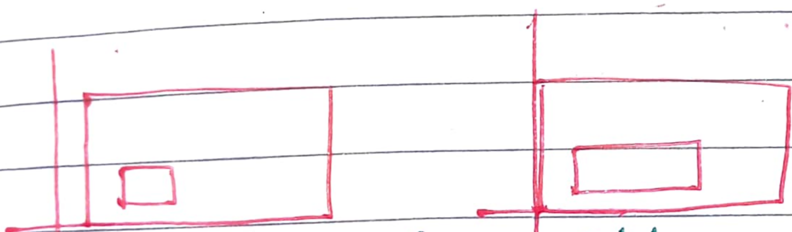
# Unit-II

## 2D-Transformations

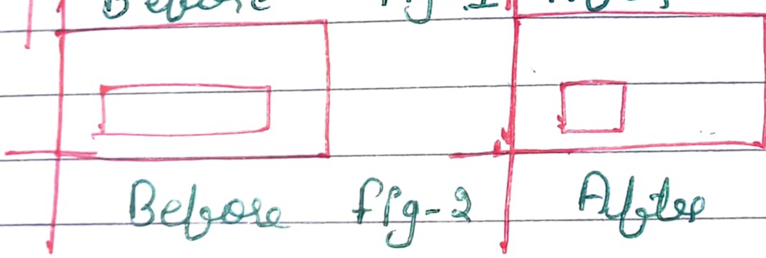
Transformation ⇒ Basically transformation means change in image.

We can modify the image by performing some basic transformations, such as

1. **Scaling** ⇒ Scaling means changing the size of the object or image.



Before fig-1 After



Before fig-2 After

$S_x$ ⇒ Scaling factor for x-co-ordinate

$S_y$ ⇒ Scaling factor for y-coordinate


$$\boxed{\begin{aligned} A' &= A \cdot S \\ B' &= B \cdot S \\ C' &= C \cdot S \end{aligned}}$$

$A' = \begin{bmatrix} x & y \end{bmatrix}$

$B' = \begin{bmatrix} x & y \end{bmatrix}$

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

$$\begin{bmatrix} x' = x \cdot S_x \\ y' = y \cdot S_y \end{bmatrix}$$

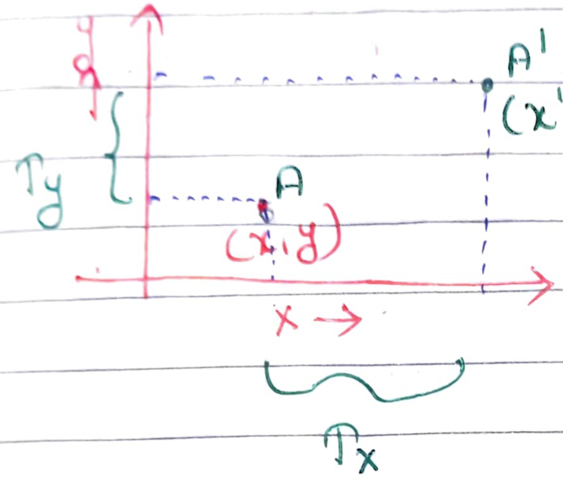
$A' = A \cdot S$

$A' = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$

$B' = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$

$C' = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$

Translation ⇒ Moving the whole image is called Translation




$x' = x + T_x$

$y' = y + T_y$

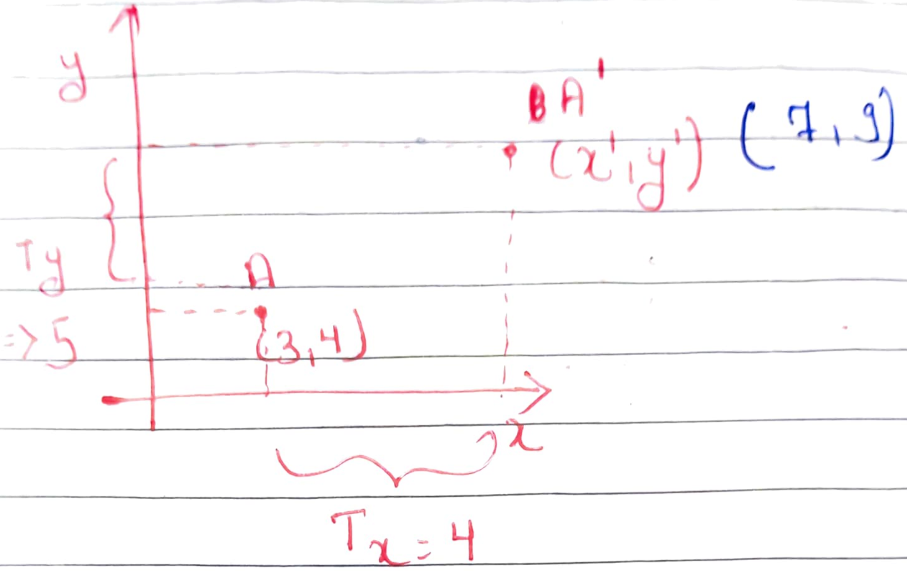
$T_x$ = Translation factor for x-axis

$T_y$ = Translation factor y-axis

$$\boxed{A' = A + T}$$

$A = \begin{bmatrix} x \\ y \end{bmatrix}$ $T = \begin{bmatrix} T_x \\ T_y \end{bmatrix}$ $A' = \begin{bmatrix} x' \\ y' \end{bmatrix}$

## Numericals :-




Sol$^n$

$A = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$ $\qquad T = \begin{bmatrix} 4 \\ 5 \end{bmatrix}$

$A' = \begin{bmatrix} x' \\ y' \end{bmatrix}$

$A' = A + T$

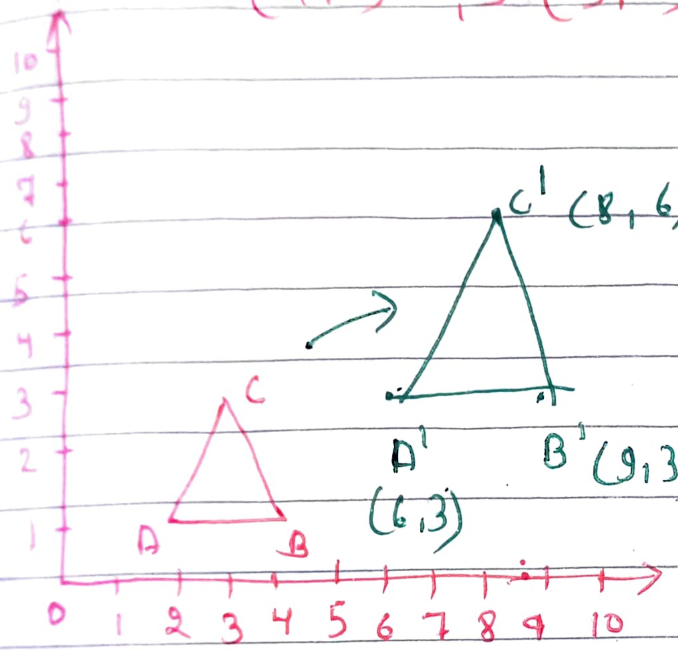
$= \begin{bmatrix} 3 \\ 4 \end{bmatrix} + \begin{bmatrix} 4 \\ 5 \end{bmatrix}$

$A' = \begin{bmatrix} 7 \\ 9 \end{bmatrix} =$

$x' = 7, \; y' = 9$

PAGE NO.:

4 units in "x" direc$^n$
2 units in "y" direc$^n$
A (2, 1) , B (5, 1) C (4, 4)




Sol$^n$

$A' = A + T$

$$A' = \begin{bmatrix} 2 \\ 1 \end{bmatrix} + \begin{bmatrix} 4 \\ 2 \end{bmatrix} = \begin{bmatrix} 6 \\ 3 \end{bmatrix}$$

$$B' = \begin{bmatrix} 5 \\ 1 \end{bmatrix} + \begin{bmatrix} 4 \\ 2 \end{bmatrix} = \begin{bmatrix} 9 \\ 3 \end{bmatrix}$$

$$C' = \begin{bmatrix} 4 \\ 4 \end{bmatrix} + \begin{bmatrix} 4 \\ 2 \end{bmatrix} = \begin{bmatrix} 8 \\ 6 \end{bmatrix}$$

# Scaling Numerical

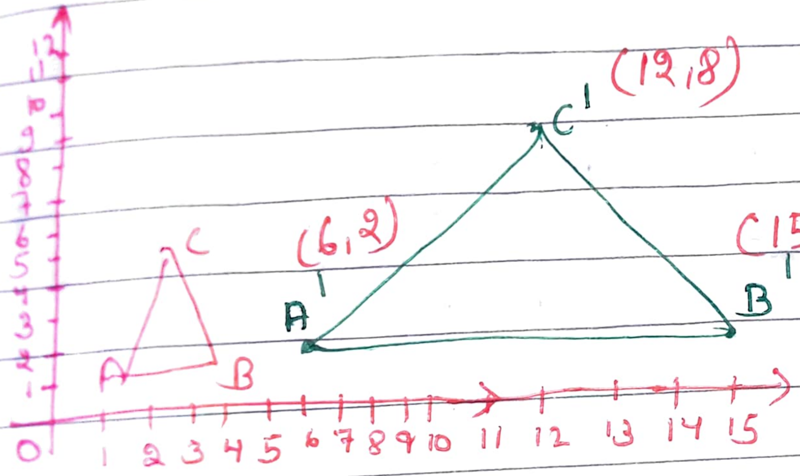


## Numericals 1

A(2,1), B(5,1) C(4,4)

3 units in "x"

2 units in "y"




**Numericals :** Given $S_x = 3$ unit, $S_y = 2$ unit

$$\boxed{A' = A \cdot S}$$

$$A' = A \cdot S$$

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

$$A' = \begin{bmatrix} 6 & 2 \end{bmatrix}$$

$$B' = B \cdot S$$

$$= \begin{bmatrix} 5 & 1 \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 15 & 2 \end{bmatrix}$$

PAGE NO.:

$C' = C \cdot S$

$= \begin{bmatrix} 4 & 4 \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}$

$= \begin{bmatrix} 12 & 8 \end{bmatrix}$

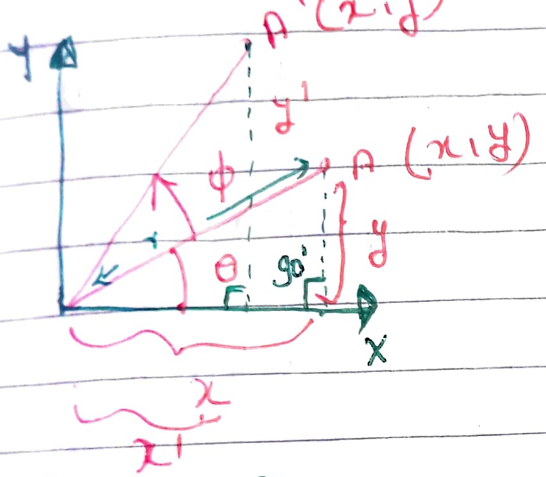
2. Scaling the square ABCD by 1.5 units in x-directⁿ and 0.5 units in y-directⁿ

3.

$\sin = \frac{P}{H}$ $\cos = \frac{B}{H}$ $\tan = \frac{P}{B}$

PAGE NO.:

**Rotation** → Repositioning an object into a circle along a circular path in x, y plane. Points can be rotated through angle θ about the origin.




anti clockwise → angle will (+)ve

clockwise → angle (−)ve

$\cos\theta = \frac{x}{r}$ $\sin\theta = \frac{y}{r}$

$\boxed{x = r\cos\theta}$ $\boxed{y = r\sin\theta}$

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

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

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

$[\cos(A+B) = \cos A\cos B - \sin A\sin B]$

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

$\boxed{x' = x\cdot\cos\phi - y\sin\phi}$

PAGE NO.

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

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

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

$$y' \Rightarrow r\sin\theta\cos\phi + r\cos\theta\sin\phi$$

$$\boxed{y' \Rightarrow y\cos\phi + x\sin\phi}$$

This $x'$ and $y'$ for anticlockwise direcⁿ

For Clockwise

$$x' = x\cos(-\phi) - y\sin(-\phi)$$

$$\boxed{x' = x\cos\phi + y\sin\phi}$$

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

PAGE NO.:

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

$= x\sin(-\phi) + y\cos\phi$

$= -x\sin\phi + y\cos\phi$

$A' = A \cdot R$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \cdot \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix}$$

for anticlock wise directn

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

for clock wise directn

Numerical 1

1) Rotate a $\Delta$ defined by $A(0,0)$, $B(6,0)$ and $C(3,3)$ by $90°$ about origin in anticlock wise direct

Ans

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

Homogeneous cordinate :- We have to use three operation as transformation namely translation, rotation and again translation. So instead of three transformation, we can accomplish this effect in single transformation. And this transformation can perform by using Homogeneous - co-ordinates.

In H.C we use 3x3 matrix instead of 2x2 and introduce additional dummy variable W.
W = 1 (non-negative value)

Generally points are represented as (x, y), but here every point is specified by a set of 3 values (xw, yw, w). The first value of homogeneous co-ordinate will be product of x and w, the second will be y & w & third will be just w.

~~The x and~~

But how we are going to represent this homogeneous point on screen because screen deals with only x and y. The x & y co-ordinates can be easily recovered by dividing the 1st & 2nd no by the third. We are not really using the 3rd variable w.

For simplicity we are keeping the value of w as 1

$$[x, y, 1] \; / \; \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

In this H.C we have to matrix multiplication concept)

For Translation :-

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

for Scaling :-

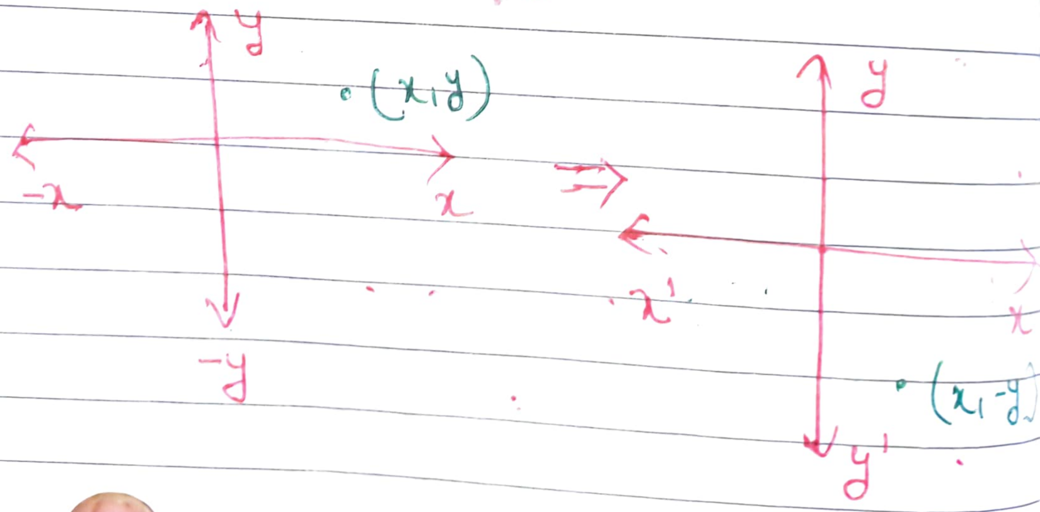
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

for rotation :-

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

Reflection ⇒ Reflection means mirror image

(i) Reflection at x-axis ⇒ We are assuming that we are keeping mirror at x-axis So if the point is $(x, y)$ then its reflection at x axis will become $(x, -y)$.

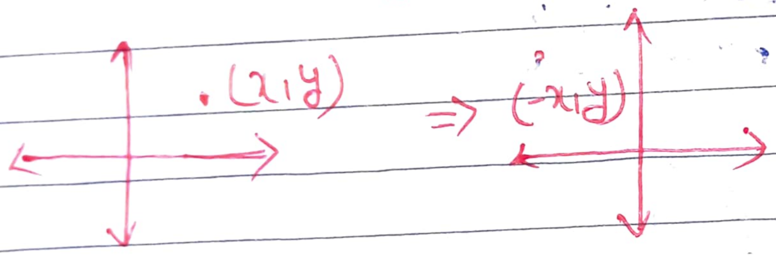


$x = x'$

$y = -y'$

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

**Reflection at y-axis** ⇒ If we have a point $(x, y)$ then the reflection at y axis will become $(-x, y)$




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

$x = -x'$

$y = y'$

$A' = R \cdot A$ [R = Reflection]

Reflection in the origin →

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

Numerical → A (3, 4)
B (2, 3)
C (4, 3)

Reflection on y-axis

$A' = \begin{bmatrix} -3 \\ 4 \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix}$

$B' = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$

$C' = \begin{bmatrix} -4 \\ 3 \end{bmatrix}$

Reflection on x-axis

$A' = \begin{bmatrix} 3 \\ -4 \end{bmatrix}$

$B' \begin{bmatrix} 2 \\ -3 \end{bmatrix}$

$C' \begin{bmatrix} 4 \\ -3 \end{bmatrix}$

shearing → The shear transformation means slanting the image. This transformation is of 2 types:-

1. x-shear
2. y-shear

1. x-shear ⇒ x-shear preserves the y-coordinate values and shift x co-ordinate value causing vertical lines to tilt.

In x-axis

$$x' = x + sh_x * y$$
$$y' = y$$

[$sh_x$ & $sh_y$ ⇒ shearing Parameters]

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

y-shear ⇒ In y-shear we are keeping x-coordinate value as it is and shifting the y co-ordinate value only, which result in tilting of horizontal lines

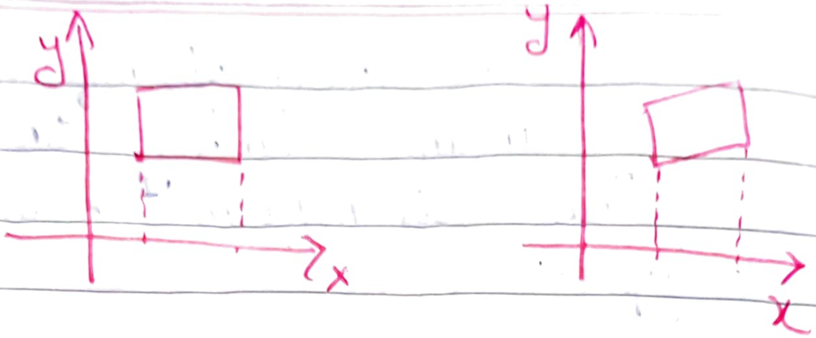


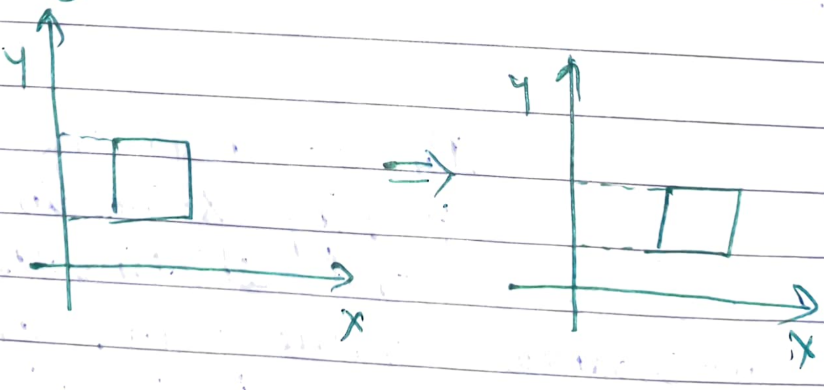

fig: y-shear

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

$$x' = x$$
$$y' = y + sh_y * x$$

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

fig: x-shear



Numericals:- Perform x-shear & y shear on a triangle having A(2,1), B(4,3), C(2,3). Consider the constant value a = b = 2.

Sol

A(2,1), B(4,3) C(2,3)

$Sh_x = 2$ $Sh_y = 2$

~~Reflect~~ ~~of~~ X-shear

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

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

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

$$B' = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$= \begin{bmatrix} 10 \\ 3 \end{bmatrix}$$

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

$$= \begin{bmatrix} 8 \\ 3 \end{bmatrix}$$

~~Reflection~~ Shearing at Y axis

$$A' = \begin{bmatrix} 1 & 0 \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 11 \end{bmatrix}$$

$$\cancel{B}\ A' = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix}$$

$$C' = \begin{bmatrix} 1 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 \\ 7 \end{bmatrix}$$

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 resultant will be identity matrix.

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

Similarly when we multiply any transformation matrix with its inverse, we will get identity matrix.

$$A^{-1} = \frac{\text{Adjoint } A}{|A|}$$

$|A| \rightarrow$ determinant A

$\hookrightarrow$ mode of A

step 1 $\Rightarrow$ To find the determinant of the matrix

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

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

Setep 2 $\Rightarrow$ finding the co-factor of the matrix

Co-factor of A will be

$$\Rightarrow \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix}$$

**Step 3** ~~To~~ After finding co-factor we have to find adjoint of matrix A. To find this adjoint we have to transpose of co-factor

[Adjoint ke liye sign change करना पड़ता है]

i.e

$$A^T = \begin{bmatrix} B_{11} & B_{21} & B_{31} \\ B_{12} & B_{22} & B_{32} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

row → column
column → row

**Step 4** Now Inverse of A

$$A^{-1} = \frac{Adj. A}{|A|}$$

$|A|$ → determinant

Inverse $\begin{bmatrix} a & d & 0 \\ b & e & 0 \\ c & f & 1 \end{bmatrix} \Rightarrow$

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

Numerical :-

If $A = \begin{bmatrix} 6 & 5 \\ 2 & 1 \end{bmatrix}$ find $A^{-1}$ by Adjoint method.

Sol$^n$ $\Rightarrow$ $|A| = \begin{vmatrix} 6 & 5 \\ 2 & 1 \end{vmatrix}$

$= (6 \times 1) - (2 \times 5)$

$= 6 - 10$

$|A| = -4$

$$A^{-1} = \frac{\text{Adj. } A}{|A|}$$

co-factor matrix A $= \begin{bmatrix} 1 & -2 \\ -5 & 6 \end{bmatrix}$

co-factor matrix $A^T$ $= \begin{bmatrix} 1 & -5 \\ -2 & 6 \end{bmatrix}$

$$A^{-1} = \frac{\text{Adj. } A}{|A|}$$

$$= \frac{1}{-4} \begin{bmatrix} 1 & -5 \\ -2 & 6 \end{bmatrix}$$

$$= \begin{bmatrix} -1/4 & 5/4 \\ 1/2 & 3/2 \end{bmatrix}$$

PAGE NO:

Composite Transformation ⇒ A sequence of transformation can be combined into single are called as composition.

We can perform ~~it~~ by sequence of 3 transfor-mation.

Composition of 2 Translation ⇒ Let $t_1, t_2, t_3, t_4$ are translation vectors. They are two translation $P_1$ and $P_2$. The matrix of $P_1$ and $P_2$ given below. The $P_1$ and $P_2$ are represented using homogeneous matrices and $P$ will be final transformation matrix obtained after multiplication.

$$P_1 = \begin{bmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{bmatrix} \qquad P_2 = \begin{bmatrix} 1 & 0 & t_3 \\ 0 & 1 & t_4 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P = \begin{bmatrix} 1 & 0 & t_1 + t_2 \\ 0 & 1 & t_3 + t_4 \\ 0 & 0 & 1 \end{bmatrix}$$

Above resultant matrix show that

2 successive translations are additive

Composition of 2 Rotation ⇒ Two rotations are also additive

$$R_A = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad R_B = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R = R_A + R_B = \begin{bmatrix} \\ \\ \end{bmatrix}$$

Composition of 2 scaling ⇒ Let $S_A$ and $S_B$ are matrix to be multiplied

$$S_A = \begin{bmatrix} S_1 & 0 & 0 \\ 0 & S_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad S_B = \begin{bmatrix} S_3 & 0 & 0 \\ 0 & S_4 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$S = S_A * S_B = \begin{bmatrix} S_1 * S_3 & 0 & 0 \\ 0 & S_2 * S_4 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

## Window and Clipping =>

**Windowing** :- The method of selecting and enlarging the portions of a drawing is called windowing.

**Clipping** :- Clipping eliminates the object or portion of object which are not visible through window. Determining which portion to omit or suppress is called Clipping.

## Viewing Transformation =>

* **World Co-ordinate System** => The object space contains the dimension as actual which is called as world co-ordinate system.

**Device co-ordinate system** => The image which we want to draw on display device must be measured in screen or device co-ordinate system.

~~Viewport~~ →

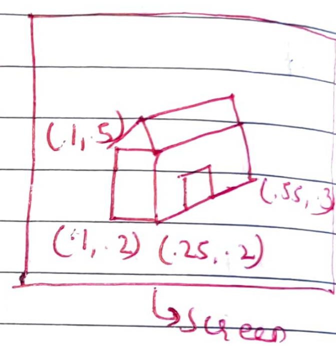




fig: Object Space
↓
(World coordinate system)

fig: Image Space
(Device coordinate system)

Viewport ⇒ We may not want to use entire screen for display. We just want some part of screen to display the image. So we will form a rectangular box on screen and in that box only we will display the image. This box is called as Viewport.

Defⁿ [An area on display device to which a window is mapped is called viewport.]

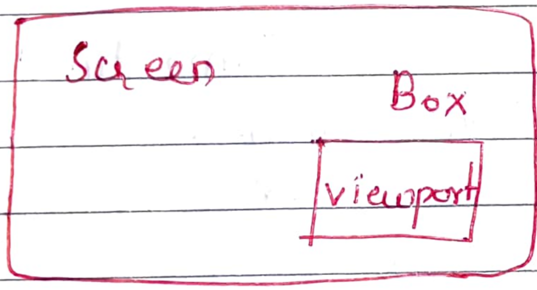



fig.:

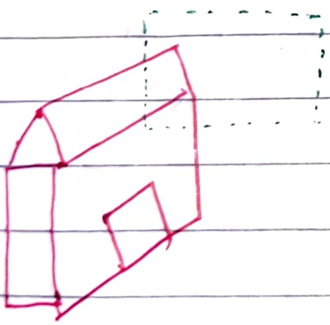



fig.: object space

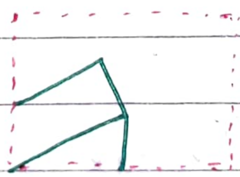


fig. Image space

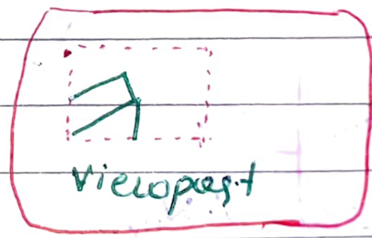



fig.: Screen

Window ⇒ A world cordinate are selected for display is called window

PAGE NO.:

(object का किसने Part दिखाना)

Window ⇒ What is to be viewed

Viewport ⇒ Where it is to be display

"The mapping of a part of world co-ordinate scene to device co-ordinate is refered as Viewing Transformation."

Some time it is also called as window – viewport transformation or Windowing transformation.

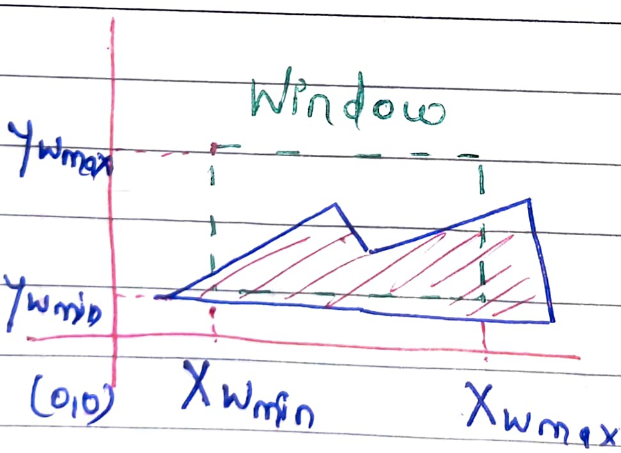
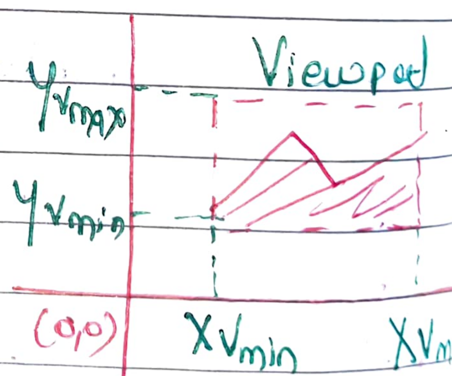



fig.. WCS



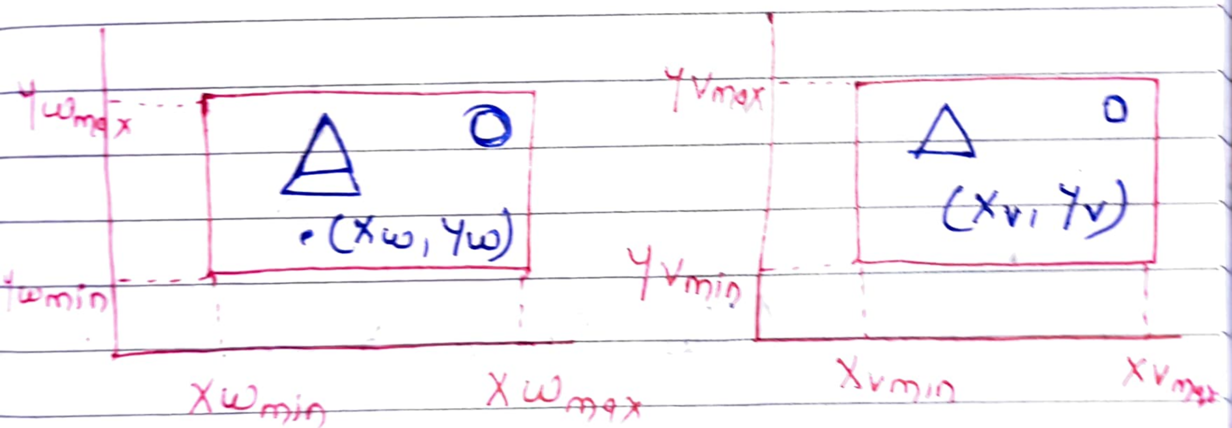

viewport

fig Device Co-ordinate

So the viewing transformation perform three steps

1. ~~Translation~~
2. Scaling
3. ~~Translation~~ Rotation.

Formula → Relative position will not change only size will change a/c to window and viewport.




Let us have $(X_w, Y_w)$ given we have to find $(X_v, Y_v)$

$$\frac{X_w - X_{wmin}}{X_{wmax} - X_{wmin}} = \frac{X_v - X_{vmin}}{X_{vmax} - X_{vmin}}$$

$$\frac{Y_w - Y_{wmin}}{Y_{vmax} - Y_{vmin}} = \frac{Y_v - Y_{vmin}}{Y_{vmax} - Y_{vmin}}$$

$$X_v - X_{vmin} = (X_{vmax} - X_{vmin})\left(\frac{X_w - X_{wmin}}{X_{wmax} - X_{wmin}}\right)$$

$$X_v - X_{vmin} = (X_w - X_{wmin}) * \underbrace{\left(\frac{X_{vmax} - X_{vmin}}{X_{wmax} - X_{wmin}}\right)}_{\text{scaling factor}}$$

$$X_v - X_{vmin} = (X_w - X_{wmin}) * S_x$$

$$\boxed{X_v = X_{vmin} + (X_w - X_{wmin}) S_x} \quad (1)$$

Similarly

$$\boxed{Y_v = Y_{vmin} + (Y_w - Y_{wmin}) . S_y} \quad (2)$$

$S_x$ & $S_y$ ⇒ Scaling factor

Clipping Algorithm ⇒ There are diff^n algorithm to clip different types of images. The images may be lines, polygons or text.

(i) Line clipping ⇒ It is ~~so~~ simplest to clip a straight line.

window

fig.: Before clipping

window

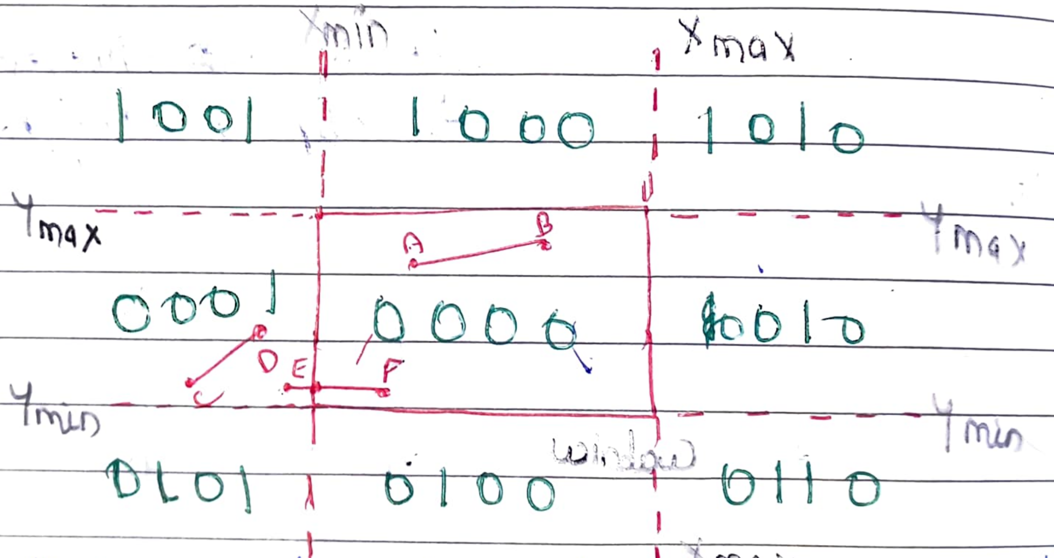
fig. After clipping

Cohen - Sutherland line clipping ⇒ Cohen Sutherland outcode algorithm is one of the popular line clipping algorithm.

This algorithm immediately removes the lines which are lying totally outside the window

This algorithm divide the plane in 9 parts and assign the outcode or binary no to each part

ABRL




End point of each line is assigned a 4 bit binary code which is called as outcode. We can give abbreviated name to this 4 bit code as "ABRL"

A ⇒ Above - Top
B ⇒ Below - Bottom
R ⇒ Right
L ⇒ Left.

PAGE NO.:

Case I ⇒ (Line inside) ⇒ Consider a line AB are surely inside the window. Therefore the outcode for end point A will be

A ⇒ 0000
B ⇒ 0000

Both end points of a line are 0000

Case II ⇒ (Line outside) Consider a line CD are surely outside the window. Therefore the outcode for end point C will be

T · I ⇒ T

C ⇒ 0001
D ⇒ 0001

AND 0001

↳ it's not equal to 0000

mean's its completely outside the window (AND result is non zero)

Case III ⇒ (Partially visible) We cannot make firm statement about a line, whether it should be visible or not visible or partially visible.

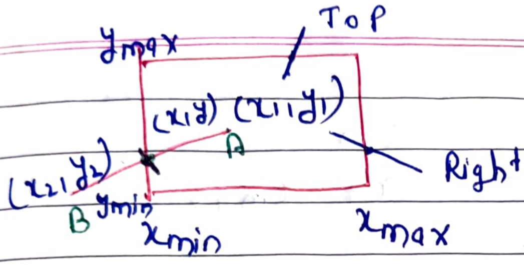
Consider a line EF

E ⇒ 0001
F ⇒ 0000

AND 0000

~~The AND result is zero. It mean the line EF is not lieing completely on any one side of the window.~~

In this condⁿ we have find the intersection point.




Window left Boundary ⇒

(Slope)

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

Left ⇒

$$x = x_{min}$$

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

$$m = \frac{y - y_1}{x - x_1}$$

$$m = \frac{y - y_1}{x_{min} - x_1}$$

$$y - y_1 = m(x_{min} - x_1)$$

$$\boxed{y = y_1 + m(x_{min} - x_1)}$$

PAGE NO.:

**Right ⇒**

$$x = x_{max}$$

$$\boxed{y = y_1 + m(x_{max} - x_1)}$$

**Top ⇒**

$$y = y_{max}$$

$$m = \frac{y - y_1}{x - x_1}$$

$$m = \frac{y_{max} - y_1}{x - x_1}$$

$$x - x_1 = \frac{y_{max} - y_1}{m}$$

$$\boxed{x = \frac{1}{m}(y_{max} - y_1) + x_1}$$

PAGE NO.:

~~Bottom~~ ⇒

Bottom ⇒

$$y = y_{min}$$

$$\boxed{x = \frac{1}{m}(y_{min} - y_1) + x_1}$$

~~Algorithm~~ ⇒ Pseudo code:

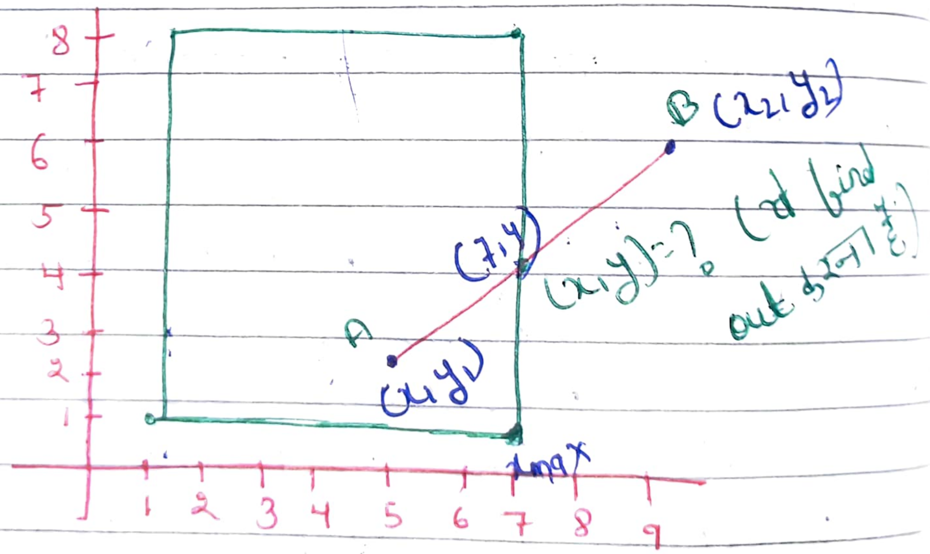
(i) Assign the region code for 2 endpoints of a given line.
(ii) If both region codes 0000 then line accepted completely
(iii) else
perform logical AND operation for both region codes.
(a) if result ≠ 0000 line is outside
(b) else line is partially inside
(i) chose an end point of the line that is outside the given rectangle
(ii) find intersection point
(iii) Replace ~~enters~~ end point with the intersection point & update region code
(iv) Repeat step2 until line is trivially accepted or trivially rejected

(V) Repeat step 1 for other lines

Numerical ⇒

Bottom, left : (1,1)
Upper, Right : (7,8)

line coordinate (5, 2) (9, 6)




$x_{min} = 1$ $\quad$ $x_1 = 5$

$y_{min} = 1$ $\quad$ $y_1 = 2$

$x_{max} = 7$ $\quad$ $x_2 = 9$

$y_{max} = 8$ $\quad$ $y_2 = 6$

It come under the partially visible.

$A = 0000$

$B = 0010$

AND $= 0000 \rightarrow$ (not completely outside and is partially visible)

The case of Right boundary of the window.

$x = x_{max}$

$x = 7$

$m = \dfrac{y_2 - y_1}{x_2 - x_1}$

$= \dfrac{4}{4} = 1$

$y = y_1 + m\,(x_{max} - x_1)$

$= 2\cancel{6} + 1\,(7 - 5)$

$= 2\cancel{6} + 1 \times 2$

$= 2\cancel{6} + 2$

$= \cancel{8}4$

$\boxed{x = 7}$ $\boxed{y = 4}$

Polygon clipping ⇒ For polygon clipping, we require an algorithm which will produce a closed figure.

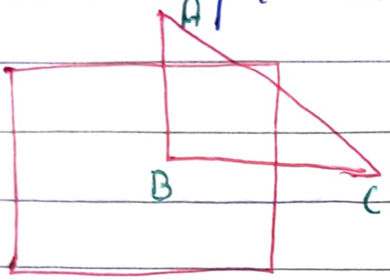



fig:- Before clipping.

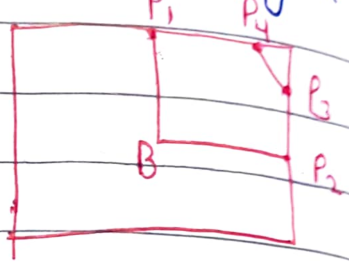
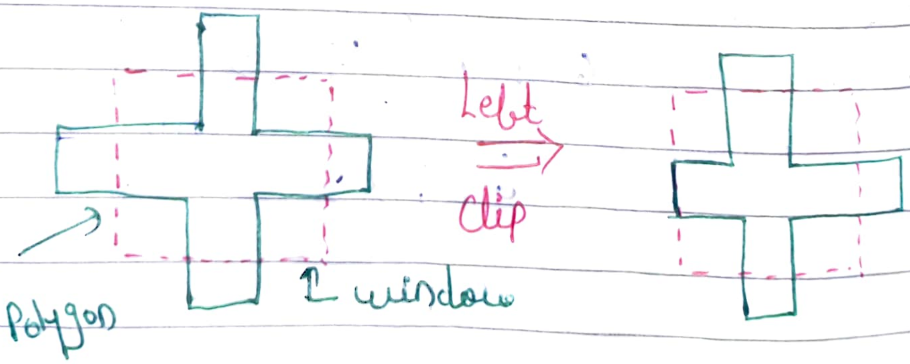


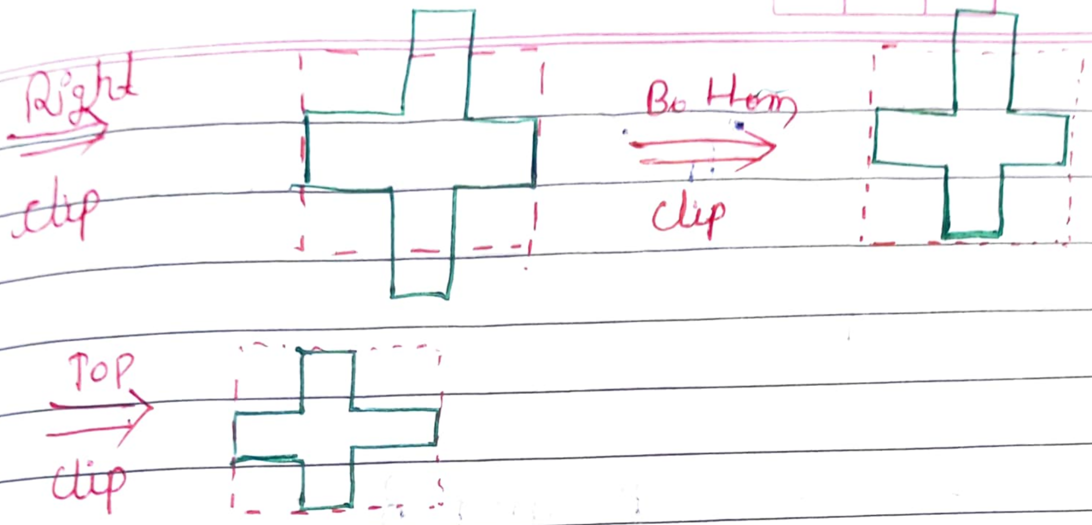

fig:- After clipping

Polygon clipping Algorithm :-

(i) Sutherland Hodgeman polygon clipping :-

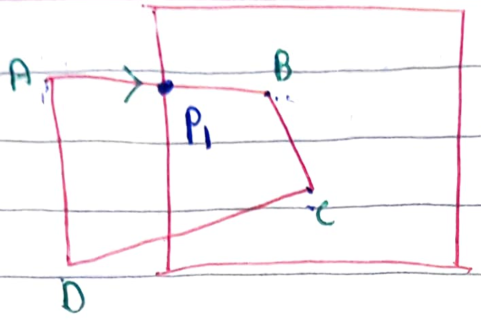
⇒ In this four different stages which is required to clip a polygon. This shown is below figure.






At the end of every clipping stage a new set of vertices is generated and this new set or modified polygon is passed to the next clipping stage. After clipping a polygon with respect to all the four boundaries we will get final clipped polygon.

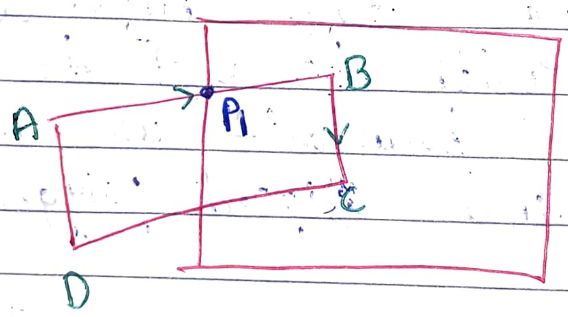
Case I ⇒ (Outside → Inside) If the first vertex is outside the window boundary & the second vertex is inside the window, then the intersection point of polygon with boundary edge of window and the vertex which is inside the window is stored in a o/p vertex list (i.e it will add as new vertex points)




$(P_1, B)$

store ⇒ $P_1$ and B in o/p vertex list.

Case II ⇒ (Inside → Inside)

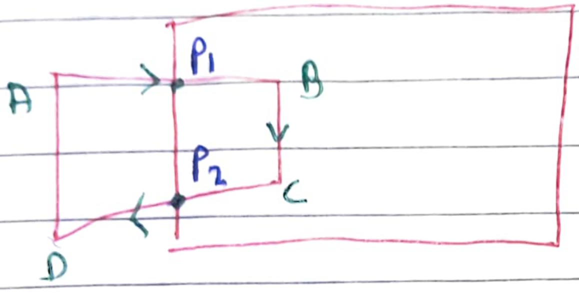



(C)

If both first and second vertex of a polygon are lying inside the window, then we have to store the second vertex only in o/p vertex list.

store ⇒ Only C in o/p vertex list.

case III ⇒ (Inside → outside)

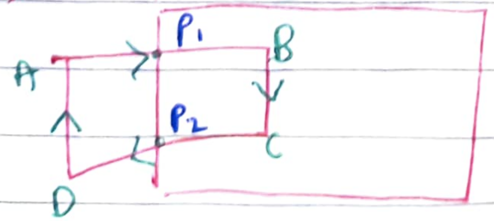



(Only $P_2$)

If the first vertex is inside the window and second vertex is outside the window i.e then we have to store only intersection point of that edge of polygon with window in o/p vertex list.

store ⇒ $P_2$ only in o/p vertex list.

Case IV ⇒ (Outside → outside)




(Nothing is stored in o/p vertex list)

PAGE NO.:

If both first and second vertex of a polygon are lying outside the window then no vertex is stored in o/p vertex list.

PAGE NO.:

Algo ⇒ Cohen Sutherland line clipping

(i) assign Region code to both end points say $C_0$ and $C_1$

(ii) if $C_0$ OR $C_1$ $== 0000$

then accepted completely (inside window)

else if

$C_0$ Logical AND $C_1 \neq 0000$ Reject it

else

Clip if line crossed $X_{min}$ or $X_{max}$

then

$$y = y_1 + m(x - x_1) \quad x = X_{max} \text{ or } X_{min}$$

else

$$x = x_1 + \frac{1}{m}(y - y_1) \quad y = Y_{max} \text{ or } Y_{min}$$

(iii) verify

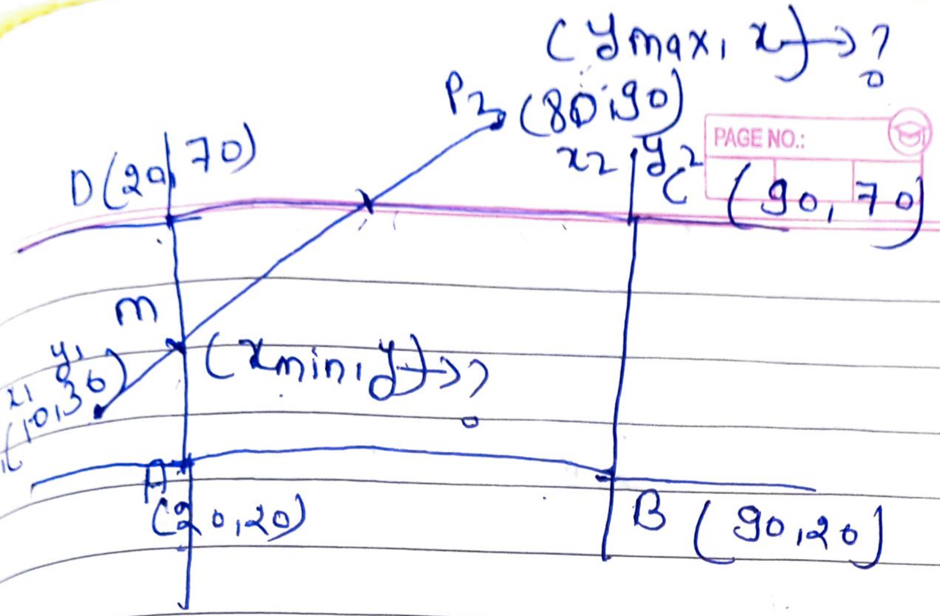
$x_{min} \le x \le x_{max}$

$y_{min} \le y \le y_{max}$

if it doesn't satisfy

then repeat (Point का x आया है की $x_{min}$ & $x_{max}$ के बिच आ रहा है या नहीं)

<u>Numerical</u> Let ABCD be the rectangular window with A(20, 20), B(90, 20), C(90, 70) and D(20, 70). Find region codes for the endpoints and use Cohen Sutherland algo to clip the line $P_1$ & $P_2$ with $P_1(10, 30)$ & $P_2(80, 90)$




$y = ?$

$x = ?$

$P_1 = 0001$

$P_2 = 1000$

AND: 

0000 → Partially visible

$x_{min} = 20$ $\qquad m = \frac{y_2 - y_1}{x_2 - x_1}$

$y_{max} = 90$ $\qquad = \frac{90-30}{80-10}$

$= 0.85$

$y = y_1 + m(x_{min} - x_1)$

$= 30 + 0.85(20 - 10)$

$\boxed{y = 38.57}$

$m = (20, 38.57)$

$y_{max} = y = 90$

$x = ?$

$x = x_1 + \frac{1}{m}(y_{max} - y_1)$

$x = \cancel{80}^{10} + \frac{1}{0.85}(\cancel{90}^{70} - 30)$

$= \cancel{8}^{1}0 + \frac{1}{0.85} \times 60$

$= 56.67$