(1-u)^2 + 9u^2(1-u) + 4u^3$ —— ①

$y_n = y_0(1-u)^3 + y_1\, 3u(1-u)^2 + y_2\, 3u^2(1-u) + y_3 u^3$

$= 0\cdot(1-u)^3 + 2\cdot 3u(1-u)^2 + 3\cdot 3u^2(1-u) + 0\cdot u^3$

$= 6u(1-u)^2 + 9u^2(1-u)$ —— ⑪





Now, substituting the values of u ~~at~~ in eq^n ① and ⑪ we get:

At $u = 0$	$x_0 = 0$	$y_0 = 0$
At $u = 1/5$	$x_1 = 0.704$	$y = 1.056$
At $u = 2/5$	$x = 1.552$	$y = 1.728$
At $u = 3/5$	$x = 2.448$	$y = 1.872$
At $u = 4/5$	$x = 3.296$	$y = 1.344$
At $u = 1$	$x = 4$	$y = 0$

Hence, the Beizer curve is:-

(0,0), (0.704, 1.056), (1.552, 1.728), (2.448, 1.872), (3.296, 1.344) and (4,0)

Assignment:

- The coordinates of four control points relative to a curve are given by P1(2,2,0), P2(2,3,0), P3(3,3,0) and P4(3,2,0). Write the equation of Bezier curve. Also find the coordinate pixels for the curve for u=0, 1/4, 1/2, 3/4, 1. Also plot Bezier curve on graph.

  Refer: https://www.youtube.com/watch?v=7zi1IafiFpo

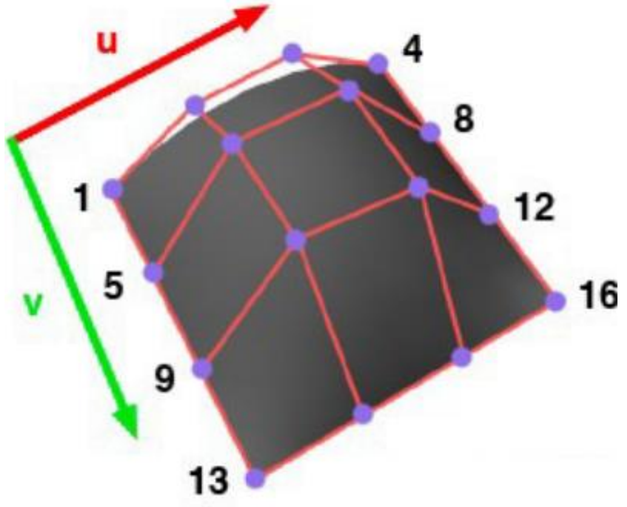
4.3.3 Non-Planer Surface: Bezier Surface

- Like Bezier curve, Bezier surface is defined by a set of control points.
- Similar to interpolation in many respects, a key difference is that the surface does not pass through the central control points; rather, it is "stretched" toward them as though each were an attractive force.
- A two-dimensional Bezier surface can be defined as a parametric surface where the position of a point p as a function of the parametric coordinates u, v is given by:

$$\mathbf{p}(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^n(u)\, B_j^m(v)\, \mathbf{k}_{i,j}$$

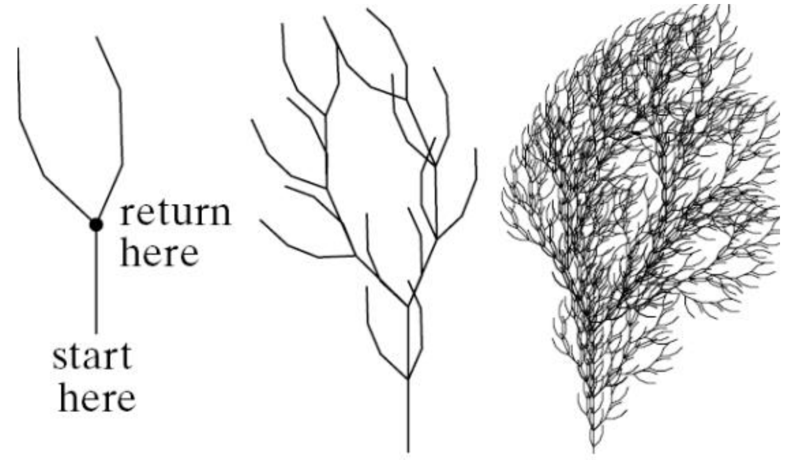
- evaluated over the unit square, where

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



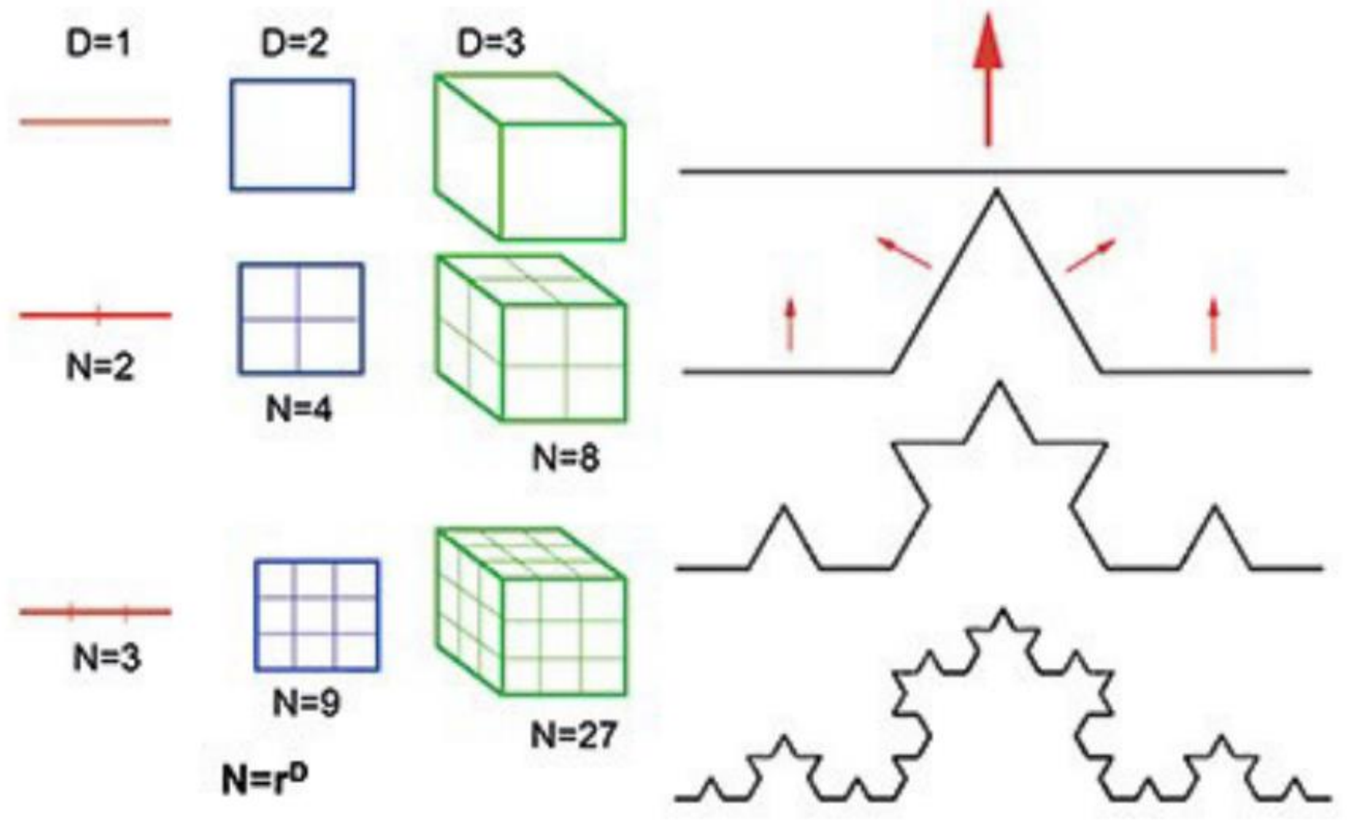
4.3.4 Fractal Geometry Method

- A fractal **is a mathematical set that typically displays self-similar patterns** or recursive operations, which means it is "the same from near as from far".
- Natural objects, such as mountains and clouds, have irregular or fragmented features, and these are described with fractal geometry methods.



- Fractional object describes and explains the features of natural phenomena with procedures or function in various dimensions that theoretically repeat an infinite number of times but finite number of steps for graphics display.
- For fractional-Generation procedure, If $P_0 = (x_0 , y_0 , z_0)$ is a selected initial point, each iteration of a transformation function F generates successive levels of detail with the calculations are:
  - $P_1 = F(P_0)$,
  - $P_2 = F(P_2)$, …

- We can describe the amount of variation in the object detail with a number called the fractal dimension

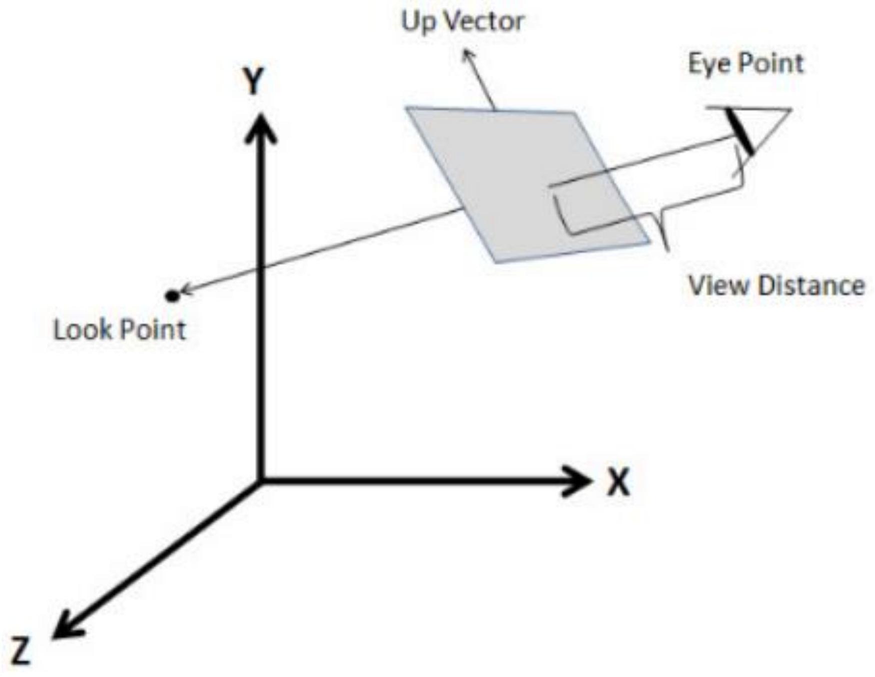


4.4 Three Dimensional Viewing Transformation and Pipeline

3D Viewing Transformation

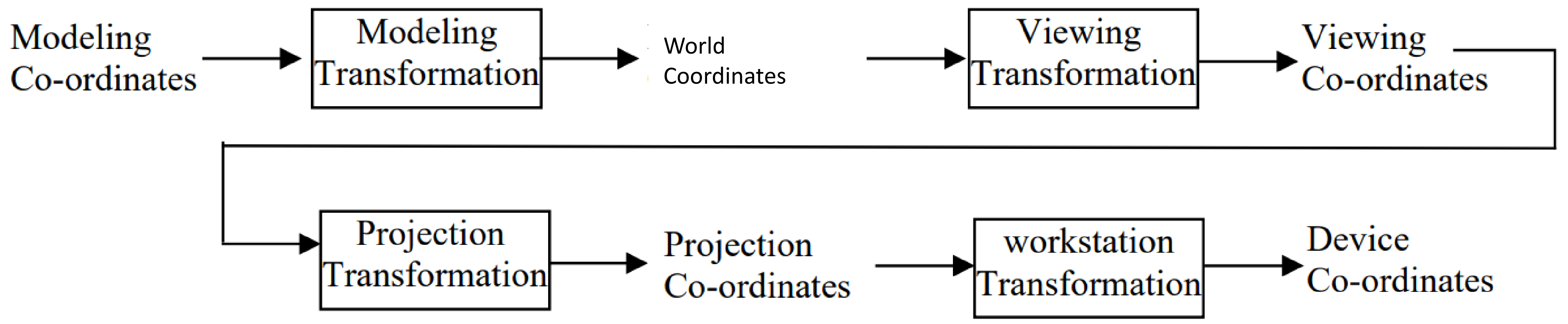
- The basic idea of the 3D viewing transformation is **similar to the 2D** viewing transformation. That is, a viewing window is defined in world space that specifies how the viewer is viewing the scene.
- A corresponding view port is defined in screen space, and a mapping is defined **to transform points from world space to screen space based** on these specifications.
- The view port portion of the transformation is the same as the 2D case. Specification of the window, however, requires additional information and results in a more complex mapping to be defined.
- Defining a viewing window in world space coordinates is exactly like it sounds; sufficient information needs to be provided to define a rectangular window at some location and orientation.
- The usual viewing parameters that are specified are:

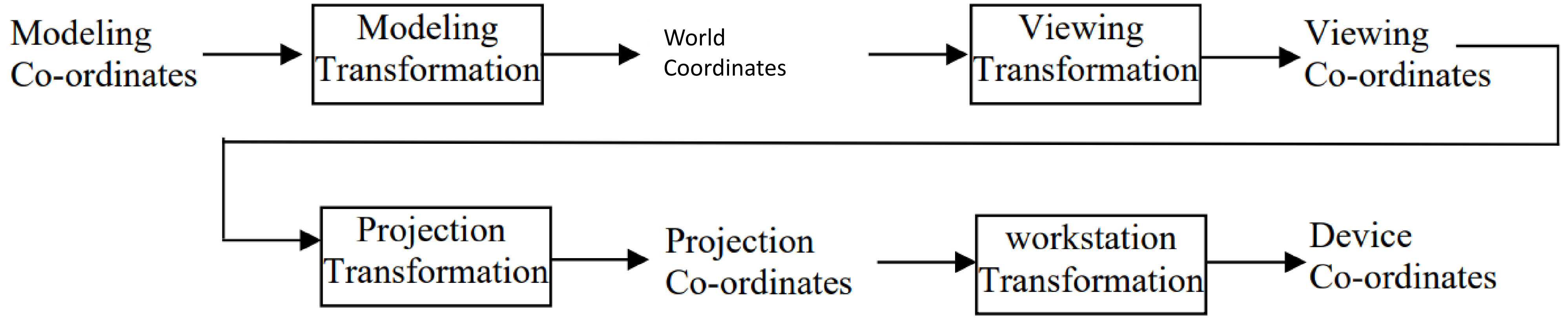
- The usual viewing parameters that are specified are:
  - **Eye Point:** The position of the viewer in world space
  - **Look Point:** The point that the eye is looking at
  - **View Distance:** The distance that the window is from the eye
  - **Window Size:** The height and width of the window in world space coordinates Up Vector which direction represents "up" to the viewer, this parameter is sometimes specified as an angle of rotation about the viewing axis.



3D Viewing Pipeline

- The following figure shows general processing steps for modeling and converting a world coordinate description of a scene to device coordinates.





- Once the scene has been modeled, **world-coordinate positions are converted to viewing coordinates.**
- The **viewing-coordinate system is used** in graphics packages as a **reference** for specifying the observer viewing position and the position of the projection plane, which we can think of in analogy with the camera film plane.
- Next, 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 specified 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 view port.**

4.5 Projections

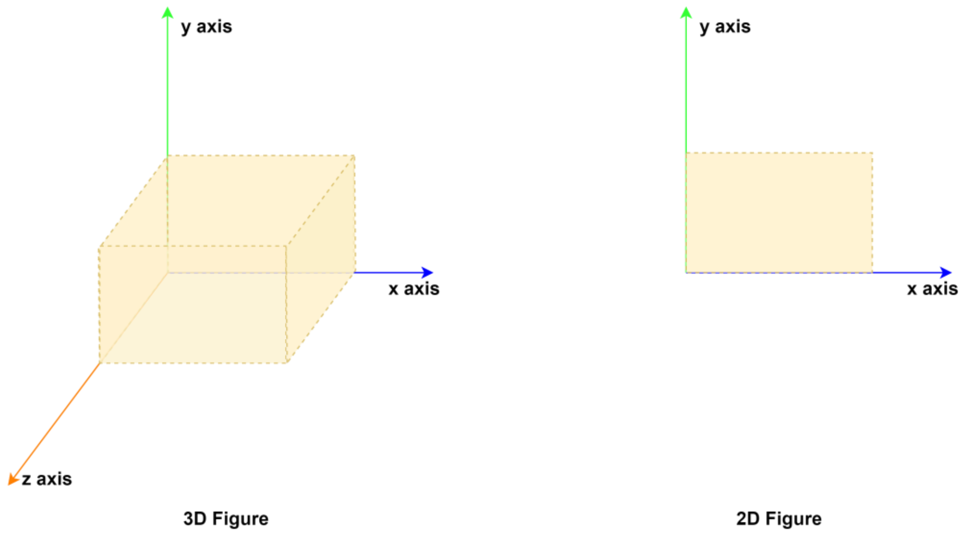
4.5.1 Parallel Projection

4.5.2 Perspective Projection

Projection

- 3D projection is any method of mapping three-dimensional points to a two-dimensional plane.
- In general, a projection transforms a N-dimension points to N-1 dimensions.
- Ideally, an object is projected by projecting each of its endpoints.
- But an object has infinite number of points. So we cannot project all those points.
- What we do is that we project only the corner points of an object on a 2D plane and we will join these projected points by a straight line in a 2D plane.
- **View Plane or Projection Plane:**

Two dimensional plane in which 3D objects are projected is called the view plane or projection plane. Simply it is a display plane on an output device



Types of Projection

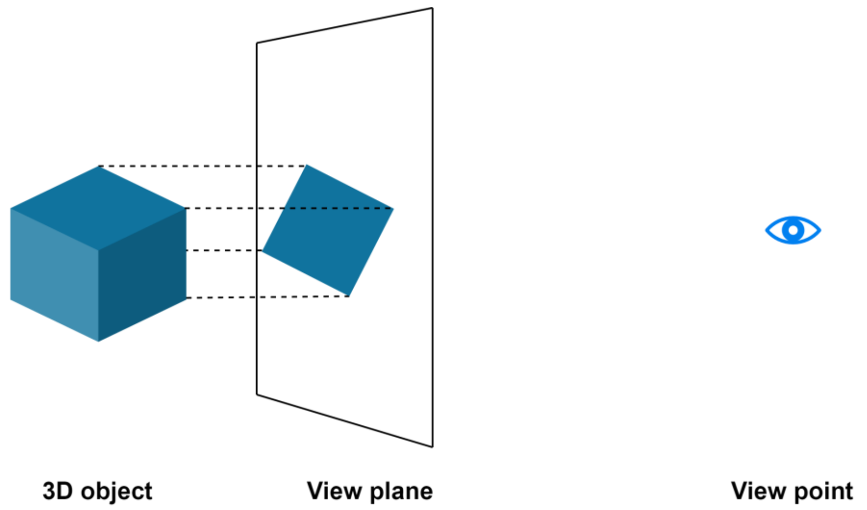
Types of Projection

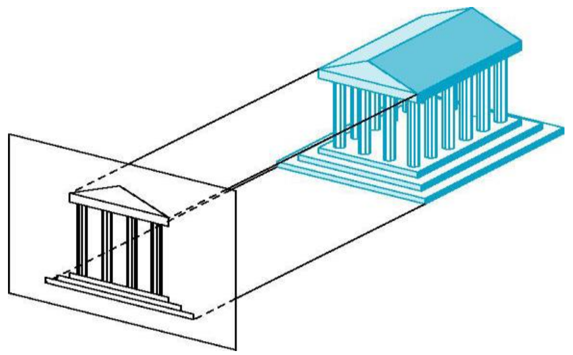
1. **Parallel Projection**
    a)Orthographic parallel projection
    b)Oblique parallel projection

2. **Perspective Projection**

i) Parallel Projection

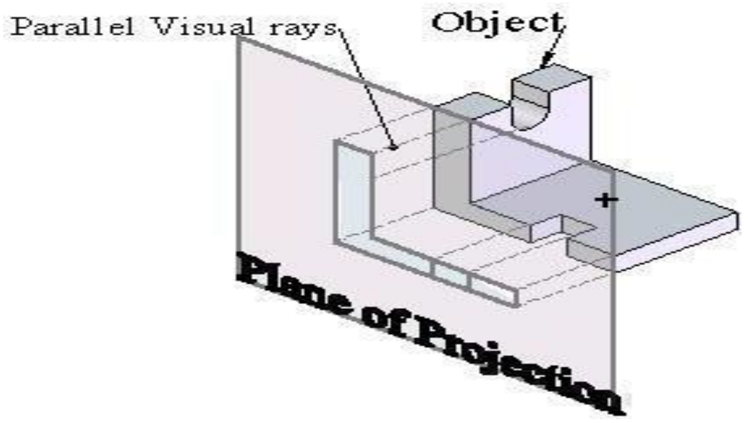
- In this projection, the co-ordinate positions are **transformed to the view plane along parallel lines.**
- the object in the real world is projected onto a plane exactly parallel to the 3D object.
- Preserves relative proportions of objects so that accurate views of various sides of an object are obtained but doesn't give realistic representation of the 3D object.
- Can be used for exact measurements so parallel lines remain parallel
- There are **two different types** of parallel projections
  - a) Orthographic parallel projection
  - b) Oblique parallel Projection

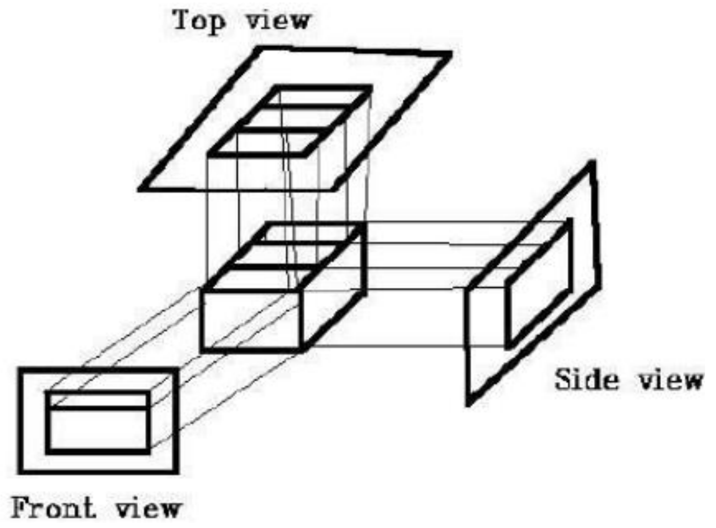




a) Orthographic Parallel Projection

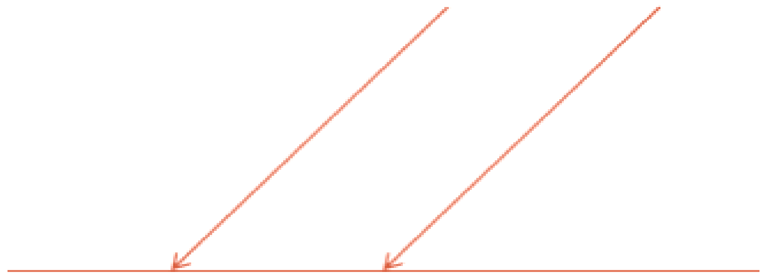
- In orthographic projection, the center of projection lies at infinity and the **projections are perpendicular** to the view plane.
- So, a **true size** and shape of a single face of object is obtained.
- In orthographic projection the direction of projection is normal to the projection of the plane.
- There are three types of orthographic projections –
  - Front Projection
  - Top Projection
  - Side Projection

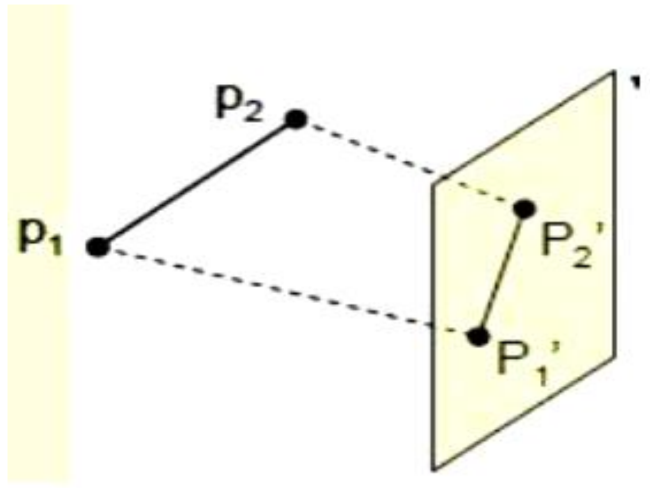




b) Oblique parallel Projection

- A projection in which the angle between the projectors and the plane of plane of **projection is not equal to 90^0** is called oblique projection.
- An oblique projection is formed by parallel projections from a center of projection at infinity that intersects the plane of projection at an oblique angle.





Derivation of Oblique projection Transformation matrix:

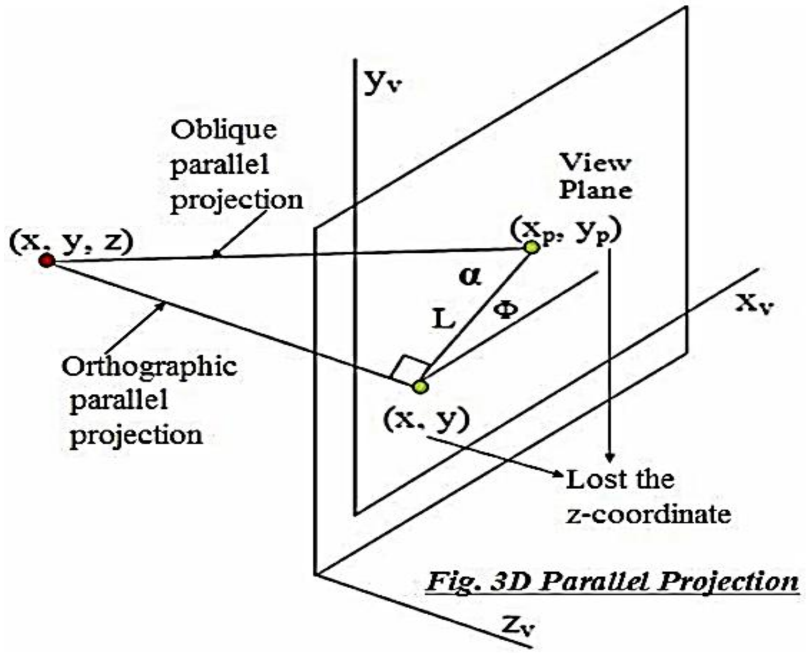


Fig. 3D Parallel Projection

- Let, Point (x, y, z) is projected to position (x_p , y_p) on the view plane.
- Let, Orthographic projection coordinated on the plane are (x, y).
- Let, Oblique projection line from (x, y, z) to (x_p , y_p) makes an angle **α** with the line on the projection plane that joins (x_p , y_p) and (x , y).
- Let, This line of length **L** is at an angle Φ with the horizontal direction in the projection plane.

- Expressing projection coordinates in terms of x, y, L and Φ as:

$$x_p = x + L \cos\Phi$$
$$y_p = y + L \sin\Phi$$

- L depends on the angle α and z coordinate of point to be projected

$$\tan \alpha = z/L$$

- Thus, L = z / tan α

$$= z L_1, \quad \text{where L1 is the inverse of } \tan \alpha$$

- So the oblique projection equations are:

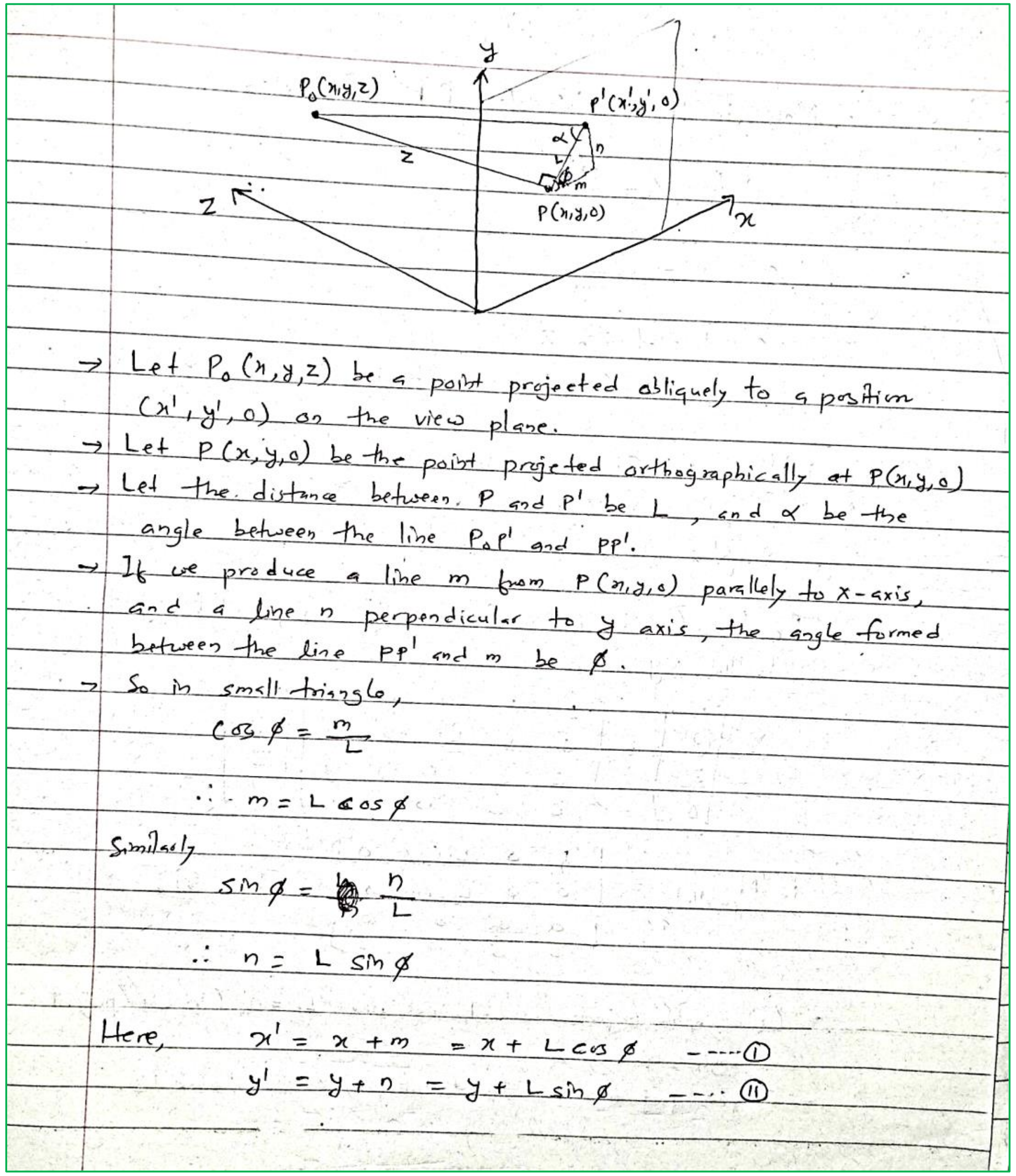
$$x_p = x + z (L_1 \cos\Phi)$$
$$y_p = y + z (L_1 \sin\Phi$$

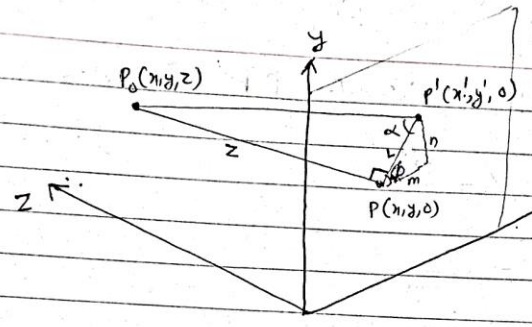
- The transformation matrix for producing any parallel projection onto the $x_v y_v$ -plane can be written as:

$$M_{parallel} = \begin{pmatrix} 1 & 0 & L_1\cos\Phi & 0 \\ 0 & 1 & L_1\sin\Phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Orthographic projection is obtained when $L_1 = 0$ (occurs at projection angle α of 90°) Oblique projection is obtained with non-zero values for L_1.

Derivation of Oblique projection Transformation matrix:





→ Let $P_0(x,y,z)$ be a point projected obliquely to a position $(x',y',0)$ on the view plane.

→ Let $P(x,y,0)$ be the point projected orthographically at $P(x,y,0)$

→ Let the distance between P and P' be L, and α be the angle between the line P_0P' and PP'.

→ If we produce a line m from $P(x,y,0)$ parallely to x-axis, and a line n perpendicular to y axis, the angle formed between the line PP' and m be ϕ.

→ So in small triangle,

$$\cos\phi = \frac{m}{L}$$

$$\therefore m = L\cos\phi$$

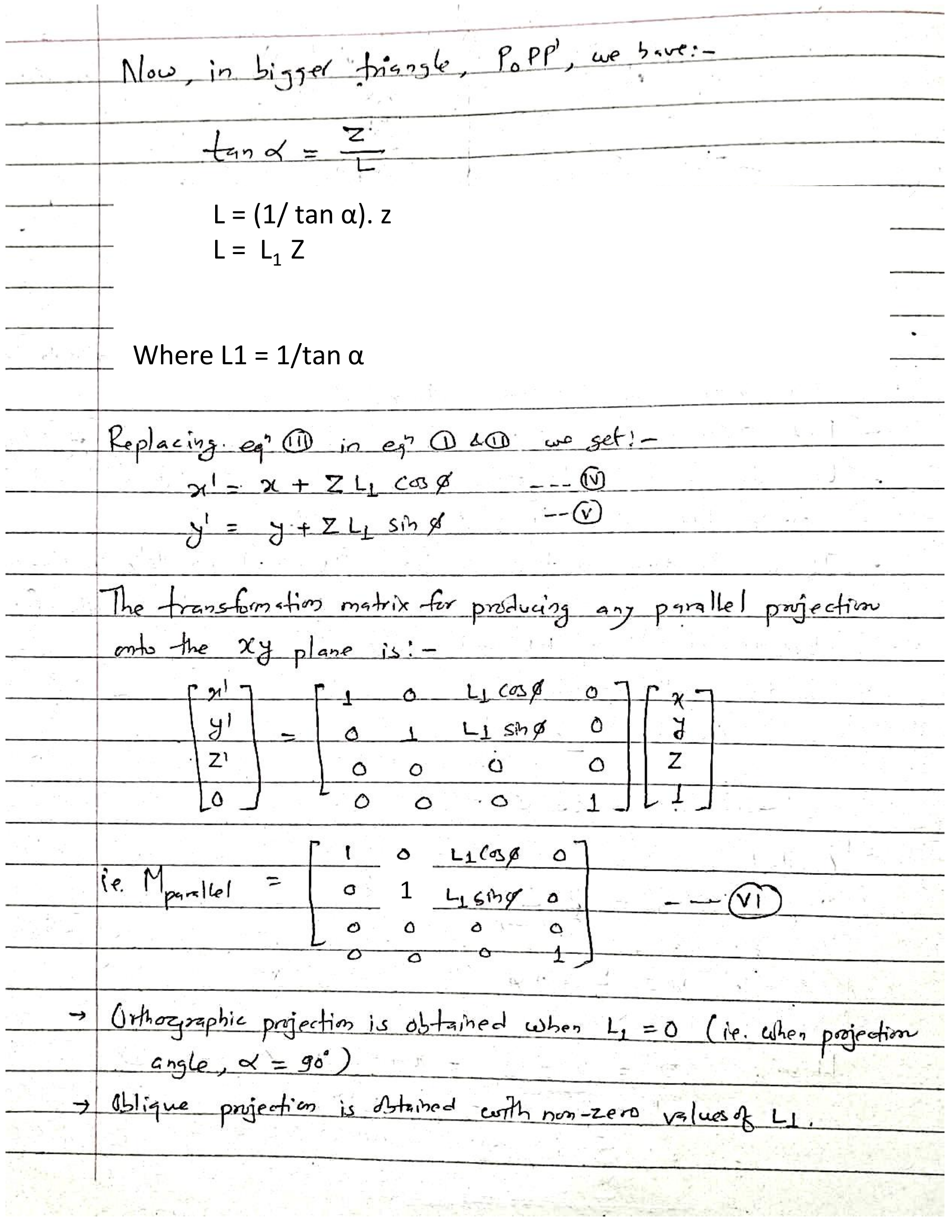
Similarly

$$\sin\phi = \frac{n}{L}$$

$$\therefore n = L\sin\phi$$

Here, $x' = x + m = x + L\cos\phi$ ----(I)

$y' = y + n = y + L\sin\phi$ ---(II)



Now, in bigger triangle, P_0PP', we have:-

$$\tan\alpha = \frac{Z}{L}$$

L = (1/ tan α). z

L = L_1 Z

Where L1 = 1/tan α

Replacing eqn (III) in eqn (I) & (II) we get:-

$x' = x + Z L_1 \cos\phi$ ---(IV)

$y' = y + Z L_1 \sin\phi$ --(V)

The transformation matrix for producing any parallel projection onto the xy plane is:-

$$\begin{bmatrix} x' \\ y' \\ Z' \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L_1\cos\phi & 0 \\ 0 & 1 & L_1\sin\phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ Z \\ 1 \end{bmatrix}$$

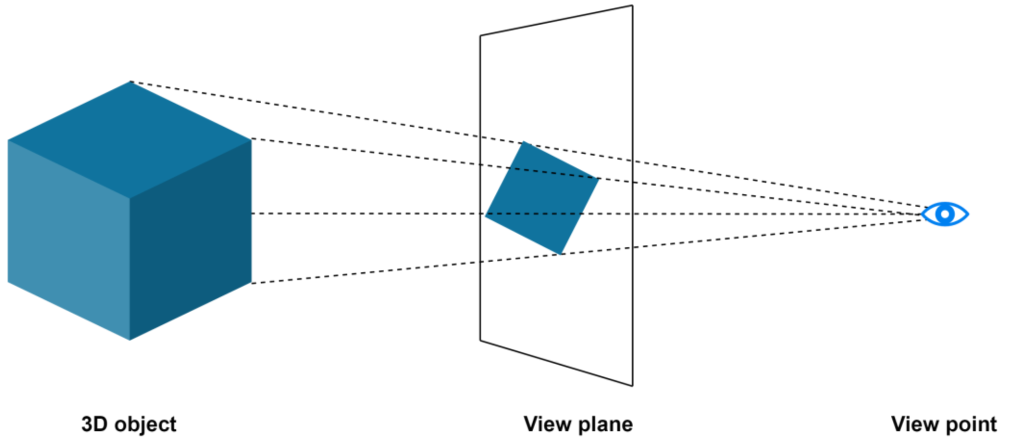
ie. $M_{parallel} = \begin{bmatrix} 1 & 0 & L_1\cos\phi & 0 \\ 0 & 1 & L_1\sin\phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ ---(VI)

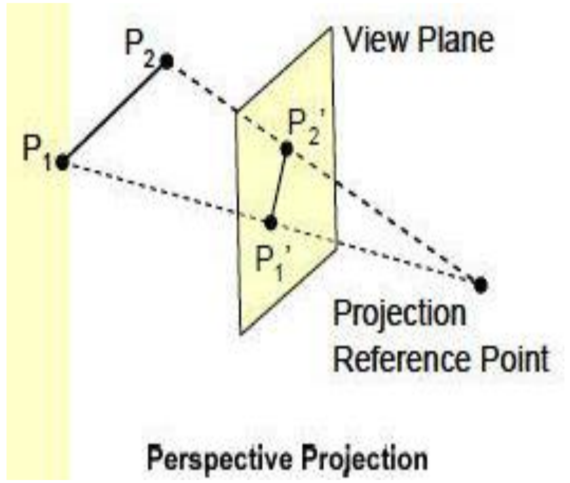
→ Orthographic projection is obtained when $L_1 = 0$ (ie. when projection angle, $\alpha = 90^\circ$)

→ Oblique projection is obtained with non-zero values of L_1.

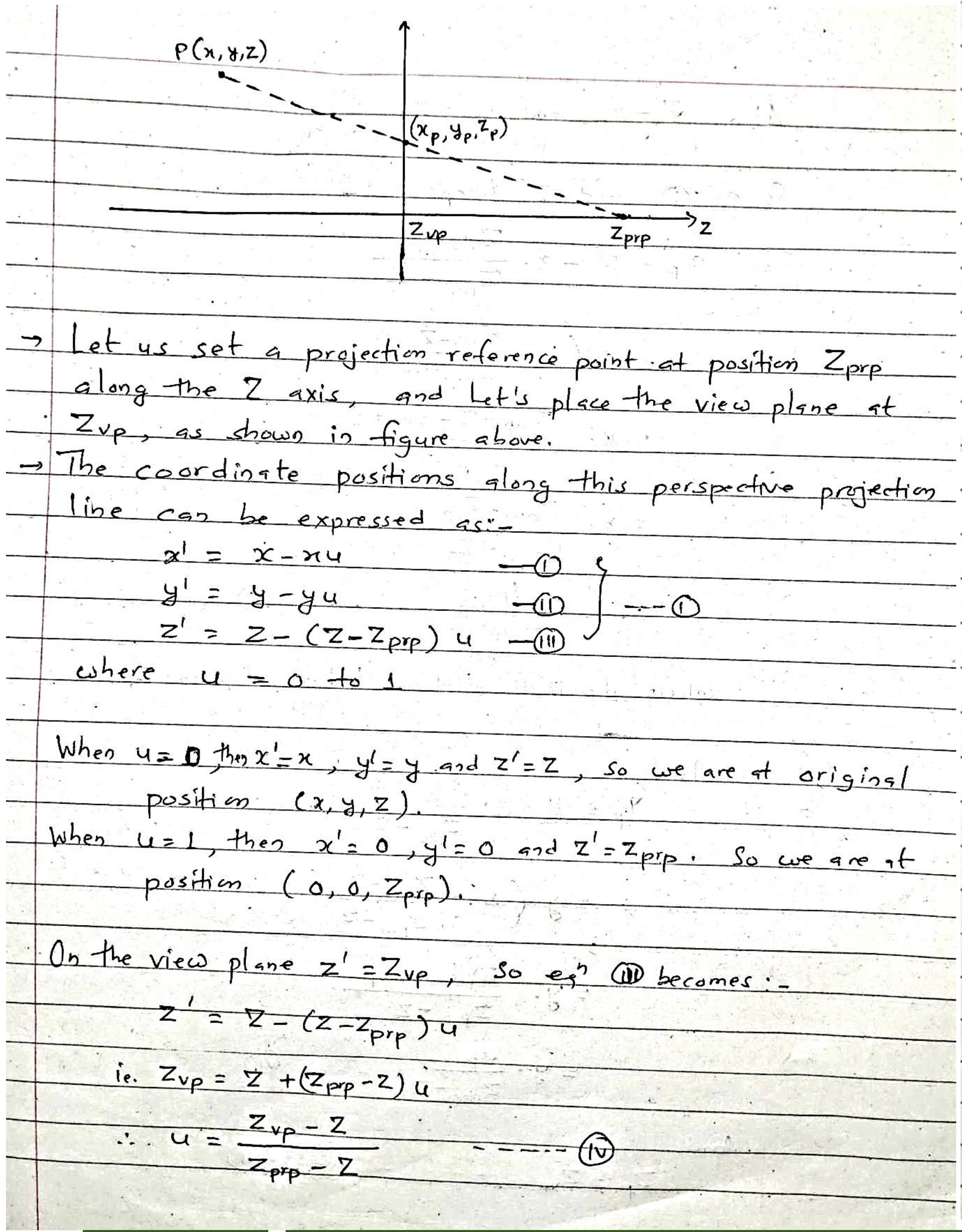
ii) Perspective projection

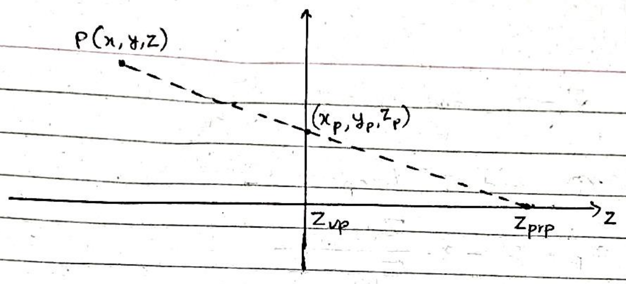
- In this projection, the co-ordinate positions are transformed to the view plane along lines that **converges to a point called as center of projection**.
- Perspective projection works by **projecting the 3D object to a reduced size on the view plane**.
- Besides, the lines of projection converge to the view plane based on the viewpoint.
- Additionally, we can also refer to the viewpoint as the center of projection.





Derivation of Perspective projection Transformation matrix:





→ Let us set a projection reference point at position Z_{prp} along the Z axis, and let's place the view plane at Z_{vp}, as shown in figure above.

→ The coordinate positions along this perspective projection line can be expressed as:-

$$x' = x - xu \quad \text{---(i)}$$
$$y' = y - yu \quad \text{---(ii)}$$
$$z' = z - (z - Z_{prp})u \quad \text{---(iii)}$$
$$\Big\} \text{---(1)}$$

where $u = 0$ to 1

When $u = 0$, then $x' = x$, $y' = y$ and $z' = z$, so we are at original position (x, y, z).

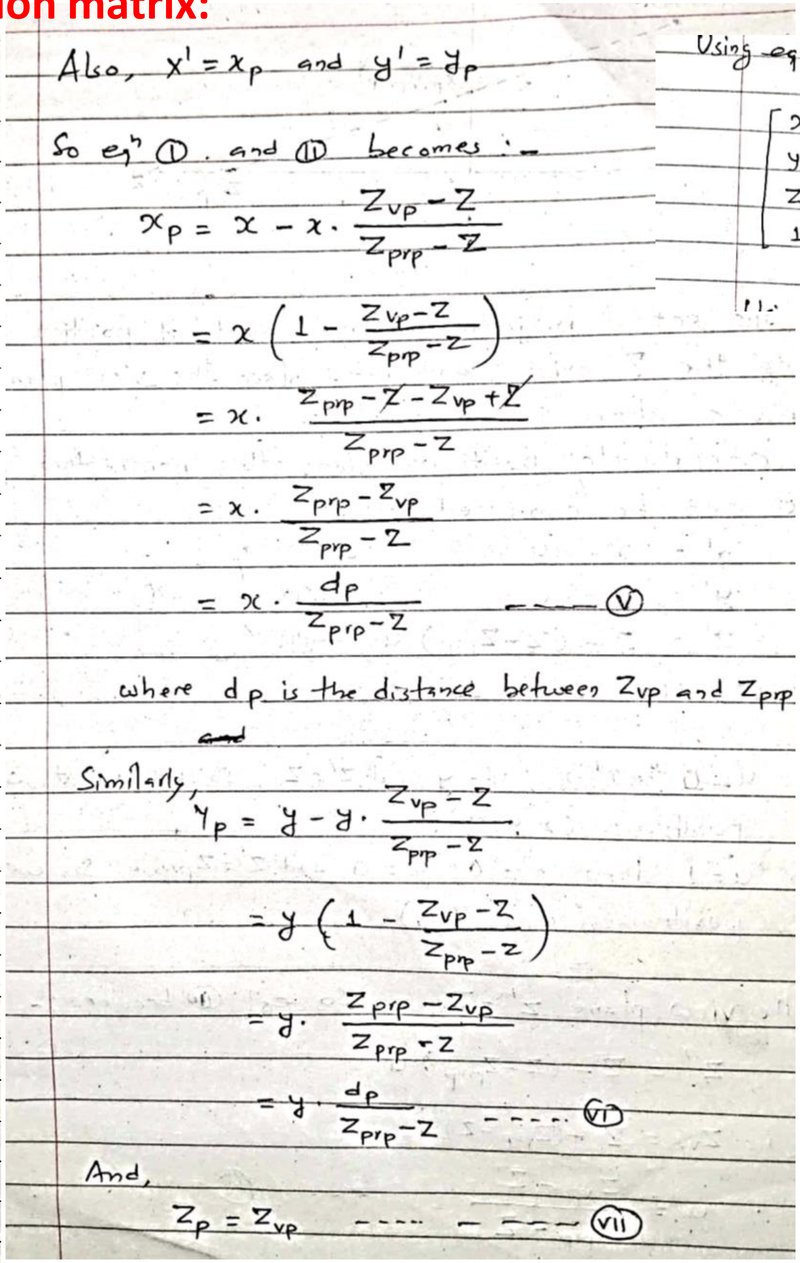
When $u = 1$, then $x' = 0$, $y' = 0$ and $z' = Z_{prp}$. So we are at position $(0, 0, Z_{prp})$.

On the view plane $z' = Z_{vp}$, so eqn (iii) becomes :-

$$z' = z - (z - Z_{prp})u$$

ie. $Z_{vp} = z + (Z_{prp} - z)u$

$$\therefore u = \frac{Z_{vp} - z}{Z_{prp} - z} \quad \text{-----(iv)}$$



Also, $x' = x_p$ and $y' = y_p$

So eqn (i) and (ii) becomes :-

$$x_p = x - x \cdot \frac{Z_{vp} - z}{Z_{prp} - z}$$

$$= x\left(1 - \frac{Z_{vp} - z}{Z_{prp} - z}\right)$$

$$= x \cdot \frac{Z_{prp} - \cancel{z} - Z_{vp} + \cancel{z}}{Z_{prp} - z}$$

$$= x \cdot \frac{Z_{prp} - Z_{vp}}{Z_{prp} - z}$$

$$= x \cdot \frac{d_p}{Z_{prp} - z} \quad \text{-----(v)}$$

where d_p is the distance between Z_{vp} and Z_{prp}

~~and~~

Similarly,

$$y_p = y - y \cdot \frac{Z_{vp} - z}{Z_{prp} - z}$$

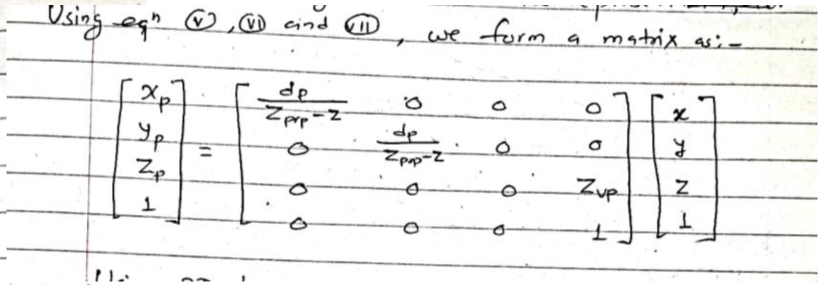
$$= y\left(1 - \frac{Z_{vp} - z}{Z_{prp} - z}\right)$$

$$= y \cdot \frac{Z_{prp} - Z_{vp}}{Z_{prp} - z}$$

$$= y \cdot \frac{d_p}{Z_{prp} - z} \quad \text{-----(vi)}$$

And,

$$z_p = Z_{vp} \quad \text{-----(vii)}$$



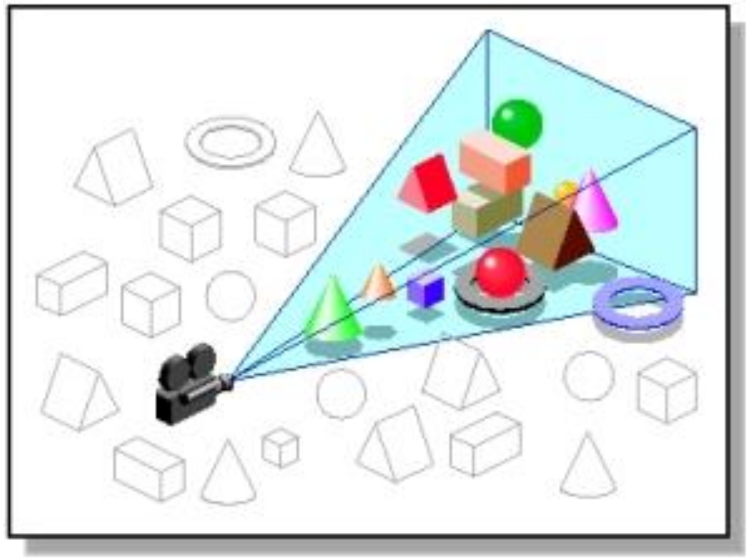
Using eqn (v), (vi) and (vii), we form a matrix as:-

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{d_p}{Z_{prp} - z} & 0 & 0 & 0 \\ 0 & \frac{d_p}{Z_{prp} - z} & 0 & 0 \\ 0 & 0 & 0 & Z_{vp} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

4.6 Clipping in Three Dimensions

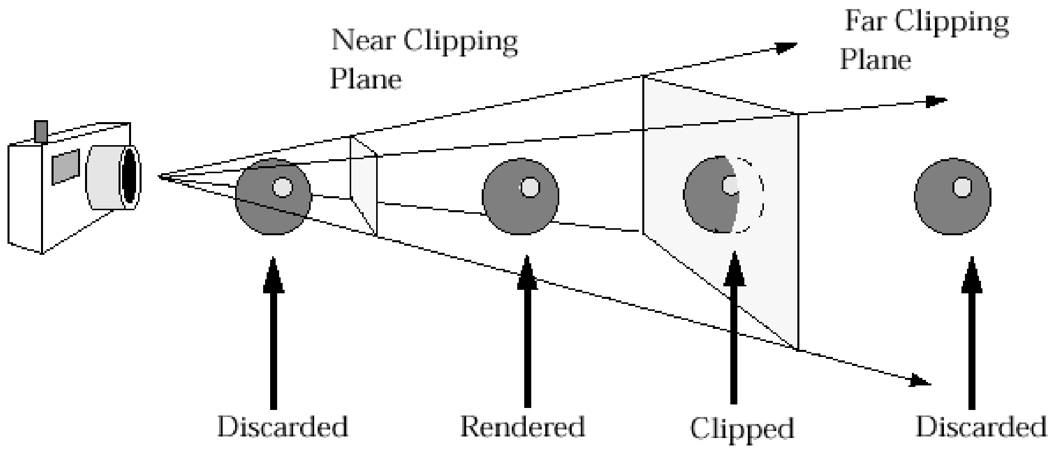
3-D Clipping

- Just like the case in two dimensions, clipping removes objects that will not be visible from the scene. All parts of objects that are outside the view volume are discarded.
- 3-D clipping is achieved in two basic steps
  - Discard objects that can't be viewed
    - i.e. objects that are behind the camera, outside the field of view, or too far away
  - Clip objects that intersect with any clipping plane
- Discarding objects that cannot possibly be seen involves comparing an objects bounding box/sphere against the dimensions of the view volume
  - Can be done before or after projection

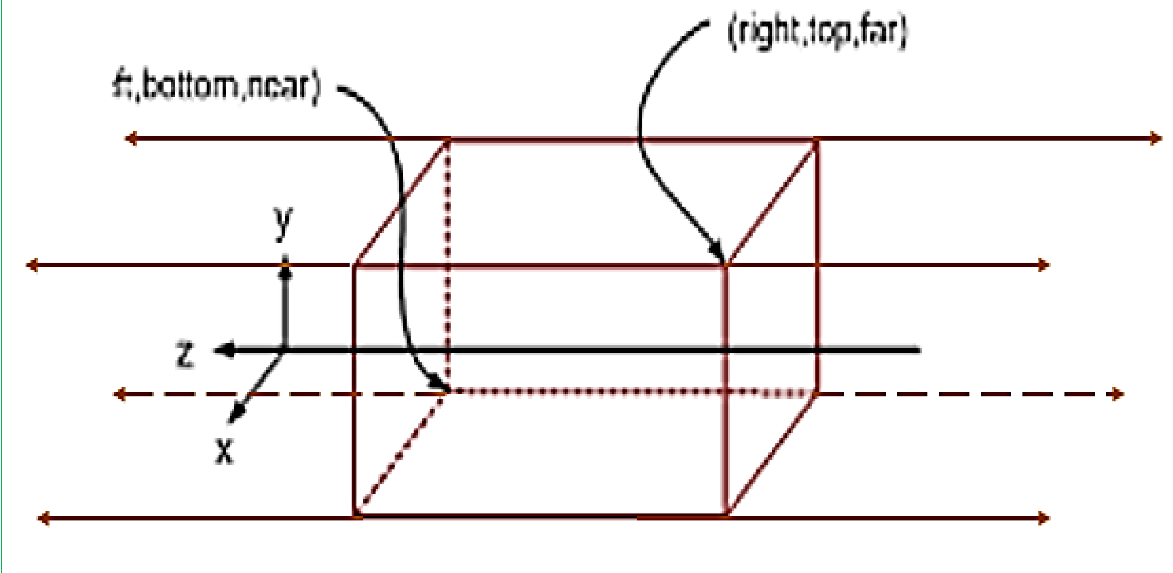


- An algorithm for three-dimensional clipping identifies and saves all surface segments within the view volume for display on the output device.
- All parts of objects that are outside the **view volume** are discarded.
- To clip a polygon surface, we can clip the individual polygon edges. To clip a line segment against the view volume, we would need to test the relative position of the line using the view volume's boundary plane equations.
- An endpoint (x, y, z) of a line segment is :
  - ➢ Outside a boundary plane if: $Ax + By + Cz + D > 0$
  - ➢ Inside the boundary plane if: $Ax + By + Cz + D < 0$

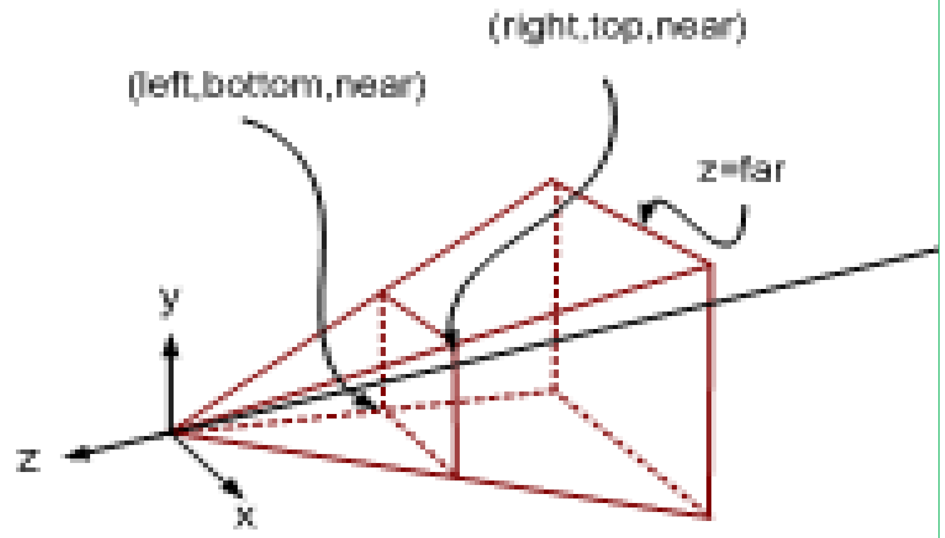
  , where A, B, C, and D are the plane parameters for that boundary.
- There are two parts to the view volume:
  - the view plane rectangle and
  - the near and far clipping planes



- In case of Parallel projections the infinite Parallelepiped is bounded by Near/front and far/back planes for clipping.
- In case of Perspective projections the semi Infinite Pyramid is also bounded by Near/front and far/back planes for clipping

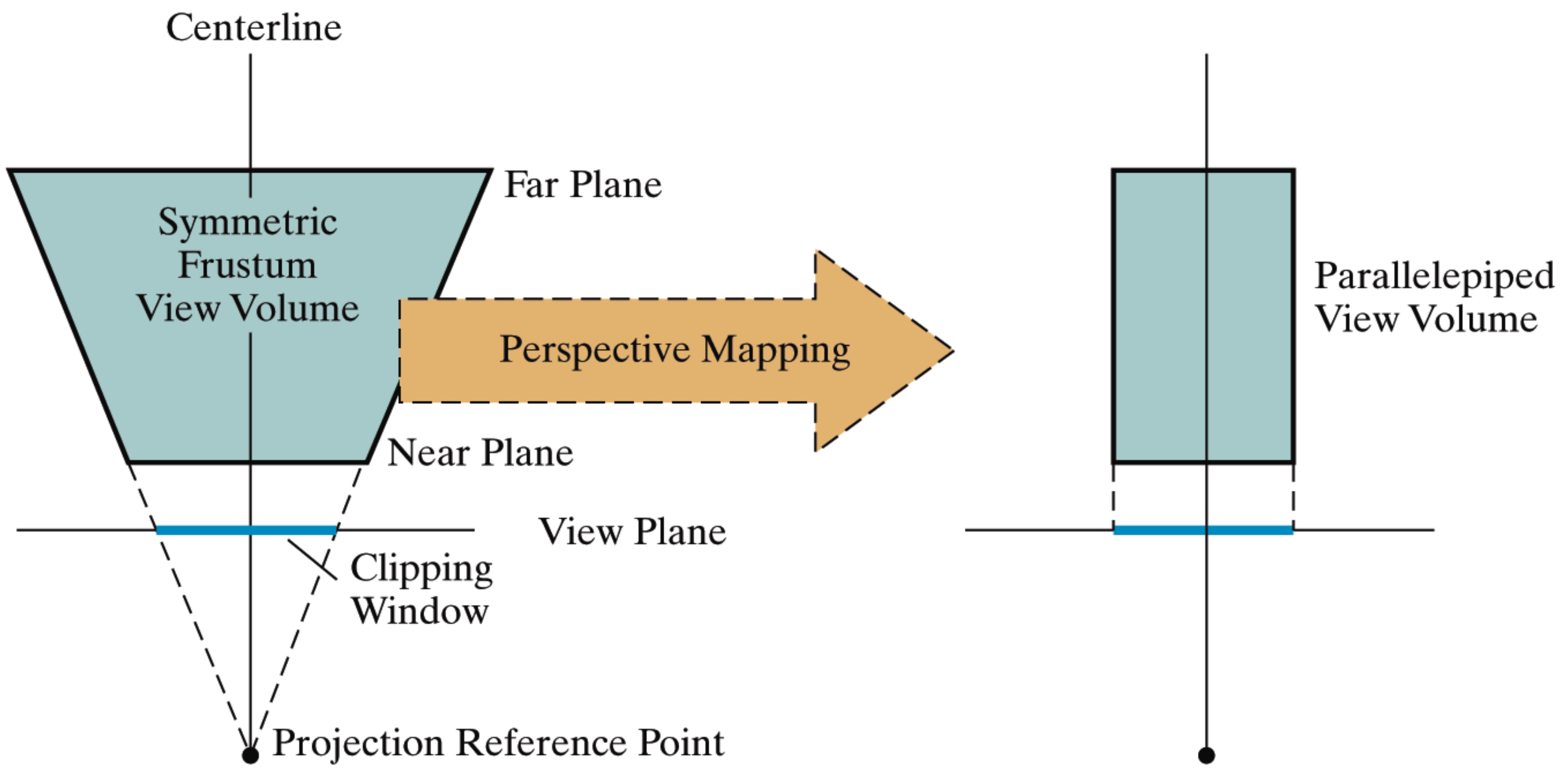


Orthographic View Volume



Perspective View Volume

- After the perspective transformation is complete the frustum shaped viewing volume has been converted to a parallelepiped - remember we preserved all z coordinate depth information


Centerline
Far Plane
Symmetric Frustum View Volume
Perspective Mapping
Parallelepiped View Volume
Near Plane
View Plane
Clipping Window
Projection Reference Point

Exam Questions

1. Explain how a 3D object is represented using polygon surface method. [2011 fall]
2. What is meant by a project? Describe the significance of making plans for project development with appropriate illustrations. [2011 fall]
3. Explain how you would rotate an object by angle Φ (phi) about arbitrary plane in 3D space. [2011 spring]
4. What is projection? Derive the expression and matrix representation for orthographic parallel projection. [2011 spring]
5. Derive a transformation matrix for perspective projection. [2012 fall]
6. How do you reflect an object about any arbitrary axis that is not parallel to any of the major coordinate axis in 3D? [2012 fall]
7. What do you mean by projection? Differentiate between parallel and perspective projection. [2012 spring]
8. What is projection? Derive the expression and matrix representation for perspective projection. [2013 fall]
9. What do you mean by projection? Differentiate between parallel and perspective projection with example. [2013 spring]
10. Derive a transformation matrix due to orthographic and oblique parallel projection. [2014 fall]

11. Derive a matrix for cubic Bezier curve formation. [2014 fall]
12. What are the issue in 3D that makes it more complex than 2D? Derive an equation for 3D translation and reflection. [2014 spring]
13. Define projection. Derive an equation for the parallel projection. [2014 spring]
14. Derive the expression and matrix representation for perspective projection. [2015 fall]
15. What role does vanishing point play in perspective projection? Explain by deriving equations for perspective projection by considering a vanishing point. [2015 spring]
16. What is the significance of homogeneous coordinate system? How can an object be reflected about an arbitrary plane in 3D? [2015 spring]
17. Describe different types of parallel projections. Derive the transformation matrix for parallel projection. [2016 fall]
18. Explain the steps of 2D viewing pipeline. How is the complexity added in 3D viewing process in comparison to 2D viewing process? [2016 fall]
19. What do you mean by perspective projection? Derive an expression for finding perspective projection of a point onto a plain surface. [2016 spring]
20. What are the issue in 3D that makes it more complex than 2D? Derive an equation for 3D translation and reflection. [2014/2017 Spring]

21. Define projection. Derive an equation for the parallel projection.
22. Define projection. Differentiate between parallel and perspective projection along with an equation. [2017 Fall]
23. Describe the **rotation of an object about an axis**, which is parallel to any of three coordinates axes of coordinate system. [2017 spring]
24. Show how to use a 3-D matrix to rotate a unit cube about the axis defined by vector (1,1,1). [2018 fall]
25. Explain Bezier method of curve drawing.
26. Define projection. Derive a matrix for a parallel projection. [2018 spring]
27. Calculate (x,y) coordinate of Bezier curve described by the following 4 control points (0,0), (1,2), (3,3), (4,0). Assume any needed values.
28. Explain and derive transformation matrix of 3D rotation about a line not parallel to any one axis. [2019 fall]
29. Derive the composite matrix for reflection of an object about an arbitrary axis in 3D space.
30. Explain Bezier curve and also specify the properties of Bezier curve. [2019 spring]

31. Explain and Derive transformation matrix for a parallel projection.
32. Derive equation for Bezier curve in quadratic polynomial and specify the blending function. [2020 fall]
33. Define projection. Differentiate between parallel and perspective projection with figure.
34. How do you represent different objects in 3D? Differentiate between parallel and perspective projection with examples. [2021 fall]
35. What is 3D transformation? What so you mean by projection? Derive a matrix to obtain the perspective projection of any arbitrary point. [2021 spring]
36. What are the different ways of representing 3D objects in computer graphics? Explain how can you represent a polygon surface as a 3D-object?
37. How do you represent 3D objects by using Polygon tables? How is the consistency of geometric data table checked and what are the rules for generating error free polygon tables? [2022 fall]

End of Chapter