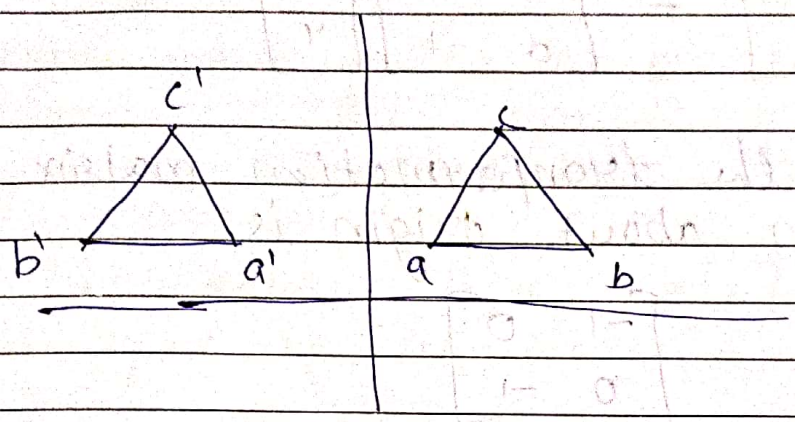
Shubhrata Kanungo

Chapter -2
(reflection wards)

Reflection about y-axis :-



$$x' = -x$$
$$y' = y$$

This transformation is identified by the reflection transformation matrix as

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

c) Reflection about the origin :

In this case we actually choose the mirror line as the axis perpendicular to the xy plane and passing through the origin. After reflection both the x and y coordinates of the object point is flipped that is x' becomes $-x$, and y' become $-y$.

$$x' = -x$$
$$y' = -y$$

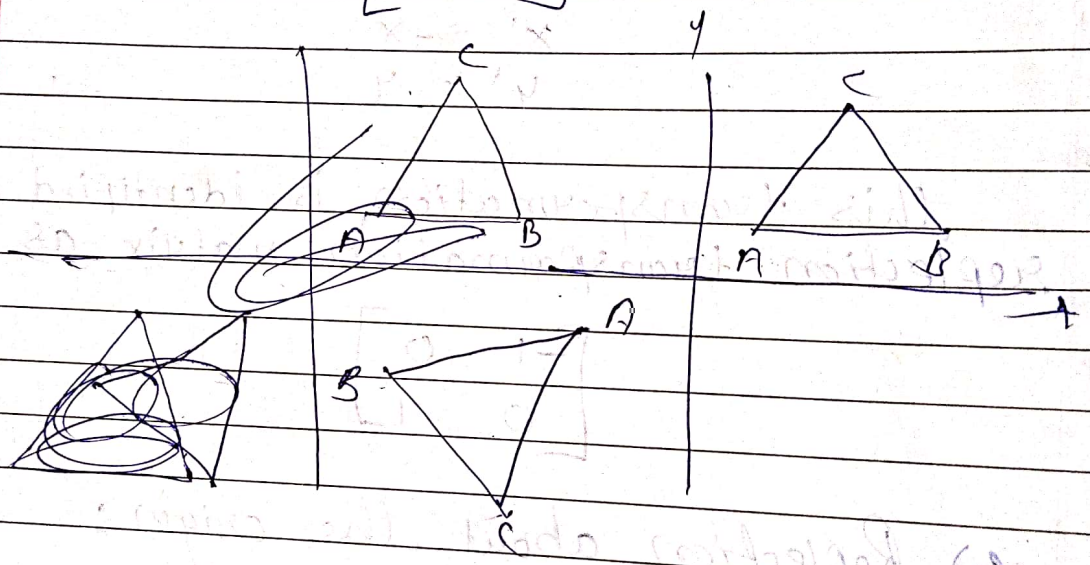
This can be represented in matrix

as -

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

Thus the transformation matrix for reflection about origin is

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



(d) Reflection about the straight line $y=x$ if we get (y, x) i.e.

$$x' = y$$
$$y' = x$$

This can be represented by

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

when $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ is transformation matrix for reflection about line $y = -x$

Scanned by CamScanner

followed by it's inverse transformation. ~~Function~~ give an object unchanged in position orientation, size and shape. we will use inverse transformation to ~~nullify the effect of~~ already applied transformation.

Inverse transformation :-

Each geometric transformation has an inverse which is describe by the opposite operation performed by the original transformation. any transformation

Translation $T^{-1}_{tx\,ty} = T_{-tx-ty}$ translation in opposite direction

$$x' = x + tx \qquad y' = y + ty$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix} = \{$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x' \\ y' \end{bmatrix} - \begin{bmatrix} tx \\ ty \end{bmatrix}$$

$$= [$$

Rotation →

$R^{-1}_{\theta} = R_{-\theta}$ rotation in opposite direction

$$P' = (R.P)^T \qquad R^{-1} = R^T$$

Clockwise :-

$$K = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

R^{-1} =

$$\begin{bmatrix} \cos(-\theta) = \cos\theta \\ \sin(-\theta) = -\sin\theta \\ \sin \end{bmatrix}$$

Scanned by CamScanner

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

Scalling :- $S_{Sx\,Sy}^{-1} = S_{\frac{1}{Sx}\frac{1}{Sy}}$

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

$x = \frac{x'}{S_x} \qquad y = \frac{y'}{S_y}$

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

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

~~Reflection : $M^{-1} = M \cdot M$~~

Composite transformation

Homogeneous Coordinate system :-
and matrix transformation

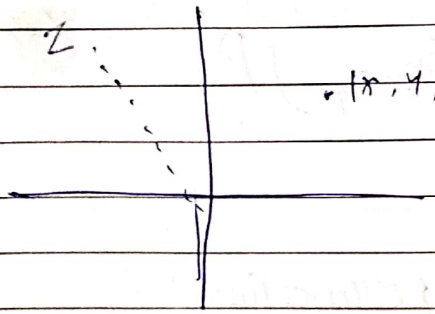
1. Each position (x, y) is represented as $(x, y, 1)$
2. All transformation can be represented as matrix multiplication
3. Composite transformation becomes easier
4. Unifying representation for transforming

Scanned by CamScanner

$$P' = PM_1 + M_2$$

↓

general representation 2D transformation



$$P(x, y) = \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

translation tran.

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

$$= \begin{bmatrix} 1 \times x + 0 \cdot y + t_x \times 1 \\ 0 \times x + 1 \times y + t_y \times 1 \\ 0 \times x + 0 \cdot y + 1 \times 1 \end{bmatrix}$$

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

$$= \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}$$

Rotation :-

$P' = R\theta(P)$

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

Scanned by CamScanner

Scaling :-

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

$$P' = S(S_x, S_y) \cdot P$$

Composite transformation :-

To control the size of an object, we need to perform matrix operations on the position vectors which defines the vertices. It is not neccesary that we get the required orientation by applying single transformation, it may require more than one transformation. since matrix multiplication is non commutative, the order of application of the transformation is important, we can set up a matrix for an sequance of transformation as a composite transformation.

① if two successive translation are applied

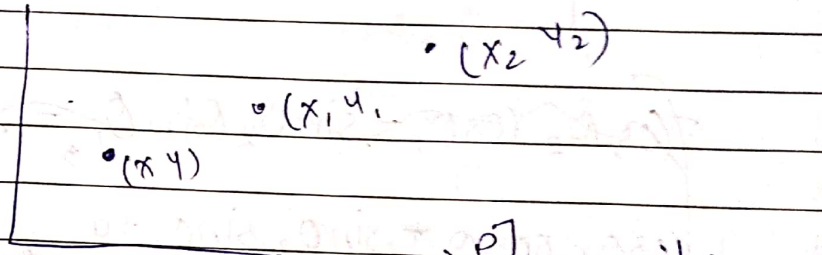
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Scanned by CamScanner

homogeneous coordinates

$$\begin{bmatrix} 1 & 0 & t_{x_1} \\ 0 & 1 & t_{y_1} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_{x_2} \\ 0 & 1 & t_{y_2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_{x_1} \\ 0 & 1 & t_{y_1} \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1\times1+0+t_{x_2}\times0 & 0 & t_{x_1}+0+t_{x_2} \\ \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} 1 & 0 & t_{x_1}+t_{x_2} \\ 0 & 1 & t_{y_1}+t_{y_2} \\ 0 & 0 & 1 \end{bmatrix}$$



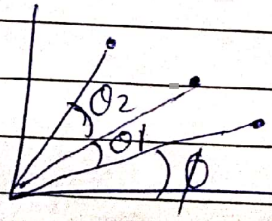
$P' = T_2(x_2 y_2)\,[T_1(x_1, y_1)\,P]$ composite translation

$P' = [T_2(x_2, y_2)\; T_1(x_1, y_1)]\,P$

commutative = A·B·C we can't do = B·A·C
X

associative = A·B = B·C

(2) If two successive rotation θ_1 and θ_2 then



$P(R_{\theta_2})\,[R(\theta_1)\cdot P]$

$P' = [R(\theta_2)\,R(\theta_1)]\,P$

$P' =$

Scanned by CamScanner

$$R(\theta) = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R(\theta) = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta_2\cos\theta_1 - \sin\theta_2\sin\theta_1 & -\sin\theta_1\cos\theta_2 - \cos\theta_1\sin\theta_2 & 0 \\ \cos\theta_2\cos\theta_1 + \sin\theta_2\sin\theta_1 + 0 & \cos\theta_2\cos\theta_1 - \cos\theta_2\sin\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & 0 \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scanned by CamScanner

$$\begin{bmatrix} \cos\theta_2\cos\theta_1 - \sin\theta_2\sin\theta_1 & -\sin\theta_1\cos\theta_2 - \sin\theta_2\cos\theta_1 & 0 \\ \sin\theta_2\cos\theta_1 + \cos\theta_2\sin\theta_1 & -\sin\theta_1\sin\theta_2 + \cos\theta_2\cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & 0 \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$P' = R(\theta_2) + R(\theta_1) P$$
$$= R(\theta_1+\theta_2) P$$

(3) If two successive scaling are applied then

$$P' = S_2(S_{x_2} \cdot S_{y_2}) \left[S_1(S_{x_1}, S_{y_1}) \cdot P \right]$$

$$= S_2(S_{x_2}, S_{y_2}) S_1(S_{x_1}, S_{y_1})$$

$$= \begin{bmatrix} S_{x_2} & 0 & 0 \\ 0 & S_{y_2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_{x_1} & 0 & 0 \\ 0 & S_{y_2} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} S_{x_1} S_{x_2} & 0 & 0 \\ 0 & S_{y_1} S_{y_2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\therefore P' = S_2(S_{x_2} S_{y_2}) \left(S_1(S_{x_1}, S_{y_1}) P \right)$$
$$= S(S_{x_1} S_{x_2}, S_{y_1} S_{y_2}) P$$

Scanned by CamScanner

Windows and clipping :-

~~Window to~~ Window to View port
~~Viewing transformation~~ transformation

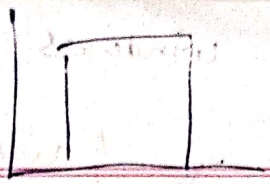
Introduction :- When drawing are too complex, they become difficult to read. In such situation, it is useful to display only those portion of drawing that are of immediate interest. This gives the effect of looking at the image through a window. Furthermore it is desirable to enlarge these portion to take full advantage is called windowing. The technique for not showing that part of the drawing which one is not interested in is called 'clipping'.

The viewing transformation :-

displaying an image of a picture involves mapping the co-ordinates of the points and lines that form the picture into the appropriate co-ordinate on the device or workstation where the image is to be displayed. This is done through the use of co-ordinate transformation known as "Viewing transformation".

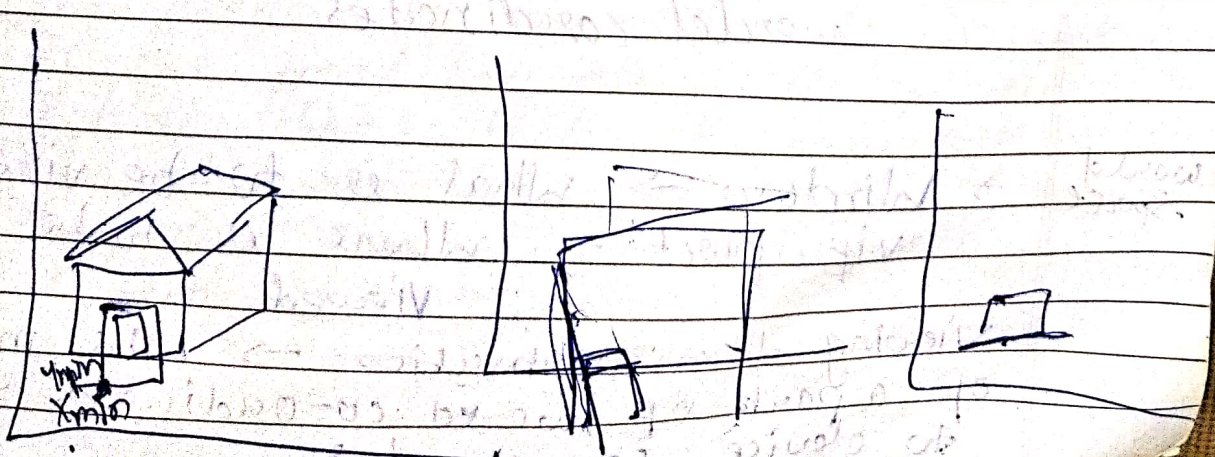
To perform a viewing transformation.

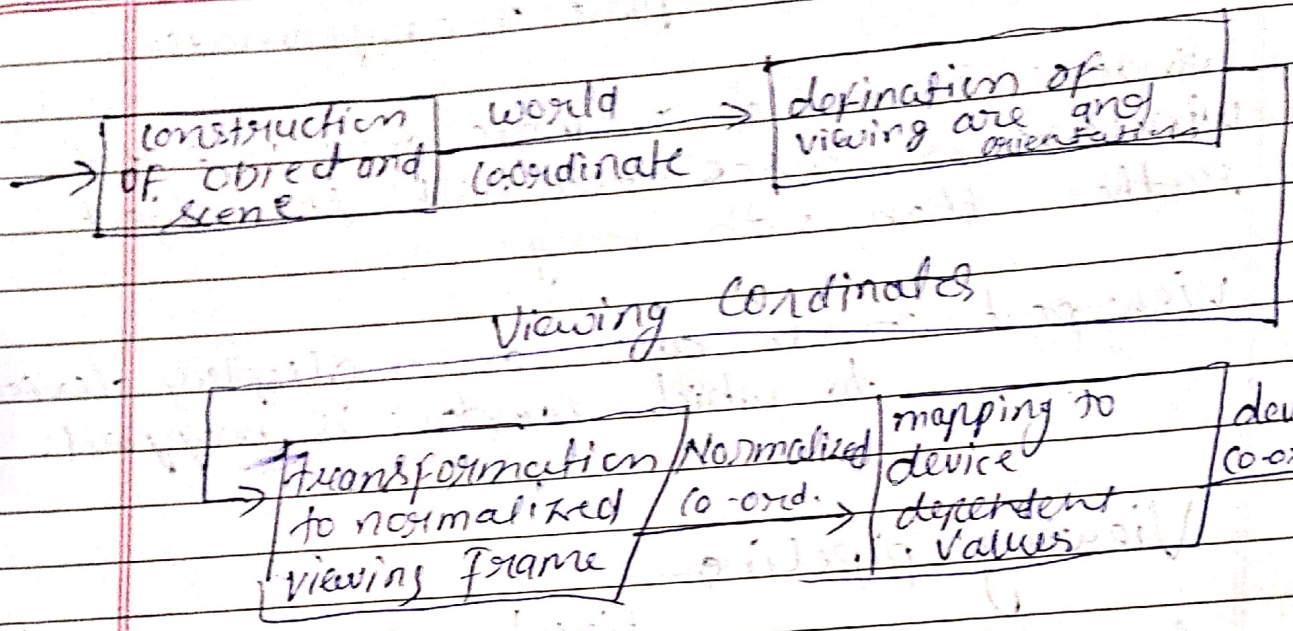
Scanned by CamScanner



we deal with a finite region in the WCS (world co-ordinate system) called a window. the window can be mapped directly on the display area of the device or on to a subregion of the display called a "viewport". Thus, a world-coordinate are selected for display is called a window and an area on a display device to which a window is mapped is called a viewport that is the window defines what is to be viewed the viewport defines where it is to be displayed.

If we are changing the position of window by keeping the viewport location constant then the different part of the object is displayed at the same position on the display device similarly if we change the location of viewport then we will see the same part of the object drawn at different places on screen





① Construct world-co-ordinate scene using modeling-coordinates transformation

② Convert world-coordinates to viewing cordinates

③ Transform viewing-co-ordinates to normalized co-ordinates

④ Normalized co-ordinates to device-coridinates.

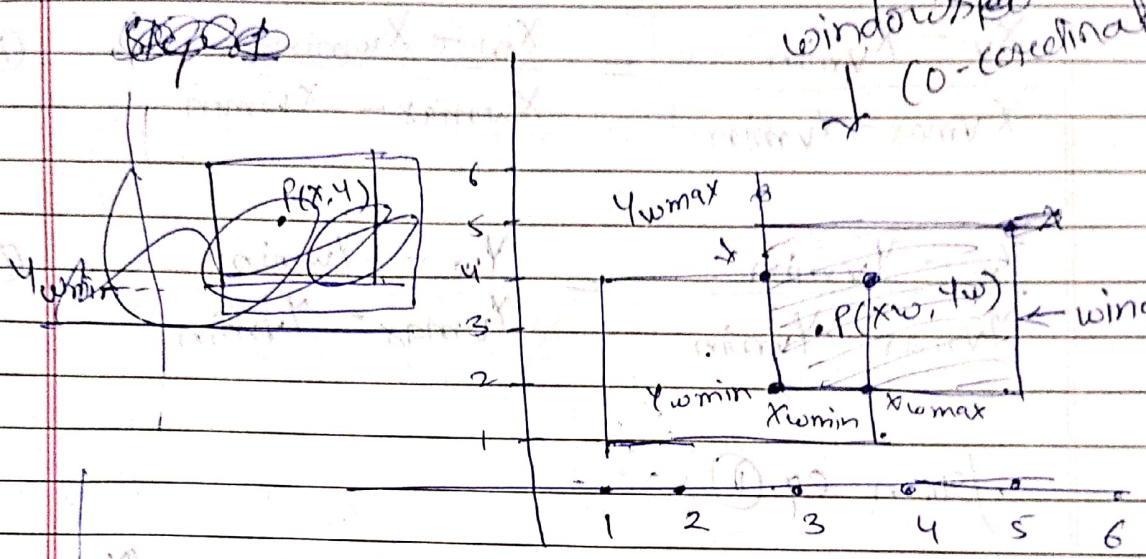
translate $\rightarrow$ $t_x = -x_{wmin}$

$t_y = -y_{min}$

$$t_x = \frac{x_{wmax}\, x_{vmin} - x_{wmin}\, x_{vmax}}{x_{wmax} - x_{wmin}}$$

Scanned by CamScanner

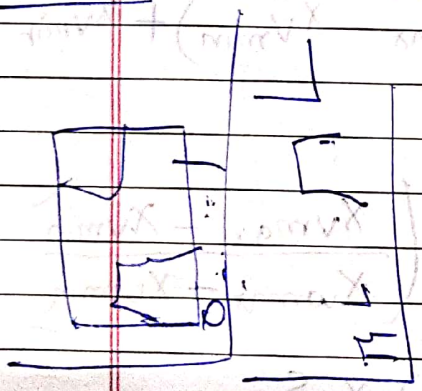
$$A_y = \frac{y_{wmax}\, y_{vmax} - y_{wmin}\, y_{vmax}}{y_{wmax} - y_{wmin}}$$



Scaling in x direction :-

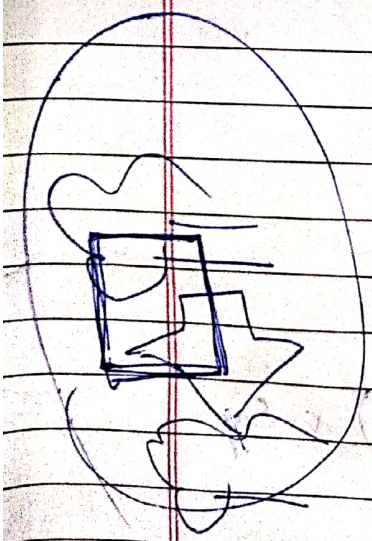
$$S_x = \frac{\text{width of viewport}}{\text{width of window}}$$

$$S_y = \frac{\text{height of viewport}}{\text{height of window}}$$



$$S_x = \frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}}$$

$$= \frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}}$$



$$S_y = \frac{y_v - y_{vmin}}{y_{max} - y_{vmin}}$$

$$= \frac{y_w - y_{wmin}}{y_{wmax} - y_{wmin}}$$

Scanned by CamScanner

$$\frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}} = \frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}} \quad \text{---(1)}$$

$$\frac{y_v - y_{vmin}}{y_{vmax} - y_{vmin}} = \frac{y_w - y_{wmin}}{y_{vmax} - y_{vmin}} \quad \text{---(1)}$$

from eq. (1)

$$x_v - x_{vmin} = \frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}} \times (x_{vmax} - x_{vmin})$$

$$x_v = \frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}} \times (x_{vmax} - x_{vmin}) + x_{vmin}$$

$$x_v = x_{vmin} + x_w - x_{wmin} \left(\frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}} \right)$$

$$= (x_{vmin} + (x_w - x_{wmin})) \times S_x$$

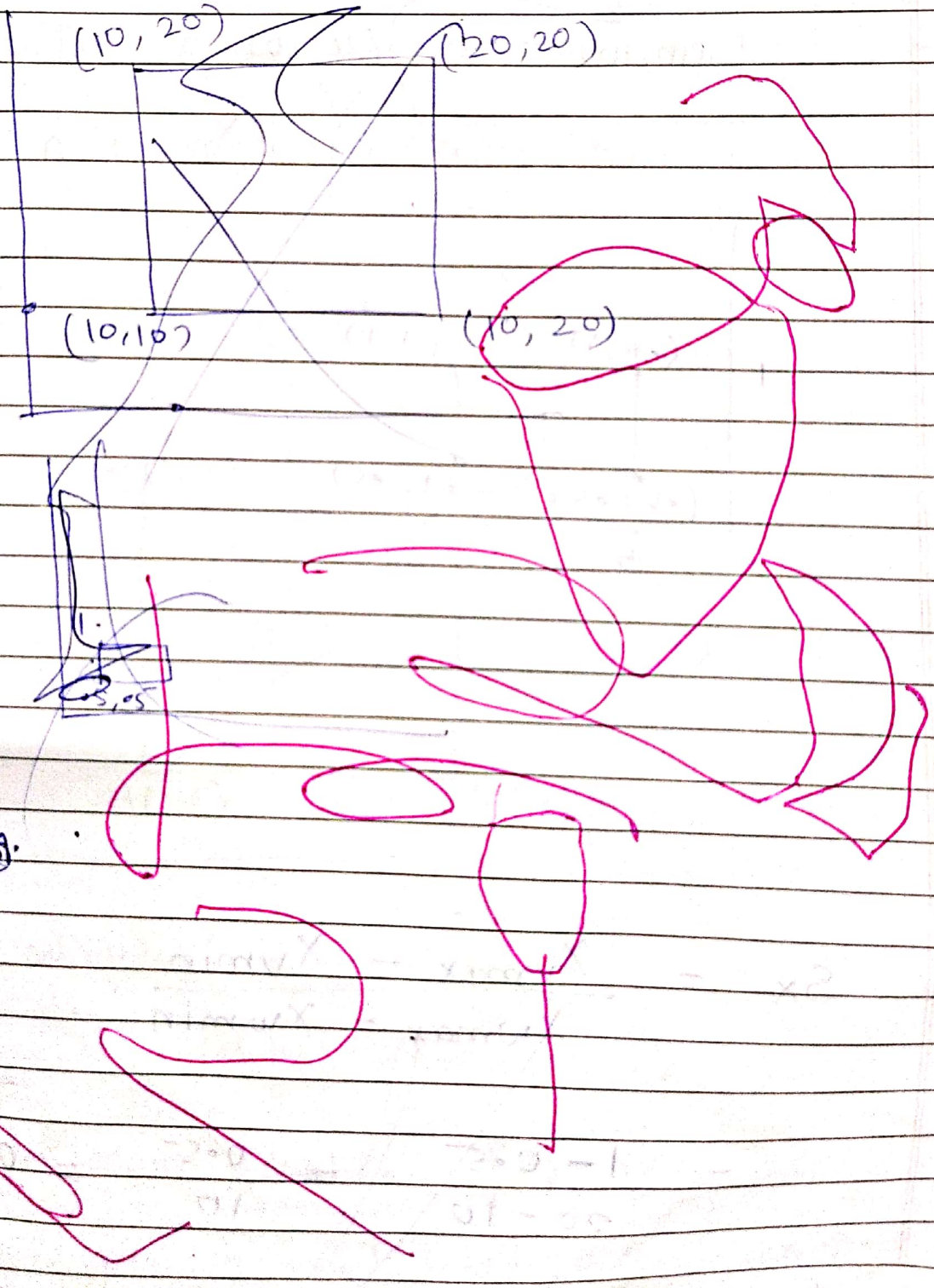
S_x = Scaling factor

$$S_x = \frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}}$$

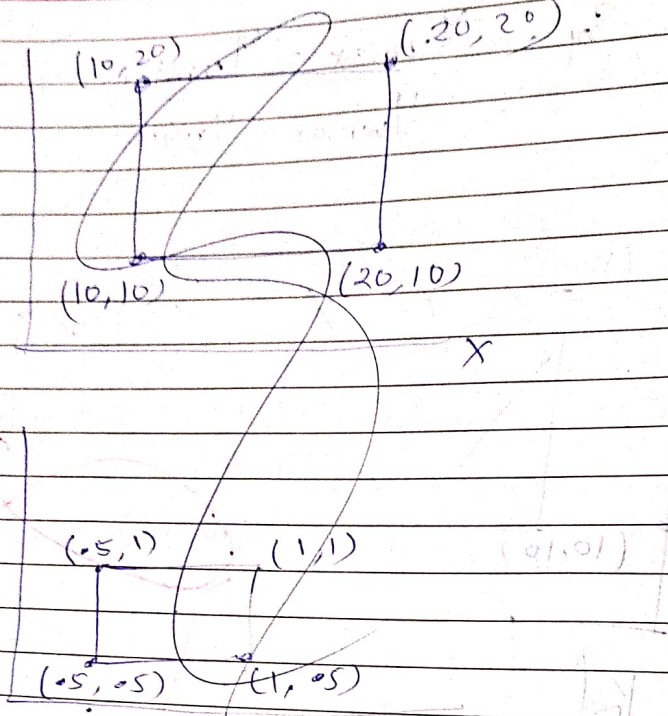
$$y_v = y_{vmin} + (y_w - y_{wmin}) \times S_y$$

Scanned by CamScanner

$$S_y = \frac{y_{vmax} - y_{vmin}}{y_{wmax} - y_{min}}$$



Scanned by CamScanner



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

$$= \frac{1 - 0.5}{20 - 10} = \frac{0.5}{10} = 0.05$$

$$S_y = \frac{1 - 0.5}{20 - 10} = \frac{0.5}{10} = 0.05$$

Scanned by CamScanner

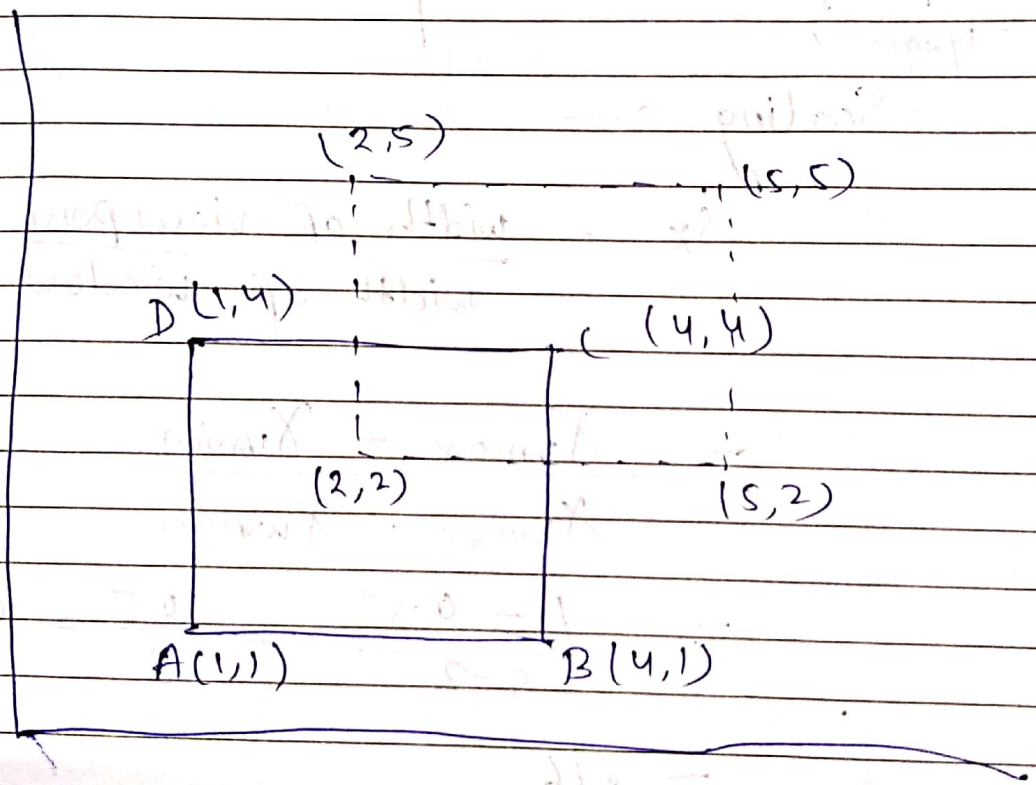
$X_v = X_{vmin} + (X_w - X_{wmin}) S_x$

$= 5 +$

Calculate viewing transformation matrix

ABCD Rectangle co-ordinate

A(1,1) B(4,1) C(4,4) D(1,4)



Window co-ordinate –

(2,2) (5,2) (5,5) (2,5)

Viewport location

(0.5, 0) (1, 0) (1, 0.5) (0.5, 0.5)

Scanned by CamScanner

Translation in negative direction ↓

$$t_x = -x_{wmin} = -2$$

$$t_y = -y_{wmin} = -2$$

$$\begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -t_{xwmin} & -t_{ywmin} & 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -2 & -2 & 1 \end{vmatrix}$$

Scaling :—

$$S_x = \frac{\text{width of viewport}}{\text{width of window}}$$

$$= \frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}}$$

$$= \frac{1 - 0.5}{5 - 2} = \frac{0.5}{3} = 0.16$$

$$= 0.16$$

$$S_y = \frac{\text{height of viewport}}{\text{height of window}}$$

$$= \frac{y_{vmax} - y_{vmin}}{y_{wmax} - y_{wmin}}$$

Scanned by CamScanner

$$= \frac{1 - 0.5}{5 - 2} = 0.16$$

Scaling :-

$$\begin{vmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

$$= \begin{vmatrix} 0.16 & 0 & 0 \\ 0 & 0.16 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Translate :-

$$\begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_{xmin} & t_{ymin} & 0 \end{vmatrix}$$

$$= \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ .5 & 0 & 1 \end{vmatrix}$$

Viewing transformation matrix —

$$\begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -2 & -2 & 1 \end{vmatrix} \times \begin{vmatrix} .16 & 0 & 0 \\ 0 & .16 & 0 \\ 0 & 0 & 1 \end{vmatrix} \times \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.5 & 0 & 1 \end{vmatrix}$$

$$= \begin{vmatrix} 0.16 & 0 & 0 \\ 0 & 0.16 & 1 \\ 0.18 & -0.32 & 1 \end{vmatrix}$$

Clipping :-

any procedure that identifies those portion of a picture that are either inside or outside of a specified region of space is reffered to as clipping algorithm or simply clipping the region against which an object is to clipped is called a clip. window.

Application of clipping include extracting part of defined scene for viewing identifying visible surface in three dimentional views. antialising a multiwindow environment and drawing and painting operation that allow parts of a picture to be selected for copying moving, erasing or duplicating

Scanned by CamScanner

Depending on application, the clip window can be applied in world a general polygon or it can even have curved boundaries. clipping algorithm can be applied in world coordinates, so that only the contents of the window interior are mapped to device co-ordinate. alternately the complete world co-ordinate picture can be mapped first to device co-ordinates or normalised device coordinates, then clipped against the viewport boundaries. world-coordinate clipping removes those primitives outside the window from further consideration thus eliminating the processing necessary to transform those primitives to device space. Viewport clipping on the other hand, can reduce calculation by allowing concatenation of viewing and geometric transformation matrices. But viewport clipping does require that the transformation to device co-ordinate is performed for all objects including those outside the window are.

Clipping may be done

types of clipping :-

(1) Point clipping (2) Line clipping

Scanned by CamScanner

③ Polygon clipping (area)
④ Curve clipping
⑤ Text clipping

1. point clipping :-

assuming that the clip window is a rectangle in standard position, we save a point $P(x,y)$ for display if the following inequalities are satisfied

$$x_{wmin} \leq x \leq x_{wmax}$$

$$y_{wmin} \leq y \leq y_{wmax}$$

if any one of these four inequalities is not satisfied, the point is clipped and not saved for display.

2. Line Clipping :-

Lines that do not intersect the clipping window are either completely inside the window or completely outside the window.

Lines may be

① visible :- Bothe end points of the line segment lie withing th window

② Non-visible :- when line definetly lies outside the window

$x_1, x_2 > x_{wmax}$ $\quad y_1, y_2 > x_{max}$

$x_2 x_2 < x_{min}$ $\quad y_1, y_2 < x_{wmin}$

③ Partially visible

The basic principle :- (assuming the clip window W)

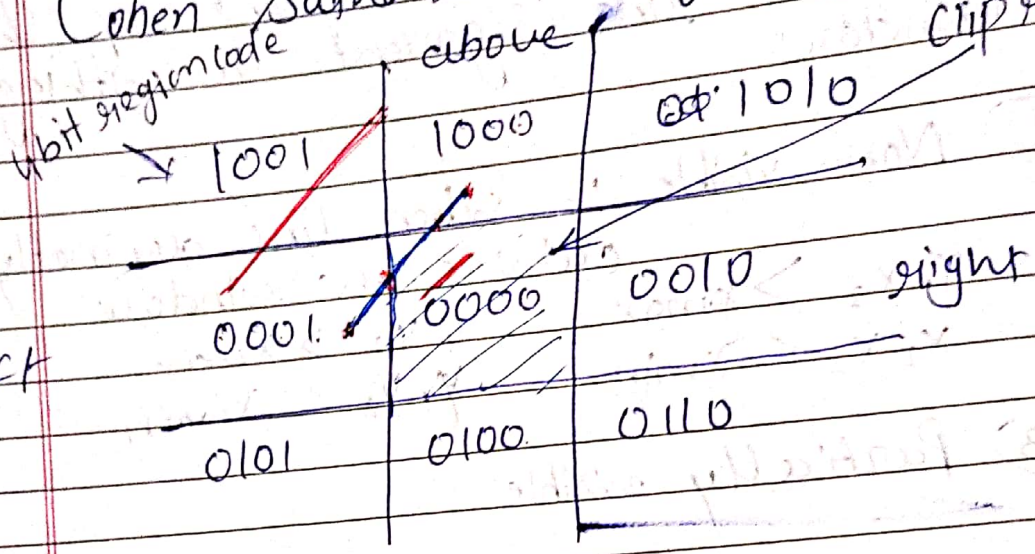
① If both endpoints of a line segment fall within W, then display the line segment.

② If both endpoints of line segment fall outside of W. ~~because they~~, then do not display the line segment.

③ If one endpoint falls withing W and another falls outside then part of the line segment is displayed.

Scanned by CamScanner

Cohen Sutherland algorithm :-



Below

Ri: L

top bottom ~~left right~~

Yes 1
No 0

L 1 0
R 1 0
T 1 0
B 1 0

algo :-

Step 1 :- assign a region code for each endpoints

Step 2 :-

If both end points have a region code 0000 then accept it (line)

Step 3 :- else perform AND operation for both region code

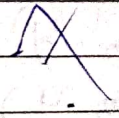
Scanned by CamScanner

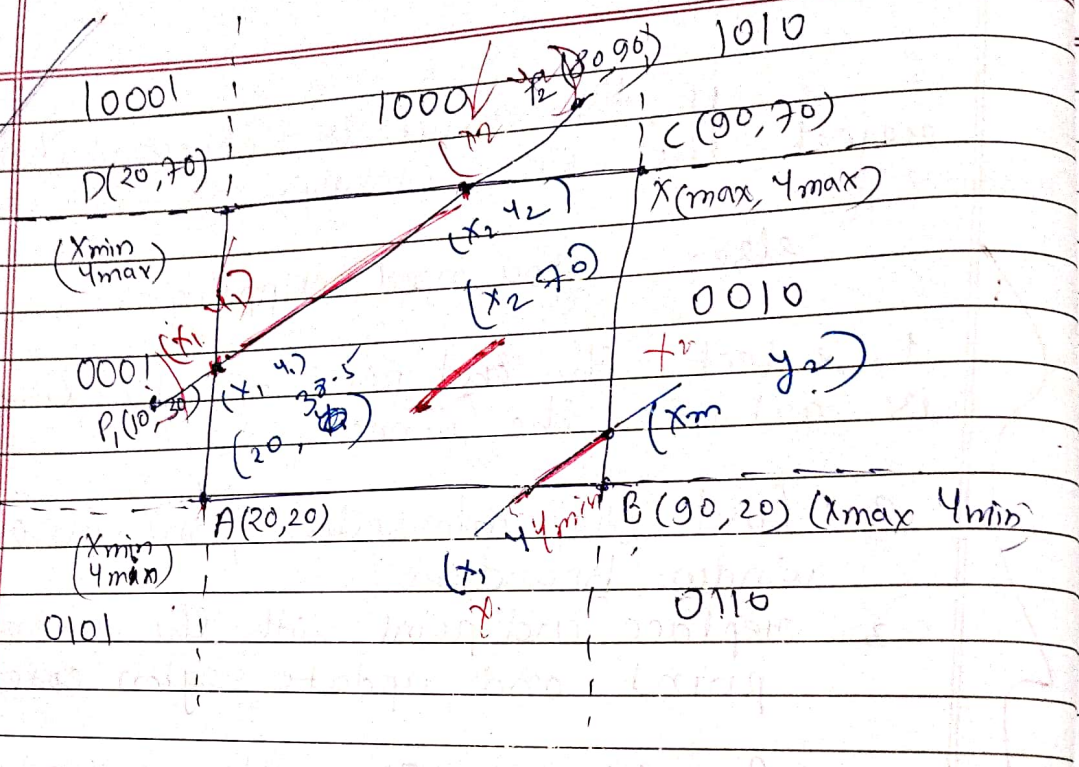
if, the result is not 0000 then reject the line (totaly)

else, you need clipping.

1. select the end point of the line that is out side the window
2. find the intersection point at the window boundries
3. replace end point with the intersection point and update region code.
4. Repeat step (2) until we find the clipped line either trivially accepted or trivially rejected.

Que) Let ABCD be the rectangular window A(20,20) B(90,20) C(90,70) and C(20,70). find the regin codes for endpoints and use cohen sutherland algorithm to clip the lines P_1, P_2 with P_1(10,30) P_2(80,90).





TBRL

P_1 (bit code) = 0001

P_2 (") = 1000

AND operation = P_1 0001

1000

0000

Since it is 0000 P_1P_2 is partially inside clipping window.

intersection formula:-

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

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

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

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

Scanned by CamScanner

$2^{10} + 2^6$

$(2^4 \times 64)$ / 2^6

256

classmate Date Page

17 million

$$m = \frac{90-30}{80-10} = \frac{60}{70} = 0.85$$

2^2

2^4

64_4

256

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

$= 0.85(20 - 10) + 30$

$y = 38.5$

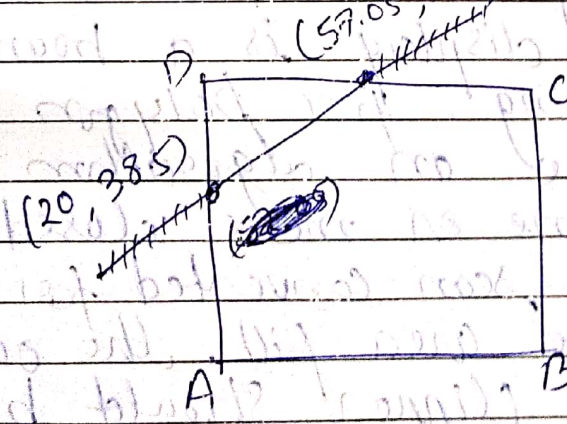
$(x_1, y_1) = (20, 38.5)$ ← x_{min}

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

$= \frac{1}{0.85}(70 - 30) + 10)$

$x = 57.05$

$(x_2, y_2) = (57.05, 70)$ ← y_{max}



Scanned by CamScanner

~~Mid-point Subdivision algorithm.~~

Liang - Barsky line clipping

use of parametric equation of line segment.

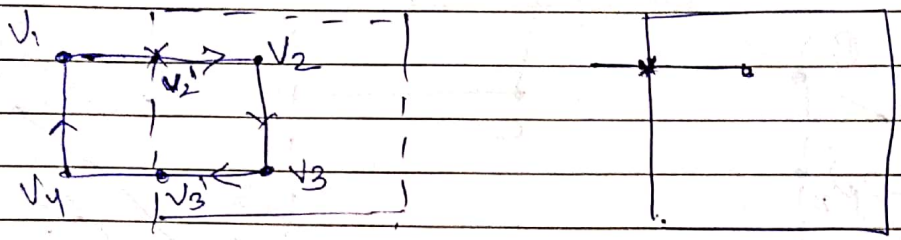
line (x_1, y_1) to (x_2, y_2)

Polygon clipping :-

To clip polygons, we need to modify the line-clipping procedures, discussed in the previous section. A boundry processed with a line clipper may be displayed as a series of unconnected line segments, dipendinding on the orientation of the polygon to the clipping window. what we really want to display is a bounded area after clipping. For polygon clipping we require an algorithm that will generate one or more closed areas that are then scan converted for the appropriate area fill. the output of a polygon clipper should be a sequence of vertices that defines the clipped polygon boundaries.

Scanned by CamScanner

Sutherland hodgman polygon clipping algorithm :-

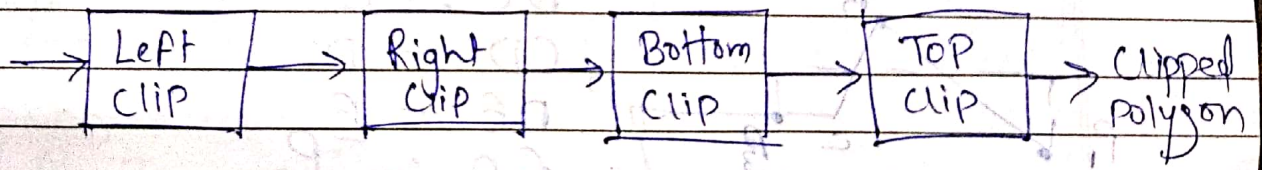


case (1) Save → V_2', V_2 (output vertex)

case (2) Save → Save V_3

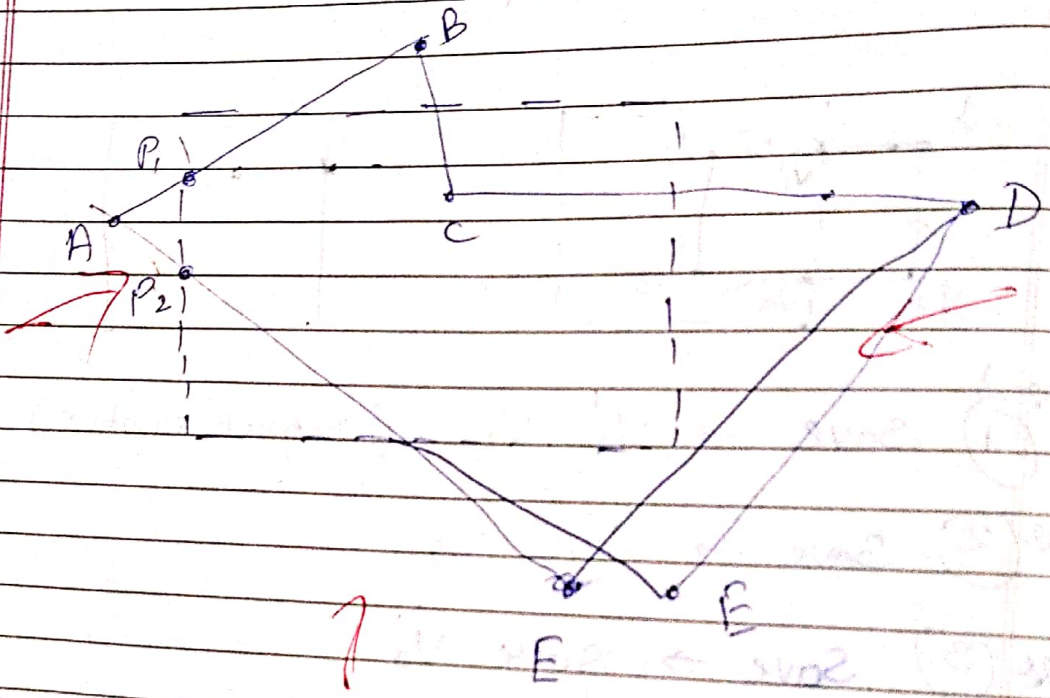
case (3) Save → Save V_3'

case (4) ⊗ don't save any vertex



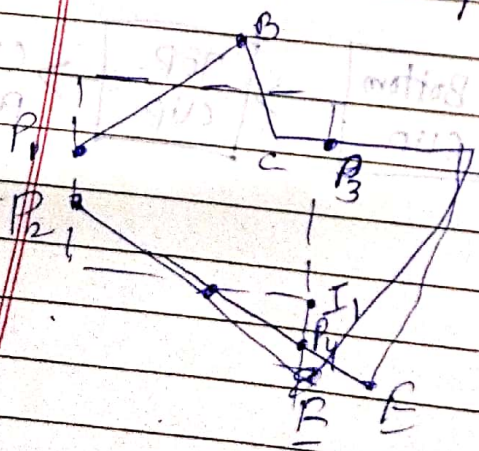
Scanned by CamScanner

Example

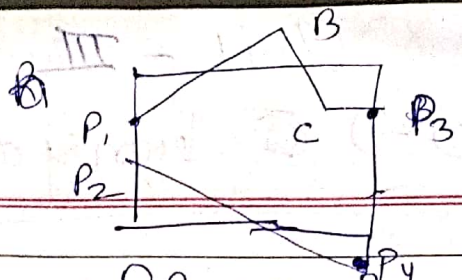


Step 1: Clip left :-

Left clip AB :- P_1B
BC :- C
CD :- D
DE - E
EA :- P_2



Scanned by CamScanner



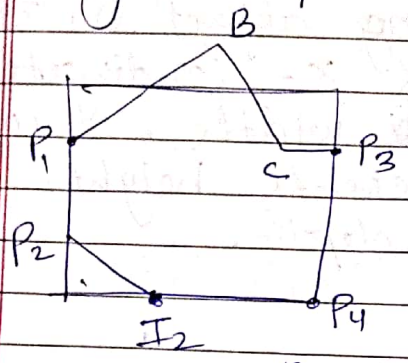
Right Clip :— $P_1B \rightarrow B$

$BC \rightarrow P_3$

$DE \rightarrow$ No vertex

$EP_2 \rightarrow P_4P_2$

$P_2P_1 \rightarrow P_1$



Bottom clip

$P_2P_1 \rightarrow P_1$

$P_1B \rightarrow B$

$BC \rightarrow C$

$CP_3 \rightarrow P_3$

$P_4P_2 \rightarrow I_2P_2$

Top Clip :→

$P_1B \rightarrow I_3$

$BC \rightarrow I_4C$

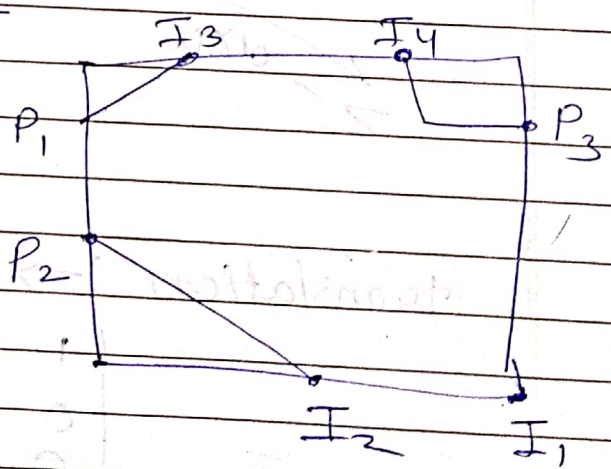
$CP_3 \rightarrow P_3$

$P_3I_1 \rightarrow I_1$

$I_1I_2 \rightarrow I_2$

$I_2P_2 \rightarrow P_2$

$P_2P_1 \rightarrow P_1$



Scanned by CamScanner

2nd Unit

Date: Page no:

2-D transformation :- (translation, Rotation

Flood fill

① Area filling is started from a point

boundary fill

Byte code
secure
Pointer से आसान

Date
1)
2.
3.

ROUGH NOTE BOOK

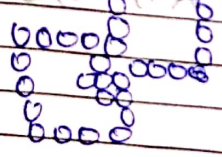
Scanned by CamScanner

and 8-connected

4-Connected :- there are techniques in which pixels are considered as connected to each other in 4-connected method the pixels may have upto 4 neighbouring pixels, as right, above, left and below of the current pixel

Similarly, in 8-connected method the pixels may have upto 8 neighbouring pixels.

Boundry Fill algorithm :- This is recursive algorithm that begins with a starting pixel called a seed, inside a region and continuous painting toward the boundry pixel. the algorithm checks to see if this boundry pixel has already filled, if answer is no, it paints the pixel and makes



Scanned by CamScanner

Date: Page no:

2-D transformation translation, Rotation.

Geometric transformation:-

Change in orientation, size and shape are accomplished with geometric transformation that alter the co-ordinates description of objects.

the basic geometric transformation are translation, rotation and scaling

Translation:-

translation consists of a shift of the object parallel to itself in any direction in the (x, y) plane, any such shift can be accomplished by a shift in x-direction plus a shift in y direction.

if the amount of x-shift

Scanned by CamScanner

Date:_____ Page no:

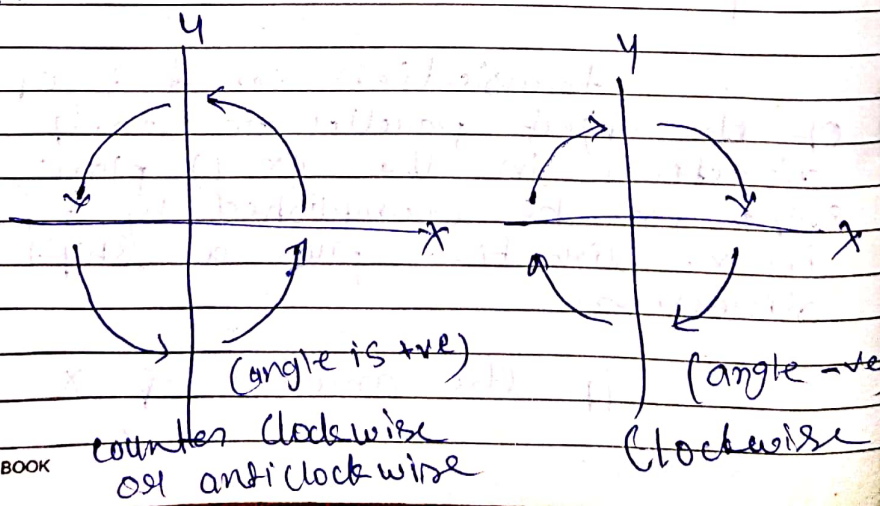
is called t_x and the amount of y-shift t_y. the translation of the point (x, y) into the point (x', y') is expressed by the formula,

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

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

To translate and object with the points, we just translate each point individually.

Rotation



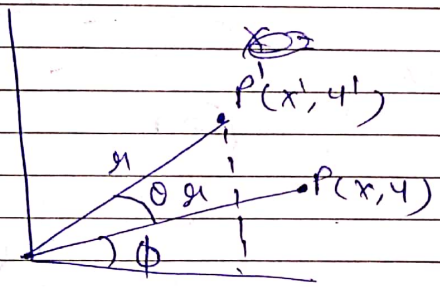
Scanned by CamScanner

$\cos\phi = \frac{3\pi}{6}$.

Date:_____ Page no:

A two dimentional rotation is applied to an object by repositioning it along a circular path in xy plane. points can be rotated through an angle θ about the origin.

The sign of angle determines the direction of rotation. Positive value for the rotation angle defines counter clockwise rotation and negative values rotate objects in the clockwise direction.



$x = r\cos\phi$

$y = r\sin\phi$

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

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

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

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

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

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

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

MY ROUGH NOTE BOOK

Scanned by CamScanner

Date:_____ Page no:

Qu 1 objectives 1 mark
10 question (10) —— 10

Qu 2 (i) —— 2
(ii) —— 3
(iii) —— 5
] 10
OR
(ii) —— 5

Qu 3 (i) —— 2]
(ii) —— 8] 10
OR
(ii) —— 8

Qu 4 (i) —— 3]
(ii) —— 7] 10
OR
(ii) —— 7

A scaling transformation alters the size of an objects this operation can be carried out of polygons by multiplying the coordinates value (x, y) of each vertex by scaling factor S_x and S_y to produce the transformed coordinates (x', y'):

Scanned by CamScanner

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

Symbolically $[x'] = [T][x]$

In the matrix form

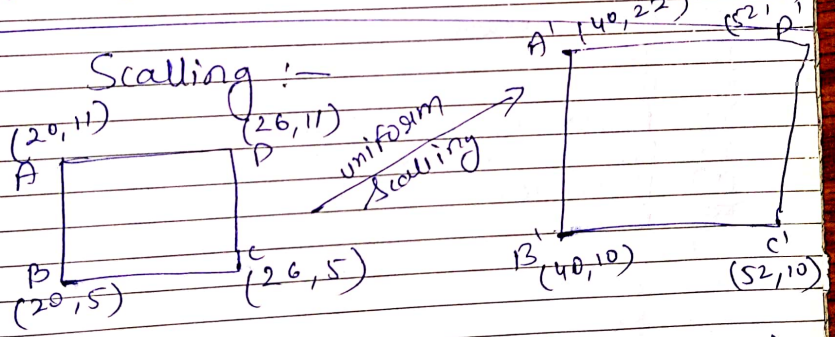
$P' = (R \cdot P)^T$ ↓ clockwise

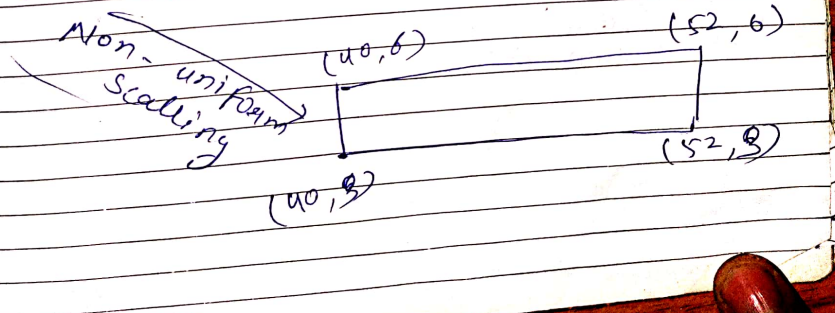
$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$ anti clockwise

$$x' = x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta$$

$$y' = y_r + (x - x_r)\sin\theta - (y - y_r)\cos\theta$$

Scalling :-





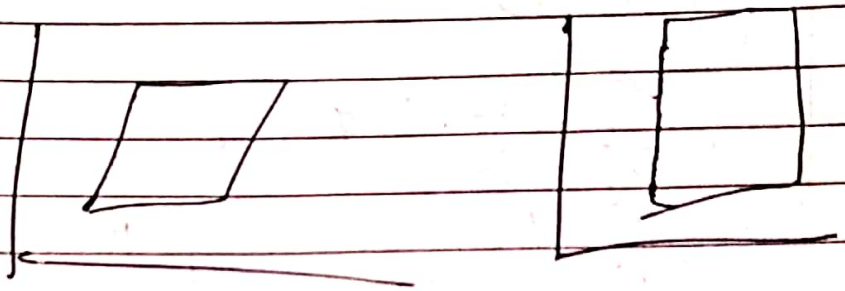
MY ROUGH NOTE BOOK

Scanned by CamScanner

Shearing

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

$$y' = y$$



$$x' = x$$

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

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

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

Scanned by CamScanner