Unit III

2D, 3D Transformations and Projections

2-D transformations: introduction, homogeneous coordinates, 2-D transformations - Translation, scaling, rotation and shear, rotation about an arbitrary point.
3-D transformations: introduction, 3-D transformations - Translation, scaling, rotation and shear, rotation about an arbitrary axis.
Projections : Parallel (Oblique: Cavalier, Cabinet and orthographic: isometric, diametric, trimetric) and Perspective (Vanishing Points – 1 point, 2 point and 3 point)

Text Books:

1. S. Harrington, "Computer Graphics"‖, 2nd Edition, McGraw-Hill Publications, 1987, ISBN 0 – 07 – 100472 – 6.
2. Donald D. Hearn and Baker, "Computer Graphics with OpenGL", 4th Edition, ISBN-13: 9780136053583.
3. D. Rogers, "Procedural Elements for Computer Graphics", 2nd Edition, Tata McGraw-Hill Publication, 2001, ISBN 0 – 07 – 047371 – 4.

Reference Books:

1. J. Foley, V. Dam, S. Feiner, J. Hughes, "Computer Graphics Principles and Practice"‖, 2nd Edition, Pearson Education, 2003, ISBN 81 – 7808 – 038 – 9.
2. D. Rogers, J. Adams, "Mathematical Elements for Computer Graphics"‖, 2nd Edition, Tata McGraw Hill Publication, 2002, ISBN 0 – 07 – 048677 – 8.

Unit : III

Page No. 1

2D, 3D Transformation projections.

[Reference : TB1]

2-D Transformations :-

Introduction :-

- The process of changing the position of object or performing certain alteration of the picture or may be any combination of these is called as "Transformation".
- Transformation allows to uniformly alter the entire picture. This process is very useful in Hand drawing techniques. where it is usually easier to change a small portion of a drawing than it is to create an entirely new picture.
- e.g. Suppose the manager wants to alter the scale of the graphs in a report or the architect wants to view a building from a different angle or the animator needs to change the position of a character all these alterations can be easily performed by using geometric transformations, because the graphic image has been coded as numbers & stored within the computer.

Matrices :-

- Computer Graphics images are generated from a series of line segments which are represented by the co-ordinates of their end points.

— Certain changes in an image can be easily made by performing mathematical operations on these co-ordinates.

Matrix multiplication :-

- example :-

$$A = \begin{vmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{vmatrix} \qquad B = \begin{vmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{vmatrix}$$

$C = 3$ $\qquad$ $R = 3$

A dimension (3×3) & B (3×2) So we multiply two matrix ~~because~~ ~~B~~ ~~has~~ the dimensions of resultant matrix C will be (3×2) & the elements in resultant matrix will be calculated as.

columns of A = Row's of B

$3 = 3$

$C(1,1) = A(1,1) \cdot B(1,1) + A(1,2) \cdot B(2,1) + A(1,3) \cdot B(3,1)$

$= (1) \cdot (1) + (4) \cdot (3) + (7) \cdot (5)$

$= 1 + 12 + 35$

$= 48$

like

$C(1,2) = A(1,1) \cdot B(1,2) + A(1,2) \cdot B(2,2) + A(1,3) \cdot B(3,2)$

$= (1) \cdot (2) + (4) \cdot (4) + (7) \cdot (6)$

$= 2 + 16 + 42$

or $= 60$

Two Dimensional Transformations :-

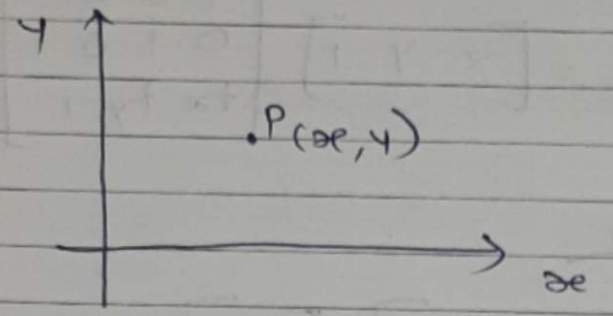
1] Translation -

— The process of changing the position of an Object is called as Translation.

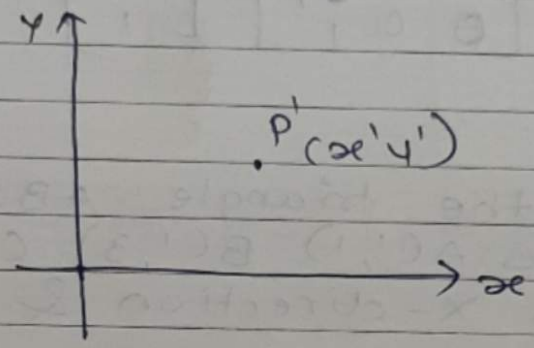
— This is done by adding to each point the amount by which the picture is

is to be shifted.

- e.g consider point P(x,y), where x, y are the co-ordinates



- This point P is to be shifted to new position P' (x', y')



- Here the point P has been shifted by some units in X-direction as well as in Y-direction. let t_x, t_y be the quantities by which the point P has to be shifted in X & Y direction respectively.

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

$t_x, t_y \rightarrow$ Translation factors.

The Translation matrix will be

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

- The transformation matrix should only contain the translation factors & no coordinate values.

— for the translation of point $P(x, y)$ to $P'(x', y')$ the translation matrix will be,

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

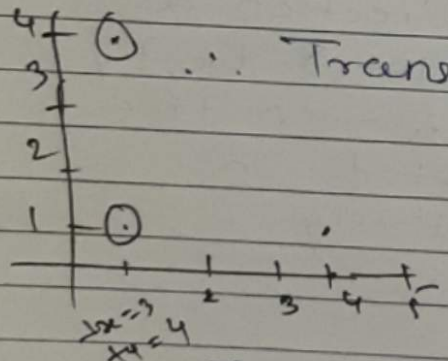
OR

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

Example :-

— Translate the triangle ABC with co-ordinates A(1,1) B(1,3) C(5,0) by 2 units in X-direction & 3 units by Y-direction.

— Translation factors are $t_x = 2$, $t_y = 3$



∴ Translation matrix

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 3 & 1 \end{bmatrix}$$

— The new point co-ordinates are

$$A' = A[T]$$

$$[x' \; y' \; 1] = [1 \; 1 \; 1] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 3 & 1 \end{bmatrix}$$

$$= (1+0+2) \; (0+1+3) \; (0+0+1)$$

$$= [3, 4, 1]$$

$[x' \; y' \; 1]$
$[x \; y \; 1]$
A = (1,1)
(1,1,1)
B = (1,3)
(1,3,1)

$$B' = B[T]$$

$$= \begin{bmatrix} 2 & 3 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 3 & 1 \end{bmatrix}$$

$$= (1+0+2)\ (0+3+3)\ (0+0+1)$$

$$= [3\ 6\ 1]$$

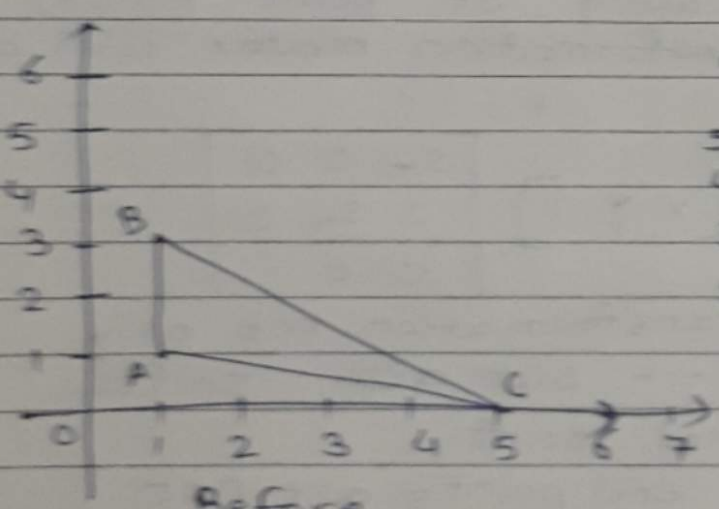
$$C' = C[T]$$

$$= \begin{bmatrix} 5 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 3 & 1 \end{bmatrix}$$

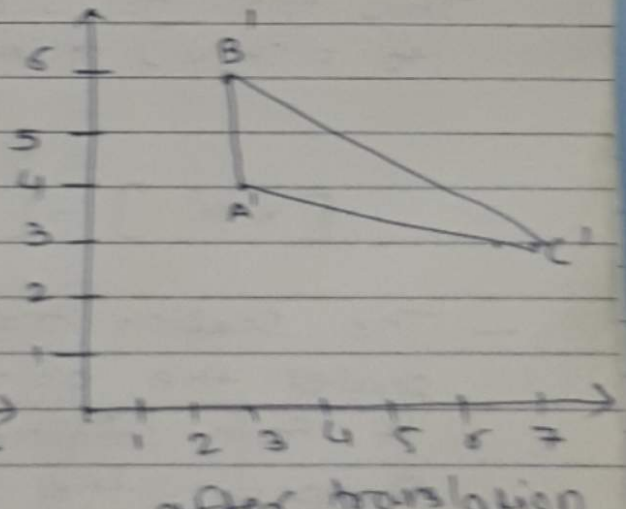
$$= (5+0+2)\ (0+0+3)\ (0+0+1)$$

$$= [7\ 3\ 1]$$

— Thus the coordinates of shifted triangle will be $A'(3,4)$, $B'(3,6)$ $C'(7,3)$



Before Translation



after translation

Scaling :-

- This transformation is used to alter the size of an object i.e either to magnify or reduce the size of an object.
- e.g Consider an object point $P(x, y)$. This point is to be scaled to $P'(x', y')$ Then the scaling factors will be S_x and S_y for the x & y co-ordinates respectively.
- To obtain the scaled object point the original points co-ordinates is to be multiplied by the scaling factor as

$$x' = x \cdot S_x$$
$$y' = y \cdot S_y$$

The scaling matrix will be

$$S = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

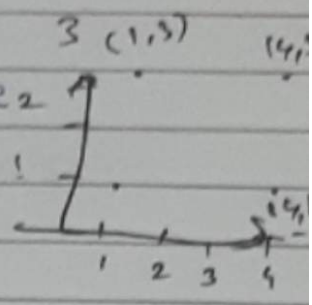
Thus, for the scaling of point $P(x, y)$ to $P'(x', y')$ transformation matrix will be

$$[x' \; y' \; 1] = [x \; y \; 1] \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- The scaling transformation not only scales the object but also shifts it from original point.
- Suppose the end points are not origin & the object is to be shifted whether right, left, up or down then it depends on the sign & magnitude of S_x & S_y.

if

$S_x > 0$, increase in size
$S_x < 0$, Reduction in size
$S_x = 0$, uniform scaling.



Example-

Magnify the triangle A(0,0), B(1,1) C (5,2) to twice its size.

sol^n:-

To magnify the triangle the scaling matrix will be needed. The scaling factors will be $S_x = 2$, $S_y = 2$

The Scaling matrix will be

$$S = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

for $A' = A[S]$

$$= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

$B' = B[S]$

$$= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 2 & 1 \end{bmatrix}$$

$$C' = C[S]$$

$$= \begin{bmatrix} 5 & 2 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 10 & 4 & 1 \end{bmatrix}$$

- the co-ordinates of scaled triangle will be $A'(0,0)$, $B'(2,2)$ $C'(10,4)$

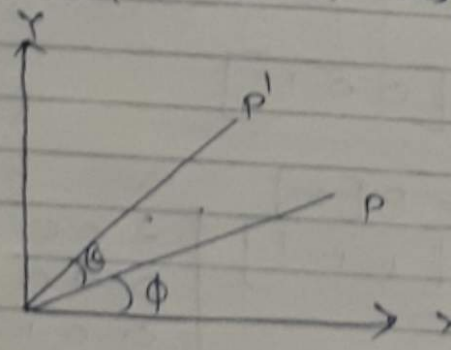
Rotation -

- This type of transformation allows the movement of an object along the circular path.
- In this case the object can be rotated by a given angle in either clockwise or counterclockwise direction.

Consider point $P(x,y)$. let the angle of rotation be θ°

(i) Transformation Matrix for Counterclockwise Direction -

To rotate the $P(x,y)$ in counterclockwise direction by θ° with respect to origin



- when the object is rotated about origin, it gets rotated with fixed radius.

$\therefore \quad x = r\cos\phi$

$y = r\sin\phi$

$\phi \rightarrow$ initial angle of P with respect to origin.

$x' = r\cos(\theta+\phi)$

$y' = r\sin(\theta+\phi)$

$x' = r\cos(\theta+\phi)$

$= r(\cos\theta\cos\phi - \sin\theta\sin\phi)$

$= r\cos\theta\cos\phi - r\sin\theta\sin\phi$

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

$(\because r\cos\phi = x$

$r\sin\phi = y)$

$y' = r\sin(\theta+\phi)$

$= r(\sin\theta\cos\phi + \cos\theta\sin\phi)$

$= r\cos\phi\sin\theta + r\sin\phi\cos\theta$

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

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

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

In the matrix form it can be represented as —

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

Thus, for rotation of point $P(x, y)$ to $P'(x', y')$ the transformation matrix will be,

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

ii) The Transformation Matrix for clockwise Direction :-

- To rotate the point p(x,y) in clockwise direction by θ° with repect to origin.
- The sign of an angle determines the direction of rotation.
- To rotate the image in a counter clockwise direction, positive angle is used hence for clockwise direⁿ negative angle will be used.

so the rotation matrix will be

$$R = \begin{bmatrix} \cos(-\theta) & \sin(-\theta) & 0 \\ -\sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$\cos(-\theta) = \cos\theta$

$\sin(-\theta) = -\sin\theta$

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

Thus for rotation of p(x,y) to p'(x',y') in clockwise direⁿ the transformation matrix will be

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

θ	0°	$30^\circ(\pi/6)$	$45^\circ(\pi/4)$	$60^\circ(\pi/3)$	$90^\circ(\pi/2)$	$180^\circ(\pi)$	$270^\circ(3\pi/2)$	$360^\circ(2\pi)$
$\sin\theta$	0	$\frac{1}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{\sqrt{3}}{2}$	1	0	-1	0
$\cos\theta$	1	$\frac{\sqrt{3}}{2}$	$\frac{1}{\sqrt{2}}$	$\frac{1}{2}$	0	-1	0	1

Example -

Rotate the triangle A (1, 1), B (2, 2) C (6, 3) in counter clockwise direction by 90°

→ To rotate the triangle in counter clockwise diren, the rotation matrix will be

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

$\therefore \theta = 90^\circ$

So $\cos 90^\circ = 0$

$\sin 90^\circ = 1$

$$\therefore R = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

for $A' = A[R]$

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

$$= \begin{bmatrix} -1 & 1 & 1 \end{bmatrix}$$ ← $(0-1+0)\ (1+0+0)\ (0+0+1)$

$= -1\ \ 1\ \ 1$

$$B' = B[R]$$

$$= \begin{bmatrix} 2 & 2 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= [-2 \quad 2 \quad 1]$$

$$C' = C[R]$$

$$= [6 \quad 3 \quad 1] \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= [-3 \quad 6 \quad 1]$$

The co-ordinates of triangle will be $A'(-1,1)$, $B'(-2,2)$ and $C'(-3,5)$

[Reference: TB1]

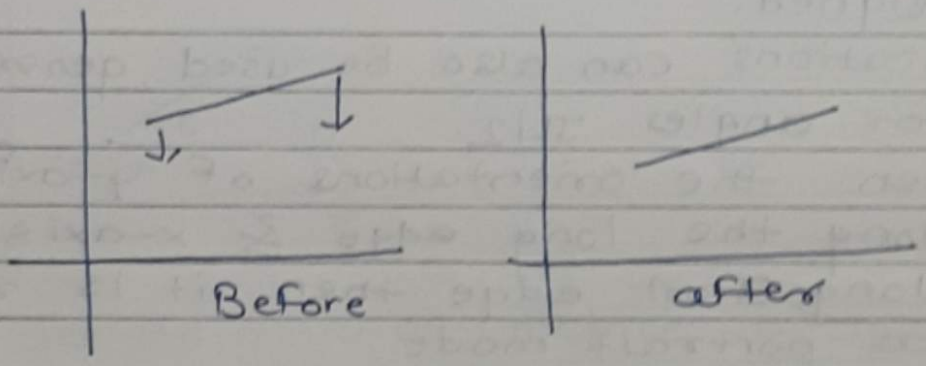
Homogeneous Co-ordinate system –

- In order to combine sequence of transformation it is required to eliminate the matrix addition associated with the translation terms in m_1.
- To achieve this the matrix m should be represented as 3×3 matrix instead of 2×2 by introducing an additional dummy co-ordinate system is called 'Homogeneous co-ordinate system'.
- it allows to express all transformation equations as, matrix multiplication.
- In this system, every point (x, y) can be expressed as $[x, y, w)$.
- for 2D transformation the value of w is 1 and thus the representation of matrix for (x, y) will be $[x, y, 1]$.

Homogeneous Coordinates	Normalized Co-ordinates
1. Real co-ordinates	1. co-ordinates defined in device independent Unit
2. Co-ordinate system designed to combine sequence of transformations	2. co-ordinates system designed to display image with unique [illegible] on the screen, indep of display resolution

Translation –

- Moving the whole image is called Translation.
- it is easy to shift the image by adding same value to all points of image by which we want to shift the image.
- If we want to shift the image down by 5 units then we have to add -5 to all the y values of that image.



- if we want to shift image up by 5 Units, then add 5 to all y values of image.
- Same thing for x values also.
- In general to translate the image we have to add some value to old (x, y) values, so that we will get new (x, y) values.

$\therefore x_2 = x_1 + t_x$ & $y_2 = y_1 + t_y$

t_x – a translation factor for x-axis

t_y – is a translation factor for y-axis.

- Depending on values of t_x & t_y we are shifting the image to [illegible]

position. But we cannot represent translation by using matrix.

Coordinate Transformation

- The co-ordinate system can be changed by using transformation.
- Transformation are of many types each of them is having a specific use.
- Translation is a transformation which is used when the origins are not aligned.
- Rotations can also be used, generally for angles $\pi/2$.
- when the orientations of y-axis is along the long edge & x-axis along short edge then it is referred as portrait mode.
- when orientation of y-axis is along the short edge & x-axis along the long edge then it is referred as landscape mode.
- The transformation from normalized co-ordinates to actual device co-ordinates is given by e.g. arithmetic use for conversion

$$x_1 \leftarrow x_x \text{ width} + \text{width} - \text{start}$$

$$y_1 \leftarrow y_x \text{ height} + \text{height} - \text{start}$$

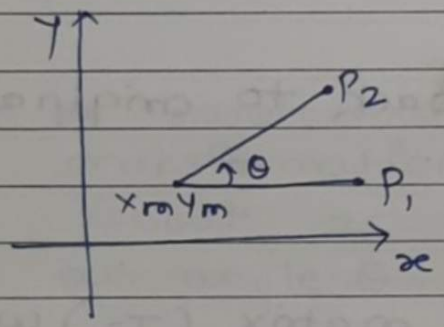
- equation represents a scale by

for x & for y which is followed by a translation by width-start & height-start.

- Transformation matrix will be,

$$\Delta = \begin{vmatrix} \text{width} & 0 & 0 \\ 0 & \text{Height} & 0 \\ \text{width-start} & \text{Height-start} & 1 \end{vmatrix}$$

Rotation about Arbitary point :-



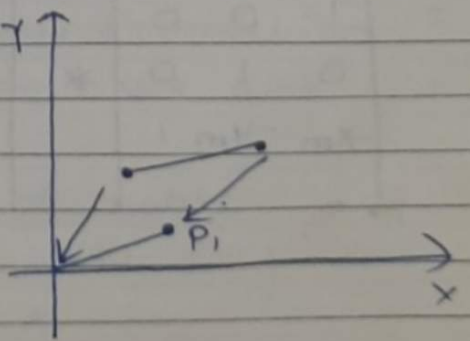
- derived rotation metrices with respect to origin. but if reference point of rotation in other than origin then in that case we have to follow series of transformation.
- Assume that we have to rotate a point P, with respect to (x_m, y_m) then we have perform 3 steps.

1) we have to translate the (x_m, y_m) to origin as in fig.

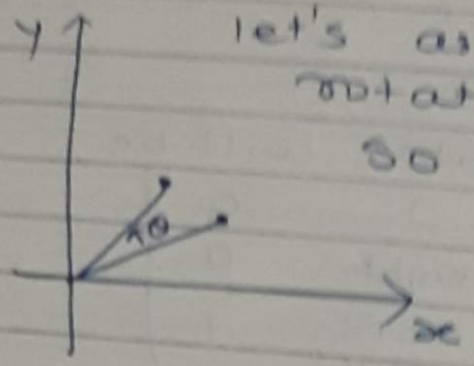
So Translation matrix (T_1) is

$$T_1 = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_m & -y_m & 1 \end{vmatrix} \Rightarrow$$

$t_x = -x_m$ & $t_y = -y_m$



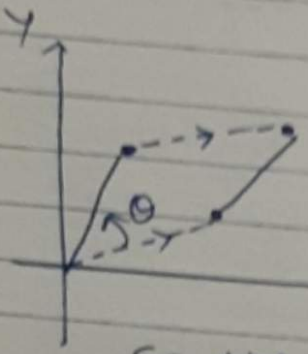
2] rotate it in clockwise or anticlock wise



let's assume as anticlockwise rotation by angle θ.
So rotation matrix will be

$$R = \begin{vmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

3]



translate back to original position.

So the translation matrix (T_2) will become

$$T_2 = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_m & y_m & 1 \end{vmatrix}$$

Now form a Combined matrix

= Translation * Rotation * Translation

= $T_1 * R * T_2$

$$= \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_m & -y_m & 1 \end{vmatrix} * \begin{vmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_m & y_m & 1 \end{vmatrix}$$

$$= \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_m & -y_m & 1 \end{vmatrix} * \begin{vmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ x_m & y_m & 1 \end{vmatrix}$$

$$= \begin{vmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ -x_m * \cos\theta + y_m * \sin\theta + x_m & -x_m * \sin\theta - y_m \cos\theta + y_m & 1 \end{vmatrix}$$

= Transformation matrix is the overall transformation matrix for rotation about arbitary point (x_m, y_m) by an angle θ in anticlockwise direction.

[Reference - TB1]

Inverse Transformation -

- The inverse transformation is very similar to matrix inversion.
- when any normal matrix (A) is multiplied with the inverse of same matrix (A^{-1}) then the resultant will be identify matrix.

$$A * A^{-1} = I$$

- when we multiply any transformation matrix with its inverse, we get identity matrix.
- steps to find inverse of matrix.

Step I] = Suppose we want to find inverse of matrix A.

$$\det(A) = \det \begin{vmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{vmatrix}$$

$$= A_{11}(A_{22} \cdot A_{33} - A_{23} \cdot A_{32}) - A_{12}(A_{21} A_{33} - A_{23} \cdot A_{31}) + A_{13}(A_{21} \cdot A_{32} - A_{22} \cdot A_{31})$$

step = 2

Finding the cofactor of matrix.

To find factor of

$$\begin{vmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{vmatrix}$$

we have to solve

$$B_{11} = \begin{vmatrix} A_{22} & A_{23} \\ A_{32} & A_{33} \end{vmatrix} \quad B_{12} = \begin{vmatrix} A_{21} & A_{23} \\ A_{31} & A_{33} \end{vmatrix} \quad B_{13} = \begin{vmatrix} A_{21} & A_{22} \\ A_{31} & A_{32} \end{vmatrix}$$

$$B_{21} = \begin{vmatrix} A_{12} & A_{13} \\ A_{32} & A_{33} \end{vmatrix} \quad B_{22} = \begin{vmatrix} A_{11} & A_{13} \\ A_{31} & A_{33} \end{vmatrix} \quad B_{23} = \begin{vmatrix} A_{11} & A_{12} \\ A_{31} & A_{32} \end{vmatrix}$$

$$B_{31} = \begin{vmatrix} A_{12} & A_{13} \\ A_{22} & A_{23} \end{vmatrix} \quad B_{32} = \begin{vmatrix} A_{11} & A_{13} \\ A_{21} & A_{23} \end{vmatrix} \quad B_{33} = \begin{vmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{vmatrix}$$

Then

$$\text{Cofactor}(A) = \begin{vmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{vmatrix}$$

— After finding co-factor next step is to find the adjoint of matrix A. To find this adjoint we have to take transpose of co-factor

i.e

$$\text{Adj}(A) = \begin{vmatrix} B_{11} & B_{21} & B_{31} \\ B_{12} & B_{22} & B_{32} \\ B_{13} & B_{23} & B_{33} \end{vmatrix}$$

$$\text{Now inverse} = \frac{\text{Adj}(A)}{\text{determinant}(A)}$$

— But in case of transformations we are having direct formula to find the inverse of transformations.

— for this we have to use homogenous co-ordinates & we have to arrange the matrix in specific order.

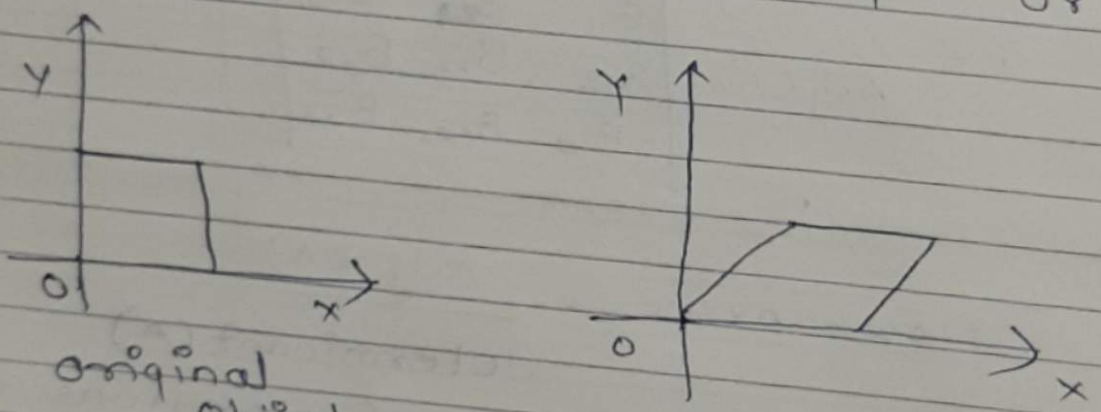
$$\text{Inverse} \begin{vmatrix} a & d & 0 \\ b & e & 0 \\ c & f & 1 \end{vmatrix} = \frac{1}{(ae - bd)} \begin{vmatrix} e & -d & 0 \\ -b & a & 0 \\ (bf - ce) & (cd - af) & (ae - bd) \end{vmatrix}$$

[Reference :- TBI]

Shear Transformation :-

— This type of transformation produces shape distortions that represents a twisting of shearing effect.
— There are two types of shearing Transformations
  — X - Shear
  — Y - Shear

— i) X- shear :- The x-shear preserves the y-coordinates but changes the x-values which causes vertical lines to tilt right or left as



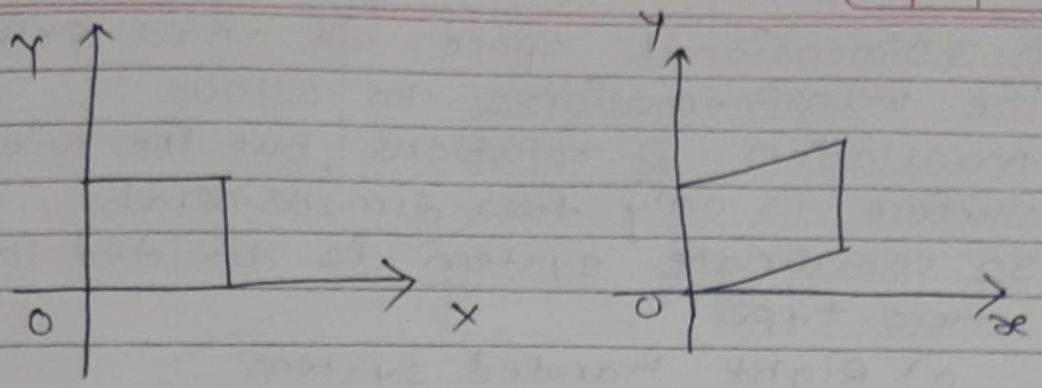
original object

$$X_{sh} = \begin{bmatrix} 1 & 0 & 0 \\ sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$x' = x + Y \cdot sh_x$$

and $y' = y$

ii) Y- shear :- The y-shear preserves the x-coordinates but changes the y-values which causes horizontal lines to transform into lines which slope up or down.



Original object Object after y-shear.

$$Y_{sh} = \begin{bmatrix} 1 & Shy & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$x' = x$$

and $$y' = y + x \cdot Shy.$$

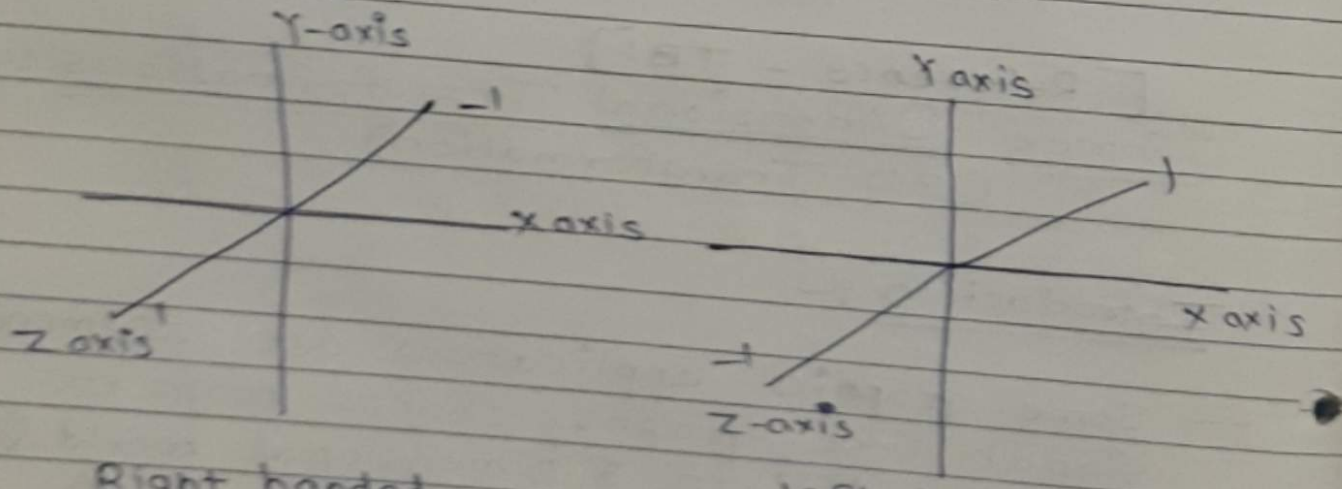
[Reference – TB1]

Three Dimensional Transformations :- (3D Transformation)

Introduction :-

- Some graphics applications are 2Dimensional like graphs, charts, maps etc. But we live in 3 Dimensional world & deal with many design appln. which describe 3 D objects.
  e.g. if architect wants to see how the structure will actually look then 3 Dimensional model can allow him to view the stru. from different viewpoint.
- Some simulation applications such as docking of spaceship or landing an airplane.

- In 3 Dimensional spare we shall extend the transformations to allow translation & rotations, but the viewing surface is only two dimensional.
- 3D coordinate system is divided into two types
  a) Right handed system
  b) left handed system
- If the thumb of right hand points in the positive z direction as one curls the fingers of the right hand from x into y, then the coordinates are called a right handed system.
- If the thumb points in the negative z direction then it is left handed system.

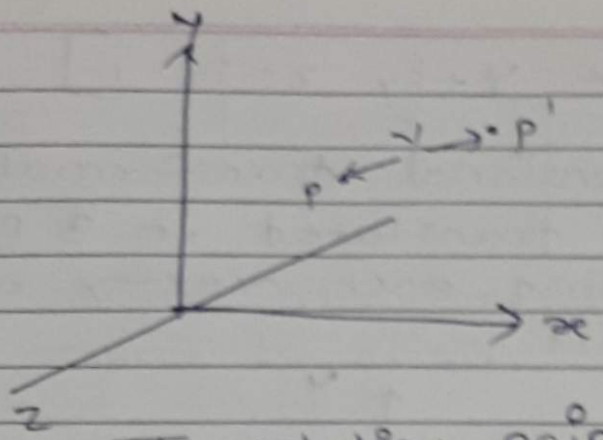


Right handed system

left handed system.

Translation -

Consider point p with the coordinates (x, y, z). To shift this point to new position $p'(x', y' z')$



Translating point

— The shifting & direction of the translation is now defined by vector $V = ai + bj + ck$

Thus,

$$x' = x + a$$
$$y' = y + b$$
$$z' = z + c$$

where a, b, c are translation factors in x, y, z direction respectively.

The matrix representation will be

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ a & b & c & 1 \end{bmatrix}$$

or

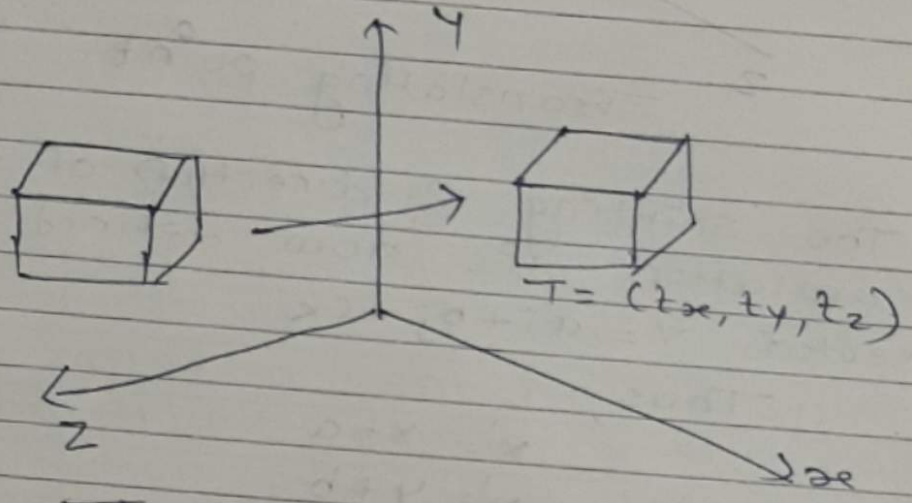
$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 0 \end{bmatrix}$$

$$P' = P \cdot T$$

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

$= [x+t_x \quad y+t_y \quad z+t_z \quad 1]$

— two dimensional transformation an object is translated in 3 Dimensional transforming each vertex of the object.



Translating object.

Scaling :-

- scaling transformation alters the size of the object. This transformation either magnifies or reduces the size depending on the value of scaling factor.
- If the scaling factor is less than 1. it reduces & if it is greater than 1 it magnifies.
- consider Point (x, y, z) which is to be scaled by S_x, S_y, S_z. Then the new coordinates will be,

$$x' = x \cdot S_y$$
$$y' = y \cdot S_y$$
$$z' = z \cdot S_z$$

The scaling matrix will be

$$S = \begin{bmatrix} S_y & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

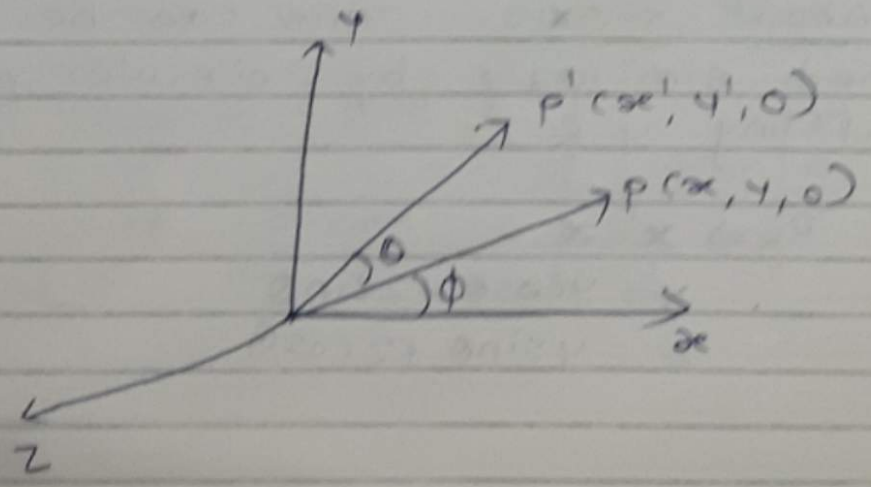
- The scaling transformation is done with respect to origin i.e the origin is kept fixed.

Rotation -

- In 2D transformation the rotation was prescribed by the angle of rotation & the point of rotation.
- In 3D rotation, the angle of rotation as well as the axis of rotation need to be mentioned.
- There are three axis, so the rotation can take place about any of these axis i.e about x-axis, y-axis & z-axis respectively.
- 3-D transformation matrix for each coordinate axes rotations with homogeneous coordinates are.

Rotation About Z-axis :-

let P be the point object in xy plane P(x, y, 0) Rotate it by an angle θ in counterclockwise direction.
The resultant point will be P'(x', y', 0).



from figure,

$$x = r\cos\phi \quad \text{---(1)}$$
$$y = r\sin\phi \quad \text{---(2)}$$

$$x' = r\cos(\theta+\phi)$$
$$y' = r\sin(\theta+\phi)$$

$$x' = r\cos\theta\cos\phi - r\sin\theta\sin\phi$$
$$y' = r\sin\theta\cos\phi + r\cos\theta\sin\phi$$

put the values of $r\cos\phi$ & $r\sin\phi$ from equations (1) & (2)

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

The resulting transformation will be,

$$R_z \Rightarrow x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$
$$z' = 0$$

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

Rotation About x-Axis:- This can be obtained similarly by circularly reshuffling y & z.

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

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

Rotation About y- axis :-

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

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

- all the above rotation matrix are for rotation in counter clockwise direction.
- To obtain the rotation matrix in clockwise direction, change the sign of "$\sin\theta$".

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

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

Rotation About An ARBITARY AXIS :-

- Any line in a space can be used as axeis of rotation.
- for deriving the transformation matrix for rotation by an angle θ° about any arbitary line in a space, The following transformation must be carried out in a sequence.

i) Translation - perform translation, so that the line will coincide with origin.

ii) Rotation - perform rotation to align with one of the coordinate axeis for example, if the line is to be aligned with z-axeis then first rotate it about x-axeis to bring it in x-z plane & then rotate it about y-axeis to align it with z-axeis, then perform rotation about z-axeis.

iii) Retranslation - Then apply inverse translation to bring the line & coordinates to their original orientation.

Consider point $P(x, y, z)$ which is to be rotated about arbitary line. The parametric equation for the line are -

$$x = x_1 + At$$
$$y = y_1 + Bt$$
$$z = z_1 + Ct$$

where

$$A = (x_2 - x_1)$$
$$B = (y_2 - y_1)$$
$$C = (z_2 - z_1)$$

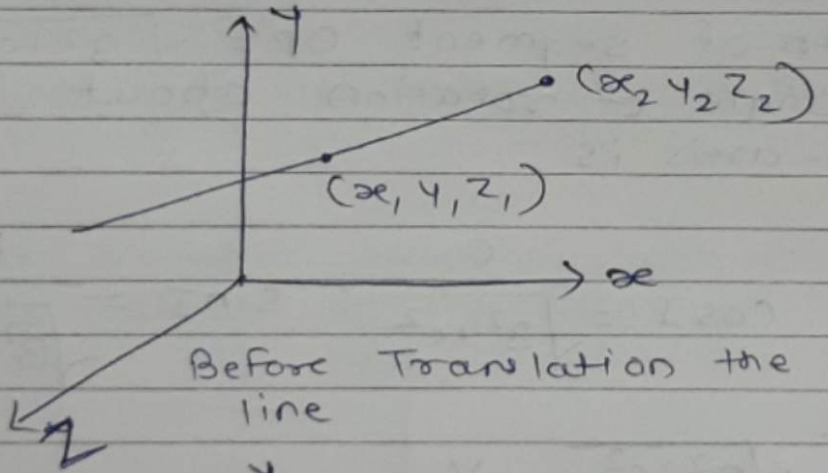
x_1, y_1, z_1 → points on the line
A, B, C → Direction vectors.

— The 1st step is to translate the line to bring it in the origin. The translation matrix will be

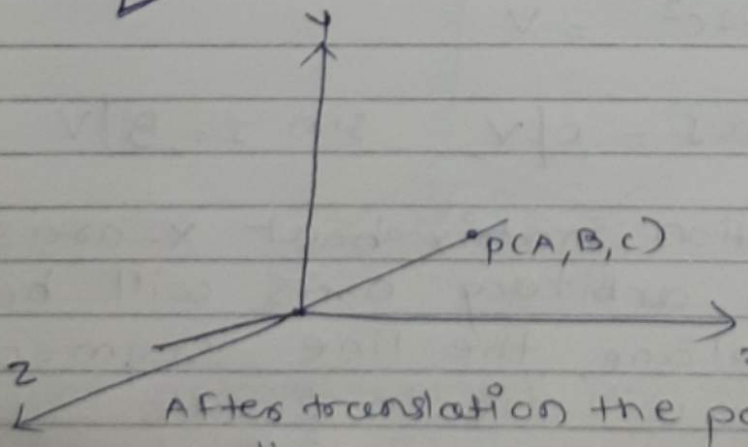
$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x_1 & -y_1 & -z_1 & 1 \end{bmatrix}$$

The inverse transformation will be,

$$T' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x_1 & y_1 & z_1 & 1 \end{bmatrix}$$



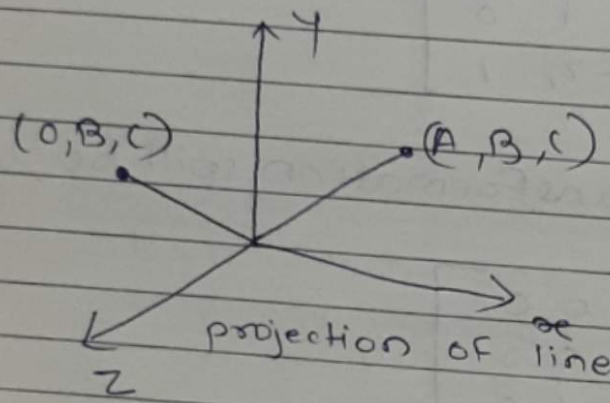
Before Translation the position of line



After translation the position of line

step 2:- is the rotation of line about x-axis to bring the line in x-z plane, for this the angle of rotation by which the line is to be rotated must be computed, for this project a point P(A,B,C) in y-z plane.

let p' be the point p in y-z plane



projection of line segment on yz plane.

- The coordinates of p' are (0,B,C). The length of segment $OP' = \sqrt{B^2+C^2}$.
The angle of rotation about x-axis is

$$\cos I = \frac{C}{\sqrt{B^2+C^2}} \qquad \sin I = \frac{B}{\sqrt{B^2+C^2}}$$

put $\sqrt{B^2+C^2} = V$

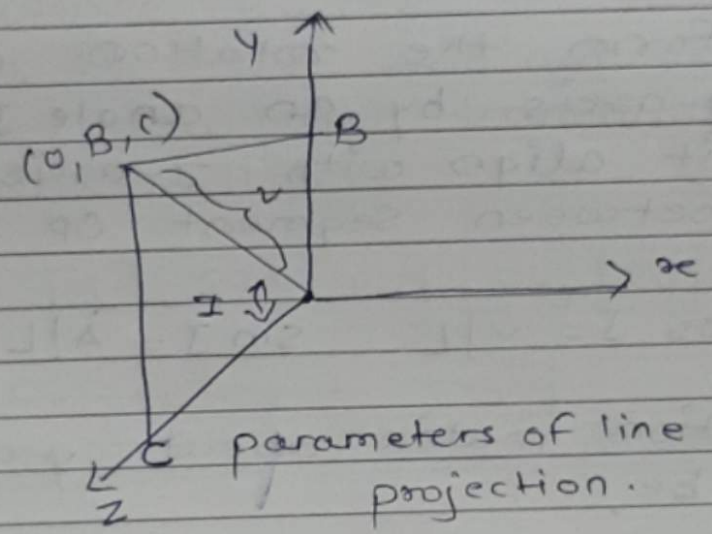
$$\cos I = C/V \qquad \sin I = B/V$$

Now, rotation matrix about x-axis, so that arbitary axis will be in xz plane, the line segment's shadow will lie in z-axis.

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/v & B/v & 0 \\ 0 & -B/v & c/v & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

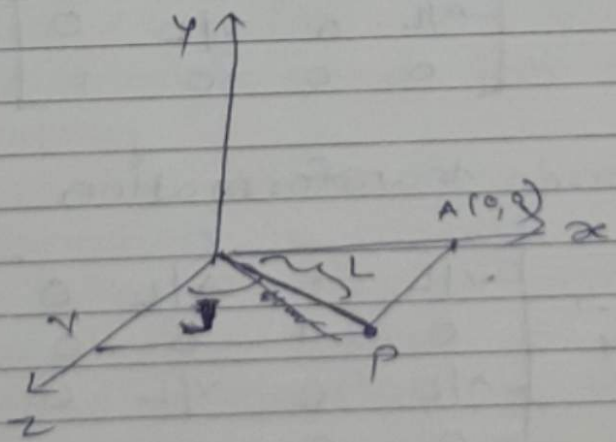
The inverse rotation will be

$$R_x^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/v & -B/v & 0 \\ 0 & B/v & c/v & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



parameters of line segment projection.

The rotation axis lying with x-z plane is shown in fig. 3.2.3



- The parameters will remain unchanged & equal to A as it is the rotation about x-axis. The y co-ordinate

become zero & z-coordinate will be

$$z = \sqrt{R^2 + c^2} = V$$

- The coordinates of point P are P(A,0,0), the length OP will be $\sqrt{A^2+B^2+c^2}$.

put, $\sqrt{A^2+B^2+c^2} = L$

$\therefore (OP) = L$

- Now perform the rotation of line about y-axis by an angle J to make it align with z-axis. is an angle between segment OP & z-axis.

$\cos J = V/L \quad \sin J = A/L$

- The rotation matrix about y axis will be,

$$R_y = \begin{bmatrix} V/L & 0 & A/L & 0 \\ 0 & 1 & 0 & 0 \\ -A/L & 0 & V/L & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- The inverse transformation will be,

$$R_y^{-1} = \begin{bmatrix} V/L & 0 & A/L & 0 \\ 0 & 1 & 0 & 0 \\ -A/L & 0 & V/L & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- after performing rotation about y-axis the line will get aligned with z-axis. Then perform the rotation about z-axis by angle θ. The matrix will be

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

- Then apply inverse transformation in sequence R_y^{-1}, R_x^{-1}, T^{-1} to rotate the axis to their original position. The resultant transformation matrix will be,

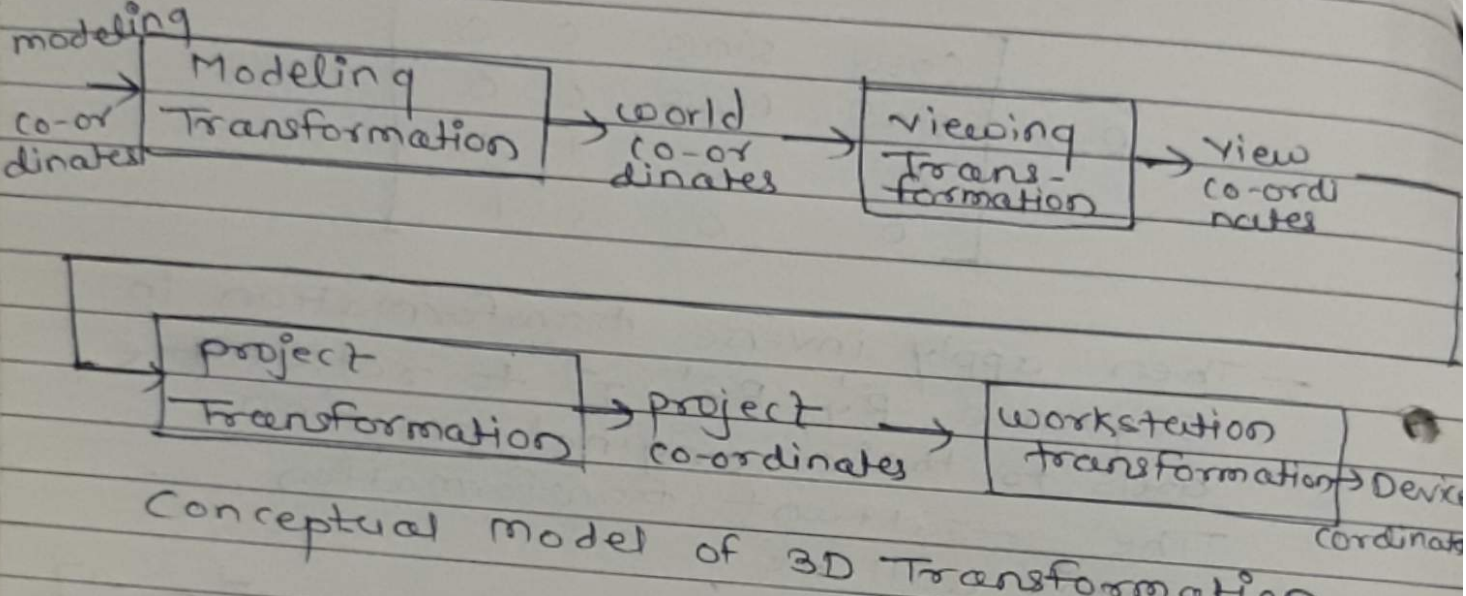
$$R_S = [T][R_x][R_y][R_z][R_y]^{-1}[R_x]^{-1}[T^{-1}]$$

[Reference – TBI]

3 Dimensional viewing Transformation :-

- In two Dimensional viewing there are 2D window & 2D viewport & the objects in the world co-ordinates are clipped against the window & are then transformed into the viewport for display.
- The Three Dimensional viewing transformation is more complex than the 2D viewing transformation.
- The complexity added because of the added dimension & the fact that even though objects are 3D the display devices are only 2D.

- The mismatch between 3D objects & 2D displays is compensated by introducing projections.
- The projections transforms 3D objects into a 2D projection plane



Conceptual model of 3D Transformation

- In 3D viewing, a view plane is specified in the world coordinates using modeling transformation.
- The world co-ordinate positions of the objects are then converted into viewing co-ordinates by viewing transformation.
- The projection transformation is then used to convert 3D description of objects in viewing co-ordinates to the 2D projection co-ordinates.
- example - To write a flight simulator program. The 1st thing to be done is to construct a model of world over which the pilot is to fly. Buildings, fields, runways, lakes & other scenes maybe constructed using 3D line & polygon primitives.
- windowing & modeling allows to use

real world dimensions. Therefore model of the world is represented using the world coordinates. Then object description is converted from world coordinates to viewing coordinates. This produces the view which the pilot can see from his airplane.

- The projection transformation is then used to convert 3D description of the object in viewing coordinates to the 2D coordinates are converted into device coordinates which are used to display the picture on the view display.

Viewing parameters :-

- To view the object from the side, top or even from the behind, it is needed to apply some rotation transformation before projection. There are two ways.

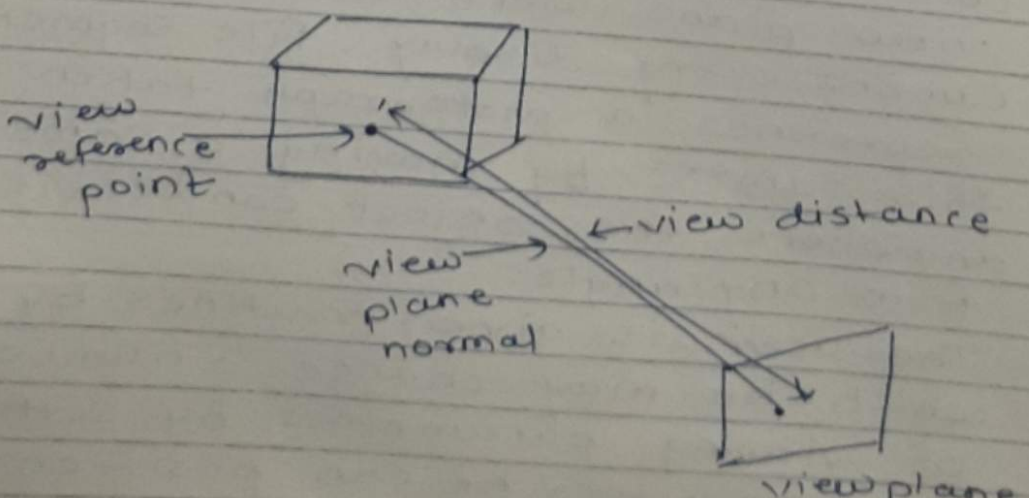
1] keeping the view plane as fixed & the object as rotated.

2] keeping the object as fixed & the view plane as repositioned.

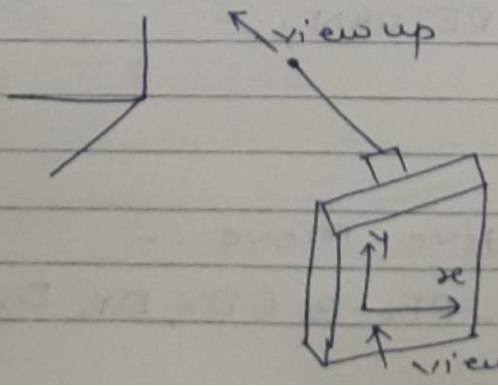
- consider the 2nd way then, if the view plane were the film in a camera, every display file segment represents a photograph taken by this camera, by moving a camera anywhere, the object can be viewed from any angle.
- The user is given routines by which he may change a number of viewing parameters. by setting the parameters, he can position the synthetic camera.

- The viewing parameters are
  a) view reference point (XR, YR, ZR)
  b) view plane normal vector (DXN, DYN, DZN)
  c) VIEW DISTANCE parameter.
  d) view up direction (XUP, YUP, ZUP)

- View reference point (XR, YR, ZR) - it is the center of attention, All other viewing parameters are expressed relative to this point. If the view is rotated, it will be at about the view reference point & not about the origin.
- view reference point can be considered as an anchor to which a string is tied.
- The synthetic camera is attached to the other end of string, By changing other viewing parameters the camera can swing through an arc or change the length of the string. one end of the string is always attached to the view reference point.



- View plane normal vector (DXN, DYN, DZN) :-
The direction of the imaginary string is given by this viewing parameters. This normal is the direction perpendicular to the view plane. i.e film in the camera.
- The camera always looks along the string towards the view reference point. The camera is pointed in the direction of view plane normal.
- View Distance parameter :- The length of the string is given by this parameter. This tells how the camera is positioned from the view reference point. The view plane is positioned VIEW-DISTANCE away from the view reference point in the direction of the view plane normal.
- View-up Direction (XUP, YUP, ZUP): This parameter fixes the camera angle. Imagine an arrow extending from the view reference point in the view up direction. looking through the camera's view finder & spinning camera until the arrow appears to be in the camera's "up" direction. changing the view reference point will change the part of the object that is shown at the origin.



PAGE No.
DATE

- There are two coordinate system. The coordinates which is used to model the object & the view plane co-ordinates. which are attached to the view object. The system must provide the user with a means of setting the parameters to the values which he desires. The values are saved as global variables.
[Reference: TBI]

3D primitives :-

- 3D primitives are used to draw points, lines & planes in 3Dimensions.

(1) 3D Absolute Move:-
MOVE_ABS_3(x,Y,Z): This is 3D absolute move command where x,Y,Z are coordinates of new position.

```
Global DF_PEN_X,
       DF_PEN_Y,
       DF_PEN_Z
       are current pen positions.

BEGIN
        DF_PEN_X ← X;
        DF_PEN_Y ← Y;
        DF_PEN_Z ← Z;
        DF_ENTER(1);
        RETURN;
END;
```

(2) 3D Relative move :-
MOVE_REL_3(DX, DY, DZ):

Arguments DX, DY, DZ - changes done in the current pen position.

```
Global  DF_PEN_X
        DF_PEN_Y
        DF_PEN_Z
        current pen position.

BEGIN

    DF_PEN_X ← DF_PEN_X + DX;
    DF_PEN_Y ← DF_PEN_Y + DY;
    DF_PEN_Z ← DF_PEN_Z + DZ;
    DF_ENTER (1);
    RETURN;
END;
```

③ Absolute line Drawing Routine :-

```
LINE_ABS_3 (X,Y,Z):
```

Arguments X, Y, Z are coordinates of point to draw the line.

```
Global  DF_PEN_X, DF_PEN_Y, DF_PEN_Z -
        current pen position.

BEGIN
        DF_PEN_X ← X;
        DF_PEN_Y ← Y;
        DF_PEN_Z ← Z;
        DF_ENTER (2);
END;
```

— There are two coordinate system. The coordinates which is used to model the object & the view plane co-ordinates. which are attached to the view object. The system must provide the user with a means of setting the parameters to the values which he desires. The values are saved as global variables.
[Reference: TBI]

3D primitives :-

— 3D primitives are used to draw points, lines & planes in 3Dimensions.

(1) 3D Absolute move :—
MOVE_ABS_3(x,Y,Z): This is 3D absolute move command where x,Y,Z are coordinates of new position.

```
Global DF_PEN_X,
       DF_PEN_Y,
       DF_PEN_Z
       are current pen positions.

BEGIN
       DF_PEN_X ← X;
       DF_PEN_Y ← Y;
       DF_PEN_Z ← Z;
       DF_ENTER(1);
       RETURN;
END;
```

(2) 3D Relative move :-
MOVE_REL_3(DX, DY, DZ):

Arguments DX, DY, DZ - changes done in the current pen position.

```
Global  DF_PEN_X
        DF_PEN_Y
        DF_PEN_Z
        current pen position.

BEGIN

    DF_PEN_X ← DF_PEN_X + DX;
    DF_PEN_Y ← DF_PEN_Y + DY;
    DF_PEN_Z ← DF_PEN_Z + DZ;
    DF_ENTER (1);
    RETURN;
END;
```

③ Absolute line Drawing Routine :-

LINE_ABS_3 (X, Y, Z):

Arguments X, Y, Z are coordinates of point to draw the line.

```
Global  DF_PEN_X, DF_PEN_Y, DF_PEN_Z -
        current pen position.

BEGIN
        DF_PEN_X ← X;
        DF_PEN_Y ← Y;
        DF_PEN_Z ← Z;
        DF_ENTER (2);
END;
```

④ 3-D Relative Line Drawing Routine :

LINE_REL_3 (DX, DY, DZ)

Arguments Dx, Dy, Dz are displacement over which a line is to be drawn.

Global DF_PEN_X, DF_PEN_Y, DF_PEN,Z – the current pen position.

```
BEGIN
    DF_PEN_X ← DF_PEN_X+DX;
    DF_PEN_Y ← DF_PEN_Y+DY;
    DF_PEN_Z ← DF_PEN-Z+Dz;
    DF_ENTER (Q);
    RETURN;

END;
```

⑤ Absolute Polygon Drawing Routine –

POLYGON_ABS_3(AX, AY, AZ, N);

Arguments N – number of polygon sides
AX, AY, AZ – array of the co-ordinates of vertices.

Global DF_PEN_X; DF_PEN_Y; DF_PEN_Z } current pen position.

```
BEGIN
    If N<3 then RETURN ERROR 'SIZE ERROR';
    DF_PEN_X ← AX[N];
    DF_PEN-Y ← AY [N];
```

```
DF_PEN_Z ← AZ[N];
DF_ENTER (N);
FOR I=1 to N
LINE_ABS_3 (AX[I], AY[I], AZ[I])
RETURN;
END.
```

⑥ 3-D Relative Polygon Drawing Algorithm :-

POLYGON_REL_3 (AX, AY, AZ, N):

- Arguments N - No. of polygon sides AX, AY, AZ - array of displacement for the polygon sides.

Global DF_PEN_X, DF_PEN_Y, DF_PEN_Z - the current pen position.

BEGIN

if N<3 then RETURN ERROR 'size error';

- DF_PEN_X ← DF_PEN_X + AX[I];

DF_PEN_Y ← DF_PEN_Y + AY[I];
DF_PEN_Z ← DF_PEN_Z + AZ[I];
Save vertex for closing polygon

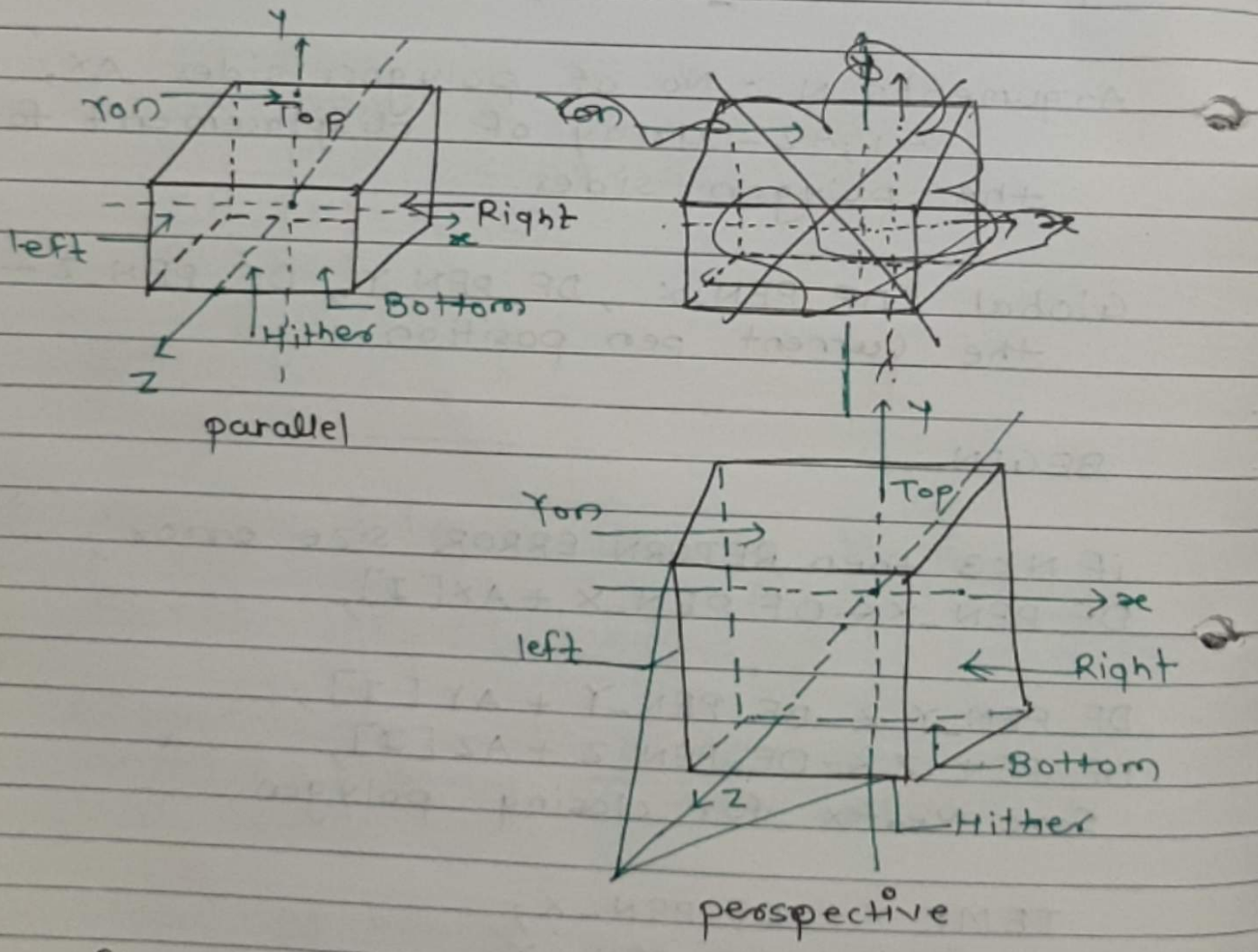
```
TEMP X ← DF_PEN_X;
TEMP Y ← DF_PEN_Y;
TEMP Z ← DF_PEN_Z;
DF_ENTER (N);
 enter polygon sides,
  FOR I=2 to N DO,
  LINE_REL_3 (AX[I], AY[I], AZ[I]),
  close the polygon.
  LINE_ABS_3 (TEMPX, TEMPY, TEMPZ)
  RETURN;
END;
```

L-R - x-coordinate
T-B - y-
F-B - z-

[Reference : TB1]

3D clipping -

- In 3D space the concept can be extended to a clipping volume or view volume.
- Two common three dimensional clipping volumes are a rectangular parallelepiped i.e a box, used for parallel or axonometric projections, & truncated pyramidal volume, used for perspective projections.
-



- fig. shows these volumes, volumes are six sided with sides
  1) left 2) Right
  3) top 4) Bottom
  5) hither (near) 6) yon (far)

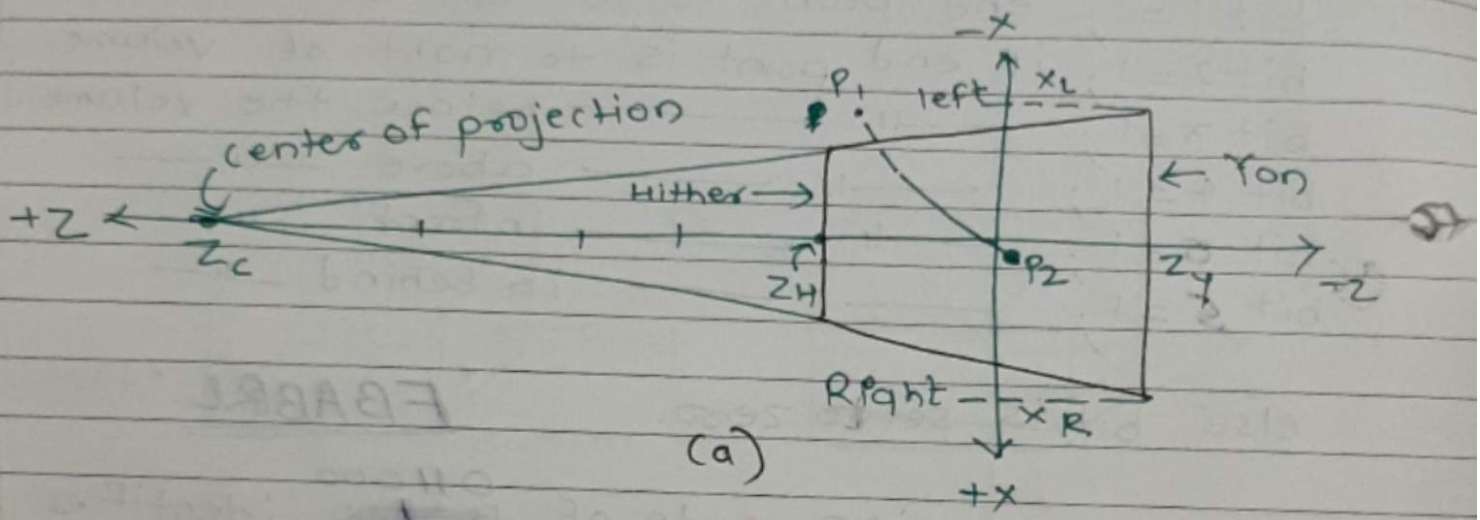
- Two Dimensional concept of region codes can be extended to 3D by considering six sides & 6 bit code instead of four sides & 4 bit code.

- like 2D, assign bit position in the region code from right to left as if
  bit 1 = 1; end point is to left of the volume.
  bit 2 = 1; if end point is to right of volume.
  bit 3 = 1; —"— below the volume.
  bit 4 = 1; —"— above —"—
  bit 5 = 1; —"— infront —"—
  bit 6 = 1; —"— is behind —"—

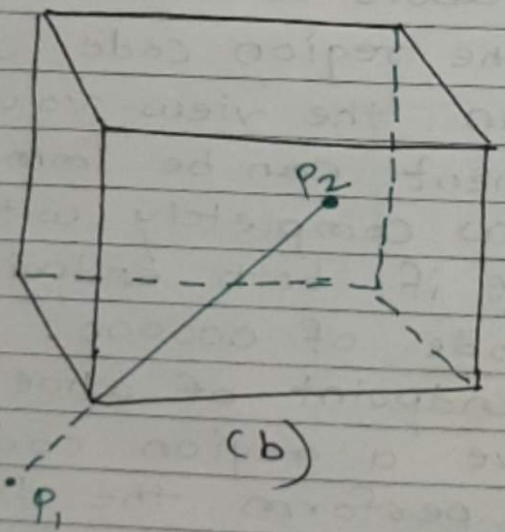
  else bit is set to zero. "FBABRL"

- e.g- a region code of 011000 identifies a point as above & behind the view volume, & the region code 000000 indicates a point within the view volume.
- A line segment can be immediately identified as completely within the view volume if both endpoints have a region code of 000000.
- If either endpoint of a line segment does not have a region code of 000000, we perform the logical AND operation on the two endpoint codes. If the result of this AND operations is nonzero then both endpoints are outside the view volume & line segment is completely invisible.
- on other hand, if the result of AND operation is zero then line Segment may be partially visible.

- it is necessary to determine the intersection of the line & clipping volume.
- we have seen that determing the end point codes for a rectangular parallelepiped clipping volume is a straight forward extension of the two dimensional algo.



(a)



(b)

- fig (a), line connecting the center of projection & the center of the ~~projection~~ perspective clipping volume coincides with z-axis in a right handed co-ordinate system.
- fig ~~(b)~~ shows top view of perspective clipping volume. The equation of the line which represents the right

hand plane in this view can be given as,

$$X = \frac{Z - Z_c}{Z_Y - Z_c} \quad x_R = Z\alpha_1 + \alpha_2$$

where $\alpha_1 = \frac{x_R}{Z_Y - Z_c}$ and $\alpha_2 = -\alpha_1 Z_c$

- The equation of right hand plane can be used to determine whether a point is to the right, on or to the left of the plane, i.e outside the volume, on the right hand plane, or inside the volume. Substituting the x & y coordinates of a point P into $x - z\alpha_1 - \alpha_2$ gives the following results.

$f_R = x - z\alpha_1 - \alpha_2 > 0$ if P is to the right of the right plane.
$= 0$ if P is on the right plane
< 0 if P is to the left of the right plane.

plane	Test functions with Results.
Right	$f_R = x - z\alpha_1 - \alpha_2$ > 0 if P is to the right of the right plane $= 0$ if P is on the right plane < 0 if P is to the left of the right plane. where $\alpha_1 = \frac{x_R}{Z_Y - Z_c}$ & $\alpha_2 = -\alpha_1 Z_c$

left $f_L = x - z \quad \beta_1 - \beta_2 < 0$ if p is to the left of the left plane.

$= 0$ if p is on the left plane

> 0 if p is to the right of the left plane.

where $\beta_1 = \frac{x_L}{z_Y - z_c}$ & $\beta_2 = -\beta_1 z_c$

Top $f_T = y - z \quad r_1 - r_2 > 0$ if p is above the top plane.

$= 0$ if p is on the top plane.

< 0 if p is below the top plane.

where $r_1 = \frac{y_T}{z_Y - z_c}$ & $r_2 = -r_1 z_c$

Bottom $f_B = y - z \quad \delta_1 - \delta_2 < 0$ if p is below the bottom plane.

$= 0$ if p is on the bottom plane.

> 0 if p is above the bottom plane.

where $\delta_1 = \frac{y_B}{z_Y - z_c}$ and $\delta_2 = -\delta_1 z_c$

Hither $f_H = z - z_H > 0$ if p is in front of the hitherplane.

$= 0$ if p is on the hither plane.

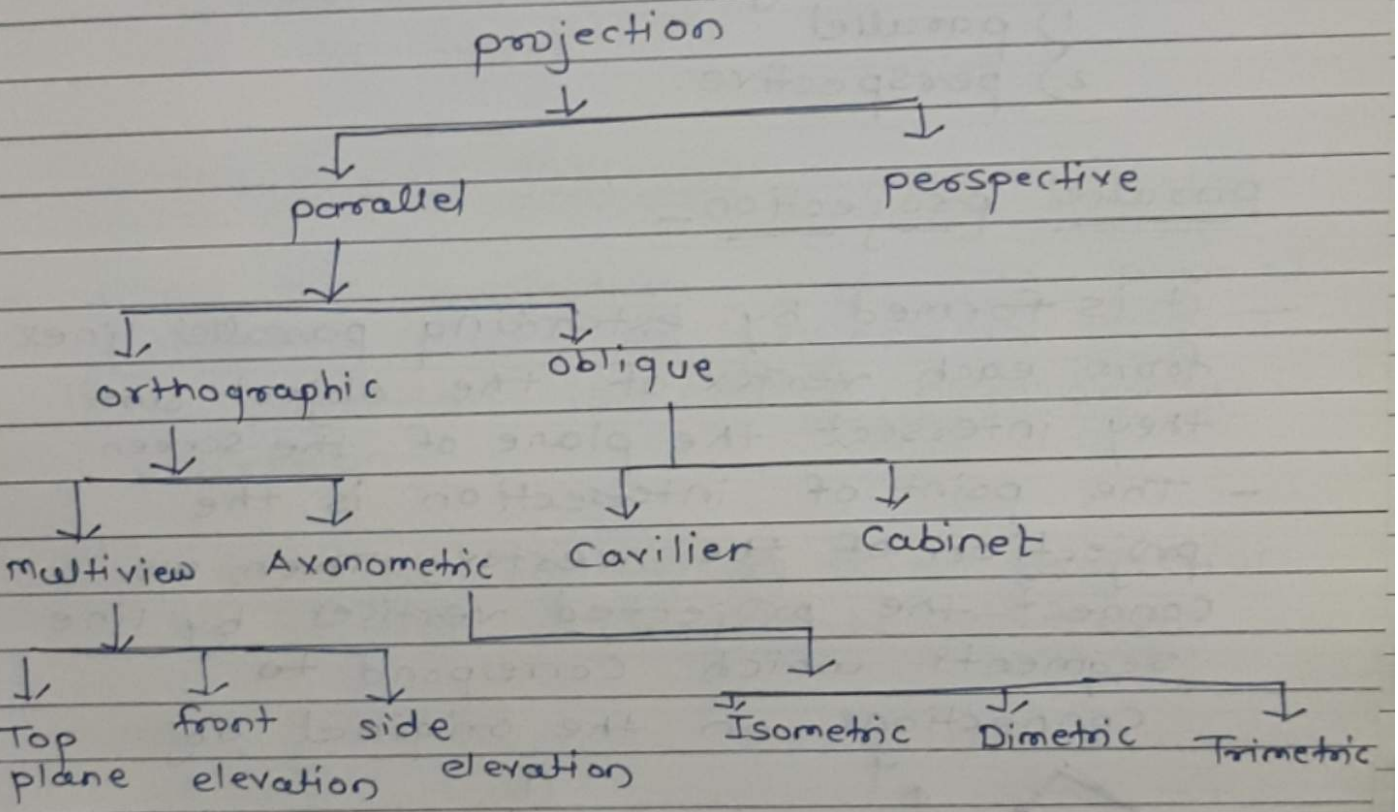
< 0 if p is behind the hither plane.

Yon $f_y = z - z_y < 0$ if P is behind the yon plane.
$= 0$ if P is on the yon plane.
> 0 if P is in front of the yon plane.

[Reference – TBI]

Projections :-

— The Process of representing a 3D object or Scene into two dimentional medium is referred as projection.



— projection is nothing but a shadow of the object.

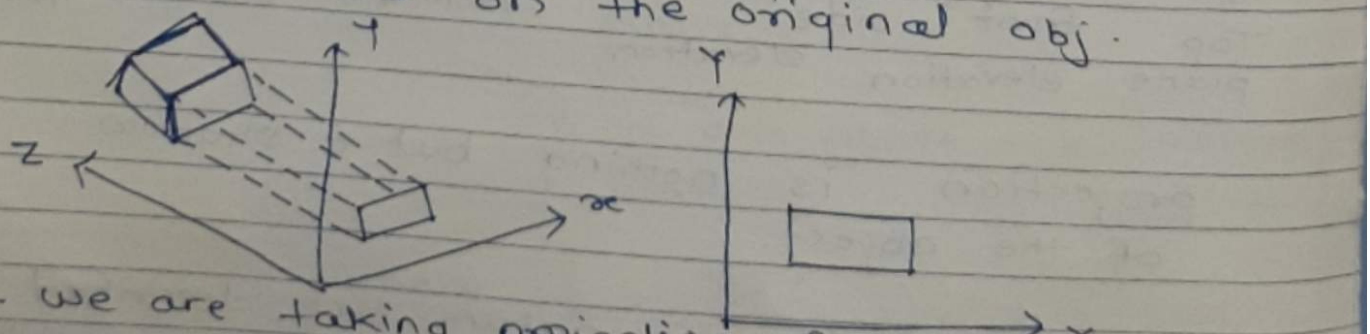
— objects available in nature are described in world coordinate system, but all graphics display media are two dimensional like screen of monitor, printout of printer.

— when we want to draw a 3D object on a monitor, we have to convert the

world coordinates into screen coordinates
- we have to project 3D object on a 2D plane.
- Ideally object is projected by projecting each of its points. as there are infinitely many points in an object. we are not producing projections of all those points. instead of that we will take projections of only corner points of an objects on 2D plane & then we will join these projected points by straight line in 2D plane.
- Two basic projection methods
  1) parallel
  2) perspective.

parallel projection -

- it is formed by extending parallel lines from each vertex of the object until they intersect the plane of the screen.
- The point of intersection is the projection of the vertex. Then we connect the projected vertices by line segments which correspond to connections on the original obj.



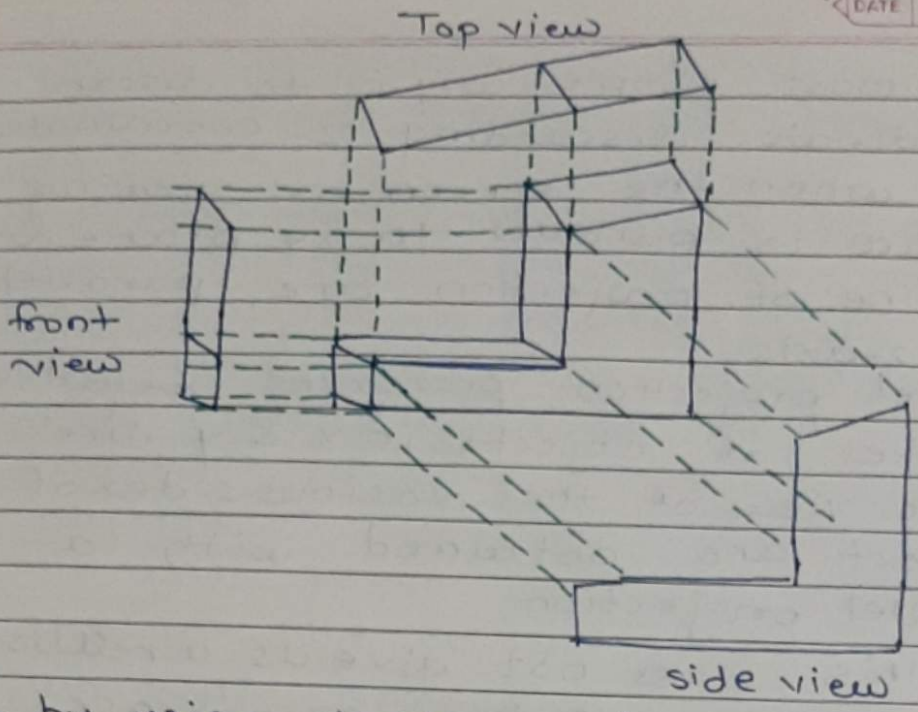
- we are taking projection of each vertex of the object till it intersects to the plane of screen.
- The p'dne on which we are taking projection is called as view plane. if we want to represent 3D object on 2D

plane, most simple way is to discard the z coordinate. discarding z-coordinates only when the screen or viewing surface is parallel to xy plane & the line of projection are parallel to z-axis.

- A parallel projection preserves relative properties of objects in x & y diren. Accurate view of the various sides of an object are obtained with a parallel projection.
- but this does not give us a realistic representation of the appearence of 3D object.

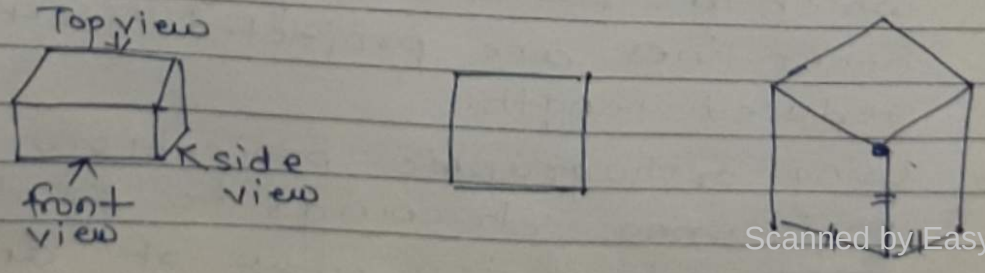
Orthographic Projection

- when the projection lines are normal (perpendicular) to projection plane the projection is called orthographic projection.
- it can be obtain ~~xy~~ this projection by setting z-coordinate to zero.
- This projection is good approximation of the actual projections that the human eye makes in the real world.
- orthographic projections do not change the length of line segments which are parallel to projection plane other lines are projected with reduced length.
- using orthographic projection in engineering drawings to produce front, side & top view of an object.



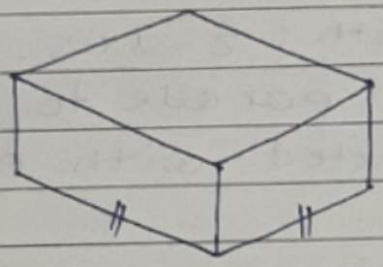
- by using orthographic projections we can draw any sides view of an object.
- we can also form orthographic projections in such a way that more than one face of an object can be displayed. Such views are called as axonometric orthographic projections.
- 3 types
  1. Isometric
  2. Diametric
  3. Trimetric

- **Isometric**
- is most commonly used projections are isometric projections.
- In this, the projections are aligned in such a way that all the edges will appear shortened by same distance.



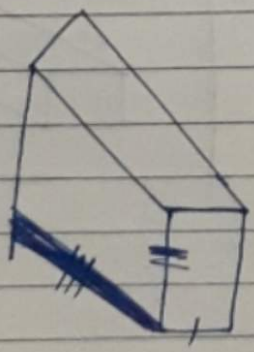
- If we consider an object as cube then side view, front view & top view will look like square. i.e we are able to see only one face at a time. But by using isometric orthographic projection we can see more than one face.
- all 3 principal axes are shortened equally so that relative proportions are maintained.

- Diametric projections :-



In this, direction of projection is in such a way that the edges parallel to only two principle axes are equally shortened.

- Trimetric Projections -



The direction of projection is in such a way that none of the 3 edges are equally shortened.

- Oblique projections :-

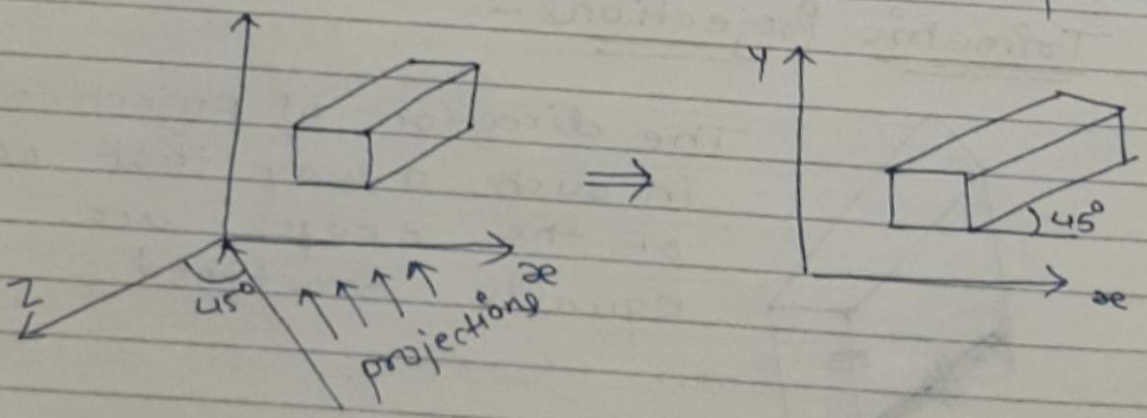
- In orthographic projections, where project lines are normal to view plane. But if the projection lines are not normal to view plane, then it is called as oblique projections.

— An Oblique projection is obtained by projecting points along parallel lines that are not perpendicular to the view plane such a projection appears in nature when sunlight casts shadows on the ground.

— oblique projections have 2 types
(1) Cavalier
(2) Cabinet.

— Cavalier Projections :- is a special case of oblique projection. This occurs when the projection vector forms an angle of 45° with z-axis. The lines which are parallel to the z-axis are projected with no change in length.

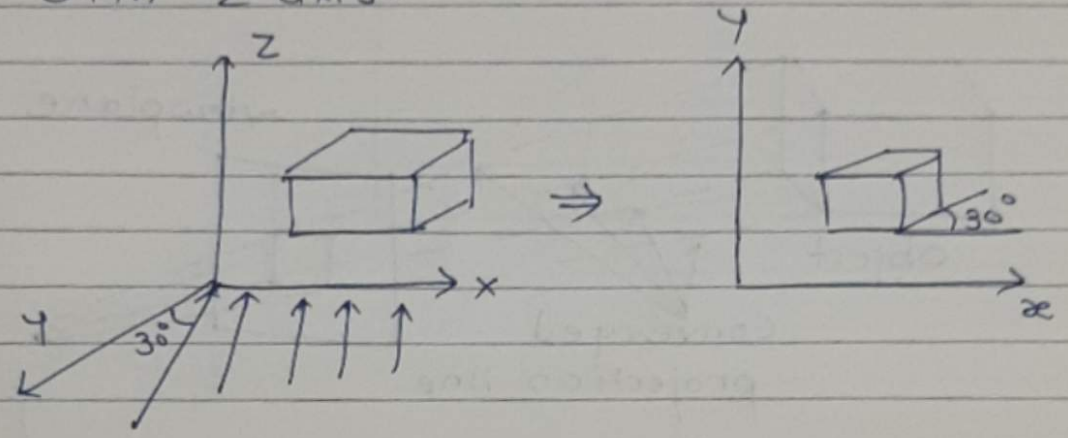
— Consider an object whose edges are parallel to axes & front face of object is parallel to view plane xy.



— As the projections are not perpendicular to view plane, we will not get only one face of object, But as angle of projection is 45°, will get z-axis values as it is but x and y-axis values get reduced. So the object will look like an elongated object.

Cabinet Projections :-

- In this we are having only half the actual z dist. along the projected axis.
- It uses a projection vector that forms an angle of approximately 26.6° with z-axis.



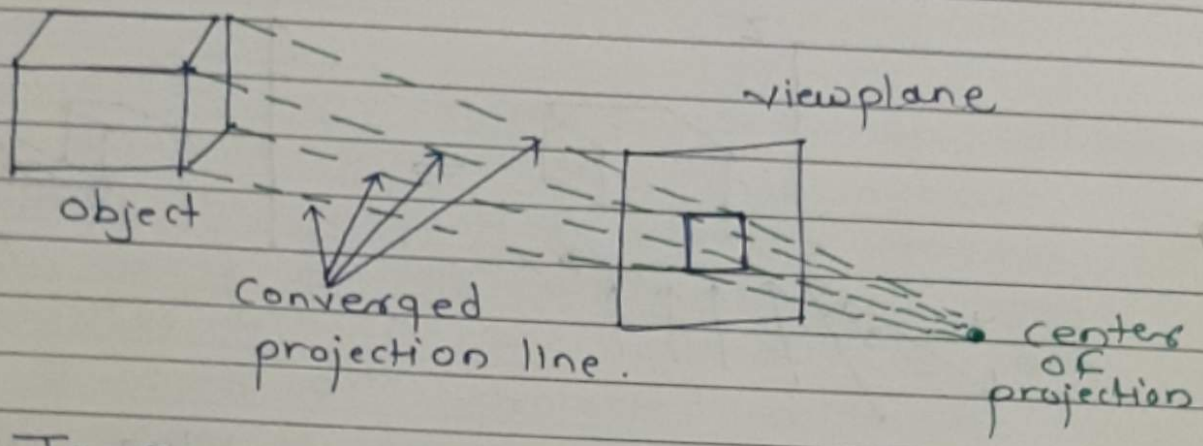
- As the projections are making angle of 26.6° (≅ 30°), z-axis values get half of its original values.
- Cabinet projections appear more realistic than Cavalier projections because of this reduction in the length of edges which are perpendicular to view plane.

Perspective Projection :-

- In this if the object is far away from the viewer then it appears smaller & it appears larger if the object is nearer to the viewer.
- This helps the viewer in determining depth cue.
- The depth cue is an indication of which portion of the image correspond to part of the object which are

close or far away -

- In this projection the lines of projection converge at a single point which is referred as center of projection.
- The intersection of lines of projection with the plane of screen determines the projected image,



- To generate perspective projection of a 3 Dimensional object, 1st of all project points along projection lines that will meet at the center of projection is selected.
- The center of projection is on the negative z-axis at a distance behind the projection plane.

Types of perspective projections

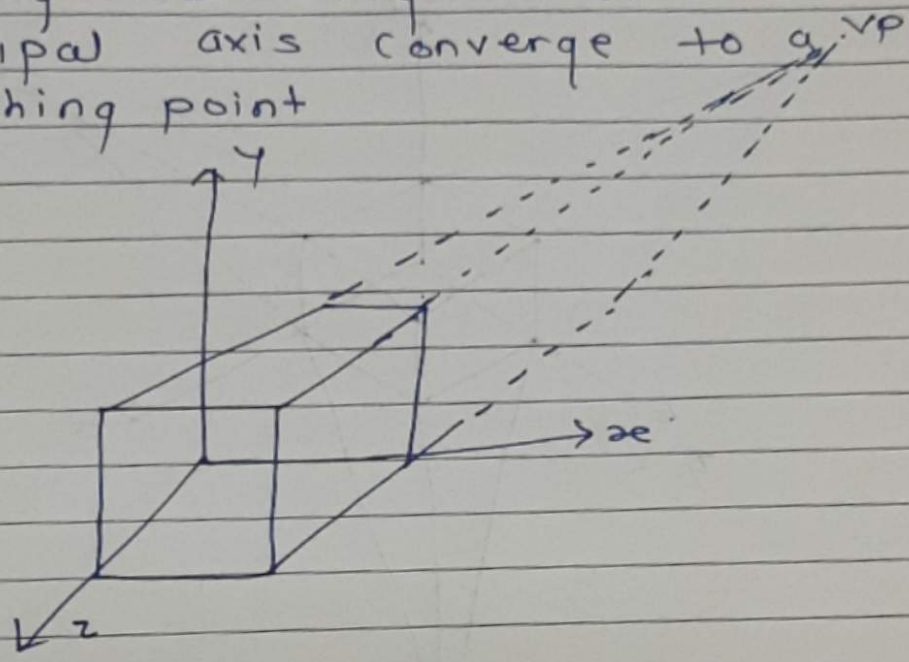
- vanishing point on view plane is referred as a point where all the lines which are not parallel to view plane are appeared to meet.
-

One point perspective

- it occurs when the projection plane is parallel to two principal axes. conversely, when the projection plane

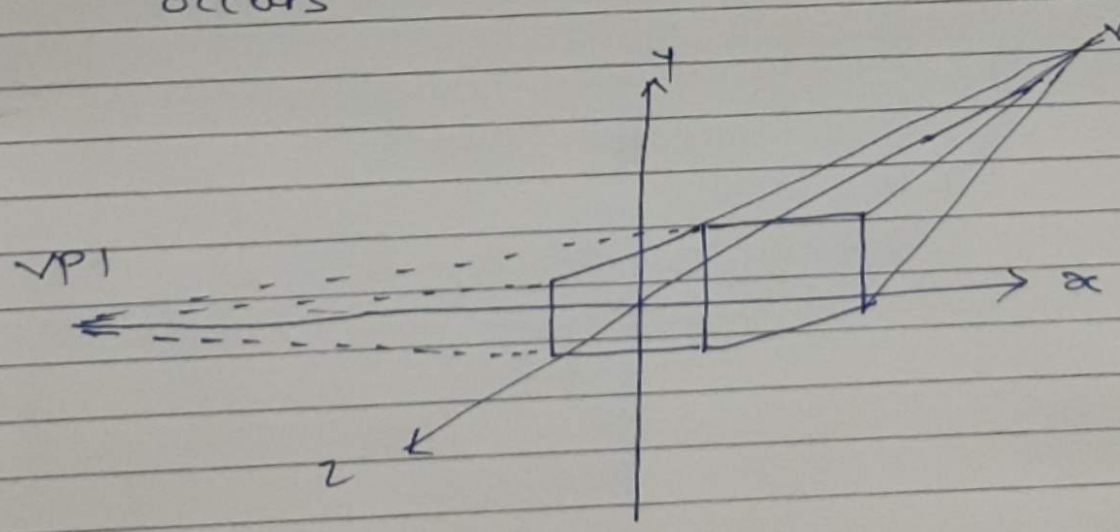
is perpendicular to one of the principal axes, one point perspective occurs

- Receding lines along one of the principal axis converge to a vanishing point



Two point perspective

if projection plane is parallel to one of the principal axes or if the projections plane intersects exactly two principal axes a two-point perspective projection occurs



Three point Perspective -

if the projection plane is not parallel to any principal axes,