Computer Graphics & multi media Systems.

UNIT - II

Basic Transformation

1. Translation
2. Scaling
3. Rotation

Transformation means a change in the orientation, size and shape of the object.

Other Transformation

1. Reflection
2. Shearing

Translation

The translation is repositioning an object along a straight-line path from one co-ordinate location to another co-ordinate location.

The translation is the rigid body transformation that saves an object without deformation.

A transformation

A translation moves to a different position on the screen.

$P(x, y)$ → Point before translation
$P'(x', y')$ → Point after translation
t_x, t_y → Translation Parameter with respect to x, y
or translation vector or Shift vector.
or distance vector.

$P'(x', y')$

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

Matrix representation

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

or

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

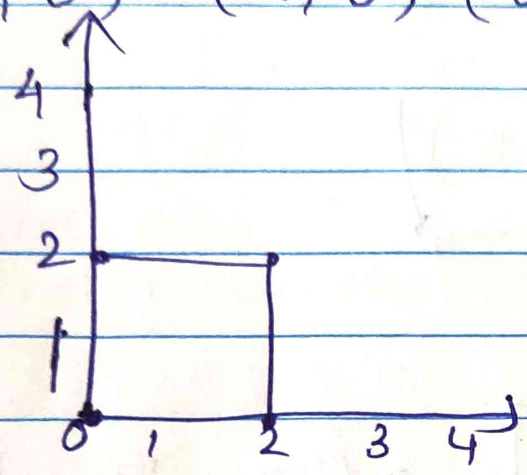
$$\boxed{P' = P + T}$$

example

A B C D

(0,0) (2,0) (0,2) (2,2)

$t_x = 2$ $t_y = 3$



before translation

A → (0,0)

$x' = x + t_x = 0 + 2 = 2$

$y' = y + t_y = 0 + 3 = 3$

(0,0) → (2,3)

B → (2,0)

$x' = x + t_x = 2 + 2 = 4$

$y' = y + t_y = 0 + 3 = 3$

(2,0) → (4,3)

C → 0,2

$x' = x + t_x = 0 + 2 = 2$

$y' = y + t_y = 2 + 3 = 5$

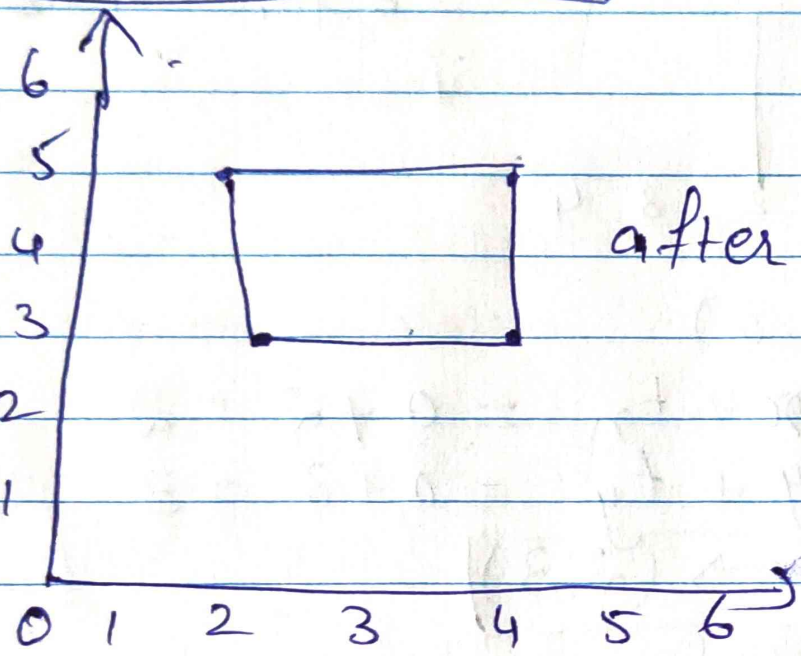
(0,2) → (2,5)

D → 2, 2

$x' = x + t_x = 2 + 2 = 4$

$y' = y + t_y = 2 + 3 = 5$

$(2, 2) \rightarrow (4, 5)$



after translation

2. Scaling

Scaling is the transformation that is used to change the object's size.

The operation is carried out by multiplying the co-ordinate value (x, y) with S_x & S_y scaling factors.

Scaling is performed about the Origin as

1) If S_x & $S_y > 1$
then enlarges the object and moves away from the origin Shape of the object also changes.

(2) If S_x & $S_y < 1$
Then reduce the object and moves towards its origin

(3) If S_x & $S_y = 1$
Then leaves the object as the same.

∴ The equation for scaling is given by

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

S_x, S_y → Scaling factor w.r.t. x & y

Matrix representation

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

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

Example: S_x & $S_y > 1$

$(0,0)$ $(2,0)$ $(0,2)$ $(2,2)$

$S_x = 2$ $S_y = 3$

$A = (0,0)$

$x' = x \cdot S_x = 0 \cdot 2 = 0$

$y' = y \cdot S_y = 0 \cdot 3 = 0$

$\boxed{(0,0) \rightarrow (0,0)}$

$B \rightarrow (2,0)$

$x' = 2 \cdot 2 = 4$

$y' = 0 \cdot 3 = 0$

$\boxed{(2,0) \rightarrow (4,0)}$

C → 0, 2

$x' = 0 \cdot 2 = 0$

$y' = 2 \cdot 3 = 6$

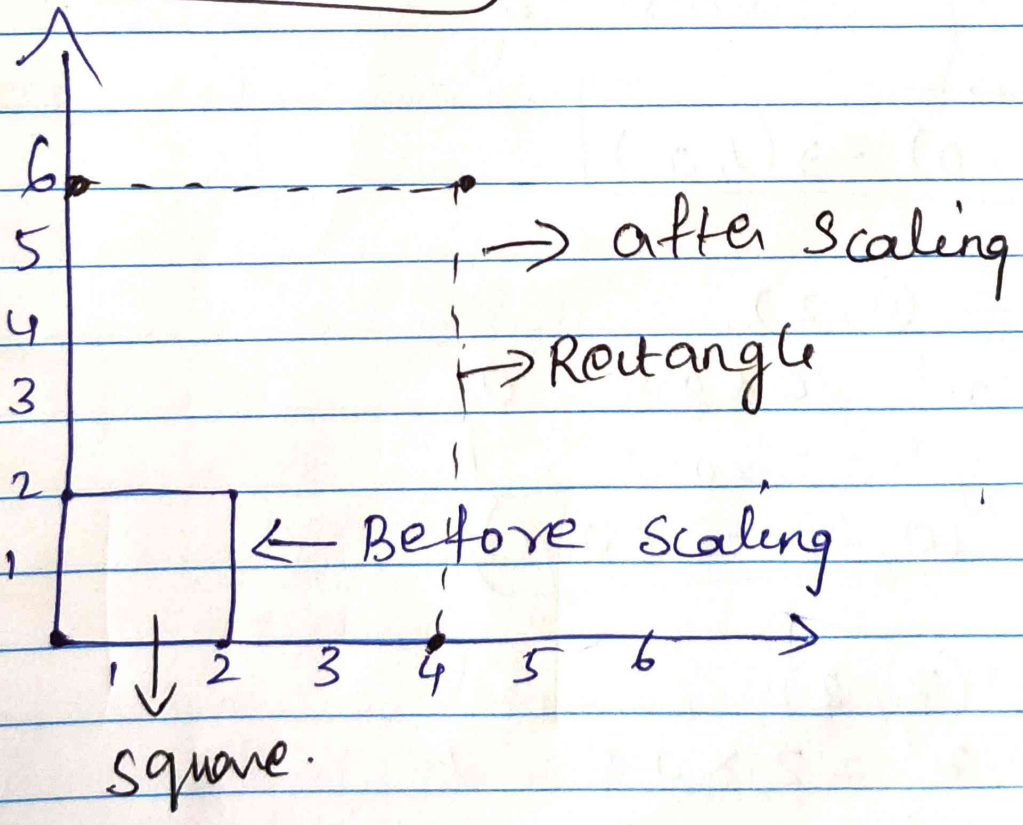
(0, 2) → (0, 6)

D → (2, 2)

$x' = 2 \cdot 2 = 4$

$y' = 2 \cdot 3 = 6$

(2, 2) → (4, 6)



Example 2: $S_x, S_y < 1$

A(0,0), B(2,0), C(0,2), D(2,2)

A 0,0

$x' = 0 \times 0.5 = 0$

$y' = 0 \times 0.5 = 0$

$(0,0) \rightarrow (0,0)$

B $\rightarrow$ (2,0)

$x' = 2 \times 0.5 = 1$

$y' = 0 \times 0.5 = 0$

$(2,0) \rightarrow (1,0)$

C $\rightarrow$ (0,2)

$x' = 0 \times 0.5 = 0$

$y' = 2 \times 0.5 = 1$

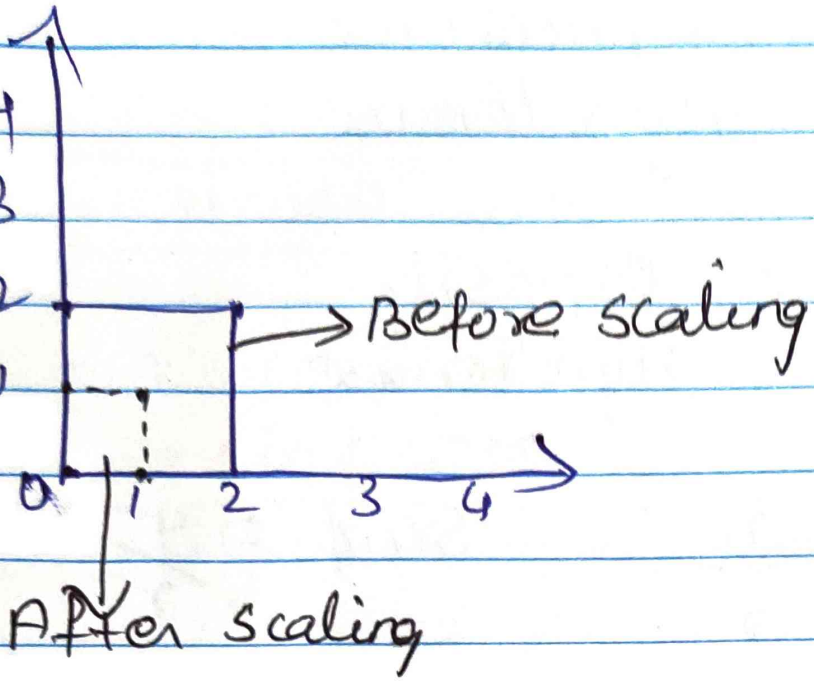
$(0,2) \rightarrow (0,1)$

D $\rightarrow$ (2,2)

$x' = 2 \times 0.5 = 1$

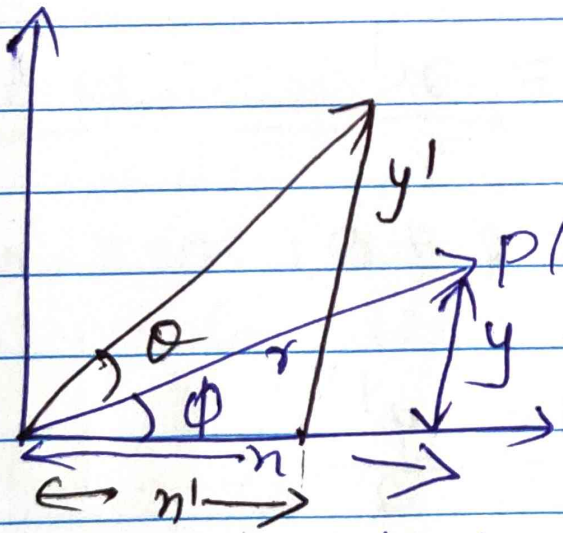
$y' = 2 \times 0.5 = 1$

$(2,2) \rightarrow (1,1)$



3. Rotation

Rotate an object with a certain angle θ either clockwise or Anti-clockwise direction



Using standard trigonometric, the original co-ordinate of points P x & y can be represented as

$$\cos\phi = \frac{\text{Adjacent side}}{\text{Hypotenuse}}$$

$$\sin\phi = \frac{\text{Opposite}}{\text{Hypotenuse}}$$

$$\cos\phi = \frac{x}{r} \qquad \sin\phi = \frac{y}{r}$$

$$x = r\cos\phi \qquad y = r\sin\phi$$

The new angle after rotation

$$P \rightarrow P' = (\phi + \theta)$$

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

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

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

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

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

$$\sin(A+B) = \sin A\cos B + \cos A\sin B$$

$$x' = r\cos(\phi+\theta)$$
$$= r\cos\phi\cos\theta - r\sin\phi\sin\theta$$
$$\boxed{x' = x\cos\theta - y\sin\theta}$$

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

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

↓

Anti-Clockwise direction.

Clockwise direction

$x' = r\cos\phi\cos\theta$.

New angle after rotation p to p'

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

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

$\sin(A+B) = \sin A\cos B + \cos A\sin B$

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

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

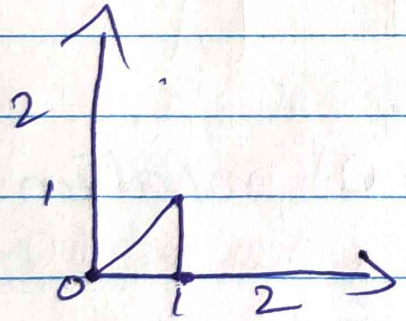
$$\begin{bmatrix} x' & y' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

Example

Triangle (0,0) (1,0) (1,1)

$\theta = 90°$ (Anti-Clock)

$\sin 90° = 1$ $\cos 90° = 0$



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

0,0

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

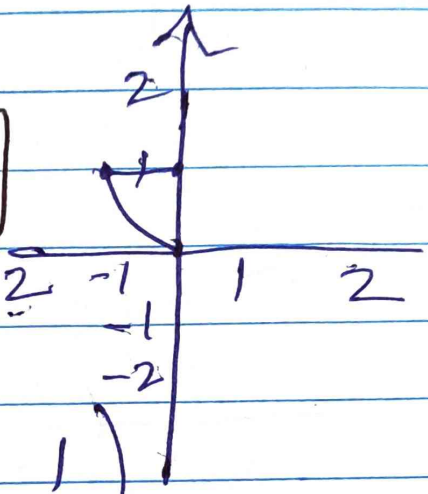
$(0,0) \rightarrow (0,0)$

$(1,0)$

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

$[x' \; y'] = [0 \quad 1]$

$(1,0) \rightarrow (0,1)$



$(1,1)$

$[x' \; y'] = (1, \; 1) \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$

$= -1, 1$

$[1, \; 1] \rightarrow [-1, \; 1]$

If the direction is not mentioned then it is a anti-clock wise direction by default.

Example for clockwise.

(0,0) (1,0) (1,1)

$\theta = 90^\circ$

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

0,0

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

$(0,0) \Rightarrow (0,0)$

(1,0)

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

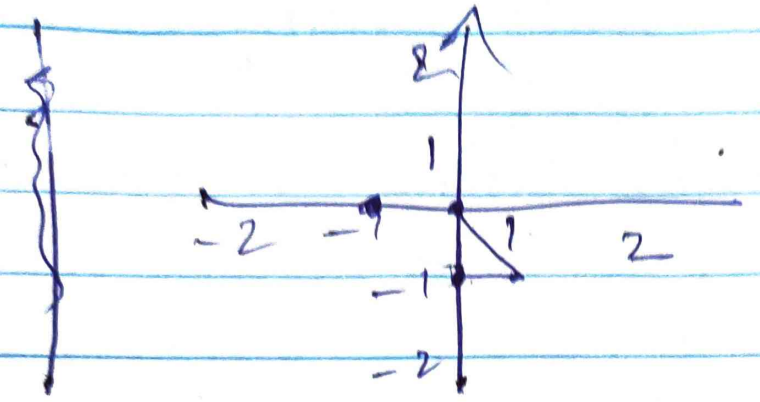
$= (0, -1)$

$(1,0) \rightarrow (0,-1)$

1,1

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

$(1,1) = (1,-1)$



4. Shearing

Shearing is the transformation used to change the shape of an existing object in the 2D plane.

The size of the object changes along the x direction as well as the y direction.

There are two types of shearing

1. X – Shear
2. Y – Shear

It is also called as Skewing

X – Shear

It preserves the y

Co-ordinates but changes the x value

∴ y remain same x value changes.

$$y' = y$$
$$x' = x + Sh_x \cdot y$$

$Sh_x \rightarrow$ Shearing parameter w.r.t x.

Matrix representation

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

Y-Shear

It Preserves the x Co-ordinates but changes the y values

$$x' = x$$
$$y' = y + Sh_y x$$

$Sh_y \rightarrow$ Shearing parameter w.r.t y

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

Example

(0,0) (0,2) (2,0) (2,2)

$Shx = 2$

$y' = y$

$x' = x + Shx \cdot y$

(0,0) $y' = 0$

$x' = 0 + 2 \cdot 0 = 0$

$(0,0) \rightarrow (0,0)$

(0,2) $y' = 2$

$x' = 0 + 2 \cdot 2 = 4$

$(0,2) \rightarrow (4,2)$

(2,0) $y' = 0$

$x' = 2 + 2 \cdot 0 = 2$

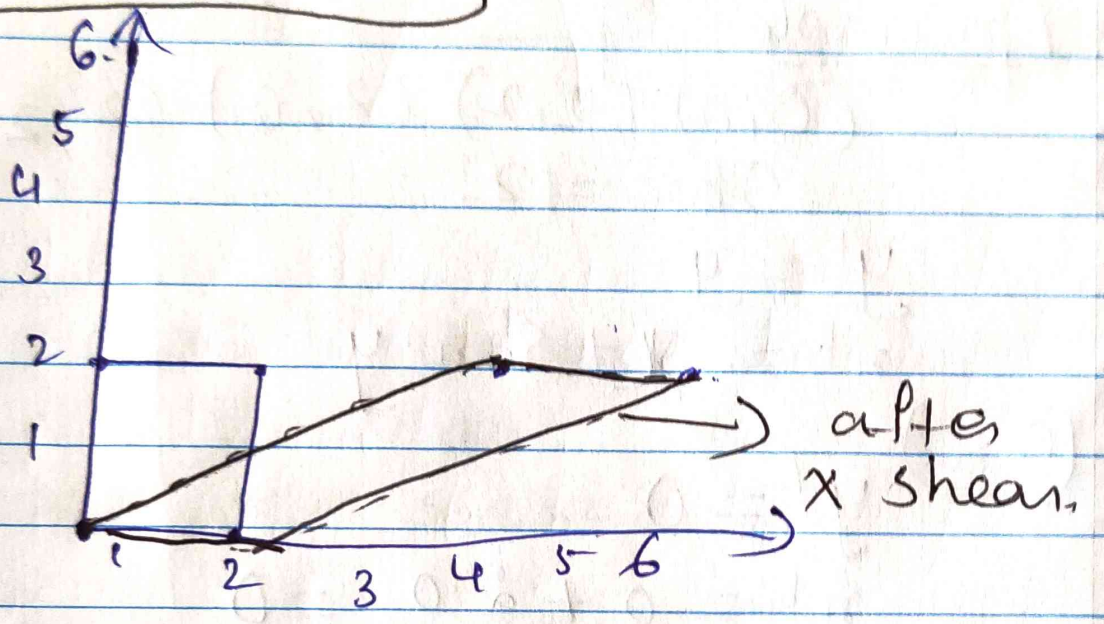
$(2,0) \rightarrow (2,0)$

(2, 2)

$y' = 2$

$x' = 2 + 2 \cdot 2 = 6$

$(2, 2) \rightarrow (6, 2)$



Example for Y-shear.

(0, 0) (0, 2) (2, 0) (2, 2)

$shy = 2$

(0, 0) $\rightarrow$ (0, 0)

$x' = x$

$y' = y + shy \cdot x$

$x' = 0$, $y' = 0 + 2 \cdot 0 = 0$

(0, 2)

$x' = 0$

$y' = 2 + 2 \cdot 0 = 2$

$(0, 2) \rightarrow 0, 2$

(2, 0)

$x' = 2$

$y' = 0 + 2 \cdot 2 = 4$

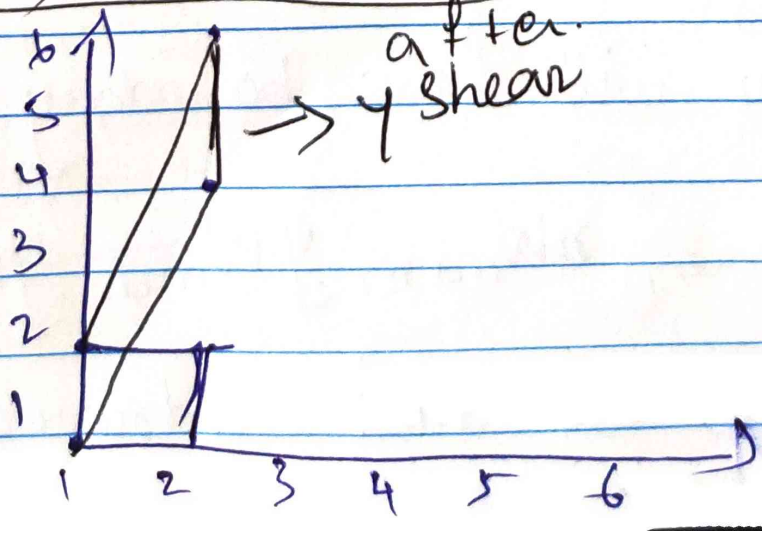
$(2, 0) \rightarrow (2, 4)$

(2, 2)

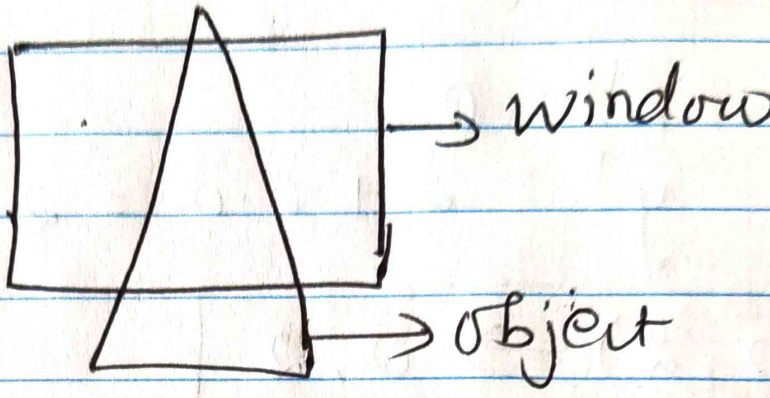
$x' = 2$

$y' = 2 + 2 \cdot 2 = 6$

$(2, 2) \rightarrow (2, 6)$



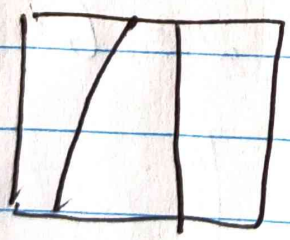
Window to viewport transformation



Window → The portion in which the object should be displayed is called window.

* What to be displayed.

viewport The place where we are displaying the object at the window.



Where to be displayed?

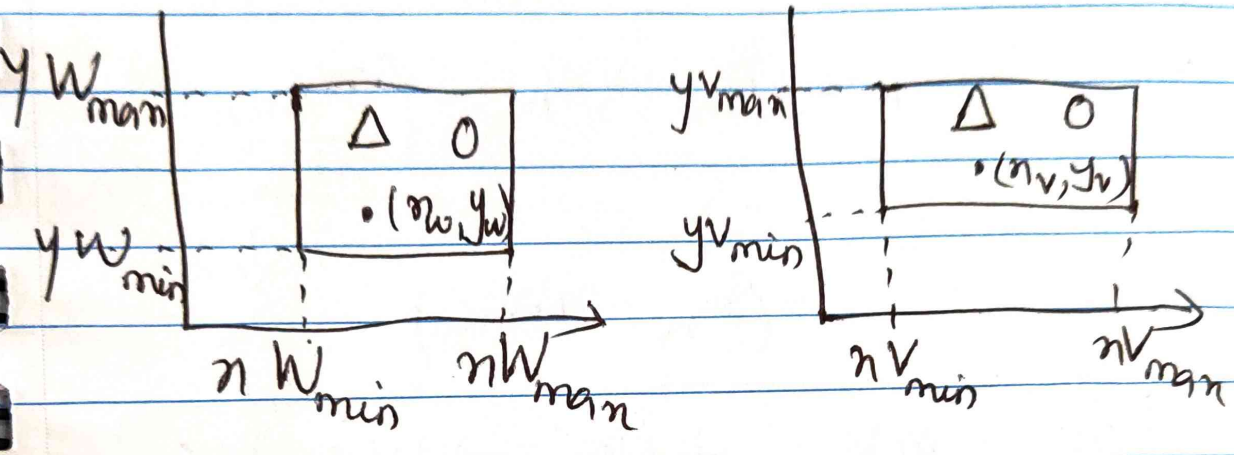
Window will have boundary values

xW_{min}, xW_{max}, yW_{min}, yW_{max}

These are also called window co-ordinates

Boundary values of viewport is called device co-ordinates.

xV_{min}, xV_{max}, yV_{min}, yV_{max}



1. Relative position of the object will not change. It is same for both window & viewport.
2. Scaling is done. ie size of the object will be changed.

$$\frac{x_w - xW_{min}}{xW_{max} - xW_{min}} = \frac{x_v - xV_{min}}{xV_{max} - xV_{min}}$$

$$\frac{y_w - yW_{min}}{yW_{max} - yW_{min}} = \frac{y_v - yV_{min}}{yV_{max} - yV_{min}}$$

$$x_v - xV_{min} = \left(\frac{x_w - xW_{min}}{xW_{max} - xW_{min}}\right) (xV_{max} - xV_{min})$$

$$x_v - xV_{min} = (x_w - xW_{min}) \left(\frac{xV_{max} - xV_{min}}{xW_{max} - xW_{min}}\right)$$

$$= (x_w - xW_{min}) \; S_x$$

$$\boxed{x_v = xV_{min} + (x_w - xW_{min}) \cdot S_x}$$

$$y_v - yV_{min} = (yV_{max} - yV_{min}) \left(\frac{y_w - yW_{min}}{yW_{max} - yW_{min}}\right)$$

$$y_v = yV_{min} + (y_w - yW_{min}) \cdot \left(\frac{yV_{max} - yV_{min}}{yW_{max} - yW_{min}}\right)$$

$$\boxed{y_v = yV_{min} + (y_w - yW_{min}) \cdot S_y}$$

$S_x, S_y \rightarrow$ scaling factors.

Example

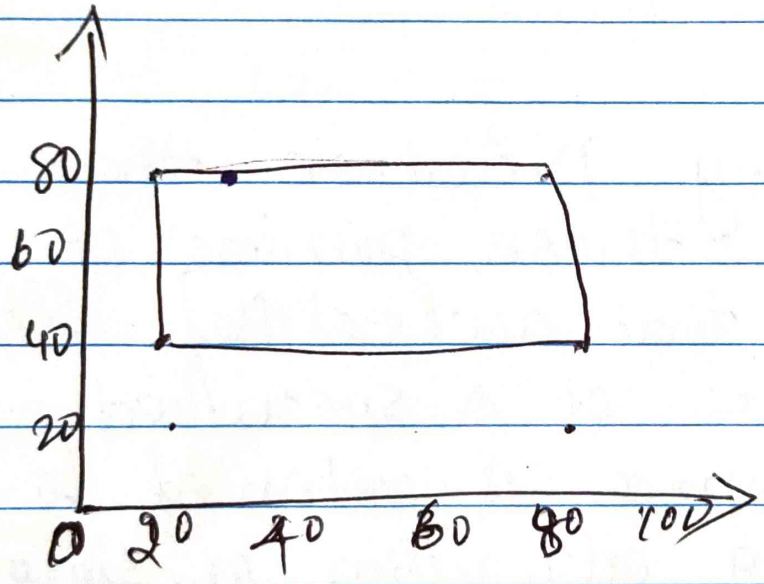
$x_{W_{min}} = 20$ $\quad$ $x_{V_{min}} = 30$

$x_{W_{max}} = 80$ $\quad$ $x_{V_{max}} = 60$

$y_{W_{min}} = 40$ $\quad$ $y_{V_{min}} = 40$

$y_{W_{max}} = 80$ $\quad$ $y_{V_{max}} = 60$

$(x_w, y_w) \rightarrow 30, 80$ $\quad$ $(x_v, y_v) = ?$



$$\frac{x_v - 30}{60 - 30} = \frac{30 - 20}{80 - 20}$$

$$x_v - 30 = \frac{10^5}{60_2} \times 30$$

$$x_v = 5 + 30 = 35$$

$$\boxed{x_v = 35}$$

$$\frac{y_v - 40}{60 - 40} = \frac{80 - 40}{80 - 40}$$

$$y_v - 40 = \frac{40}{40} \times 20$$

$$\boxed{y_v = 60}$$

T.

Clipping

Any procedure that identifies those portions of a picture that are either inside or outside of a specified region space is referred to as a clipping algorithm or simply clipping.

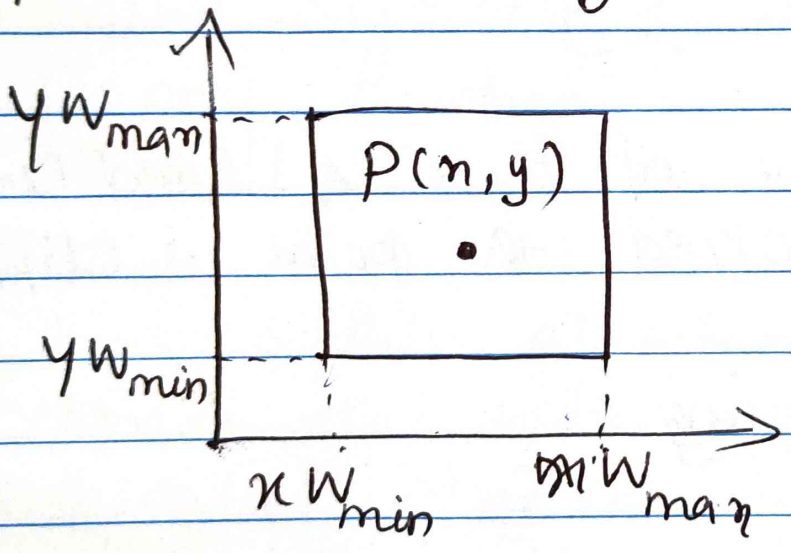
The region against which an object is to be clipped is called a clip window.

Clipping is the process to identify the picture either inside or outside of the displaying area.

Types of clipping

1. Point clipping
2. Polygon clipping
3. Line clipping
4. Text clipping
5. Curve clipping

1. Point clipping



Point clipping is used to determine whether the point is inside the window or not.

Assuming that clip window is a rectangle in the standard position we have a point P(x,y)

for display, if the following conditions are satisfied.

$$xW_{min} \leq x \leq xW_{max}$$
$$yW_{min} \leq y \leq yW_{max}.$$

1. $xW_{min} \leq x$
2. $xW_{max} \geq x$
3. $yW_{min} \leq y$
4. $yW_{max} \geq y$

If any of these 4 conditions is not satisfied the point is clipped.

---

2. Line Clipping

The line clipping is a process in which we can cut the part of the line, which lies outside the view plane.

Only those lines are visible, which lie inside the view plane

The Line clipping process is implemented by following line clipping algorithms

1. Cohen - Sutherland Line clipping
2. Liang - Barsky Line clipping
3. Midpoint Subdivision Line clipping

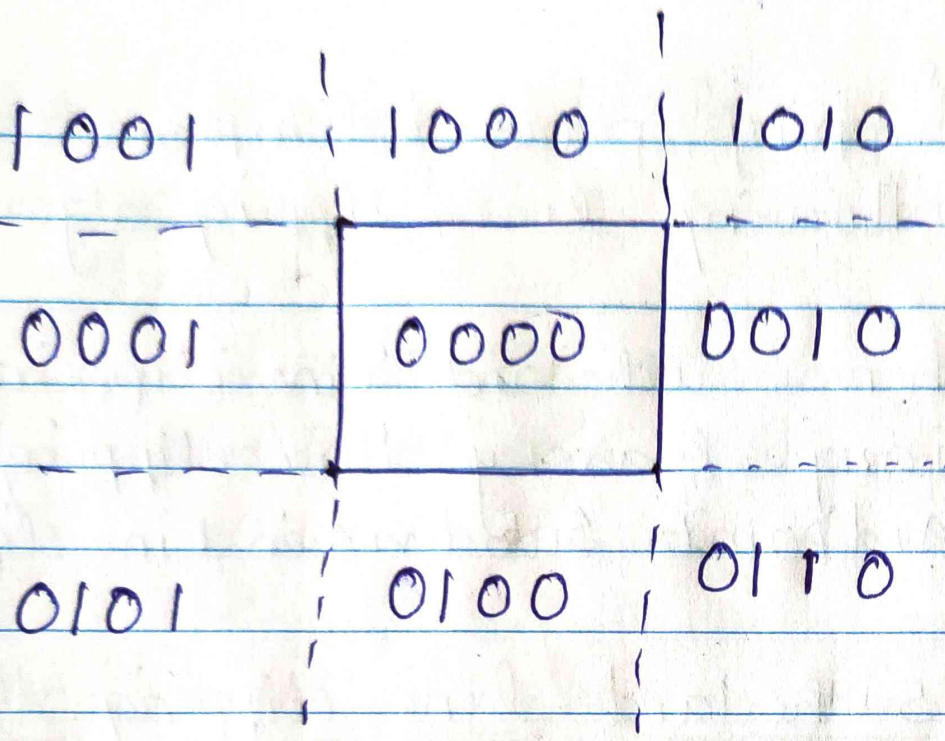
Cohen Sutherland line clipping Alg

The algorithm was named by "Danny Cohen" and "Ivan Sutherland".

=> In this algorithm we will divide the view plane into 9 equal segments. that only save the viewport.

=> Next each segment is represented using 4 bit region code TBRL

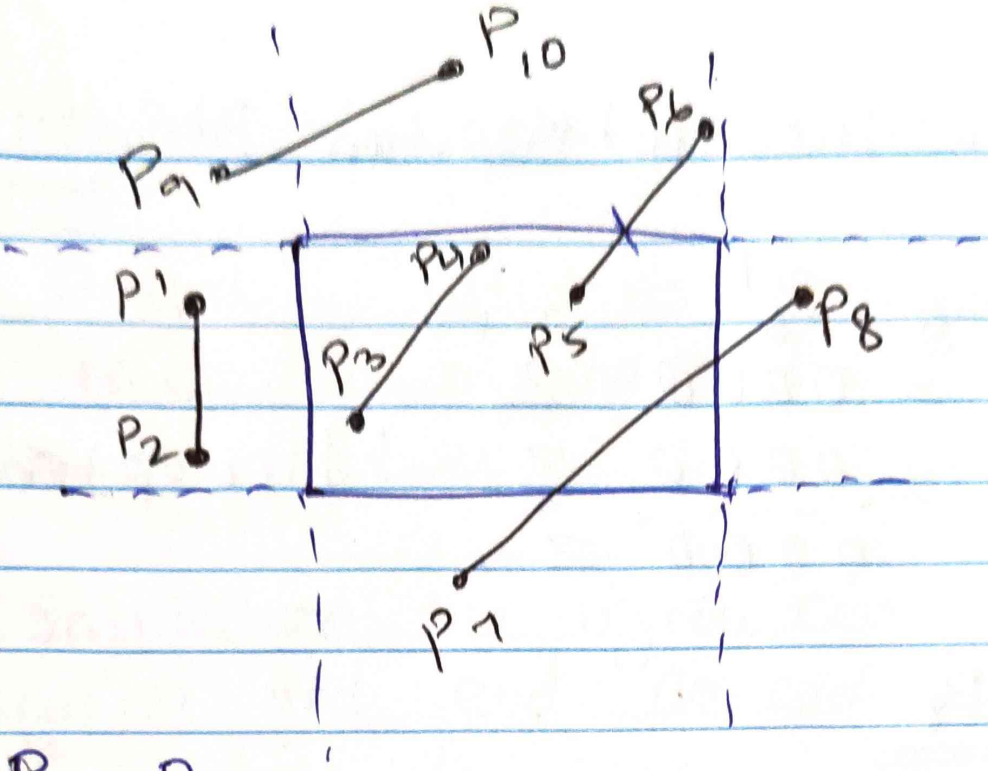
TBRL -> Top, Bottom, Right Left



1. If the line is inside the window then no clipping required completely accept the line.

2. If the line is completely outside the window then no clipping required. Completely reject the line.

3. Partially inside the window then clipping is required and find the intersection point



P_1, P_2

$P_1 = 0\ 0\ 0\ 0$ NZ

$P_2 = 0\ 0\ 0\ 1$ NZ Rejected

$0\ 0\ 0\ 1$ NZ

P_3, P_4

$P_3 = 0\ 0\ 0\ 0$ Z

$P_4 = 0\ 0\ 0\ 0$ Z Accepted.

$0\ 0\ 0\ 0$ Z

P_5, P_6

$P_5 = 0\ 0\ 0\ 0$

$P_6 = 1\ 0\ 0\ 0$ NZ Partially inside

$0\ 0\ 0\ 0$ Z

Find the intersection pt P_6'

P_5, P_6'

$P_5 = 0000$ Z

$P_6 = \underline{0000}$ Z Accepted.

0000 Z

P_7, P_8

$P_7 = 0100$ NZ Partially

$P_8 = \underline{0010}$ NZ inside.

0000 Z

Find the intersection pt P_7', P_8'

$P_7' = 0000$ Z

$P_8' = \underline{0000}$ Z Accepted.

0000 Z

P_9, P_{10}

$P_9 = 1001$ NZ

$P_{10} = \underline{1000}$ NZ Rejected.

1000 NZ

Algorithm for Cohen-Sutherland

1. Define a window or view plane get the co-ordinates from the user of a line.

2. Initialize the region code for initial and end co-ordinates of a line to 0000

3. Check whether the line lies within partially or outside the window.

   (i) Now assign the region code for both the initial & end co-ordinates.

   ii) After assigning, if both the end points give 0000 then the line is completely within the window.

   Perform the AND operation to find it is zero region or none zero region.

iii) Else, if the result is not 0000 or non-zero then the line is not inside the window and that line would not be considered for clipping.

iv) Else the line is partially inside the window.

4. After confirming the line is partially inside the window, then the next step to find the intersection point at the window boundary By using

1. If the line passes through the top

$$x = x + \frac{(y_{wmax} - y)}{m}$$

$$y = y_{wmax}$$

2. If the line passes through the bottom

$$x = x + \frac{(yW_{min} - y)}{m}$$

$$y = yW_{min}$$

3. If the line passes through the left region

$$y = y + (xW_{min} - x)\, m$$

$$x = xW_{min}$$

(4) If the line passes through the right region

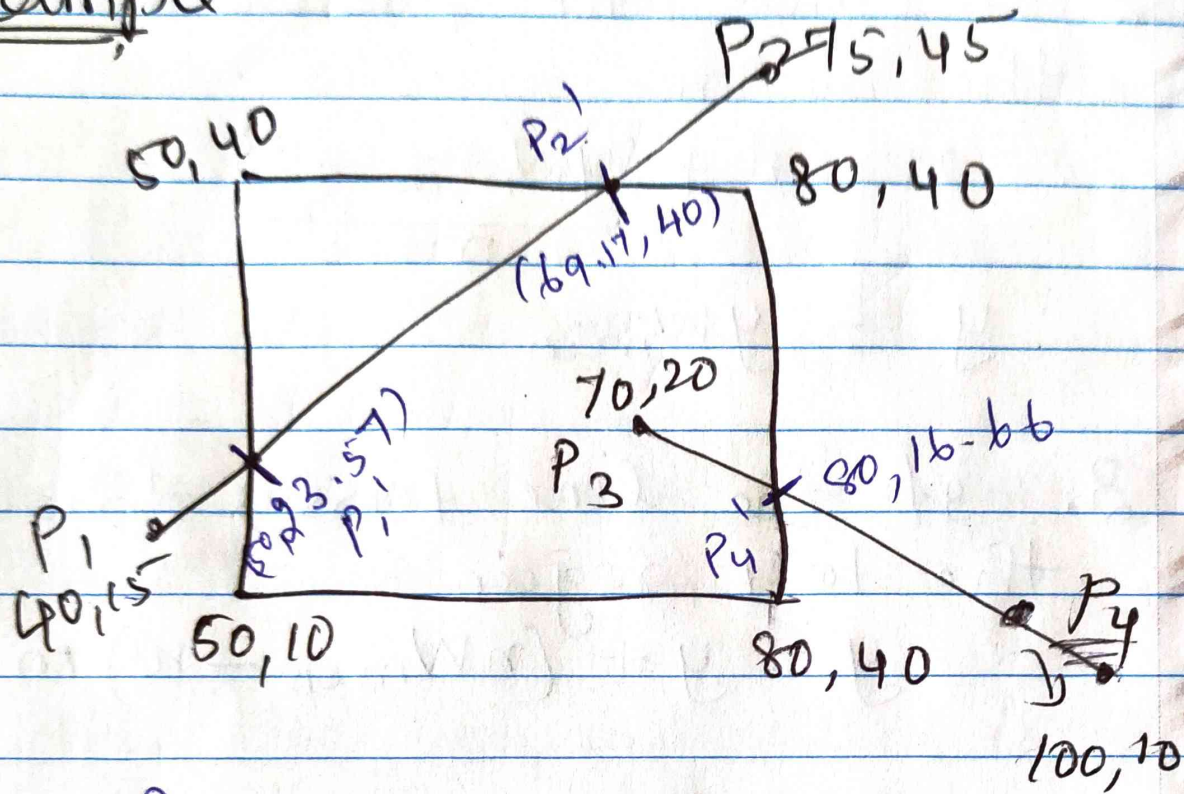
$$y = y + (xW_{max} - x)\, m$$

$$x = xW_{max}.$$

Now, overwrite the endpoint with a new one & update it.

Repeat 4th step till the line doesn't get clipped completely.

Example



P_1 , P_2

$P_1 = 0001$ NZ

$P_2 = 1000$ NZ

0000 Z

Partially inside clipping is required.

Line Passes through Left region.

$$y' = y + (x W_{min} - x)\, m$$

$$x' = x W_{min}$$

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

$x_1, y_1 \rightarrow 40, 15$

$x_2, y_2 \rightarrow 75, 45$

$$m = \frac{45 - 15}{75 - 40} = \frac{30}{35}$$

$$\boxed{m = 6/7}$$

Clipping for the point P_1

$$y = 15 + (50 - 40)\, 6/7$$

$$= 15 + 10 \times 6/7$$

$$\boxed{\begin{aligned} y &= 23.57 \\ x &= 50 \end{aligned}}$$

Clipping for the point P_2

Line Passes through the top

$$x = x + \frac{y W_{max} - y}{m}$$

$$y = y W_{max}$$

$$x = 75 + \frac{(40 - 45)}{6/7}$$

$$= 75 + (40 - 45) \times 7/6$$

$$= 75 - 5 \times 7/6$$

$$\boxed{\begin{aligned} x &= 69.17 \\ y &= 40 \end{aligned}}$$

P_3 , P_4

$P_3 = 0000$ Z

$P_4 = 0000$ NZ Partial

$\overline{0000}$ Z

clipping required for P_4

$P_3 = 70, 20$ $P_4 = 100, 10$

$$m = \frac{10 - 20}{100 - 70} = \frac{-10}{30}$$

$$\boxed{m = -1/3}$$

$P_4 \rightarrow$ line passes through the right region

$$y = y + (W x_{max} - x)\, m$$

$$x = W x_{max}.$$

$$y = 10 + (80 - 100)(-1/3)$$
$$= 10 - 20(-1/3)$$
$$= 10 + 20/3 = 16.66$$

$x = 80$
$y = 16.66$

Polygon Clipping Algorithm

Polygon

A polygon can be described as the enclosed collection or group of the lines.

In a Polygon all lines

are connected.

Lines can be a combination of edges and vertices, which together form a polygon.

Polygon clipping

It is a process in which we only consider the part which is inside the view plane or window

We will remove or clip the part that is outside the window

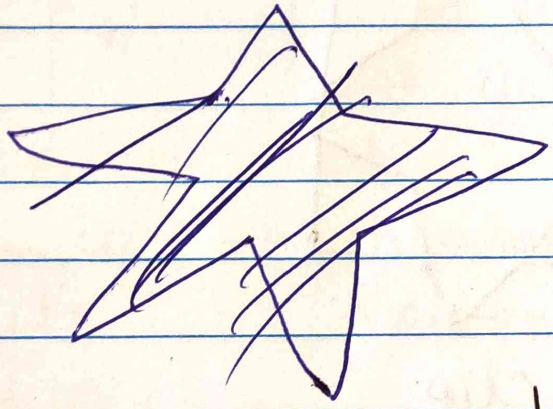
Sutherland - Hodgeman polygon clipping algorithm

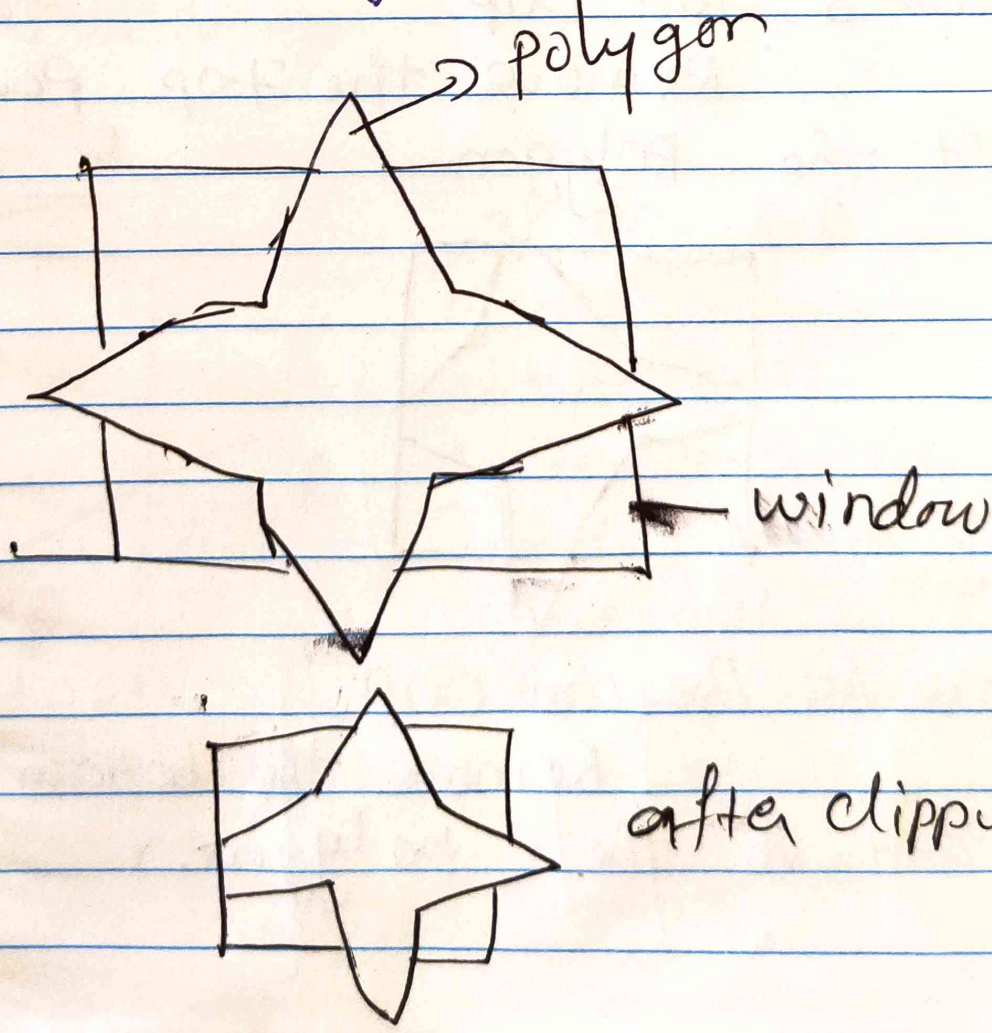
The polygon clipping algorithm deals with 4 different clipping cases.

The output of each case is input to the next case.

Case 1 : Left clip

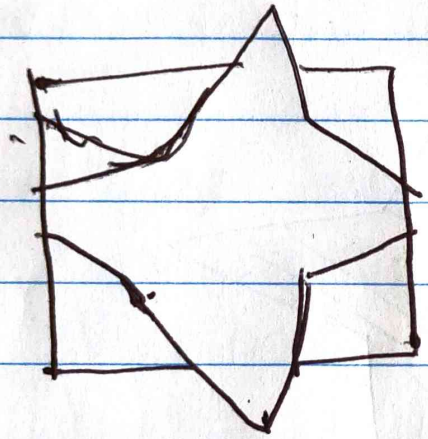
Removes the left part of the polygon, which is outside the window. Save only the portion which is inside the window





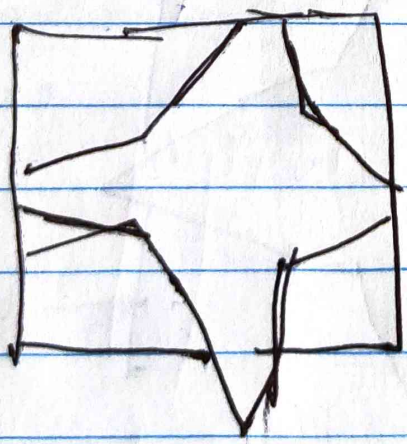
Case 2: Right clip

Remove the right part of the polygon, which is outside the window.



Case 3: Top clip

Remove the top part of the polygon.

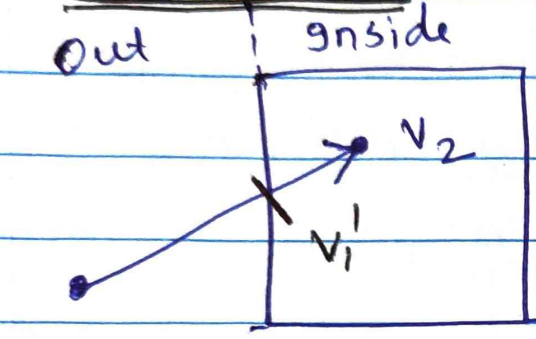


Case 4: Bottom clip

Remove the bottom part of the polygon.

There should be the following Conditions while we clip a polygon.

Condition 1



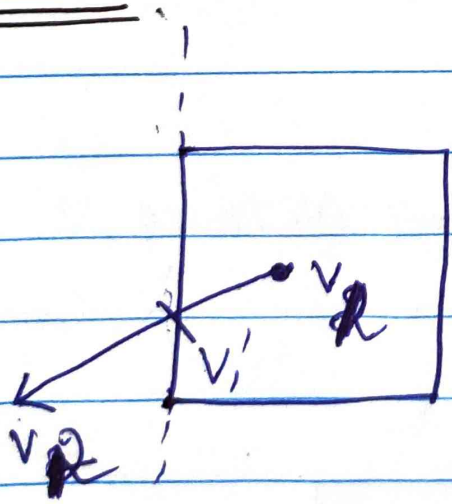
out — in

Output

v_1', v_2

If the 1st vertex of the polygon is outside and the second vertex is inside the window then the o/p will be intersection pt & 2nd vertex.

Case 2



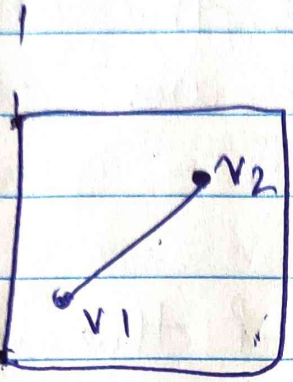
Output

in — out

v_1'

If the first vertex of the polygon is inside & the second vertex is Outside the window then the output will be the intersection point

Case 3



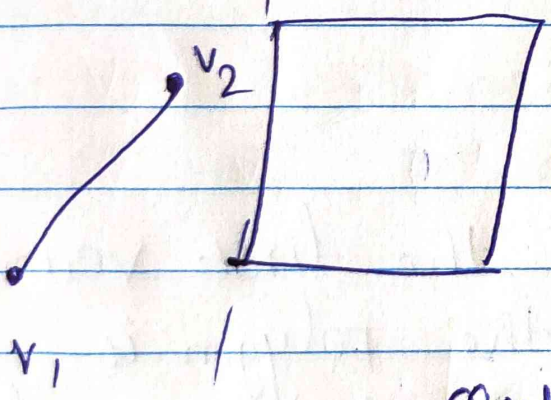
If both vertices of the polygon are inside the window. Then the o/p will be 2nd vertex.

Output

in - in

V_2

Case 4

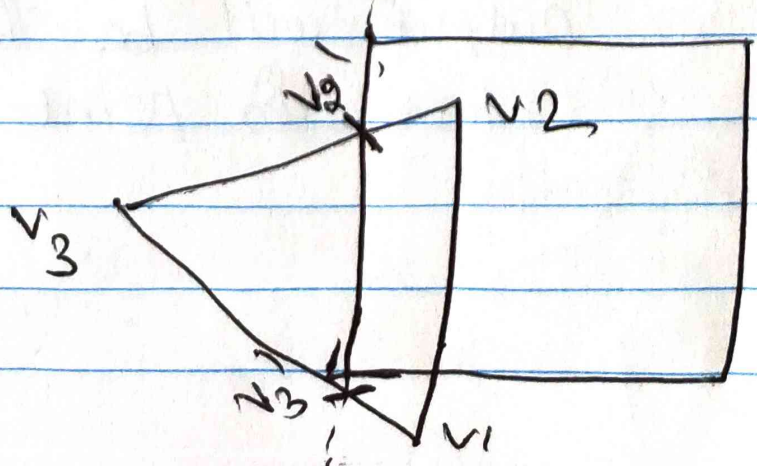


If both vertices of the polygon are outside the window. Then the output will be nothing

out - out

Output → Nothing

Example

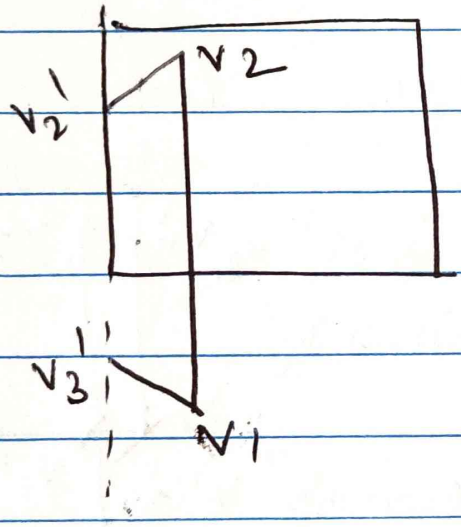


Left clip

$v_1, v_2 \rightarrow$ in – in $\rightarrow v_2$

$v_2, v_3 \rightarrow$ in – out $\rightarrow v_2'$

$v_3, v_1 \rightarrow$ out – in $\rightarrow v_3', v_1$



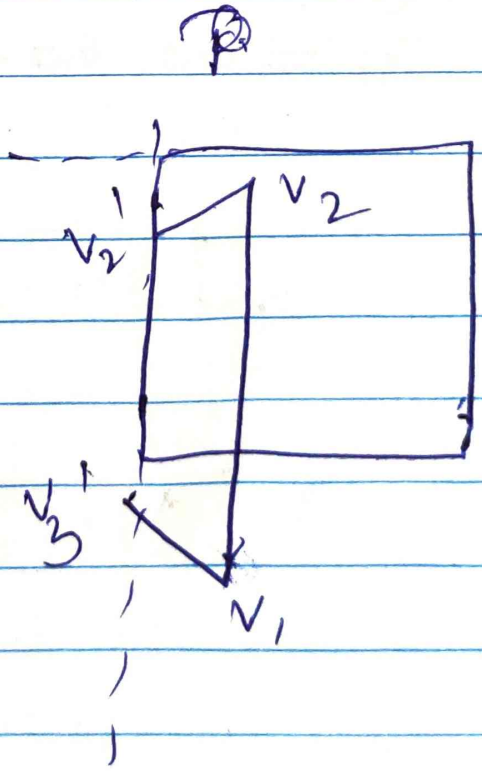
Right clip

$v_1, v_2 \rightarrow v_2$

$v_2, v_2' \rightarrow v_2'$

$v_2', v_3' \rightarrow v_3'$

$v_3', v_1 \rightarrow v_1$



Top clip

$v_1, v_2 \rightarrow v_2$

$v_2, v_2' \rightarrow v_2'$

$v_2', v_3' \rightarrow v_3'$

$v_3', v_1 \rightarrow v_1$

Bottom clip

$v_1, v_2 \rightarrow$ out - in $\rightarrow v_1', v_2$

$v_2, v_2' \rightarrow$ in - in $\rightarrow v_2'$

$v_2', v_3' \rightarrow$ in - out $\rightarrow v_2''$ v_3'

$v_3', v_1 \rightarrow$ out - out - null