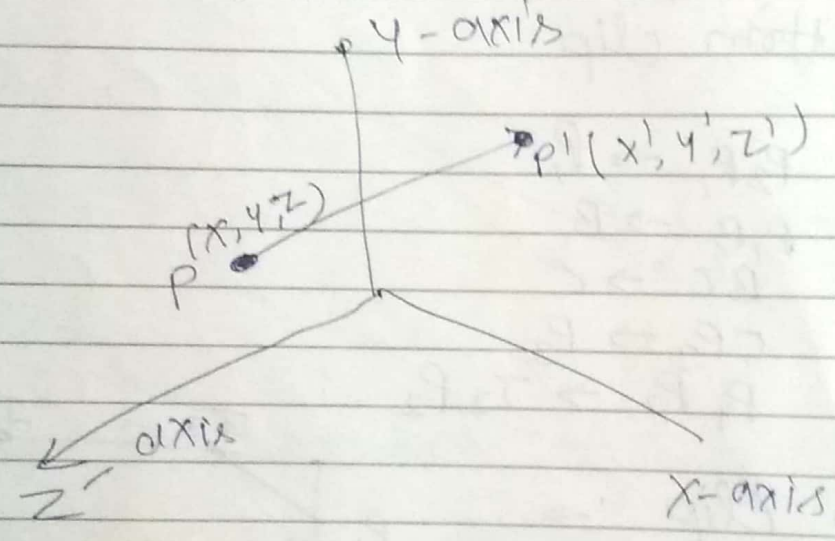
3-D Transformations

To represent any object in 3D we need 3 parameters x-coordinate which represents width, y-coordinate represents height.
z-coordinate is depth.

3D translation :-



translation :→

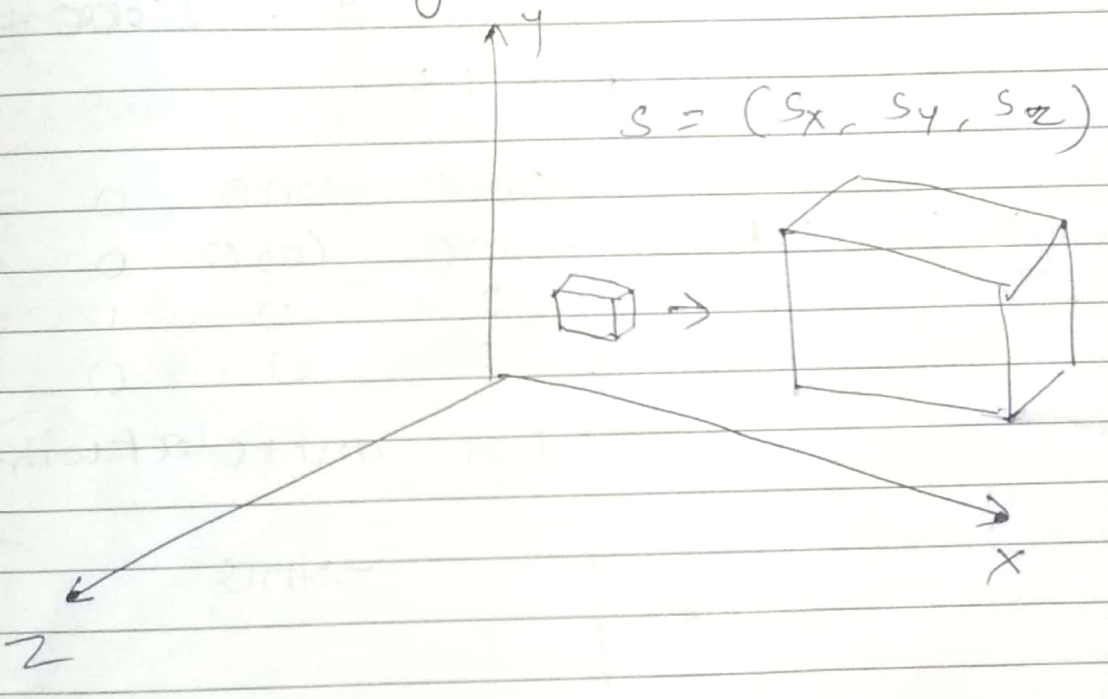
$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

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

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

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

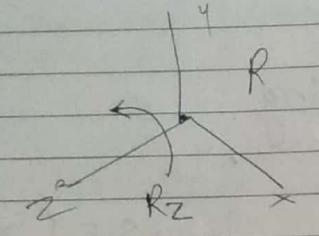
3D Scalling



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

3D Rotation :-

Rotation about z-axis :- z unchanged



$z' = z$

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

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

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

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

→ for anticlockwise

$$\times \begin{bmatrix} & -\sin\theta & \\ & & \\ & & \end{bmatrix}$$

for clockwise →

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

Rotation about x axis

x - unchanged

$x' = x$

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

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

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

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

$P' = R_x(\theta) P$

Rotation about y axis

y unchanged

$y' = y$

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

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

$$R_y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

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

Three diamentional projection:—

Projection :— when we want to draw a 3D object on a monitor, we have to convert the world co-ordinate into screen coordinate. For this we have to ~~convert~~ project a 3D object on a 2D plane. So projection is a process of representing a three dimentional object or scene into two diamentional medium ~~that~~

" projection is nothing but a shadow of the object.

Importatant terms related to projection:—

1) Center of projection :— the point where projection is taken. it can either be light source or eye position

2) Projection plane :— The plane on which projection of the object is formed.

3) Projectors :— lines emerging from center of projection and hitting the projection plane after passing through a point in the object to be projected

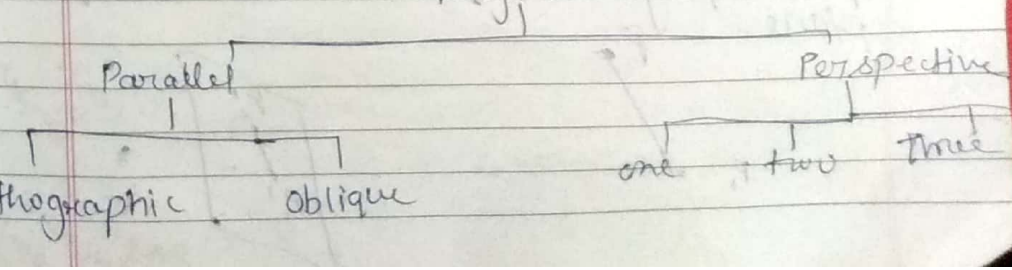
So, the plane geometric projections or simply projection of the object are formed by the intersection of lines called projectors on a plane called the projection plane. projectores are lines from an arbitary point called the center of projection, through each point in an object

" If the center of projection is at a finit distance in three diamensional space, the result is a perspective projection."

" If the center of projection is an infinity, all the projectors are parallel and the result is parelel projection."

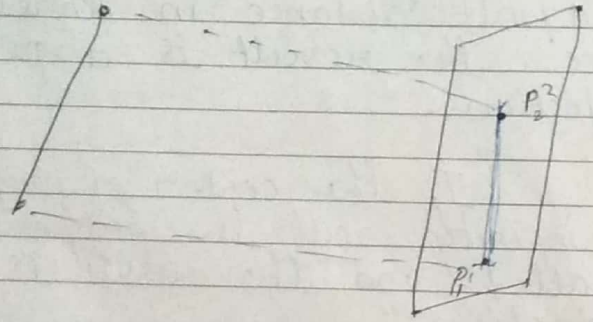
So we have two type of projections.

1. Parallel projection
2. Perspective projection.

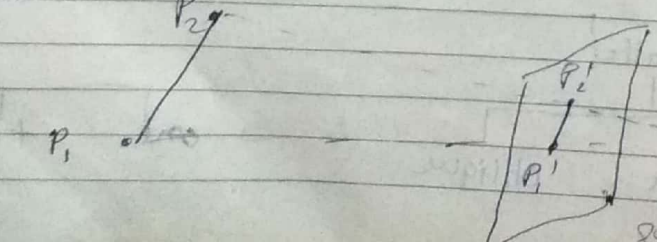


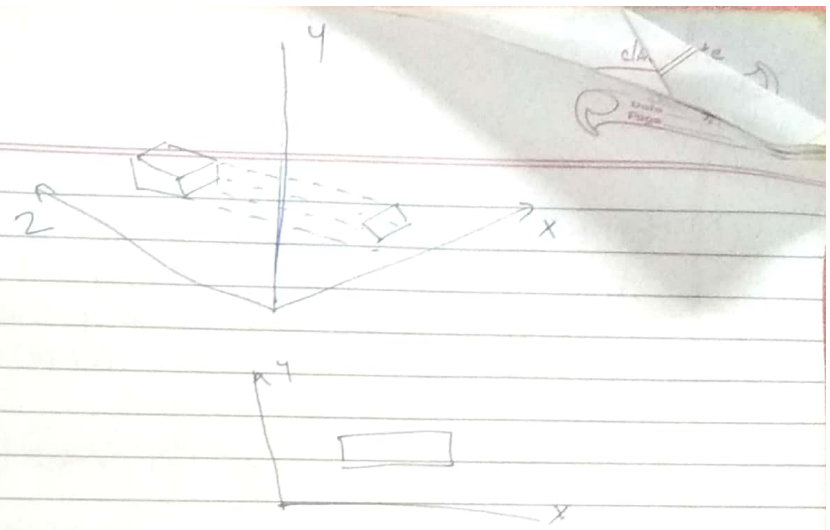
(a) Parallel projection :-

in a parallel projection coordinates position, ~~coordinate~~ are transformed to the view plane along parallel lines. these are linear transforms that are useful in blueprint to produce scale drawing of three-dimentional objects.



(2) Perspective projection :- For a perspective projection, object are transformed to the view plane along lines the converge to a point called ~~to~~ center of projection. The projected view of an object is determined by calculating the intersection of the projection lines with the view plane.

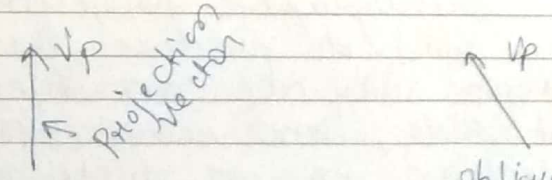




Parallel projection :-

we can specify a parallel projection with a projection is perpendicular to the view plane, we have an orthographic parallel projection. Otherwise we have an oblique parallel projection.

There are two type of parallel projection :-



Orthographic projection (a)

(b)

algorithm for parallel ~~from~~ Projection vector

Begin $x \leftarrow x - z * S_{xp}$

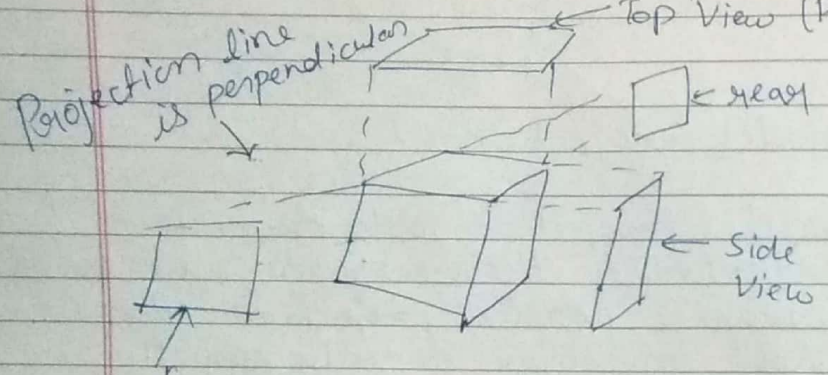
$y \leftarrow y - z * S_{yp}$

Return;

End;

orthographic parallel project

it is a part of the parallel projection in which the center of projection lies at infinity.



Front & Side view → elevation

Orthographic projection are most often used to produce the front, side and top view of an object, front, side, and rear orthographic projection of an object are called elevation and top orthographic projection is called plan view.

Orthographic :-
1. Isometric
2. Dimetric
3. Trimetric

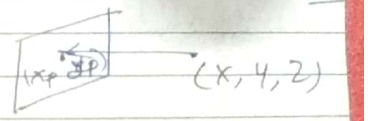
oblique :-
1. Cavalier
2. Cabinet

1. Orthographic

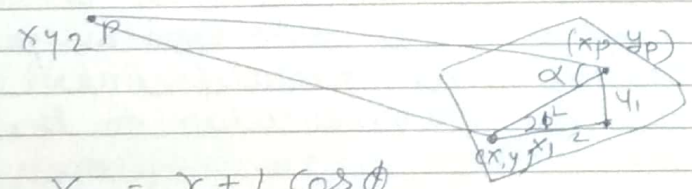
$x_p = x$

$y_p = y$

[because of perpendicular



2. Oblique projection [line projection line is not Perpendicular]



$x_p = x + L\cos\phi$

$y_p = y + L\sin\phi$

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

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

$$L = \frac{z}{\tan\alpha} \qquad L = zL_1$$

$$x_p = x + z(L_1\cos\phi)$$

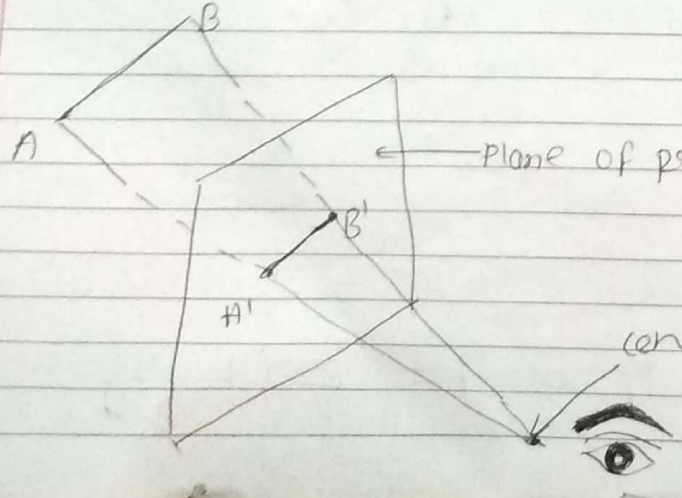
$$y_p = y + z(L_1\sin\phi)$$

$$M_{parallel} = \begin{bmatrix} 1 & 0 & L_1\cos\phi & 0 \\ 0 & 1 & L_1\sin\phi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

oblique projection — an oblique projection is obtained by projecting points along parallel lines that are not perpendicular to the projection line.

— x — x —

Perspective Projection :-

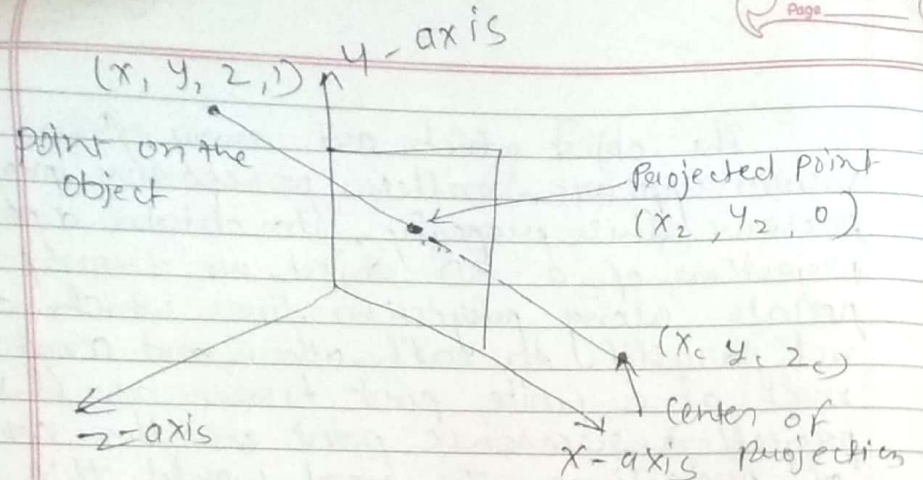


the object which are away from the viewer appear smaller. perspective projection preserve this property. to obtain a perspective projection of a 3D object, we transform points along projection lines which are not parallel to each other and converge to meet at a finite point known as the projection reference point or the center of projection. In real world this center of projection is human eye. The projected view is obtained by calculating the intersection of the projection lines with the view plane.

transformation matrix for perspective projection.

Let us consider the center of projection is at (x_c, y_c, z_c) and and the point on object is (x_1, y_1, z_1) then the parametric equation for the line containing these point can be given as :-

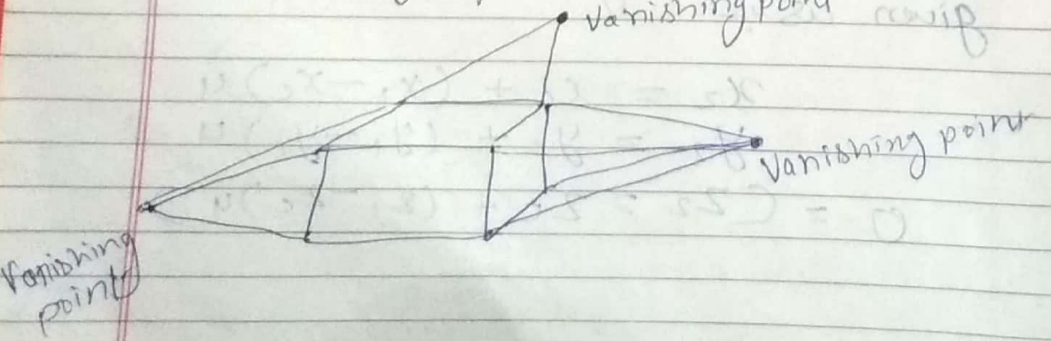
$$x_2 = x_c + (x_1 - x_c)u$$
$$y_2 = y_c + (y_1 - y_c)u$$
$$0 = (z_2 = z_c + (z_1 - z_c)u)$$



So the homogeneous matrix can be written as :-

$$[x_2 \; y_2 \; z_2 \; 1] = [x, y, z, 1] \begin{bmatrix} -z_c & 0 & 0 & 0 \\ 0 & -z_c & 0 & 0 \\ x_c & y_c & 0 & 1 \\ 0 & 0 & 0 & -z_c \end{bmatrix}$$

Vanishing points :-



Certain set of parallel lines appear to meet at some point on the projection plane. these points are called Vanishing points

Three type of perspective projection :-

1. one - point projection
2. Two - point projection
3. Three - point projection

projection vary from a minimum of one to a maximum of three.

(1) Single - point :- one - point perspective occurs when one principal axis intersects the plane of projection

(2) Two point perspective :- Two point or two principal vanishing point perspective projection occurs when the plane of projection intersects exactly two of the principal axis.

(3) Three - point :- Three point projection occurs when projection plane intersects all three of principal axis.

	Parallel	Perspective
1)	Co-ordinate position of the object are transformed into the view plane along the parallel line	1) Co-ordinate position of the object are transformed into the view plane along to the point called center of projection.
2)	Relative proportion of object are maintained	2) it does not maintane Reative proportion
3)	it gives accurate view of the objects	3) it does not give accurate view of the object.
4)	It does not give realistic view	4) it gives realistic view of the objet.
5)		

Hidden Surface elimination :-

The surface that are blocked or hidden from view must be 'removed' in order to construct a realistic view of the 3D scene. The identification and removal of these is called the "hidden surface problem".

The problem of hidden-surface removal or equivalently visible surface determination is solved by applying different hidden-surface algorithm. These algorithm determines the lines edges surface or volumes that are visible or invisible to an observer located at a specific point in the space. These algorithm are broadly classified according to whether they deal with object difinations directly or with their projected images.

These two approaches are called :-

① object space methods or object precision method a

② Image space method

① object space method :- O.S.M is implemented in the visual-co-ordinate system, considering the geometrical

relationships between the actual objects it composes objects and parts of objects to each other with in scene definition to determine which of the surface is seen we should label as visible very precise results, generally to the precision of the machine are available ~~as visible~~. These result can be satisfactory enlarged many times. objectspace method are generally used in line-display algorithm.

Image Space method :-

image space method is implemented in the screen co-ordinate system, considering the relationships between the image of the object in their final configuration at the pixel level. Thus in an image-space algorithm, visibility is decided point by point at each pixel position on the view plane. most hiddem line/surface algorithm use image-space method.

① Object Space-method

(1) Back face detection method
② Painter's algorithm
③ Roberts "

② Image Space methods :-

i) Area subdivision method
~~ii)~~ Depth buffer method
iii) Octree method
iv) Scan line method
(v) Ray tracing method.

Back face detection method :-

Back-face detection is a fast object space algorithm based upon the inside-outside test for identifying the back-face of a polyhedron. Suppose a point (x,y,z) is inside a polygon surface with plane parameters A, B, C and D. If

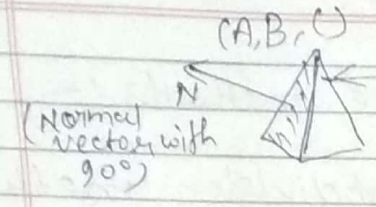
$$Ax + By + Cz + D < 0$$

when an inside point is along the line of sight to the surface, the polygon must be a back face.

the direction of the light face can be identified by examining the result

$N.V > 0$ where

N is the normal vector to the polygon surface with Cartesian Component A, B, C.



Viewing direction (V)

Inside outside test

Condition for inside point :-

$$Ax + By + Cz + D < 0 \Rightarrow \text{inside}$$

plane equation

$V \cdot N > 0$ surface backface

$V \cdot N < 0$ frontface

$\vec{N} = A\hat{i} + B\hat{j} + C\hat{k}$ ($\hat{i},\hat{j},\hat{k}$ = unit vector)

$\vec{V} = V_z\hat{k}$ (1, if it not visible in V.N)

if both are positive

$\vec{N} = C\hat{k}$

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

upside = -Negative

downword = +ve

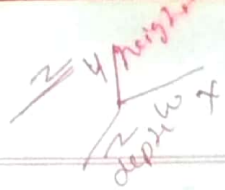
If $\vec{N} = -2\hat{k}$

$\vec{V} = +3\hat{k}$

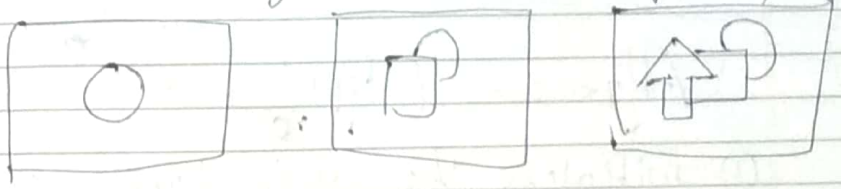
$\vec{N} \cdot \vec{V} = -6 < 0$ then front face

$N \cdot V > 0$ Same direction (back face)

$N \cdot V < 0$ opposite " (front face)



Painter's algorithm (depth sort or priority algo)



The painter's algorithm is also called as depth sorting algorithm or priority algorithm. The basic idea of the depth sort algorithm developed by Newell and Sancha, is to paint the polygon into the frame buffer in order of decreasing distance from the view point this process involves following basic fuctions :-

1. Sorting of polygons in order of decreasing depth
2. Resolving any ambiguities this may cause when the polygon's z extents overlap splitting polygons z extents overlap it is splitting polygons if necessary.

Step 1 Sorting the surface according to the z-value

Step 2 Scan conversion of sortest surfaces

3. Scan conversion of polygon in order, starting with the polygon

of greatest depth.

Algo :- (depth buffer)

① initialize the depth buffer and refresh buffer so that for all buffer position.

depth (x, y) = 0

and

refresh (x, y) = $I_{backgnd}$

② for each position on each polygon surface, compare depth values to previously stored values in the depth buffer to determine visibility

i) Calculate the depth z for each (x, y) position on the polygon

ii) If z > depth (x, y) then set

depth (x, y) = z

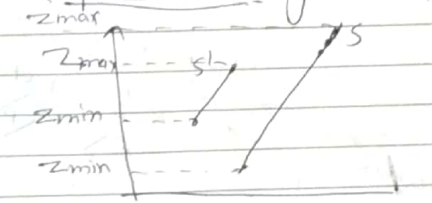
refresh (x, y) = I_{surf} (x, y)

Where $I_{backgnd}$ is the value for the background intensity and I_{surf} (x, y) the projected intensity value for the

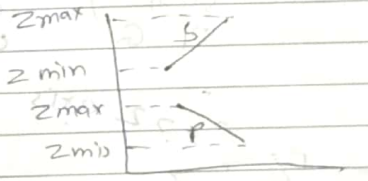
depth buffer

surface at pixel position (x, y). after all surface have been processed the depth buffer contains the corresponding intensity values for those surface.

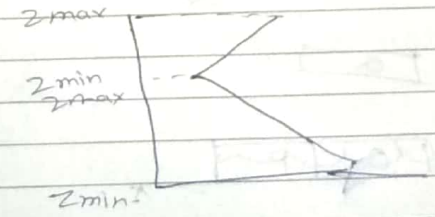
depth sorting or painter's algorithm

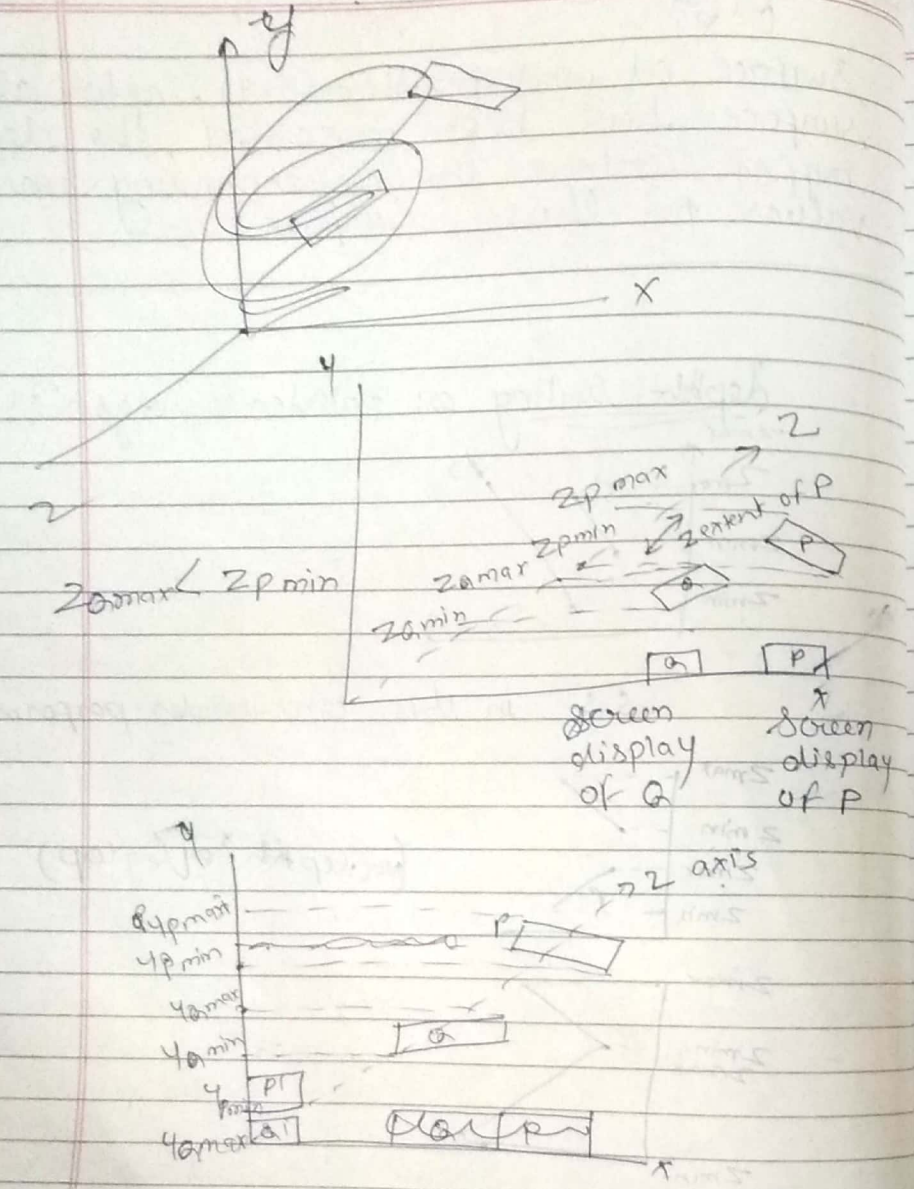


s s' in this can reorder perform



(No depth overlap)





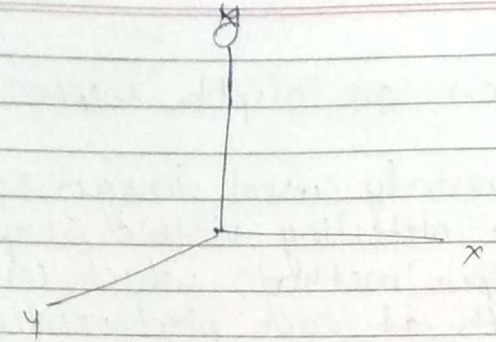
Abhinav missing DCS, PS
Sanjay - OS

Z-buffer or depth buffer :-

A commonly used image-space approach to detecting visible surface is "depth buffer method" which compares surface depth at each pixel position on the projection plane. This procedure is usually measured from the viewing system. This technique was originally proposed by CATMULL.

Z buffer is a simple extension of the same size as the frame buffer is set up which holds depth information each element of the depth buffer corresponds to a pixel in the framebuffer. and initially holds the maximum depth in the scene. The frame buffer is cleared to the background colour.

As each polygon is scan-converted the depth at each pixel is calculated and compared with corresponding depth in the depth buffer. if the depth is less than that stored in the depth buffer. then that ~~buffer~~ pixel is set to the polygon depth. if the polygon depth is greater than the depth buffer depth ~~of~~ then that pixel is not written to the frame buffer.



buffer (x,y) → z value

refresh buffer → intensity of xy

z (0-ordinate (Normalized)

→ $0 < z < 1$ ← front clipping plane

back cliping plane

② ~~Initialize~~

for calculating z value

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

↗ plane value

$$z = \frac{-Ax + By + D}{C}$$

(x,y) (x+1, y)

$$z' = \frac{-A(x+1) - By - D}{C}$$

$$z' = \frac{-Ax - A - By - D}{C}$$

$$z' = z - \frac{A}{C}$$

advantages :-

① it is easy to implemant
② It can be implemented in hardware to overcome the speed problem
③ Since the algorithm processes objects one at a time, the total number of polygone in a picture can be arbitrarily large.

disadvantage :-

① it requires an additional buffer and hence the large memory.
② it is time consuming process