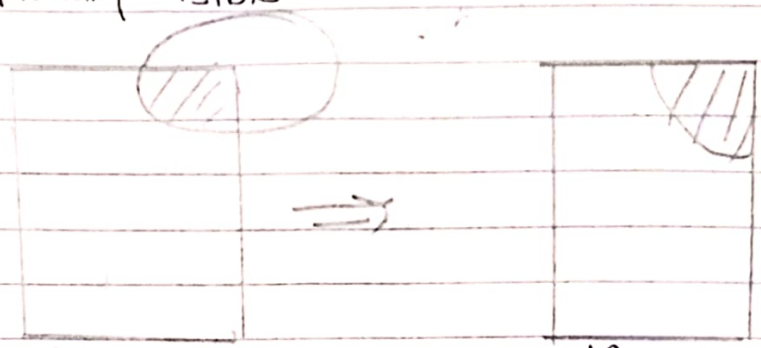
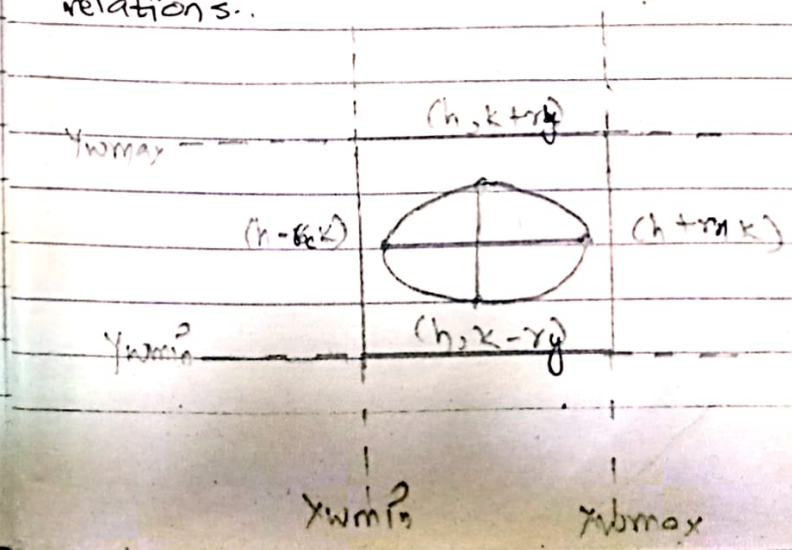
Ellipse clipping algorithm

- A procedure used to extract visible part of ellipse according to window specification is called as ellipse clipping algorithm.
- The ellipse must be filled before clipping otherwise distinct segments of curve are displayed.
- A ellipse must be divided into 3-classes
  (a) visible
  (b) Invisible and
  (c) Partially visible



unclipped ellipse clipped ellipse

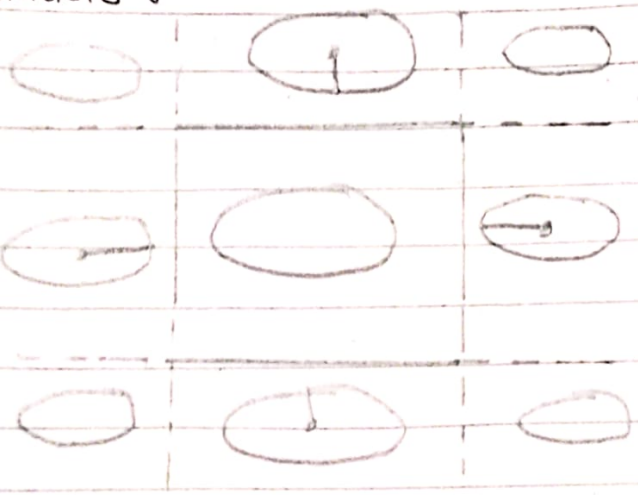
- The partially visible ellipse requires clipping operation.
- An ellipse is said to be visible only when end points lies inside the window frame i.e. it must satisfy the following relations..



$h - r_x \geq x_{wmin}$ and
$x_{wmax} \geq h + r_x$

$k - r_y \geq y_{wmin}$ and
$y_{wmax} \geq h + r_y$

A circle is invisible if and only if it satisfy the following relations :



(1) $h+r < x_{wmin}$ or $h-r > x_{wmax}$

and

(2) $k+r < y_{wmin}$ or $k-r > y_{wmax}$

If an ellipse doesn't satisfy stated above 2 conditions then, it is called partially visible ellipse.
It requires clipping operation.
For clipping, the intersection point between window boundry and ellipse can be calculated by solving equation of ellipse and window frame.

The inside outside test is performed for each points of ellipse during rasterization.
If the point lies inside the ellipse, then it is accepted otherwise rejected from operation.

Polygon clipping algorithm

A procedure used to extract the visible parts of the regular polygon according to window frame is called polygon clipping algorithm. A polygon is said to be regular if and only if it is constructed by collection of points and lines.

The polygon must be filled before clipping to get exact information otherwise, discrete segment of lines are displayed after clipping.

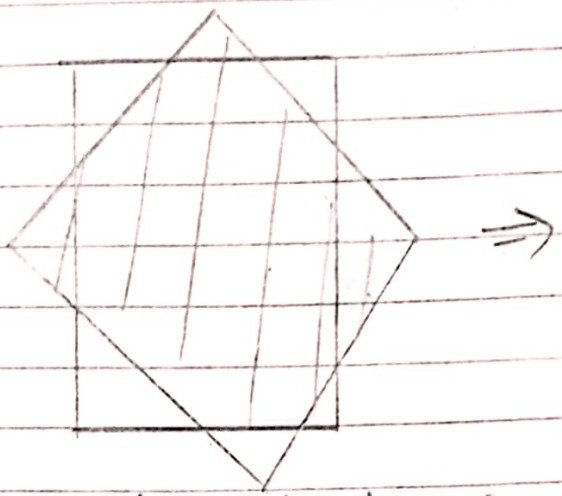


fig: Unclipped polygon

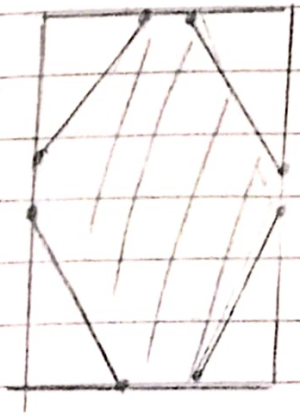
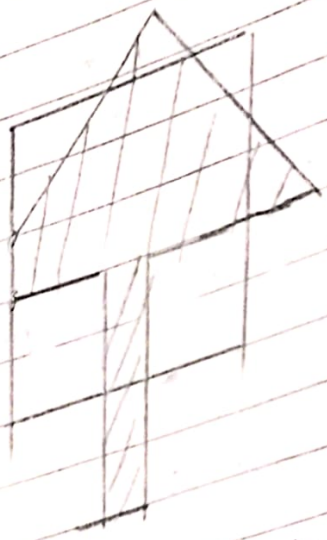


fig: clipped polygon

Sutherland Hodgeman polygon clipping

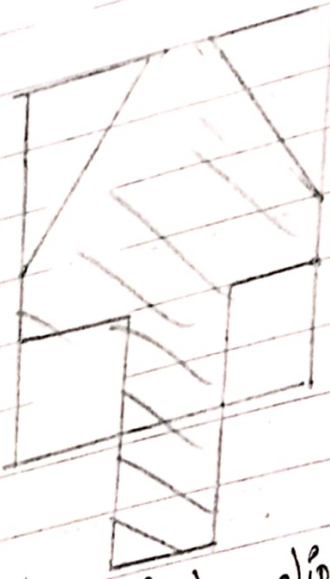
A clipping procedure applied to clip a regular polygon based on line clipping procedure provided by cohen sutherland is Sutherland Hodgeman polygon clipping. This procedure follow the clockwise approach ie. the left part of window is clipped followed by top, right and buttom.

unclipped polygon



left clipped polygon

top clipped polygon



Right clipped polygon

bottom clipped polygon.

The intersection point between window frame and polygon edges can be calculated as :

for left clipping,

$$I_x = x_{wmin}$$
$$I_y = y_1 + m(I_x - x_1)$$

for top clipping,

$$I_y = y_{wmax}$$
$$I_x = x_1 + \left(\frac{I_y - y_1}{m}\right)$$

for right clipping,

$$I_x = x_{wmax}$$
$$I_y = y_1 + m(I_x - x_1)$$

for bottom clipping,

$$I_y = y_{wmin}$$
$$I_x = x_1 + \left(\frac{I_y - y_1}{m}\right)$$

Area filling

A procedure used to render a pattern or color inside the area covered by a polygon is called as area filling. The fix

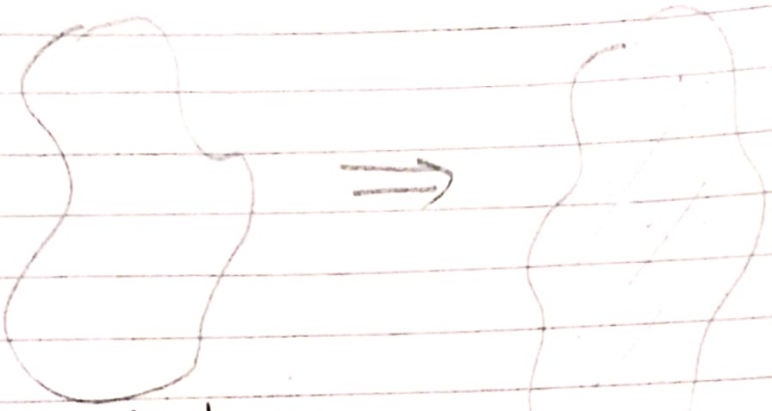
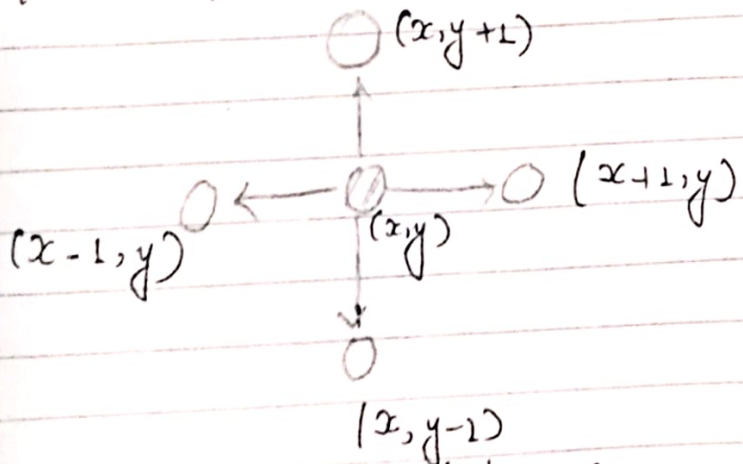


fig 6 unfilled area

fig 6 filled area.

Two algorithms either flood fill or boundary fill be used to fill the irregular or regular polygon with a patterns. This procedure use the connectivity between the pixels to access the area covered by polygon. The connectivity may be 4 point or 8 point.



4- connected region

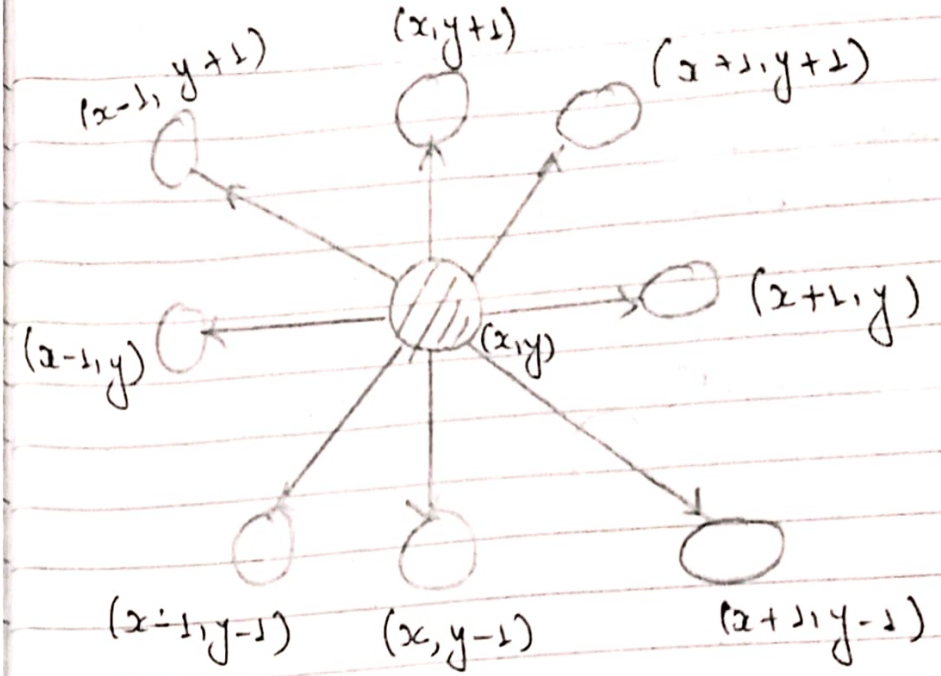


fig 6 8-connected region

Boundary fill algorithm

A polygon boundry is filled with specified color and inside area be filled by taking boundry color as a decision making variable. The area be accessed by forming 4-connected or 8-connected regions. The algorithm for 4-connected region is :-

~~Boundary fill-4 (x, y, fill_color, boundry-color integer);~~

```
{
Boundryfill_4 (int x, int y, fill_color, boundry-color integer);
{
int current-color = getpixel(x, y);
if (current_color != fill_color & current_color != boundry_color)
```

```
{
 putpixel (x,y, fill_color);
Boundryfill_4 (x+1,y, fill_color, boundry_color);
Boundryfill_4 (x, y+1, fill_color, boundry_color);
Boundryfill_4 (x-1, y, fill_color, boundry_color);
Boundry_fill_4 (x, y-1, fill_color, boundry_color);
}
}
```

The algorithm for 8 connected region is :

```
Boundry fill_8 (int x, int y, fill_color, boundry_color & Integer)
    {
    int current_color = getpixel (x,y);
    if (current_color != fill_color & current_color !=
                                    boundry_color)
      {
      putpixel (x,y, fill_color);
     Boundry fill_8 (x+1, y, fill_color, boundry_color);
     Boundryfill_8 (x+1, y-1, fill_color, boundry_color);
     Boundryfill_8 (x, y-1, fill_color, boundry_color);
     Boundry fill_8 (x-1, y-1, fill_color, boundry_color);
    Boundryfill_8 (x-1, y, fill_color, boundry_color);
     Boundry fill_8 (x-1, y+1, fill_color, boundry_color);
     Boundryfill 8 (x, y+1, fill_color, boundry_color);
     Boundryfill_8 (x+1, y+1, fill_color, boundry_color);
```

```
}  }
}
```

→ Floodfill algorithm

This procedure used the old color used by polygon area as a decision making variable. It is also used to fill the area with different color patterns. It also access the area covered by a polygon by taking 4 or 8 connected regions.

```
void floodfill (int x, int y, int oldcolor, int fillcolor)
{
int currentcolor = getpixel (x,y);
If (currentcolor != fillcolor && currentcolor == oldcolor
{
putpixel (x,y, fillcolor);
floodfill_4 (x+1, y, oldcolor, fillcolor);
floodfill_4 (x, y+1, oldcolor, fillcolor);
floodfill_4 (x-1,y, oldcolor, fillcolor);
floodfill-4 (x, y-1, oldcolor, fillcolor);
}
}
```

projection Transformation

A tool used to convert n dimensional co-ordinate into n-1 dimension is called as projection transformation. It is used to map the three dimensional object into 2dimensional viewing area by maintaining image property.

The thin lines are used to provide a reference form object transformation is called as projectors. and the point where projectors are intersect is projection reference point.

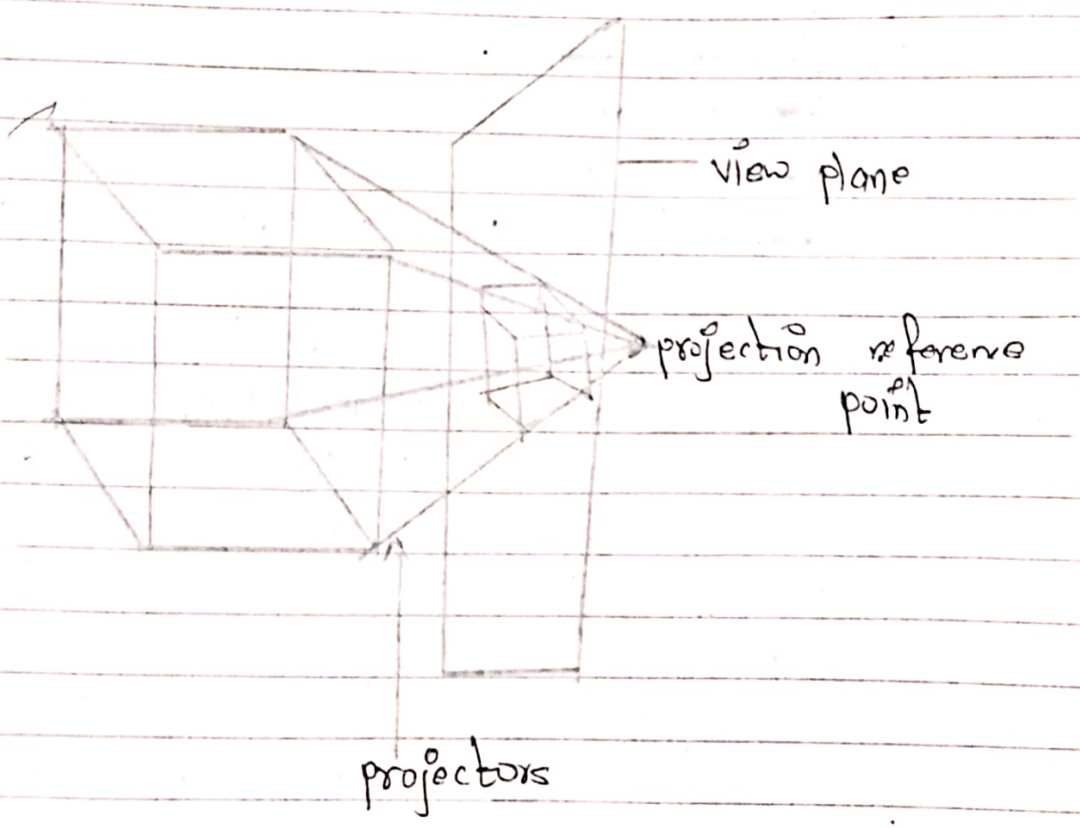


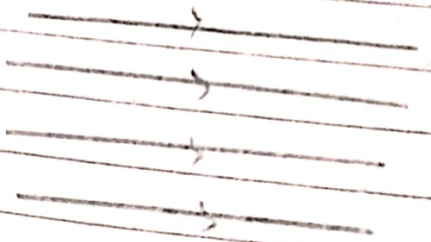
fig: projection Transformation.

On the basis of points of projection reference point, there are 2 types of projection :

1. parallel projection.
2. perspective projection

1. parallel projection :

When projectors are parallel to each other and meet at a infinite distance ie. projection reference point lies at a infinite distance from the projection plane called as parallel projection



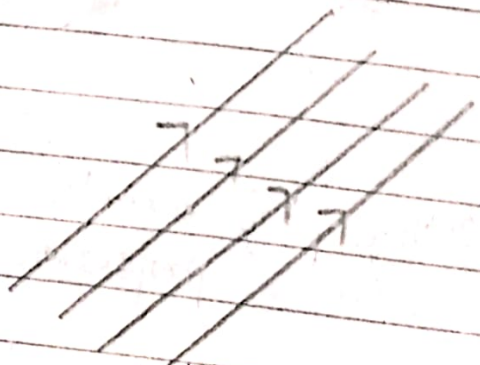


fig 6 parallel projections.

On the basis of allignment of projectors, there are 2 types of parallel projections.

1. orthographic parallel projection
2. oblique parallel projection.

(A) Orthographic parallel projection:

A type of parallel projections where projectors are orthogonal with view plane i.e. angle between projector and view plane = 90°. is called orthographic parallel projection.

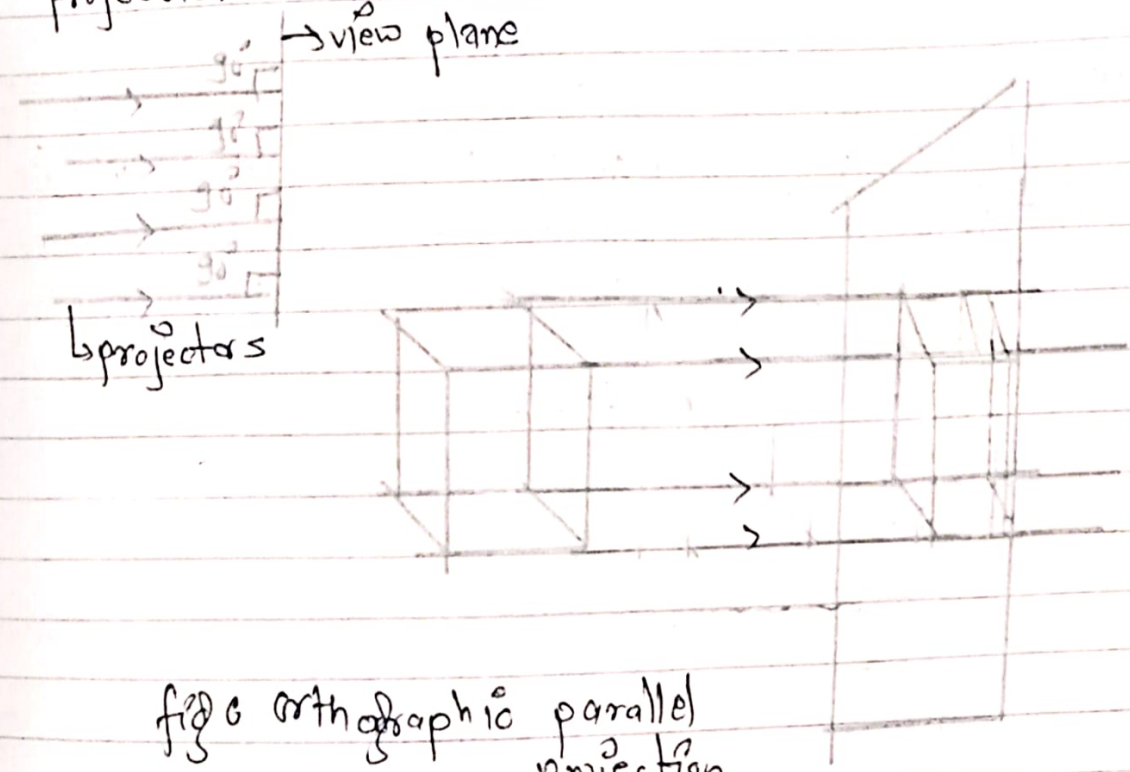
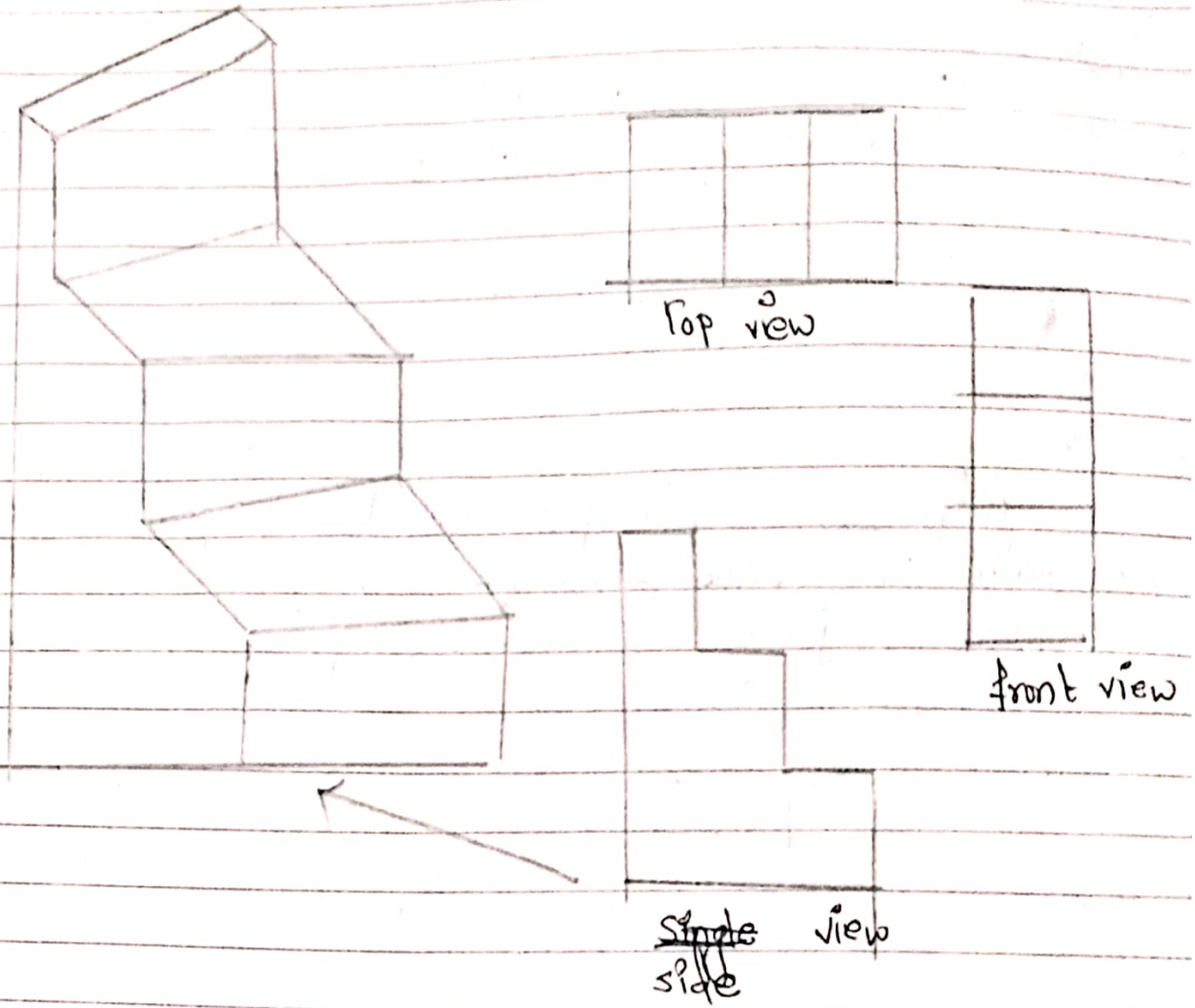


fig: orthographic parallel projection

The orthographic parallel projection always maintain the relational pro relative proportion on shape and size. So, the view is not realistic such that as mostly used in engineering drawing r machine drawing instead of computer graphics.

t is used to produce the front top and side view of 3D bjects in engineering drawings.



Say (x, y, z) be a point is transformed into view plane along z direction as (x', y', z') then, for orthographic projection,

$$x' = x$$
$$y' = y$$
$$z' = 0$$

Now,

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

where,

$$M_{orth} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ is called matrix of orthogonal projection.

(b) oblique parallel projection :

A type of parallel projection where projectors are inclined with view plane at any angle θ, then it is called as oblique parallel projection.

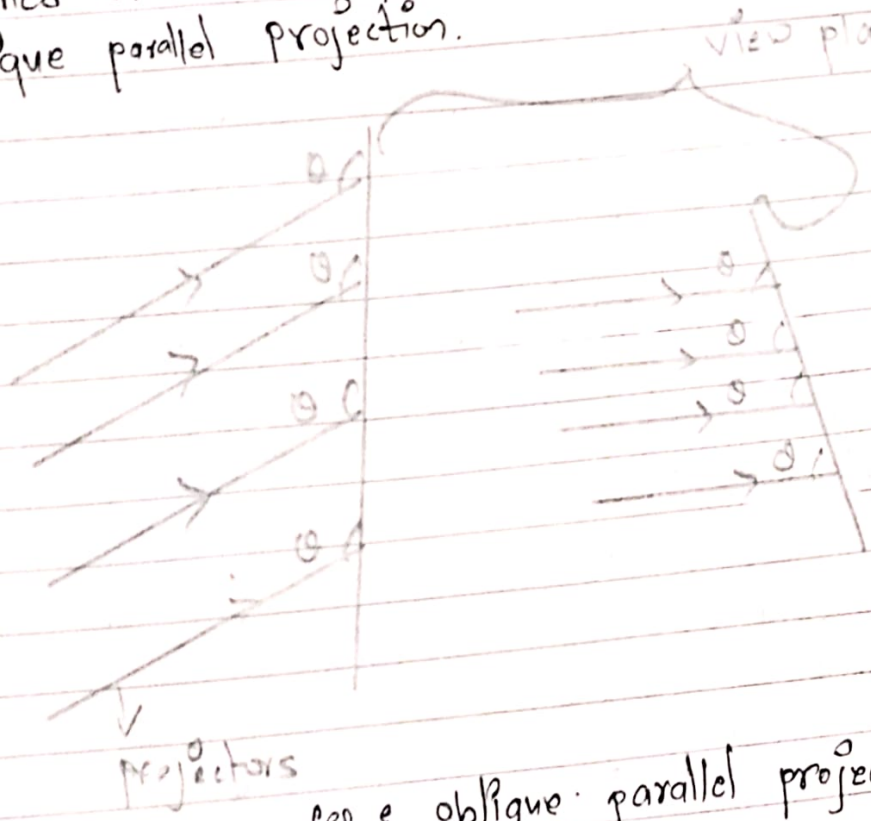


fig : oblique parallel projection

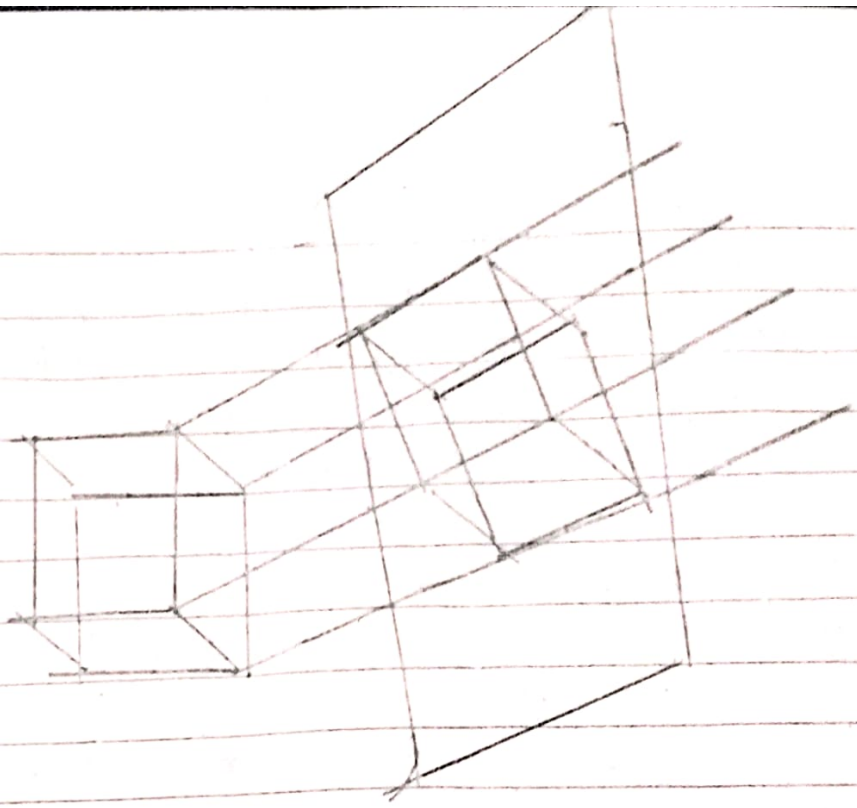
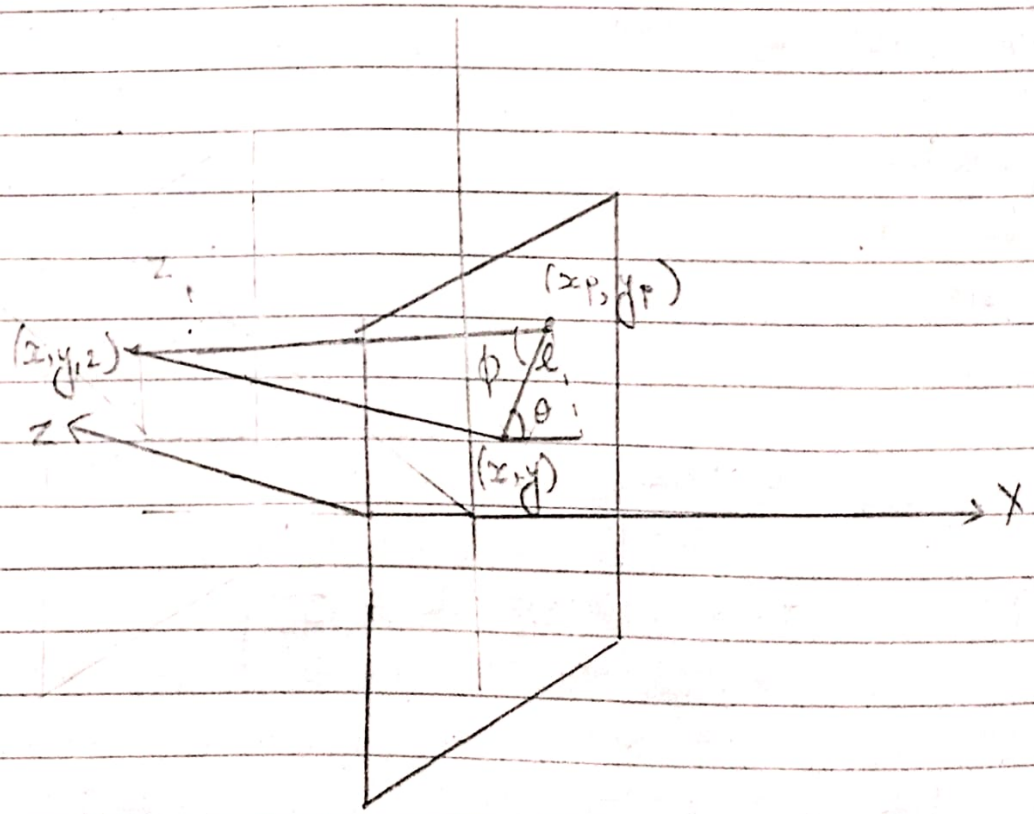


fig 6 of oblique parallel projection

Mathematics of oblique parallel projection



Suppose a point $p(x,y,z)$ is projected on projection plane (x,y) as (x_p, y_p) when projectors are parallel to each other but inclined with any angle with projection plane. (x,y) be the point on projection plane obtained through orthographic parallel projection.

The projectors are defined by 2 angles ϕ and θ where
ϕ = angle of line (x,y) and (x_p, y_p) with projection plane.
θ = angle of line (x,y) and (x_p, y_p) with x-axis in projection plane.

Then,

$$\cos\theta = \frac{x_p - x}{L}$$

$$x_p - x = L\cos\theta$$

$$x_p = x + L\cos\theta \quad \text{——— (1)}$$

$$\sin\theta = \frac{y_p - y}{L}$$

$$y_p = y + L\sin\theta$$

$$\tan\phi = \frac{z}{L} \Rightarrow L = \frac{z}{\tan\phi}$$

~~$z = L\tan\phi$~~

$= L_1 z$ where $L_1 = \frac{1}{\tan\phi}$

Now, new points are:

$$x_p = x + L_1 \cos\theta$$
$$y_p = y + L_1 \sin\theta$$
$$z_p = 0$$

In matrix form,

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & L_1\cos\theta & 0 \\ 0 & 1 & L_1\sin\theta & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

where, $M_{oblique} = \begin{bmatrix} 1 & 0 & L_1\cos\theta & 0 \\ 0 & 1 & L_1\sin\theta & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ is called matrix of oblique parallel project.

(2) Perspective projection:

A type of projection in which projectors are intersect at a finite ~~plane~~ points from the projection plane is called perspective projection.

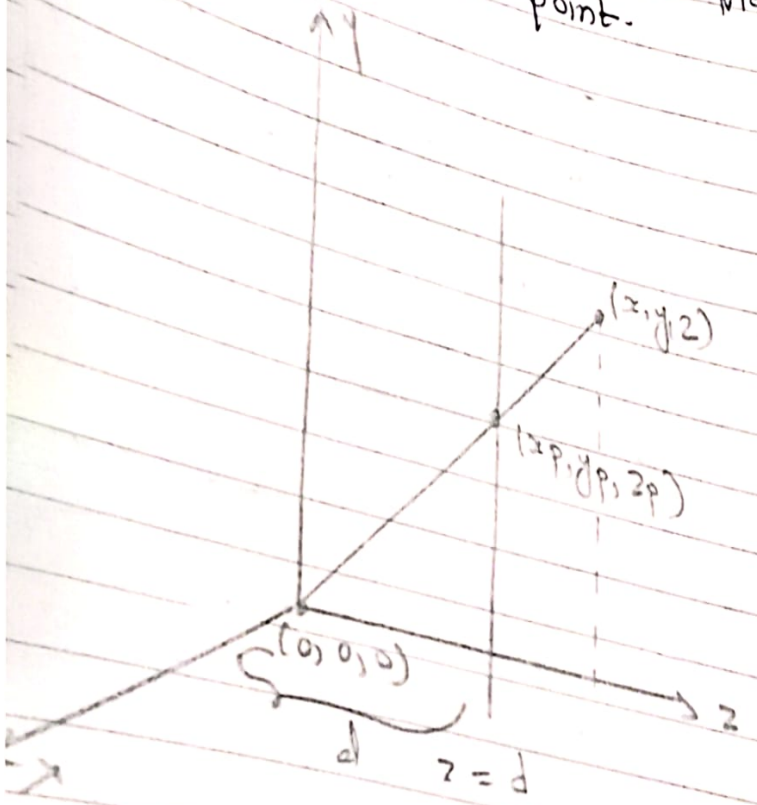
- It is also called as realistic project as it produce projected image ~~area~~ analogous to the eye view. This type of projection is most popular in computer graphics.

On the basic of projection reference point, taken to produce the projected image, there are 3 types of perspective projection.

(a) one point perspective projection.

Two point perspective projection
Three point perspective projection

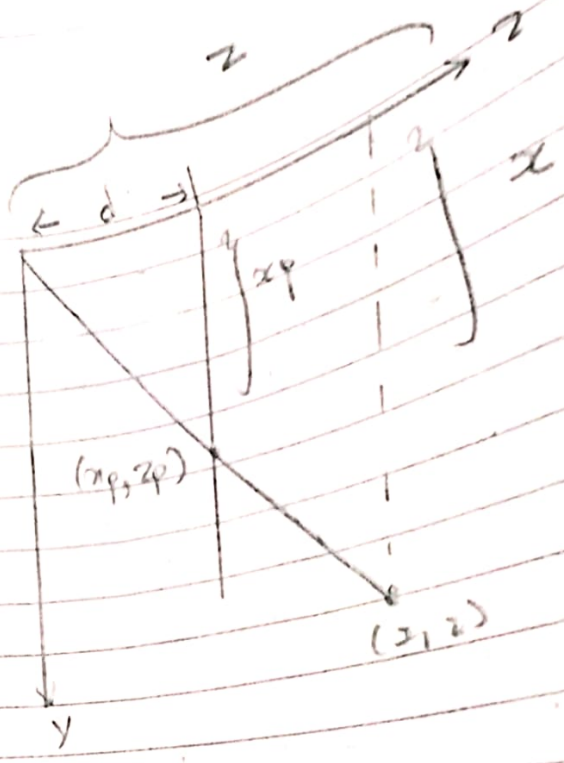
Mathematics of perspective projection
Say a point $p(x,y,z)$ is projected on view plane placed at a distance d from origin along z-direction. (x_p, y_p, z_p) is a projected point on view plane taking origin as a projection reference point.



when projected to xz plane,

from similar triangle properties

$$\frac{d}{z_p} = \frac{z}{x}$$

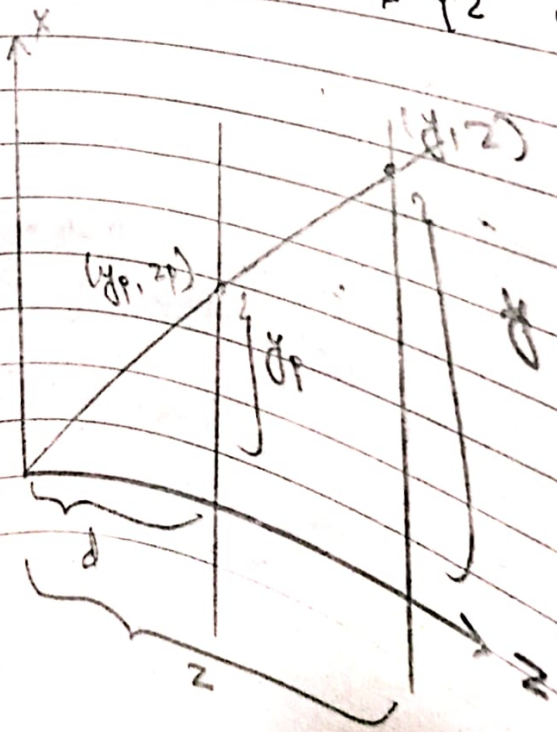


$$\Rightarrow \frac{x_p}{d} = \frac{x}{z}$$

$$\Rightarrow x_p = \frac{x}{z} \cdot d$$

$$= \frac{x}{(z/d)} \quad \text{——— (1)}$$

when projected on yz plane,



From similar triangle properties,

$$\frac{d}{y_p} = \frac{z}{y}$$

$$y_p = \frac{y}{z} d = \left(\frac{y}{z/d}\right) \quad \text{— (iii)}$$

Also, $z_p = d = \frac{z}{(z/d)}$ — (iii)

From above relations,

$$(x_p, y_p, z_p, 1) = \left(\frac{x}{(z/d)}, \frac{y}{(z/d)}, \frac{z}{(z/d)}, 1\right)$$

when, this point is denormalized then,

$$(x_p, y_p, z_p, 1) = (x, y, z, z/d)$$

In matrix form,

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$\therefore M_{pers} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$ is called matrix of perspective projection.

3D object representation

Any 3 dimensional object available in Real environment can be represented by collection of points, lines, surfaces, ~~called~~ segments, curve surfaces, etc. Generally, 3D object representation are categorized into 2 classes :

(a) Boundry representation.
(b) Space partitioning representation

(a) Boundry Representation :

In this representation, any 3D object surface is represented by collection of polygon surfaces and a polygon surface is represented by collection of points, lines and curve segments in sequence.

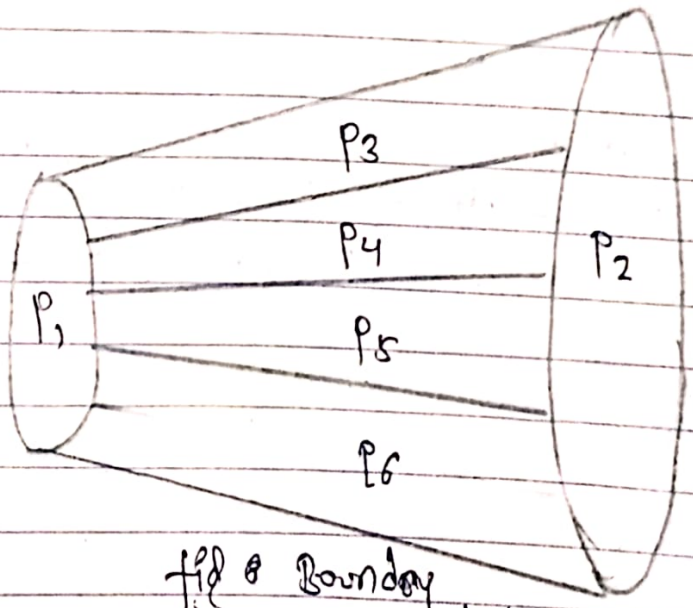


fig : Boundry Representation.

(4) Space partitioning representation :
In this representation, any 3D object can be represented by collection of individual 3D objects. For example :

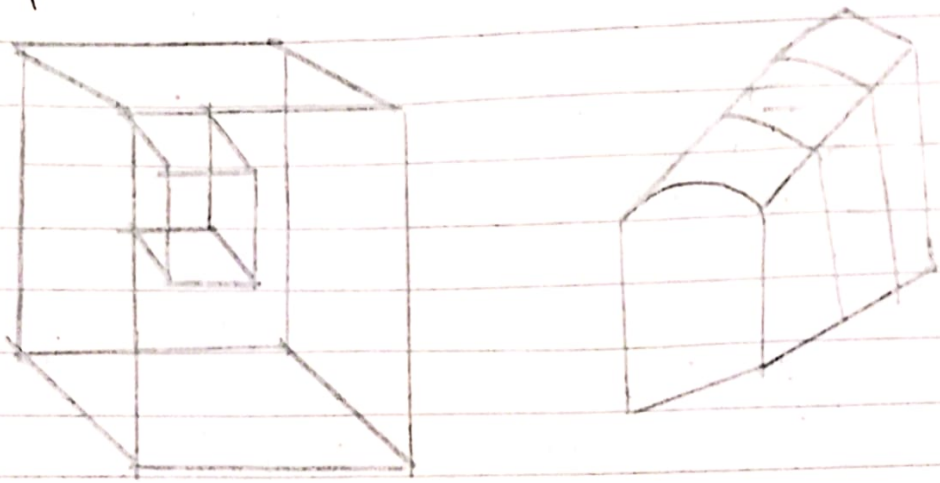


fig : space partitioning representation.

Polygon surface Representation
A ~~irregular~~ polygon surface is represented by collection of points and lines in a sequence. The polygon surface representation are :

(1) polygon table
(2) polygon messes
(3) plane equation

(1) ~~Polygon~~ table :
A polygon table is a data structure about the 3D object that provides the information about geometrical structure as well as attributes. ~~of~~ A polygon table is a combination of

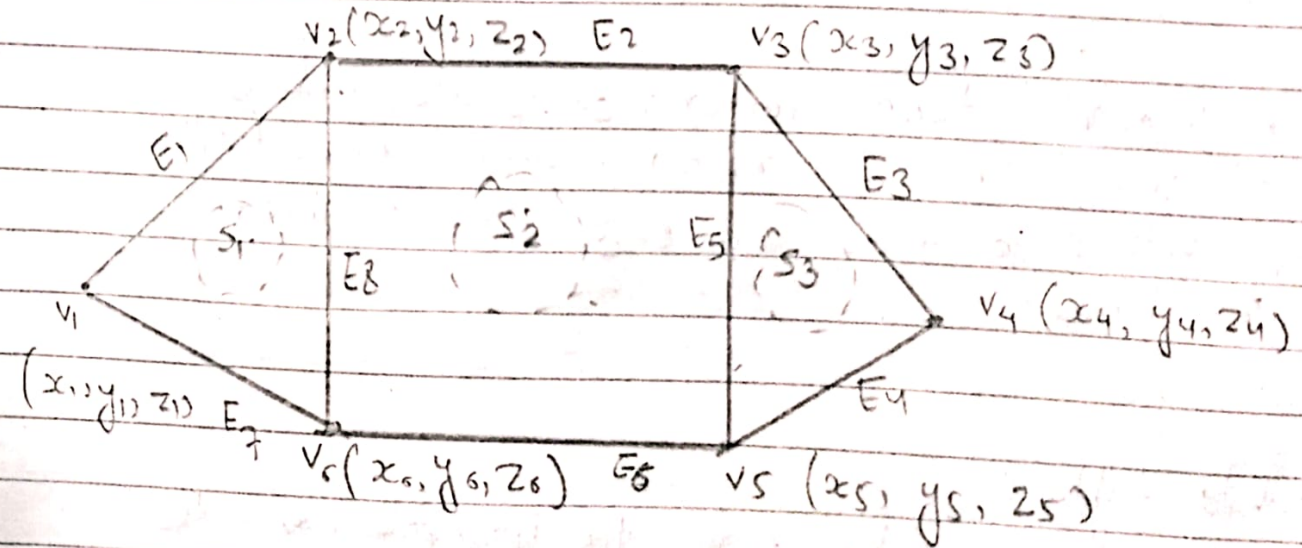
2 tables :

1. Geometrical tables
2. Attribute Table

(1) Geometrical

It provides the information about geometrical data required to form a polygon surface including vertices, edges and surfaces i.e. a geometrical table is a combination of vertex table, edge table and surface table. The vertex table stores the information about spartial co-ordinates required to represent the vertex along with space vertex pointer. Edge tables stores the information about ~~edges~~ required to form ~~up a polygon surfaces~~ edge.
vertex pointer

The surface table stores the information about edge pointers required to form a surface. Eg :



Vertex table	
Vertex	Co-ordinates
V_1	(x_1, y_1, z_1)
V_2	(x_2, y_2, z_2)
V_3	(x_3, y_3, z_3)
V_4	(x_4, y_4, z_4)
V_5	(x_5, y_5, z_5)
V_6	(x_6, y_6, z_6)

Edge table	
Edges	vertex
E_1	V_1, V_2
E_2	V_2, V_3
E_3	V_3, V_4
E_4	V_4, V_5
E_5	V_3, V_5
E_6	V_5, V_6
E_7	V_1, V_6
E_8	V_2, V_6

Surface Table	
Surface	Edges
S_1	E_1, E_7, E_8
S_2	E_2, E_8, E_6, E_5
S_3	E_5, E_4, E_3

Here, in geometrical tables, vertex, edges and surface table forms a linked list structure. where surface table is linked with edge table and edge table is linked with vertex table.

(2) Attribute table:

A attribute table stores the information about the rendering properties of 3D objects like plane eqn coefficients, specular reflection parameter, refractive index, RGB values,

opacity parameter, etc.

(2) Polygon Meshes :-

- a polygon that form by collection of some type of polygon that satisfies "a vertex is shared by at least two vertex, an edge is shared by at most two polygons and a polygon is a closed sequence of edge" is called polygon meshes.

- A polygon mesh is used to represent a polygon surface defined by the mesh function as :-

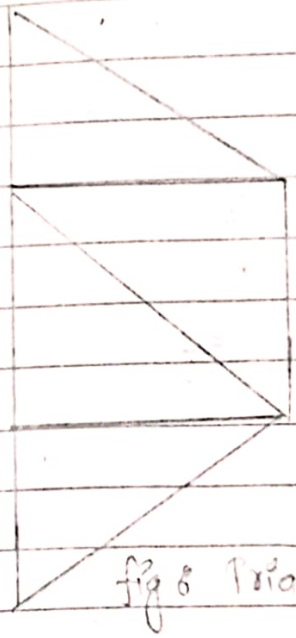


fig :- Triangular mesh function

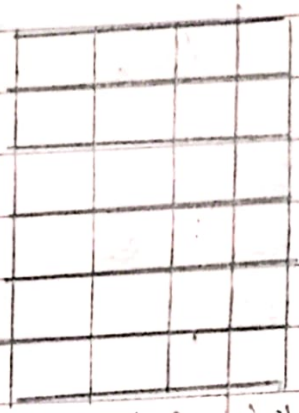


fig :- Quadrilateral mesh

(3) Plane equation

It is a mathematical representation of polygon surface a 3D object is represented by collection of polygons and a polygon is represented by a plane equation. Say

(x, y, z) be a point on plane then a plane equation is

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

where, A, B, C, D are the plane equation coefficients, they specifies the spatial orientation of the polygon surface. To represent a polygon surface we have to know the coefficient of plane equation. say (x_1, y_1, z_1), (x_2, y_2, z_2) and (x_3, y_3, z_3) bare three points on plane. Then,

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

Solving these equation using cramers rule

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

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

Here, a table stores the all coefficients of plane equation they are useful for polygon surface representation.

Wireframe representation

a type of 3D object representation in which a 3D object is represented by collection of points and lines only.

- It is not a realistic representation so it is not suitable in CG but popular on design and drafting.

- It is used to represent the skeleton of 3D object and requires less memory space.

- a wireframe represented object is defined by the table that stores the information of edges with start and end vertex.

- for example :

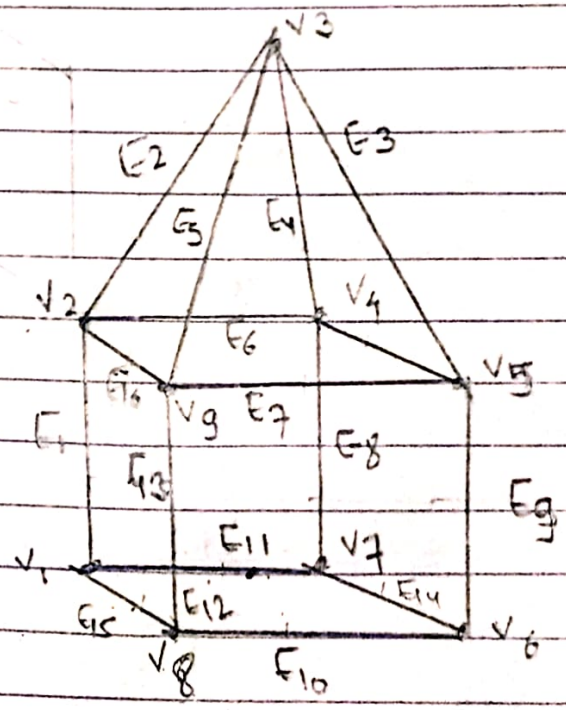


fig : Wireframe represented object

Wireframe table

Edge	vertex	
	start	end
E_1	V_1	V_2
E_2	V_2	V_3
E_3	V_3	V_5
E_4	V_3	V_4
E_5	V_3	V_9
E_6	V_2	V_5
E_7	V_9	V_5
E_8	V_4	V_7
E_9	V_5	V_6
E_{10}	V_8	V_6
E_{11}	V_1	V_7
E_{12}	V_9	V_8
E_{13}	V_9	V_8
E_{14}	V_7	V_6
E_{15}	V_1	V_8
E_{16}	V_2	V_9

→ Blobby object representation

Parametric cubic curve

- a parametic polynomial of order n is represented as,

$$P(t) = a_n t^n + a_{n-1} t^{n-1} + \dots + a_1 t + a_0$$

for $0 \le t \le 1$.

where, $P(t) = \{x(t), y(t), z(t)\}$

Here, $x(t)$, $y(t)$ and $z(t)$ are the X, Y, Z are components of $P(t)$.

Here, higher degree polynomial provides the better result but requires more memory space for storage as well as time

- The above polynomial becomes curve if and only if $n \geq 2$.
- The more suitable degree for representation of curve is $n = 3$ so,

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

where,

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

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

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

The curve represented by above relation is called parametric cubic curve.

Parametric cubic curve representation (Algorithm)

- To represent a parametric cubic curve, we need to find the coefficient of parametric cubic curve and that can be done through method of matrix interpolations.

- a parametric cubic polynomial is represented by,

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

In matrix,

$$p(t) = \begin{bmatrix} a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}$$

$$= C \cdot T.$$

where

$C \cdot T = \begin{bmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix}$ is called matrix of coefficient.

$T = \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix}$ is called matrix of parameter

Here, the coefficient matrix can be represented by combination of 4x4 and 4x1 matrix as C

$$\Rightarrow C = M \times P$$

where,

$$M = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix}$$

is called basis matrix and used to represent the characteristics of curve.

$$P = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \end{bmatrix}$$

is called control vector matrix and helps to gurde the curve shape.

The basis matrix is used to derive a blending function of a cubic curve as B.

$$u \cdot (t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix}$$

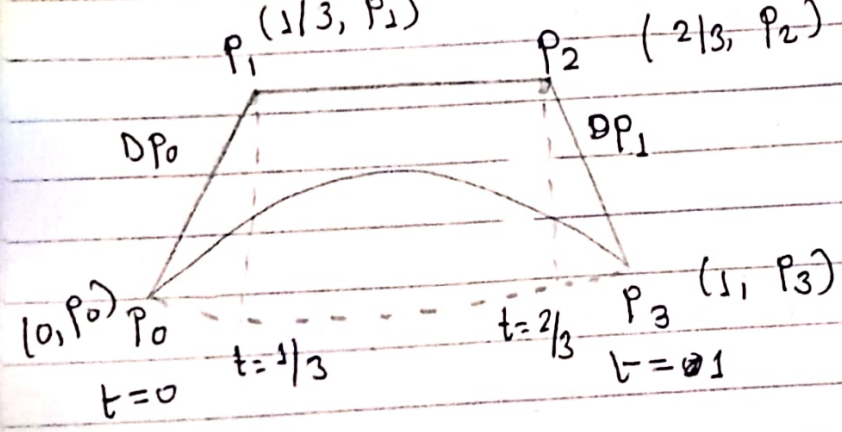
The parametric cubic curve function can be calculated as below :

$$\Rightarrow \boxed{P(t) = u(t) * P}$$

Bezier Curve

- A curve control through the two end points and tangent end point is called as bezier curve.

- More flexible to design a curve surface to represent a real environment object surface.



Say, a parametric cubic curve,

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

$$= \begin{bmatrix} t^3 & t^2 & t^1 & t \end{bmatrix} \begin{bmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix}$$

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

$= T \cdot C.$

where,

$C = \begin{bmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix}$ matrix of coefficient.

for $0 \le t \le 1$,

Now,

at $t = 0$,

$P(0) = a_0$

$P'(0) = a_1$

$a_0 = P(0) = P_0$

$a_1 = P'(0) = \frac{P_1 - P_0}{1/3 - 0}$

$= 3(P_1 - P_0)$

$\therefore a_1 = 3P_1 - 3P_0$

for $t = 1$,

$P(1) = a_3 + a_2 + a_1 + a_0$

or $P(1) = a_3 + a_2 + 3P_1 - 3P_0 + P_0$

or $a_3 + a_2 = P_3 - 3P_1 + 2P_0$ ——— (1)

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

or $3a_3 + 2a_2 + a_1 = \frac{P_3 - P_2}{1 - 2/3}$

$$\text{or } 3a_3 + 2a_2 + a_1 = 3(p_3 - p_2)$$
$$\text{or } 3a_3 + 2a_2 = 3p_3 - 3p_2 - (3p_1 - 3p_0) \quad \text{—— (ii)}$$
$$\text{or } 3a_3 + 2a_2 = 3p_3 - 3p_2 - 3p_1 + 3p_0$$

Solving eq^n (i) and (ii), we get,

$$[a_3 + a_2 = p_3 - 3p_1 + 2p_0] \times 3$$
$$3a_3 + 2a_2 = 3p_3 - 3p_2 - 3p_1 + 3p_0$$
$$- \quad - \quad + \quad - \quad -$$
$$a_2 = 3p_2 - 6p_1 + 3p_0$$

Also,

$$a_3 = p_3 - 3p_1 + 2p_0 - a_2$$
$$= p_3 - 3p_1 + 2p_0 - (3p_2 - 6p_1 + 3p_0)$$
$$= p_3 - 3p_1 + 2p_0 - 3p_2 + 6p_1 - 3p_0$$
$$= p_3 + 3p_1 - p_0 - 3p_2$$

We have,

$$a_0 = p_0$$
$$a_1 = 3p_1 - 3p_0$$
$$a_2 = 3p_2 - 6p_1 + 3p_0$$
$$a_3 = p_3 - 3p_2 + 3p_1 - p_0$$

In matrix form,

$$C = \begin{bmatrix} q_3 \\ q_2 \\ q_1 \\ q_0 \end{bmatrix} = \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_3 \\ P_2 \\ P_1 \\ P_0 \end{bmatrix}$$

$\therefore C = M_B \times P$

Here,

$$M_B = \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

P_B is called basis matrix for bazier curve.

Blending function for bazier curve Q_t is

$$u(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & -3t^3+t^2 & 3t^3-6t^2-3t & -t-3=3+1 \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & -3t^3+t^2 \end{bmatrix}$$

$$= \left[t^3 \quad (3t^2 - 3t^3) \quad (3t^3 - 6t^2 + 3t) \quad (3t^2 - t^3 - 3t + 1) \right]$$

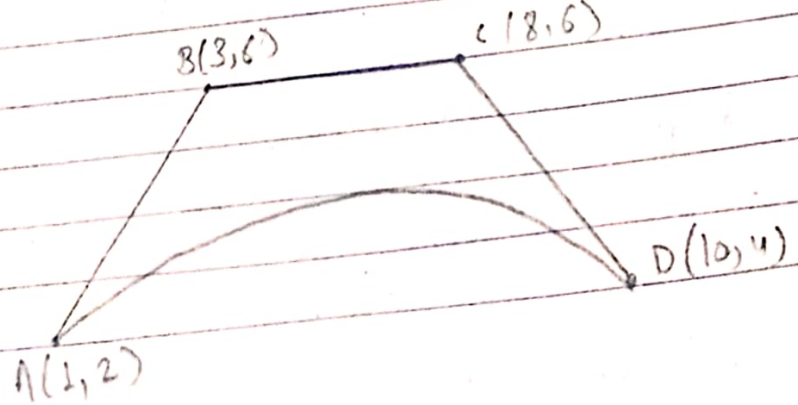
also,
bazier curve function is,

$$P_b(t) = u(t) \times P$$

$$= \left[t^3 \quad (3t^2 - 3t^3) \quad (3t^3 - 6t^2 + 3t) \quad (3t^2 - t^3 - 3t + 1) \right] \begin{bmatrix} P_3 \\ P_2 \\ P_1 \\ P_0 \end{bmatrix}$$

$$= \left[P_3 t^3 + P_2 (3t^2 - 3t^3) + P_1 (3t^3 - 6t^2 + 3t) + P_0 (3t^2 - t^3 - 3t + 1) \right]$$

Construct the bazier curve defined over the rectangle A(1,2), B(3,6), C(8,6), D(10,4)



We know,

$$P_t = a_3 t^3 + a_2 t^2 + a_1 t + a_0$$

when, ~~P_t~~ $P_t = (x_t, y_t)$ $\{x(t), y(t)\}$

~~x_t~~ = for x-component,

$x_0 = 1, \quad x_1 = 3, \quad x_2 = 8, \quad x_3 = 10$

$$y_x(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & (3t^2 - 3t^3) & (3t^3 - 6t^2 + 3t) & (3t^2 - t^3 - 3t + 1) \end{bmatrix}$$

$$x(t) = y(t) * \begin{bmatrix} x_3 \\ x_2 \\ x_1 \\ x_0 \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & (3t^2 - 3t^3) & (3t^3 - 6t^2 + 3t) & (3t^2 - t^3 - 3t + 1) \end{bmatrix} \begin{bmatrix} x_3 \\ x_2 \\ x_1 \\ x_0 \end{bmatrix}$$

$$= x_3 t^3 + (3t^2 - 3t^3) x_2 + (3t^3 - 6t^2 + 3t) + x_0 (3t^2 - t^3 - 3t + 1)$$

$= 10t^3 + 8(3t^2 - 3t^3) + 3(3t^3 - 6t^2 + 3t) + 1(3t^2 - t^3 - 3t + 1)$

$= 10t^3 - 24t^2 - 24t^3 - 9t^3 - 18t^2 + 9t + 3t^2 - t^3 - 3t + 1$

$= 10 - 6t^3 + 9t^2 + 6t + 1$

For y-component,

$y_0 = 2,\ y_1 = 6,\ y_2 = 6,\ y_3 = 4$

$$y(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & 3t^2 - 3t^3 & 3t^3 - 6t^2 + 3t & 3t^2 - t^3 - 3t + 1 \end{bmatrix}$$

$$y(t) = u(t) * \begin{bmatrix} y_3 \\ y_2 \\ y_1 \\ y_0 \end{bmatrix}$$

$$= \begin{bmatrix} t^3 & 3t^2 - 3t^3 & 3t^3 - 6t^2 + 3t & 3t^2 - t^3 - 3t + 1 \end{bmatrix} \begin{bmatrix} 4 \\ 6 \\ 6 \\ 2 \end{bmatrix}$$

$= [4t^3 + 6(3t^2 - 3t^3) + 6(3t^3 - 6t^2 + 3t) + 2(3t^2 - t^3 - 3t + 1)]$

$= 4t^3 + 18t^2 - 18t^3 + 18t^3 - 36t^2 + 18t + 6t^2 - 2t^3 - 6t + 2$

$= 2t^3 - 12t^2 + 12t + 2$

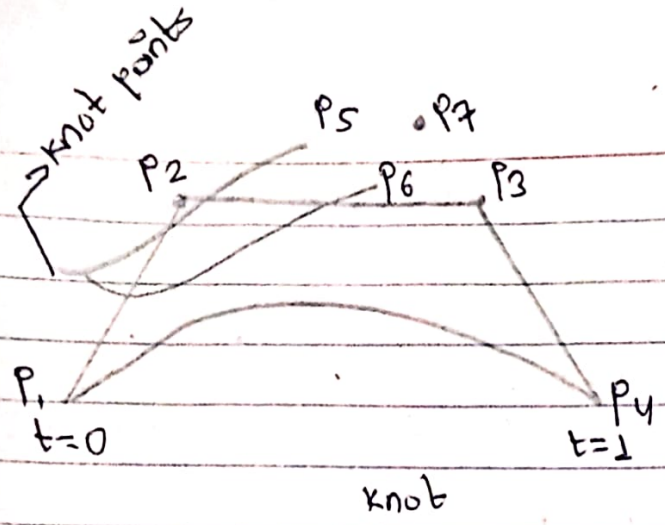
Spline curve

- A flexible curve used to design the metal strip, faces of airlines, car and real environmental 3D objects surface is called spline curve.

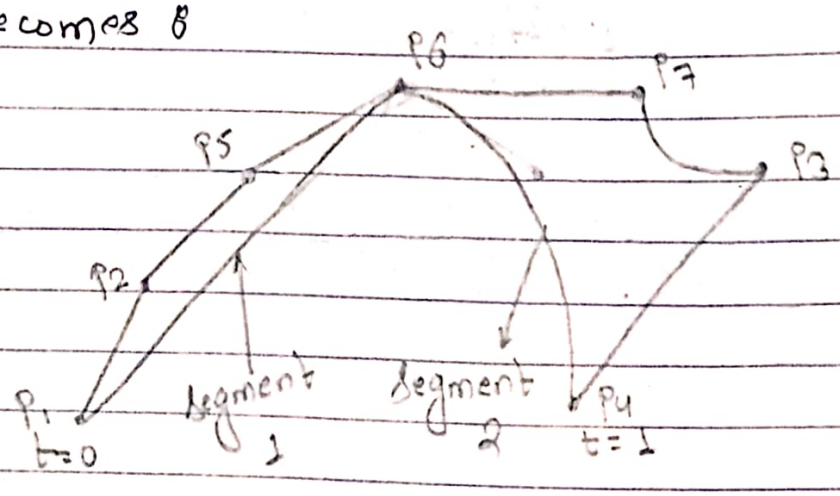
flexible metal strip

real 3D object surface

- provides the higher degree of flexibility since, it use the intermediate points to control the shape of curve over the interval those points inserted between the interval called as knot points.



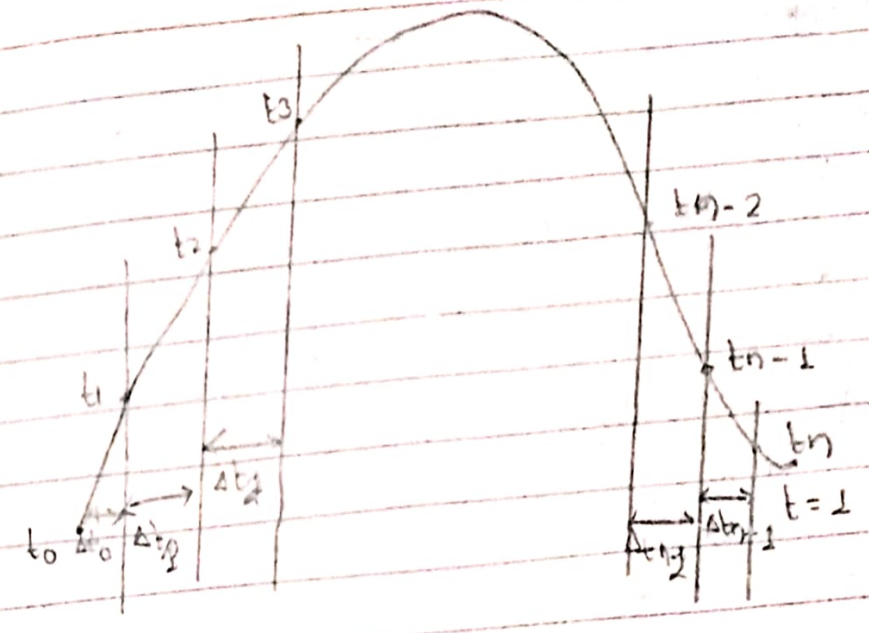
When the two points are inserted between points P_1, P_2, P_3 and P_4 as P_5 and P_6. Then, the curve becomes :



There are 2 types of B-spline curve :

(a) Uniform / periodic B-spline
(b) non-uniform / aperiodic B-spline

- a spline curve defined over an interval is divided into n equal segments by inserting sufficient number of knot points is called uniform b-spline.

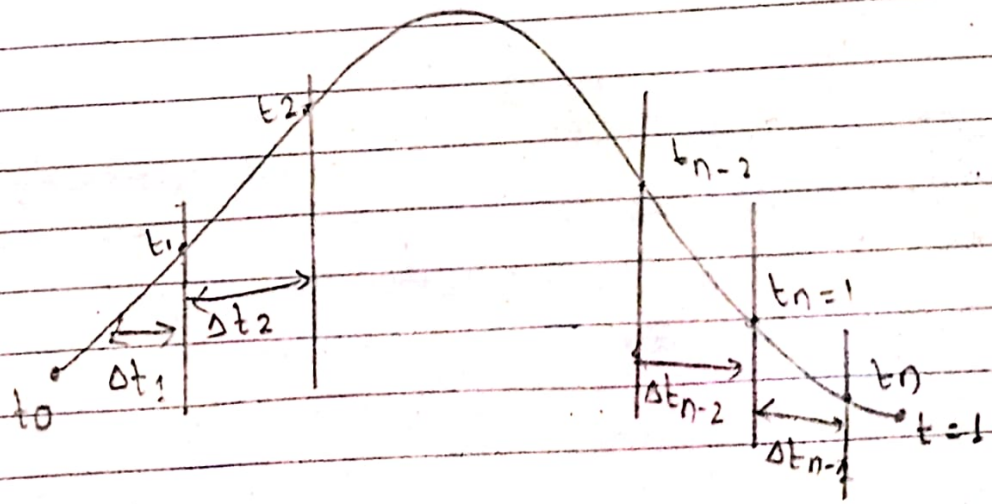


$$\Delta t_0 = \Delta t_1 = \Delta t_2 = \dots = \Delta t_{n-1}$$

~~that is not~~ not

- a spline curve defined over an interval is divided into n - equal segments is called non-uniform b-spline.

$$\Delta t_0 \neq \Delta t_1 \neq \Delta t_2 \neq \dots \neq \Delta t_{n-1}$$



A bazier curve surface (In Book)

Continuity conditions for parametric cubic curve

It specifies nature of 2 curves at joining points. There are 2 types of continuity :

(a) Geometric continuity
(b) Parametric continuity

(a) Geometric continuity :

It specifies the joining conditions between 2 curves ~~using~~ interms of order of derivative and their magnitudes. It is used to define the geometrical shape of curve obtained by joining 2 curves. It is noted through G^n where n specifies ~~of~~ order of continuity.

- order of ~~Geog~~ Geometric continuity are :

(1) zero order (G^0)
(2) First order (G^1)
(3) Second order (G^2)
(4) n^{th} order (G^n).

(1) Zero order geometric continuity (G^0)

When 2 curve segments are joined to each other then it is called as zero order geometric continuity.

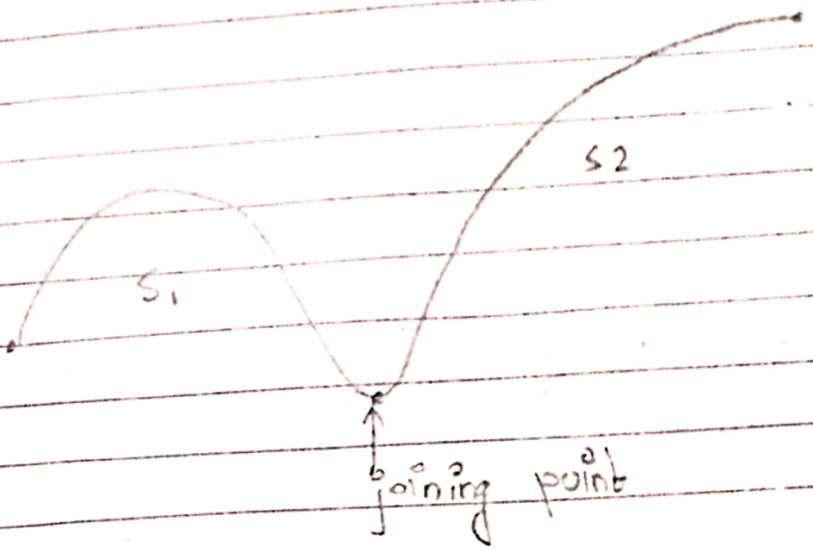


fig: Zero order geometric continuity

(2) First order geometric continuity (G^1)

A curve that satisfy the condition of G^0 and equal magnitude of at of first order derivative (slope of tangent) at joining point is called first order geometric continuity (G^1).

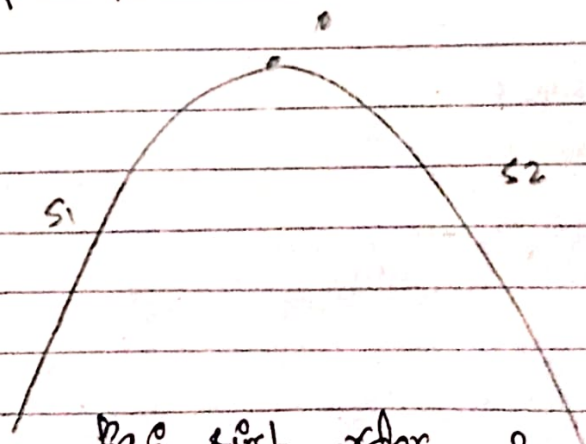


fig: first order geometric continuity.

(2) G^2 continuity :

A curve that satisfy the condition of G^1 and have equal magnitudes of second order derivative at joining points is called second order geometric continuity.

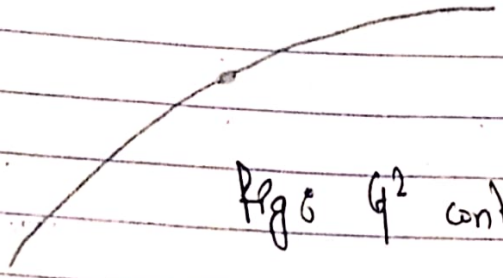


Fig : G^2 continuity

(3) n^{th} order geometric continuity :

A curve that satisfy the condition of G^{n-1} and have equal magnitudes of $(n-1)^{th}$ order derivative at joining points is called n^{th} order geometric continuity.

Parametric continuity conditions :

Notation : G replace by C.

- equal magnitude as well as direction.

sweep representation

Visible surface ~~det~~ detection (VSD)
is (hidden surface removal)
procedure that used to find the visible part of 3D object according to the viewing position is called VSD. It is need necessary to optimize the rendering process in terms of computational time and storage.

various algorithms can be used to find the visibility of 3D object. They are divided into 2 classes.
(1) Image space method (ISM)
(2) object space method (OSM)

(1) ISM:
It visibility be find by comparing each pixels ie. pixels by pixel or by point by point, comparison be made.

- More accurate
- more efficient
- requires more computational time
- requires more storage

eg: Z-buffer method

(2) Object space method (OSM):
Visibility be find by comparing group of pixels at a time ie. surface by surface

comparison be made.

- less accurate
- less efficient
- less computational time
- less storage

Eg: back-face detection

Back-face detection method
- is a object space method use to find the visibility of 3D object.
- based on plane equation that represents the orientation of surface say (x, y, z) be a any point on plane then,

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

where A, B, C, D are the coefficients of plane equation that specifies the spatial orientation of object surface.

- The surface Normal vector about the plane is

$$\vec{N} = A\vec{i} + B\vec{j} + C\vec{k}$$

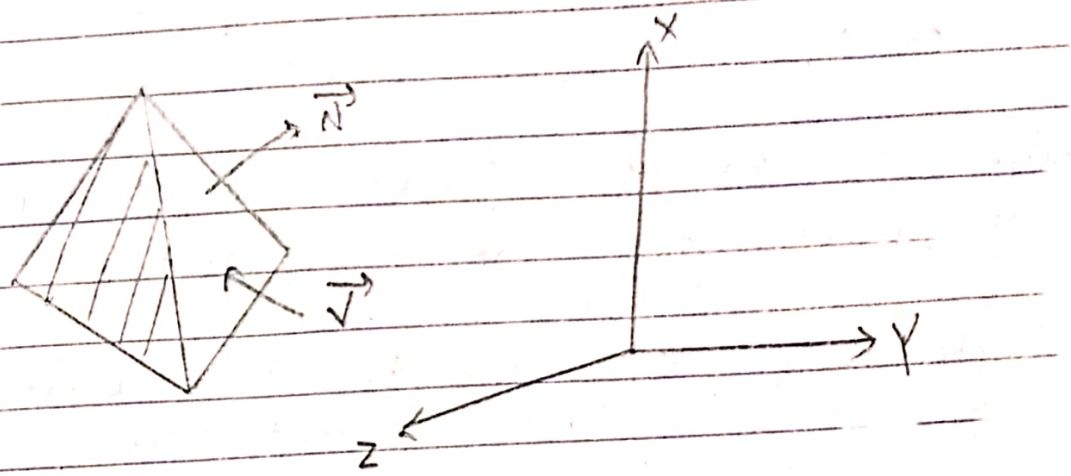
- when object is view along z-direction then viewing vector towards viewing is

$$\vec{V} = -V_z \vec{k}$$

Now, $\vec{N} \cdot \vec{V} = |\vec{N}|\,|\vec{V}| \cos\theta$

where, $\vec{N}$ and $\vec{V}$ are unit vectors then,

$$\cos\theta = \frac{\vec{N}\,\vec{V}}{|\vec{N}|\,|\vec{V}|} = \vec{N} \cdot \vec{V} = -V_z C$$



The object surface is front face if $0 < \theta < 90^\circ$

The object surface is back face if $90^\circ < \theta < 180^\circ$

Hence,

$\cos\theta = -V_z C \gtrless 0 \Rightarrow C < 0$ then surface is front face.

$\cos\theta = V_z C \gtrless 0 \Rightarrow C > 0$ surface is back face

Z- Buffer (Depth Buffer) Algorithms

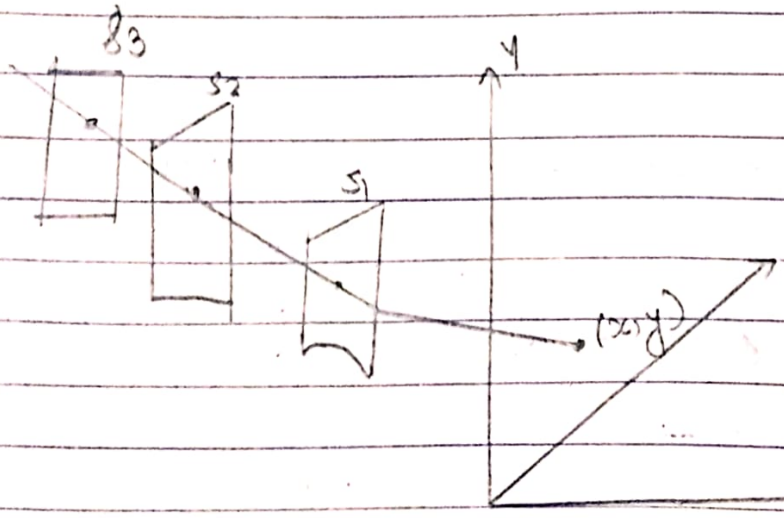
- an image space method that compares the each pixel depth (z-value) to find the visibility.

- a appropriate method for visible surface detection that uses the two dedicated buffer to store the information about depth (z- value) and intensity value. They are :

  (1) Depth Buffer
  (2) Refresh buffer

- Depth buffer stores the information about depth computed for each pixel position of projected image surface.

- Refresh buffer use to store the information about rendering intensity value (color) for that pixel.



- fig above shows the projected surfaces on xy plane

where each surface use co-planner with each other. The co-ordinate (x,y) represents the projected point and corresponding z-value is used to define the depth.

Algorithms:

1. Initiate the buffer area as:

$$\text{Depth}(x,y) = 0$$
$$\text{Refresh}(x,y) = I_{back}$$

where, I_{back} is the intensity value of background surface.

2. for each (x,y) position compute the depth value (z-value)

then,

if $z < 0$ then, update the buffer area as:

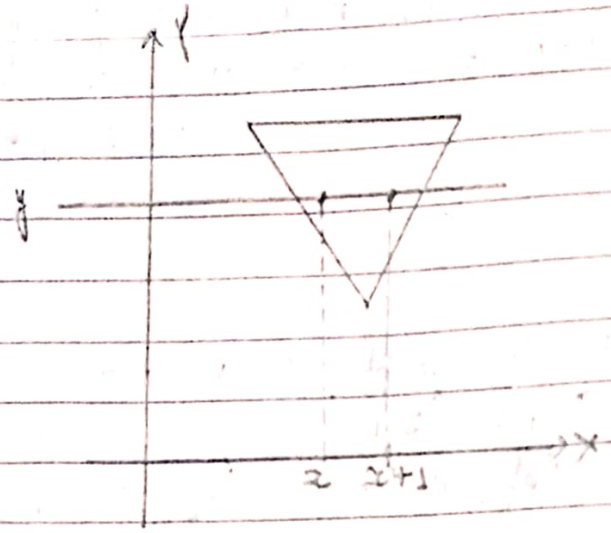
$$\text{Depth}(x,y) = z$$
$$\text{Refresh}(x,y) = I_{surface}$$

where, $I_{surface}$ is the intensity value for surface.

else,

do nothing

Computation of depth



Say (x, y, z) be a point on object surface then,

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

or, $$z = -A/C\,x - B/C\,y - D/C \quad \text{---(1)}$$

for horizontal scanning, $x \to x+1$, $y \to y$, $z = z$

$$z' = -A/C\,(x+1) - B/C\,y - D/C$$

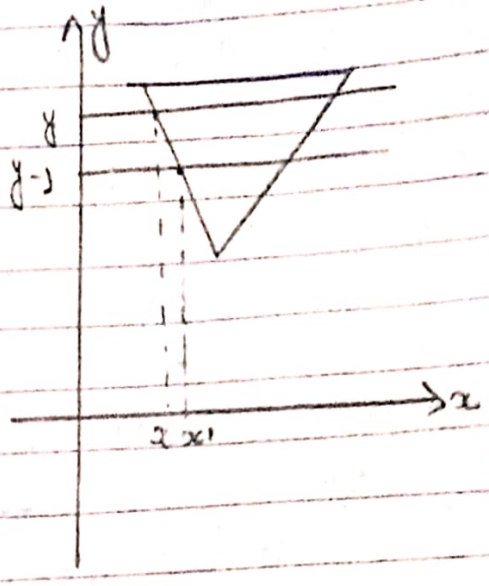
or $$z' = -A/C\,x - A/C - B/C\,y - D/C$$

or $$z' = (-A/C\,x - B/C\,y - D/C) - A/C$$

$$\boxed{z' = z - A/C}$$

for vertical scanning,

$x = x'$, $y \to y-1$, $z \to z'$



If $y = mx + c$ be the equation of edge then,

$$\frac{y-c}{m} = x$$

or, $x' = \frac{y-1-c}{m}$

or $x' = \frac{y-c}{m} - \frac{1}{m}$

or $x' = x - \frac{1}{m}$

$$\boxed{\text{or, } x' = x - \frac{1}{m}}$$

Now

$$z' = -A/C\left(x - \frac{1}{m}\right) - B/C\,(y-1) - D/C$$

$$= -A/Cx + \frac{A}{Cm} - \frac{B}{C}y + \frac{B}{C} - D/C$$

$$= \left(-A/C\,x - B/C\,y - D/C\right) + \frac{A}{Cm} + B/C$$

$$\boxed{z' = z + A/Cm + B/C}$$

(~~method~~ most efficient algorithm of VSD)

Scan line method

- a method based on both image space and object method.
- ~~Are~~ Generally used find the overlapping area between two surfaces
- Scan line are used to access the area covered by the object surface so it is called scan line method.
- This algorithms used the information stored on polygon table to define a active edge list.
- The active edge list ~~specifies~~ stores the edges or crosses of a projected scan line.

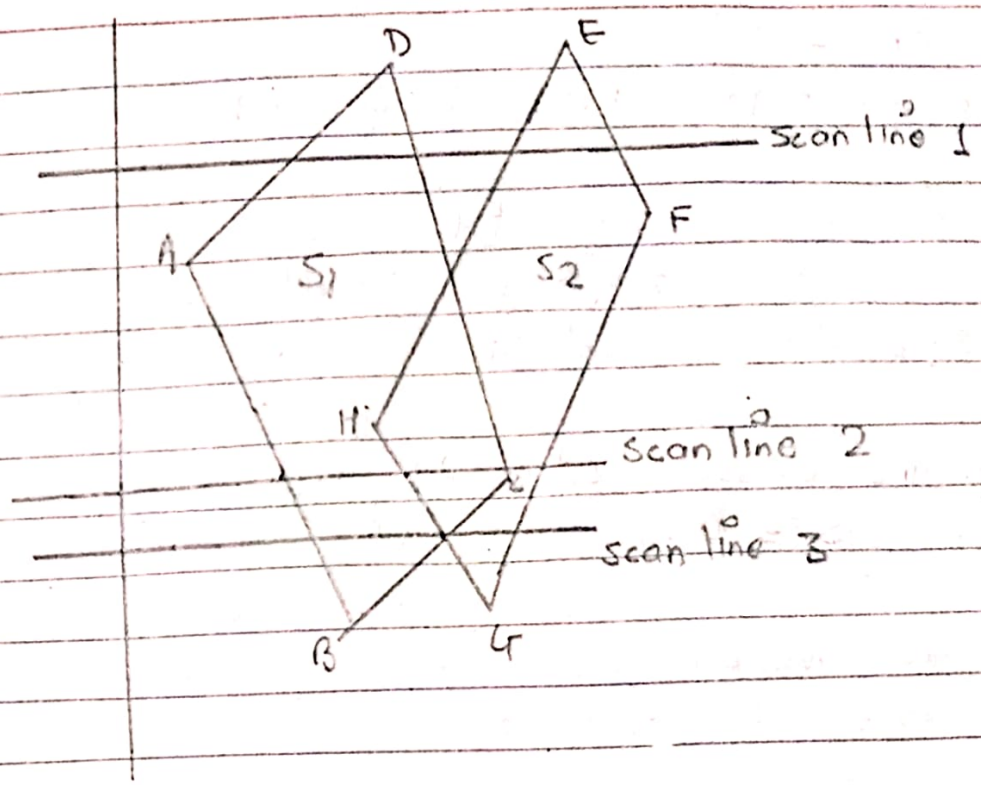
- flag bits are made to define the status of inside/outside point on surface.

- The inside/outside area be determined through the even/odd rule.

- Even/odd rule specifies "when the no. of edges cross by a line is odd then it is inside the surface otherwise it is outside surface."

- The method determines the overlapping area by observing the status of flags.

- If two or more than two flags are on at a time than it belongs to overlapping surface otherwise it belongs to single surface.



for scan line 1,

- active edge list contains edges AD, CD, EH and EF
- when scan line crosses edge AD then S_1 = ON, S_2 = OFF
- when scan line crosses edge CD then S_1 = OFF, S_2 = OFF
- when scan line crosses edge EH then, S_1 = OFF, S_2 = ON
- When scan line crosses edge EF then S_1 = OFF S_2 = OFF

So, there is no any overlapping since, only one flag is active at a time.

ON = odd
off = even

for scan line 2,
- Active edge list contains FG, GH, CD, AB

-when scan line crosses edge FG then,
~~S_2 = ON,~~ ~~S_3 = ON~~
S_3 = OFF S_2 = ~~ON~~ OFF

-when scan line crosses edge AB then,
S_1 = ON, S_2 = OFF

-when scan line crosses GH, then,
S_1 = ON, S_2 = ON

-when scan line crosses CD then,
S_1 = OFF, S_2 = ON

There is a overlapping surface between GH and CD.

for scan line 3,
- Active edge list contains AB, BC, GH, GF

-when scan line crosses edge AB then,
S_1 = ON, S_2 = OFF

-when scan line crosses edge BC then,
S_1 = ~~ON~~, OFF S_2 = ~~off~~ ON

- when scan line crosses edge GH then,
  S_1 = ON, S_2 = ON

- when scan line crosses edge GF then,
  S_1 = OFF, S_2 = OFF

This method use back face detection algorithm for single surface and z-buffer algorithm for overlapping surface to find visibility.

Depth-sort method (least priority method)
- A method based on image space method
- The depth values are computed and stores on the buffer.
- depth values stored on buffer are sorted in ascending or descending order.
- The depth value above a reference value are belongs for front face and less for back face.
- This method prioritize the pixel with less and higher care groups so, this algorithm is said to be painter algorithms.

- It is called as painter algorithms since it provides the higher care for front pixel as computer to low which is analogous to painters procedure us to generate a paint.

Algorithms :

1. Compute the depth for each pixel position and stored on buffer.

depth (x, y) = z

2. sort the depth in ascending order.

3. If z > zref then it belongs to front face else it belongs to back face.

- → area subdivision method
- → active method
- → Bsp Tree method

→ do yourself

Illumination model

- Source of light - a source that able to produce the light rays or reflect them for illumination.

-two

✓ point light source : emits the light from a point towards a direction. It has directional characteristics so, mostly used in CG.
for example : electric bulb, sun, moon, etc.

✓ Distributed light source : emits the light rays from a area. It has no any directional characteristics.

eg: florescence light, tubelight

- A procedure used to calculate the amount of intensity value of reflected light from a real 3D object surface is called illumination model.

- The calculated intensity value is applied on rendering of object. There are 3 types of basic illumination model.

1. Background light model (Ambient light model)
2. Diffuse reflection model
3. specular (phong shading) reflection model

(1) Background light model:

Basic In this approach, the 3D object surface is illuminated by light rays available on environment.
- It is a basic model for all illumination models.
- Say I_a be the intensity of environmental light rays then, the intensity value obtained from basic ambient model is:

$$I_{amb} \propto I_a$$

$$I_{amb} = k_a I_a$$

where, k_a is proportionality constant that ~~specifies~~ is called as ambient reflection coefficient.

It's value lies between 0 to 1.

(a) Diffuse reflection model :

A type of reflection in which reflector able to reflect the equi intense light in all over the direction. Is called Diffuse reflection and corresponding reflector is called idle reflector.

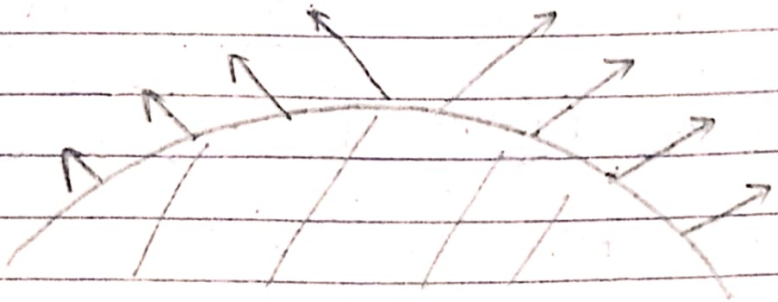


fig : Diffuse reflection

An illumination model in which 3D object surface is illuminated by light rays obtained from environment as well as diffuse reflection then it becomes Diffuse reflection model.

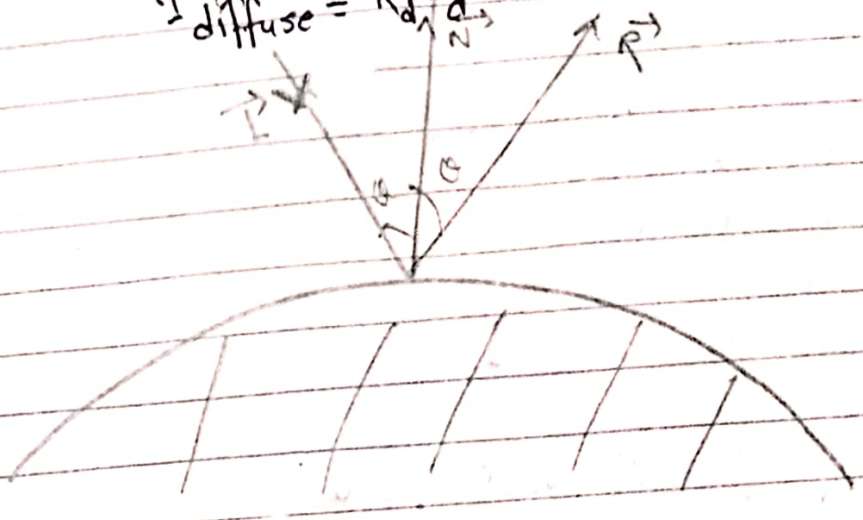
i.e.

$$I = I_{amb} + I_{diffuse}$$

Say, I_d be the intensity of light used on diffuse reflection then,

$$I_{diffuse} \propto I_d$$

$$I_{diffuse} = K_d I_d$$



According to Lambert cosine rule, the intensity value of obtained from diffuse reflection is corrected by $\cos\theta$ where θ is angle of incident

Now, $I_{diffuse} = K_d I_d \cos\theta$

where,

K_d is called as diffuse reflection coefficient whose value lies in between 0 to 1.

Say, $\vec{d}$ = unit vector directed towards light source.
$\vec{N}$ = unit surface normal vector.
$\vec{P}$ = unit vector directed towards direction of reflection.

Now,

$$\vec{N} \cdot \vec{L} = |\vec{N}|\,|\vec{L}| \cos\theta$$

$$\cos\theta = \frac{\vec{N} \cdot \vec{L}}{|\vec{N}|\,|\vec{L}|}$$

$$= \vec{N} \cdot \vec{L}$$

$$\therefore I_{diffuse} = K_d I_d (\vec{N} \cdot \vec{L})$$

The total intensity for diffuse reflection model is

$$I = I_{amb} + I_{diffuse}$$

$$\therefore I = K_a I_a + K_d I_d (\vec{N} \cdot \vec{L})$$

(3) Specular reflection model :

A type of reflection in which reflector reflects the bright spot of light or high intense light towards a particular direction. that is called as specular reflection. for eg : disco light, light rays from shinny surface, etc.

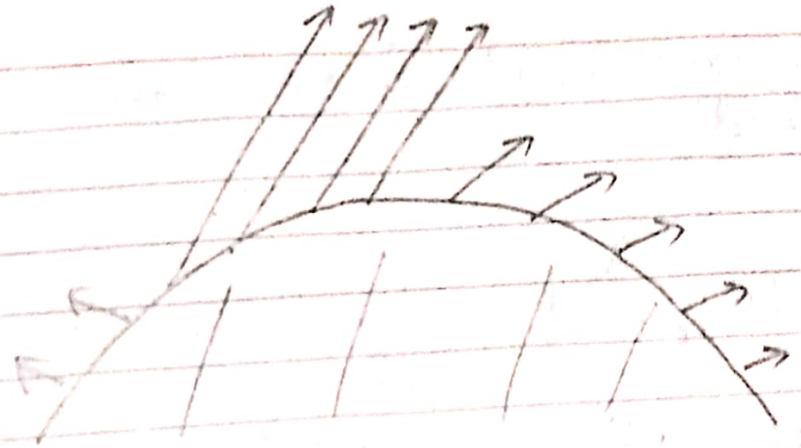


fig 6 specular reflection

A type of illumination model in which light rays obtained from environment, diffuse reflection as well as specular reflection are used to illuminate 3D object surface. Then, such model is called specular reflection model.

$$I = I_{amb} + I_{diffuse} + I_{spe} \quad \text{——— (1)}$$

Let I_L be the light rays used on specular reflection then, $I_{specular} \propto I_L \cos^{n_s}\phi$

~~$I_{spe} = w(\theta) I_L$~~

where,

η_s = Specular reflection parameter

$\eta_s \simeq 0$ for rough surface and

$\eta_s = 1$ for shiny surface

i.e. η_s lies between 0 to 1.

ϕ = angle between direction of viewing and reflection.

Now,

$$I_{spe} = W(\theta) I_l \cos^{n_s} \phi$$

where,

$W(\theta)$ = specular reflection coefficient and it depends on angle of incident.

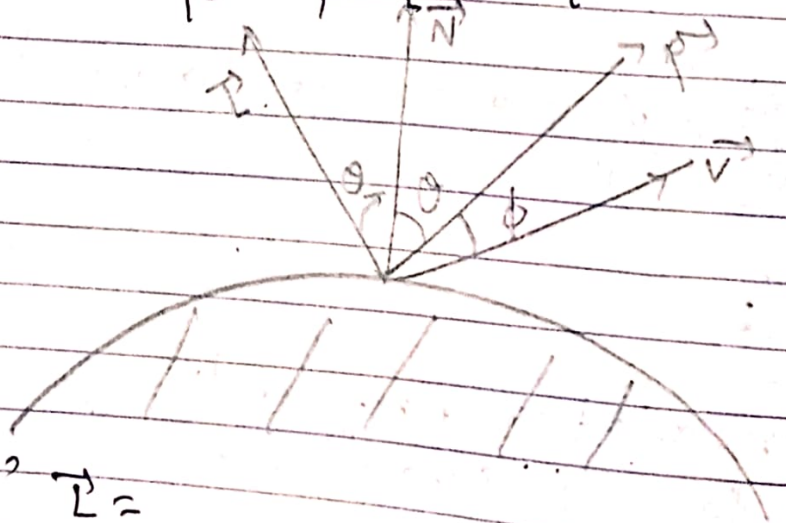
when $\theta \simeq 0$, then $W(\theta) \simeq 0$.

when $\theta \simeq 90°$, then $W(\theta) \simeq 1$.

Here, $W(\theta)$ lies between 0 to 1 and can be replaced by K_s.

Then,

$$I_{spe} = K_s I_l \cos^{n_s} \phi$$



Say,

$\vec{L}$ =

$\vec{N}$ =

$\vec{R}$ =

$\vec{V}$ =

Now, $\vec{R} \cdot \vec{V} = |\vec{R}||\vec{V}| \cos\theta$

$$\cos\phi = \frac{\vec{R} \cdot \vec{V}}{|\vec{R}||\vec{V}|}$$

$$= \vec{R} \cdot \vec{V}$$

$$= \vec{V} \cdot \vec{R}$$

Now, $I_{spe} = k_s I_l (\vec{V} \cdot \vec{R})^{n_s}$

Hence,

Total intensity value on specular reflection model is

$$I = I_{amb} + I_{diff} + I_{spe}$$

$$= k_a I_a + k_d I_d (\vec{N} \cdot \vec{L}) + k_s I_l (\vec{V} \cdot \vec{R})^{n_s}$$

Surface rendering method <polygon rendering>

{3D clipping} {3D viewing}

What do you mean by solid modeling? Explain the process for solid modeling with example.

Ans 6 The process of representation of solid object unambiguously is called solid modeling. It represent only one object.

For solid representation, we divide each space into two regions interior and exterior to it separated from each other by boundry of object.

Various methods of solid modeling are 6

(a) Sweep representation 6

Used for creating solid with uniform thickness in a particular direction and axis symmetric solid by translating and rotation sweeping. Sweeping requires a surface to be moved on a trajectory.

(b) Boundry representation 6

In this representation, object is defined in terms of its surface boundaries, vertices, surface, edges and faces. It is commonly used in practice.

(c) Spatial partitioning representation 6

It is a process of diving a space into two or more disjoint subset. In other word, space partitions, divide space into non-overlapping region. Any point in the space can be then identified to lie in exactly one of

the region.

(2) How can you represent 3D viewing ? Explain the with equation and practical application.

Ans : In 2D, we specify a window on 2D world and a view port on 2D view surface. object in world are clipped against window and one then transformed into view port for display.
Added complexity caused by :
(a) Added dimension
(b) Display device are only 2D.

Solution is accompolished by introducing projection that transform 3D object in 2D plane.
In 3D, we specify view volume (only the object within view volume will appear in display on output device. one are (clipped from display) in world, projection onto projection plane and view port on view surface.

So, object in 3D world are clipped against 3D view plane. So, then are projected. The content of projection of view volume onto projection plane called window are then transformed into view port for display.

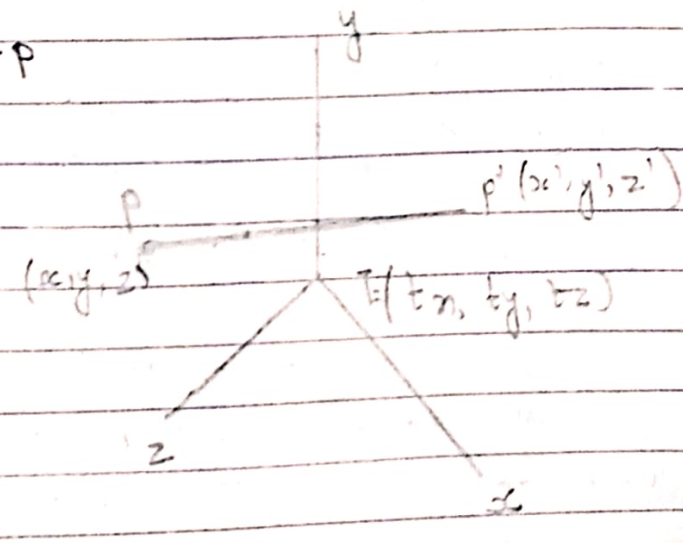
(3) Explain following term with practical application.

(a) 3D translation :

A point is translated from $P(x, y, z)$ to $P'(x', y', z')$ with matrix operation.

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

or, $P' = TP$



(2067)

(1) Computer graphics. application of computer graphics.
(2) Derive window to view port transformation coefficient matrix. application of this matrix.
(3) Explain (a) 3D rotation
(b) 2D shear
(4) line clipping algorithm and applications.
(5) DDA ? How can you draw line using this algorithm ?
(6) How can you represent 3D object ? How can you draw line using this algorithm.
(7) Explain about plain equation method. explain which algorithm is better for hidden surface removal.

OR

Explain about depth buffer method. Justify that is better than plane equation method.

(8) How curves be generated ? Explain it with suitable algorithm.
(9) Consider 256 pixel x 256 scan lines images with 24-bit true color. If 10 minutes video is required to capture, calculate the total memory required ? Why intensity assignment is required ?
(10) Why shedding is required in the CG ? Explain in detail about constant intensity shading.

OR

List different types of shading models. Explain in detail about Gouraud shading model.

(2067)

(1) random scan display system ? Block diagram and explain in detail.

(2) Homogeneous coordinates ? explain with equation and application.
(3) Explain : (a) 3D mirror
(b) 2D Rotation
(4) Explain circle clipping algorithm. Where do you require circle clipping algorithm ?
(5) How can you draw circle ? Explain with algorithm.
(6) Explain about polygon table. How can you apply in case of computer animation. ?
(7) polygon mesh ? explain application of polygon mesh with eg.
(8) Justify that hidden surface removal is required in CG. Explain in detail about depth buffer method.
(9) OR
Explain about scan line method. Just that it is better than depth buffer method.
(10) Consider 256 pixels X 512 scan lines image with 24-bit true color. If 20 minutes vide is required to capture, calculate the total memory required ? What is the color intensity model ?
(11) Explain about phong shading. How can you modify phong shading model ? OR
Explain about Gourand shading model. compare with phong shading model.

2068

(1) W' Raster scan display system. block diagram and explain.
(2) How can you represent 3D viewing ? Explain with eq^n & application.
(3) Explain : (a) 3D Translation
(b) 2D mirror
(4) Where do you require ellipse clipping algorithm ? Explain about ellipse clipping algorithm.

(5) How can you draw circle & Explain with algorithm.
(6) Explain about polygon table & How can you apply in case of virtual reality?
(7) B-spline modeling & Explain its process with example.
(8) Hidden surface removal is required in computer graphics is very important, Justify. Explain in detail about scan line method.

OR

Explain in detail about scan line method. Justify that it is better than plane equation method.

(9) Consider 1024 pixels x 1024 scan lines images with 24-bit true color. If 10 minutes video is required to capture, calculate the total memory required & How can you incorporate shadow in computer graphics.
(10) Difference diffuse reflection and specular reflection. Why do we require shading model & Explain.

OR

explain about phong shading model. Compare with Gouraud shading Model.

(2069)

(1) random scan display system with adv and disadv.
(2) Why homogeneous co-ordinate used for transformation computations in CG? Explain
(3) Difference window port and view port. How are lines grouped into visible, invisible and partially visible categories in 2D clipping & Explain.
(4) Polygon. Types. Explain.
(5) Difference periodic B-spline curves and non-periodic B-spline curve.

6. Explain z-buffer algorithm for removing hidden faces?
7. Difference Incremental algorithm over DDA with eg.
8. Define (any 2)
   a) Video controller
   b) 3D viewing
   c) Raster graphics
   d) List priority.
9. Explain simple illumination model with eg.

   or

   Explain gourand shading model with eg.
10. Explain virtual reality and applications in CG.

(2070)

1. Random scan system? Explain operation of it with diagram.
2. procedure to fill interior of ellipse with specific pattern.
3. Show that 2 successive reflection about any line passing through the co-ordinate origins equivalent to a single rotation about origin.
4. Line clipping? procedures.
5. Illustrate window to view point transformation with eg.

   or

   procedure to implement highlighting as blinking operation.
6. Why polygon description is considered as standard graphics objects? Explain imp of polygon table.
7. Model Bezier curve. Explain imp. of bazier curve in graphical modeling

   or

   procedure to perform 2-point perspective projection of an object

(8) Solid modeling? Explain its procedures.
(9) Explain area sub-division method for visible surface detection
(10) explain steps for computer animation.

(2071)

(1) Computer graphics and applications
(2) Explain scan conversion algorithms with eg.
(3) Explain 2D and 3D transformations.
(4) Explain window to view port transformation with application.
(5) Explain with algorithm of generating curves.
(6) set up procedure for establishing polygon tables for any input set of data points defining an object.
(7) Define algorithm for visible surface detection.
(8) Define (any 2) & (a) 3D viewing
(b) polygon meshes
(c) Boundry Representation
(d) Sweep Representation
(9) Explain visible surface detection with algorithms.
(10) Virtual reality and animation? Explain.

OR

scan line algorithm with eg.

(2072)

(1) Computer graphics. applications.
(2) Use BLA to draw a line having end points (25, 20) and (15, 10).
(3) Construct Bazier curve of order 3 and 4 polygon vertices A(1, 1), B(2, 3), C(4, 3) and D(6, 4).

Derive mid-point algorithm.
Differentiate flood fill and boundry fill method
explain with applications: (a) 2D shearing (b) 3D viewing
homogeneous co-ordinates & Rotate a triangle A(5,6), B(6,2) and C(4,1) by 45° about an bitrary pivot point (3,3).

OR

Explain about Diffuse Reflection model.
projection & Differentiate betn parallel and prospective projection.
Given clipping window P(0,0), Q(30,20), S(0,20) use Cohen Sutherland algorithm to determine the visible portion of line A(10,30) and B(40,0).
Explain steps for computer animation & its application in computer science.

(2073)

Raster scan display system with its architeracture.
Derive an equation to draw a line using DDA algorithm when slope is greater than 1.
Derive an expression for BZA application
How can you draw circle using mid-point circle algorithm? Explain with algorithm.
Explain scan line algorithm for removing hidden surfaces.
Explain with practical application of
a) 2D Rotation
b) Computer animation

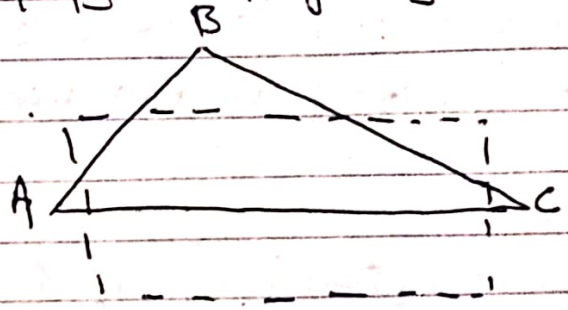
(8) Derive window to view port transformation coefficient matrix explain the application matrix

or

Globby objects & How is it represented & explain wireframe representation object.

(8) Calculate total memory required to store a 10 minute video in a SVGA system with true color and 25 fps.

(9) Explain polygon clipping in detail. By using sutherland-Hodgemen polygon clipping algorithm, clip the following polygon



(10) Virtual reality & explain importance of virtual reality and its application.

(2074)

(1) Digitize end points (10, 18), (15, 8) using DLA.

(2) What are object space and image space method of hidden surface removal & Describe the ~~scaling~~ back face detection method of hidden surface removal.

(3) perform scaling transformation to the triangle and vertices A (6, 9), B (10, 5), C (4, 3) with scaling factor $S_1 = 3$ and $S_2 = 2$.

(4) Explain about parametric cubic curve. Describe the properties of Bazier curve.

(5) Explain the visual effect that occurs when during animation of a Gourand shading polyhedron, the centre of highlight moves from one vertex to another along an edge.

OR

Illustrate difference between orthographic (parallel) and perspective projections.

(6) Define window and view port. Describe 3 dimension windows to view port transformation with matrix representation for each step.

(7) consider a raster scan system having 12 inch by 10 inch screen with resolution of 100 pixels per inch in each direction. If the display controller of this system refreshes the screen at the rate of 50 frames per second, how many pixels could be accessed per second and what is the access time per pixel of the system?

(8) Find the composite transformation matrix for reflection about a line $y = mx + c$.

(9) construct a polygon table for a object with six vertex, 8 edge and 3 surface

OR

· Explain role of computer graphics on animation. Define clipping operation with example.

(10) Digitize an ellipse with center (20, 20) and x-radius = 8 and y-radius = 6.

OR

Find the new co-ordinates of a unit cube 90° rotated about an axis defined by its end points A(2, 1, 0) and B (3, 3, 1).