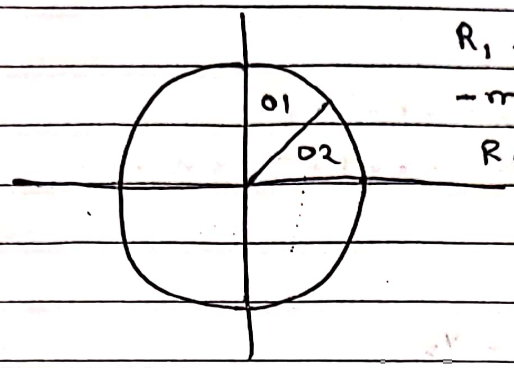
Amey Thakur B50
B 50 CG

Terna Engineering College, Navi Mumbai

Q1. Derive mid point ellipse drawing algorithm.



$R_1 - m < 1$

$- m = 1$

$R_2 - m > 1$

$m < 1$	$m > 1$
x - unit int	y - unit int.
y - ?	x - ?

$(x_{k+1}, y_k) / (x_{k+1}, y_{k-1})$ | $(x_k, y_{k-1}) / (x_{k+1}, y_{k-1})$

Equation of ellipse is $\frac{x^2}{rx^2} + \frac{y^2}{ry^2} = 1$

$\therefore x^2 ry^2 + y^2 rx^2 - rx^2 ry^2 = 0$

$m = \frac{dy}{dx}$

$2ry^2 \geq 2rx^2 y$

Region 01

$m < 1 \quad x = x_{k+1} \quad y = y_k / y_{k-1}$

$(x_k + 1, y_k) \quad (x_{k+1}, y_{k-1})$

Apply midpoint in eq^n.

$P_{1k} = (x_k + 1)^2 ry^2 + (y_k - 1/2)^2 rx^2 - rx^2 \cdot ry^2$

$P_{1\,k+1} = (x_k + 1)^2 ry^2 + (y_{k+1} - 1/2)^2 rx^2 - rx^2 ry^2$

$= \underline{((x_k + 1) + 1)^2} ry^2 + (y_{k+1} - 1/2)^2 rx^2 - rx^2 y^2$

2

$P_{k+1} - P_{1k} = (x_{k+1})^2 ry^2 + r^2 y + r(x_k + 1) ry^2 - y_{k+1}^2 rx^2 +$

$\frac{1}{4} rx^2 - y_{k+1} rx^2 - rx^2 ry^2 - (x_k + 1)^2 ry^2$

$- y_k^2 rx^2 - 1/4 \, rx^2 + y_k rx^2 + rx^2 ry^2$

$P_{1k+1} = P_{1k} + ry^2 + 2(x_{k+1}) ry^2 + rx^2 (-y_{k+1}^2 - y_k^2)$
$- rx^2 (y_{k+1} - y_k)$

Substitute $(0, r)$ in P_{1k}

$P_{1k} = 1 \cdot ry^2 + (ry - 1/2)^2 rx^2 - rx^2 ry^2$
$= ry^2 + ry^2 rx^2 + rx^2/4 - ry^2 rx^2 - rx^2 ry^2$

$P_{1k} = ry^2 + rx^2/4 - ry\, rx^2$

$P_k \le 0$

$x = x_k + 1$, $y = y_k$

else

$x = x_k + 1$ $y = y_k - 1$

Region 2:

$m > 1$, y - unit int, $x = ?$

$(x_k, y_k - 1)$ (x_{k+1}, y_{k-1})

Apply midpoint in eqn.

$P_{2k} = (x_k + 1/2)^2 ry^2 + (y_k - 1)^2 rx^2 - rx^2 ry^2$

$P_{2k+1} - P_{2k} = (x_{k+1} + 1/2)^2 ry^2 + ((y_k - 1) - 1)^2 rx^2 - rx^2 ry^2$
$- (x_k + 1/2)^2 ry^2 - (y_k - 1)^2 rx^2 + rx^2 ry^2$

$= x_{k+1} \cdot ry^2 + ry^2/4 + x_{k+1} \cdot ry^2 + (y_k - 1)^2 rx^2 + rx^2$
$- 2(y_k - 1) rx^2$

$- rx^2 ry^2 - x_k^2 ry^2 - ry^2 - ry^2/4 - x_k ry^2 - (y_k - 1)^2 rx^2 + rx^2 ry^2$

$P_{2k+1} = P_{2k} + rx^2 - 2 rx^2 (y_k - 1) + ry^2 (x_{k+1}^2 - x_k^2)$
$+ ry^2 (x_{k+1} - x_k)$

If $P_{2k} \le 0$ $x = x_k$, $y = y_k - 1$

else $x = x_k + 1$ $y = y_k - 1$

Terna Engineering College, Navi Mumbai

Q2. Difference between random scan and raster scan displays

Random Scan	Raster Scan
① It draws a picture one line at a time	① It draws a picture frame by frame
② Refresh rate depends on number of lines to be display	② Refresh rate is 60 fps.
③ It is used in drawing application	③ It generates stair step appearance in straight line
④ It does not display realistic shaded picture	④ It is used to display realistic shaded picture
⑤ Resolution is high	⑤ Resolution is low.
⑥ Scan conversion speed is high	⑥ Scan conversion speed is low.

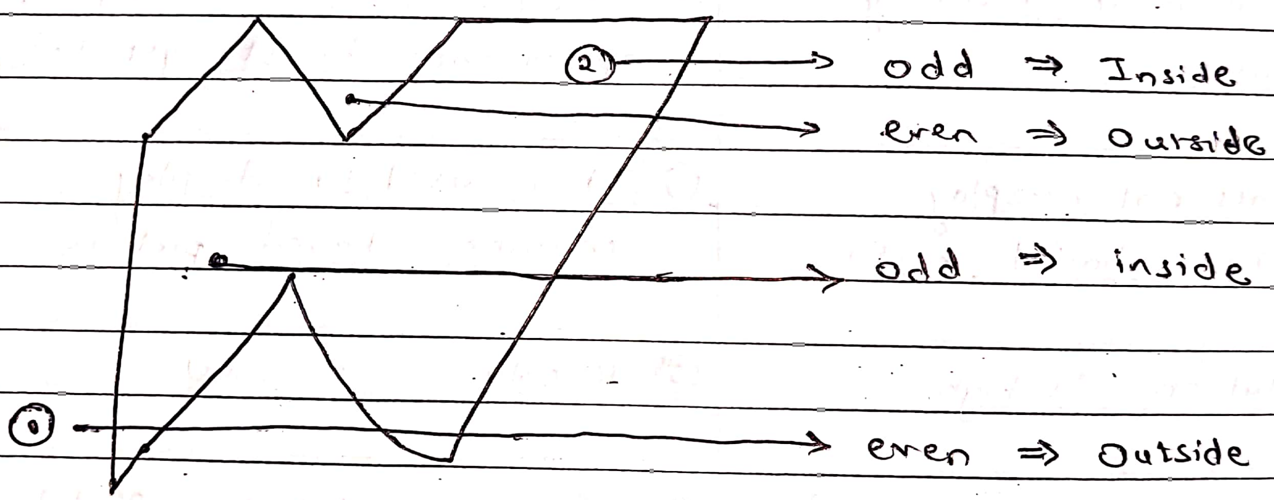
Q3. Explain inside outside test.

We can determine whether a point is inside or outside in polygon. It is possible using 2 methods

1. Even odd rule
2. Non-zero Winding number rule.

1. Even odd rule

We have to test whether a point is inside or outside.

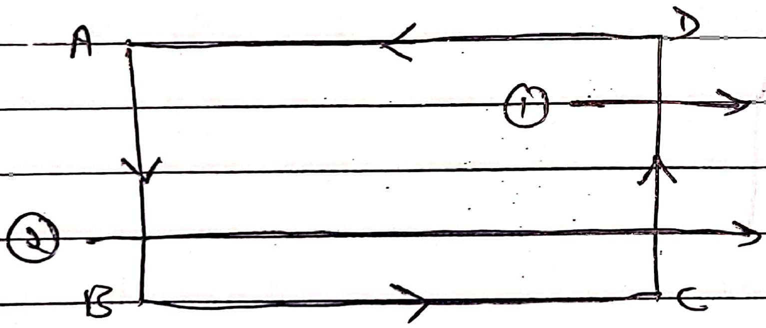


- Make a line parallel to x axis and which crossed the polygon vertices.
- Then count how many intersections are occurring with polygon boundry by the line till the point i.e line is reached.
- If there is odd number of intersections then point is inside else it is outside of polygon.
- This is even odd method.

Terna Engineering College, Navi Mumbai

② Non zero winding rule

- In non-zero winding number system we give direction number to each boundry line, which is crossed by this line and we sum these direction number
- Direction number indicates the direction in which polygon edges are drawn
- The side which start at above the ~~drawn~~ drawn line and crosses line and then ends below the line.
- In that case, we design that side direction number as 1.
- If the edges of polygon start below the drawn line and then ends above the line, we give -1 to direction number.
- Then find sum of there number if it is non zero then point is inside. If the sum is zero then point is outside



Rules:

Winding number $w = 0$

$R \to L = w = w+1$

$L \to R = w = w-1$

$w \neq 0 \Rightarrow$ Inside

$w = 0 \Rightarrow$ Outside

Point 1

CD $= w = 1$ $\Rightarrow$ Inside

Point 2

CD $= R \to L = w = w+1 = w$

AB $= L \to R = w-1 = 1-1 = 0$

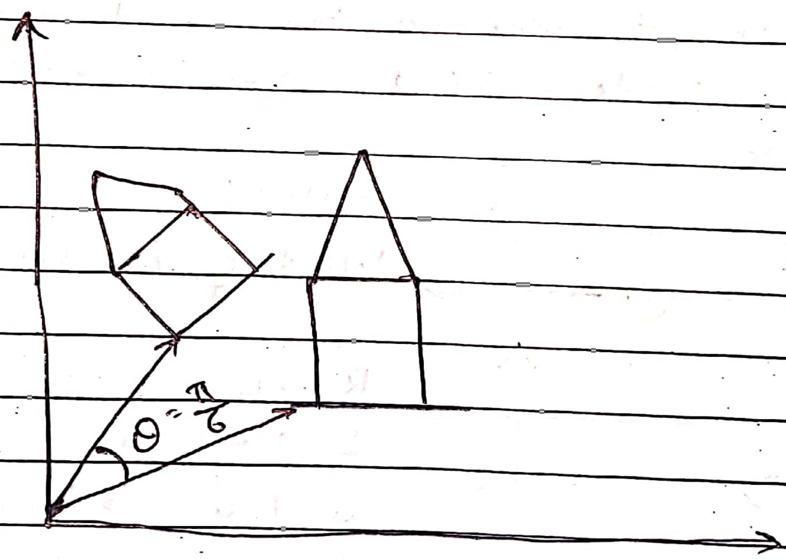
$\Rightarrow$ Outside

Q4. Explain 2D rotation about arbitrary point

- Transformation means change in image.
- We can modify the image by performing some basic transformation such as
  → Scaling
  → Rotation
  → Translation

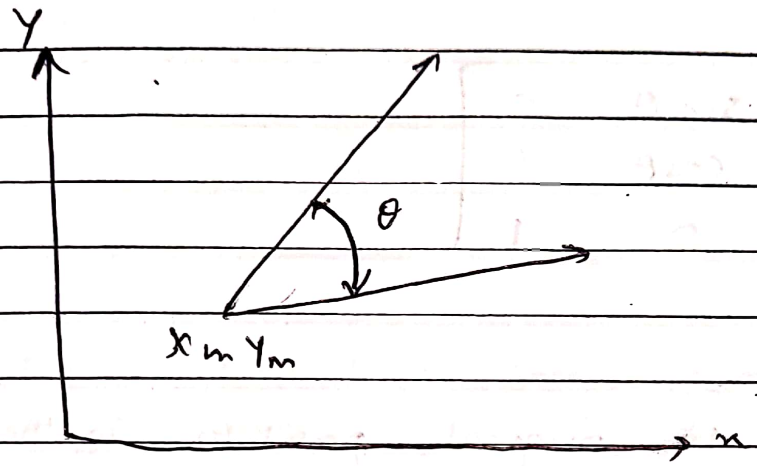
Rotation :

For rotation we need trigonometry logic, suppose we have point $P1 = (x_1, y_1)$ and we rotate it about the original by an angle θ to get a new position $P_2 = (x_2, y_2)$ as shown.



Rotation about arbitrary point :

Suppose the reference point of rotation is other than origin, then in that case we have to follow series of transformation



Assume that we have to rotate a point P1 with respect to (x_m, y_m) then we have to perform 3 steps

① Translation :

First we need to translate the (x_m, y_m) to origin

Translation matrix (T_1) will become

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

Here, $T_x = -x_m$ and $T_y = -y_m$

② Rotation :

Rotate it in clockwise or anticlockwise direction. Let's assume anticlockwise direction by θ angle. So rotation matrix will be

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

③ Translation :

Translate back to original position. So the translation matrix (T_2) will be

$$T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ X_m & Y_m & 1 \end{bmatrix}$$

Now combined matrix

$$M = \text{Translation} * \text{Rotation} * \text{Translation}$$

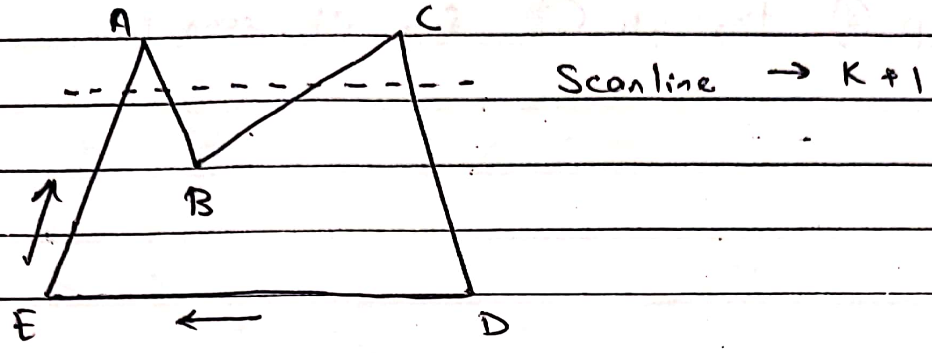
$$= T_1 * R * T_2$$

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

$$= \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ -X_m * \cos\theta + Y_m * \sin\theta + X_m & -X_m * \sin\theta - Y_m * \cos\theta + Y_m & 1 \end{bmatrix}$$

Q5. Explain scan line polygon fill algorithm.

Step 1 : Accept number of vertices



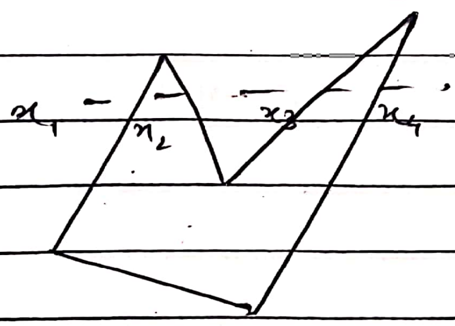
Step 2 : AEL is prepared for each scan line which contains list of intersecced edge.

AEL =

AB
BC
CD
ED

Step 3 : For each polygon in AEL intersecting points are calculated

Step 4 : Sort all intersecting points in increasing order in values.



x_1	→	Left
x_2	→	Right
x_3	→	Left
x_4	→	Right

First x value = left boundry

Second x value = Right boundry

Step 5 : Fill all pixel position between L & R boundry with specified color value.

step 6 : Repeat step ② through ⑤ for all scan line.

step 7 : STOP

Terna Engineering College, Navi Mumbai

Q6. Explain Cohen Sutherland line clipping algorithm

Step 1: Accept line segments (x_1, y_1) & $x_2, y_2)$ and window $(x_{w\,min}, y_{w\,min})$ & $(x_{w\,max}, y_{w\,max})$

Step 2: Find 4 bit region code

if	x_1	<	$x_{w\,min}$	B_1	= 1	else	$B_1 = 0$
if	x_2	>	$x_{w\,max}$	B_2	= 1	else	$B_2 = 0$
if	y_1	<	$y_{w\,min}$	B_3	= 1	else	$B_3 = 0$
if	y_2	>	$y_{w\,max}$	B_4	= 1	else	$B_4 = 0$

1001	1000	1010
0001	0000	0010
0101	0100	0110

TBR2

Step 3: (a) If both end point codes are 0000 visible display and stop

(b) If logical AND of both end point codes is not 0000, invisible and discard line & STOP.

(c) If logical AND of end point code is 0000 then line is clipping candidate

Step 4: Determine intersecting point coordinates (x', y')

$$x' = x_1 + \frac{1}{m}(y' - y_1) \text{ where}$$

$$y' = y_{w\,min} / y_{w\,max}$$

$$y' = y_1 + \frac{1}{m}(x' - x_1) \text{ where}$$

$$x' = x_{w\,min} / x_{w\,max}.$$

Step 6: Go to step 2, until you get totally visible or invisible line segment

Step 7: STOP.