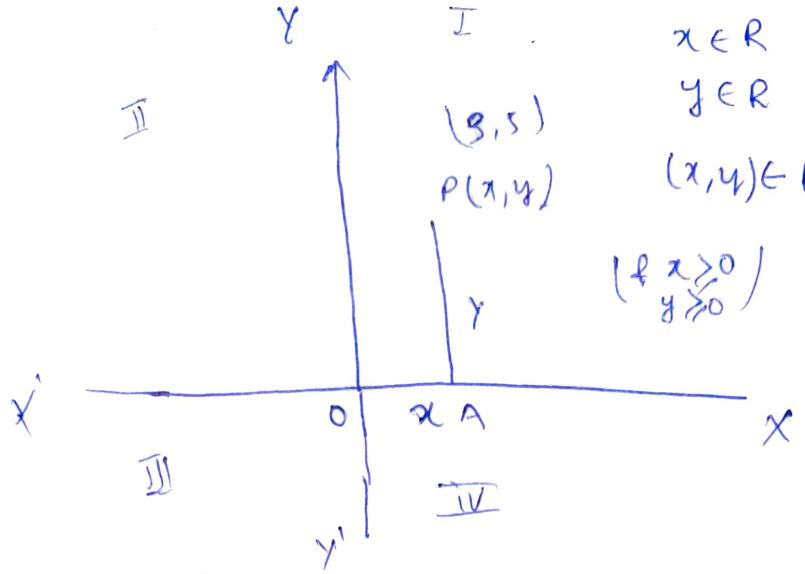
06-02-24



$x \in R$

$y \in R$

$(x,y) \in R \times R = R^2$

$(if\ x \geq 0,\ y \geq 0)$

* General Equation of Line

$ax + by + c = 0$

$\rightarrow y = mx + c$ - in

$\rightarrow \frac{dy}{dx} = m$ { differential eq of line degree 1

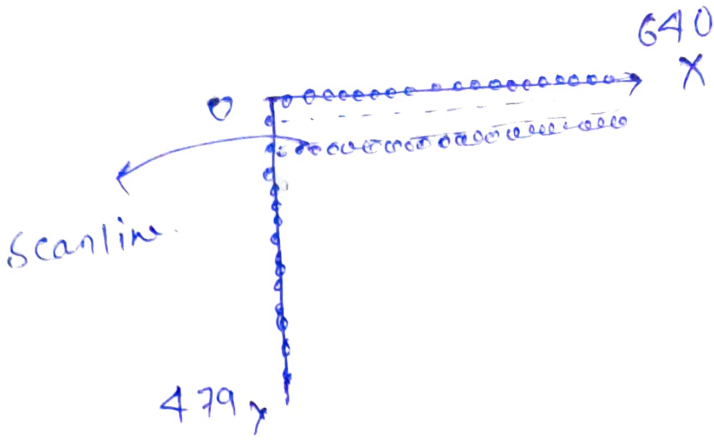
$\frac{\Delta y}{\Delta x}$

$\rightarrow \frac{d^2y}{dx^2} = 0$ { degree 2

* computer screen :-

i) computer screen is nothing but the matrix (raster) collection of pixels, pixel is smallest geometric with has width and height.

- The no. of pixel horizontally in a line and no. of pixel vertical line gives resolution of screen. the very common resolution of computer screen is 640×480

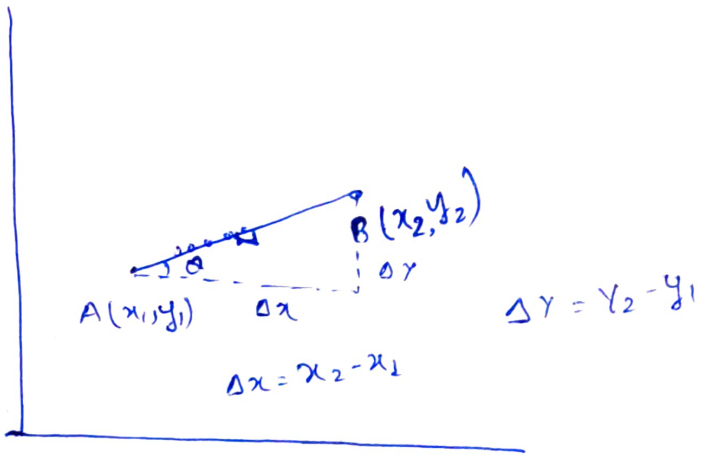


→ row of pixcell in computer screen is called scanline.

* D.D.A (digital defference Analyzer)

→ pixcell position are intiger value.

→ from two point we draw only one line.



Slop $\Delta x \Rightarrow \frac{y_2 - y_1}{x_2 - x_1} = m$

$m = \tan\theta^\circ \quad \tan 90^\circ = \infty$

Case I

* Now consider the case :-

$(0 < m < 1)$

→ if $x_1 < x_2$ then movement of pixel on line is left to right. (L → R)

$$m = \frac{\Delta y}{\Delta x} < 1 \Rightarrow \Delta y < \Delta x$$

→ Scince Δx > Δy so, at every step of pixel determination the change in x co-orclinate will be larger then the change in y -co-orclinate, and the maximum possible increment which we can give to x or y co-orclinate will be 1.

In this case we will give maximum possible increment to x - coordinate is 1.

Suppose we are at pixel x and y (x_n, y_n) and the next position is (x_{n+1}, y_{n+1})

then,

$$x_{n+1} = x_n + 1$$

and $m = \frac{\delta}{1}$ (figure) ∴ $\boxed{m = \frac{\delta}{1} \quad \delta = m}$

$$\delta = m \quad \text{(increment in y)}$$

$$y_{n+1} = y_n + m$$

Thus we have

$$\left\{ \begin{array}{l} x_{n+1} = x_n + 1 \\ y_{n+1} = y_n + m \end{array} \right\} \quad \text{--- (1)}$$

To get the pixcell position corresponding to a point we will be used to the round function in this algorithms.

e.g- round(4.7) = 5
round(4.29) = 4

2* Case = II

If $m > 1$ $(L \to R)$ $(x_1 < x_2)$

$\frac{\Delta y}{\Delta x} > 1 \Rightarrow \Delta y > \Delta x$

Then the next position in these case

$$y_{n+1} = y_n + 1$$

Again $m = \frac{1}{\sigma}$, ~~there σ is the change~~
there σ is the change

$\Rightarrow \sigma = \frac{1}{m}$ { mx can at step of next position determine

then,

$$x_{n+1} = x_n + \frac{1}{m}$$

$$\left\{ \begin{array}{l} x_{n+1} = x_n + 1/m \\ y_{n+1} = y_n + 1 \end{array} \right\} \quad \text{--- (2)}$$

3* case III (L←R) $(x_1 > x_2)$

$(0 < m < 1)$ $m = \frac{\Delta y}{\Delta x} < 1$

$\Rightarrow \Delta y < \Delta x$

then eqn $\begin{cases} x_{n+1} = x_n - 1 \\ y_{n+1} = y_n - m \end{cases}$ — (3)

4* case IV

$(m > 1)$ $(R \to L)$ $(x_1 > x_2)$

$\frac{\Delta y}{\Delta x} > 1 \Rightarrow \Delta y > \Delta x$

then $\begin{cases} y_{n+1} = y_n - 1 \\ x_{n+1} = x_n - 1/m \end{cases}$ — 4

$\Rightarrow$ using eq 1, 2, 3, 4 (as per case algo. determines the pixel position all computer screen and eliminate them to generate the line on computer screen.

1. Q Digitize the to line to joining points (10,10) to (20,16) using DDA Algo and plot the pixcell on cartesian grid.

where

$(x_1, y_1) \equiv (10,10)$

$(x_2, y_2) \equiv (20,16)$

$(x_1 < x_2)$

$\Delta x = x_2 - x_1 = 20 - 10 = 10$

$\Delta y = y_2 - y_1 = 16 - 10 = 6$

$m = \frac{\Delta y}{\Delta x} = \frac{6}{10} = 0.6$

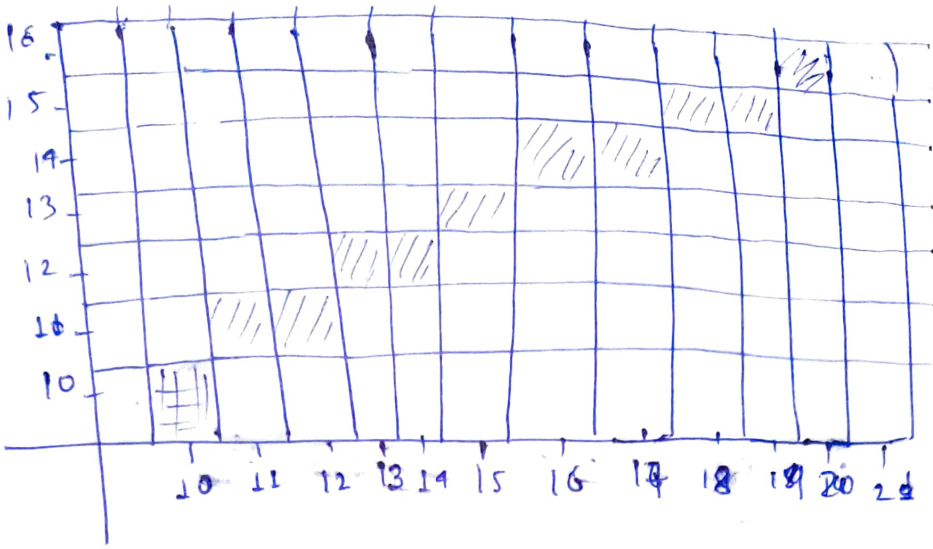
Solve.

$(0 < m < 1)$ & $(x_1 < x_2)$ thus it is case I of DDA.

$x_{n+1} = x_n + 1$

$y_{n+1} = y_n + m$

C.P	draw, graph C.pixcell	N.P	round N. pixel
(10.10)	(10,10)	11, 10.6	(11, 11)
(11, 10.6)	(11, 11)	(12, 11.2)	(12, 11)
(12, 11.2)	(12, 11)	(13, 11.8)	(13, 12)
(13, 11.8)	(13, 12)	(14, 12.4)	(14, 12)
(14, 12.4)	(14, 12)	(15, 13)	(15, 13)
(15, 13)	(15, 13)	(16, 13.6)	(16, 14)
(16, 13.6)	(16, 14)	(17, 14.2)	(17, 14)
(17, 14.2)	(17, 14)	(18, 14.8)	(18, 15)
(18, 14.8)	(18, 15)	(19, 15.4)	(19, 15)
(19, 15.4)	(19, 15)	(20, 16)	(20, 16)
(20, 16)	(20, 16)		



Q Digitize the to Line to join points (10,10) to (18,20) using DDA Algo. and plot the pixel on cartesian grid.

where,

$(x_1, y_1) \equiv (10, 10)$

$(x_2, y_2) \equiv (18, 20)$

$(x_1 < x_2)$

$\Delta x = x_2 - x_1 = 18 - 10 = 8$

$\Delta y = y_2 - y_1 = 20 - 10 = 10$

$m = \frac{\Delta y}{\Delta x} = \frac{10}{8} \Rightarrow 1.25$

Hence,

$(m > 1)$ & $(x_1 < x_2)$ and increment in L→R so it is II case of DDA

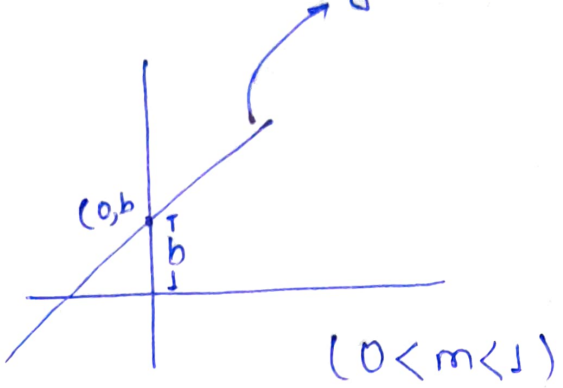
$\frac{\Delta y}{\Delta x} > 1 \Rightarrow \Delta y > \Delta x$.

$y_{n+1} = y_n + 1$

2. Algorithms :-

Breshem's line generating Algo.

Eq of line $y = mx + b$



$$m = \frac{\Delta y}{\Delta x} < 1$$

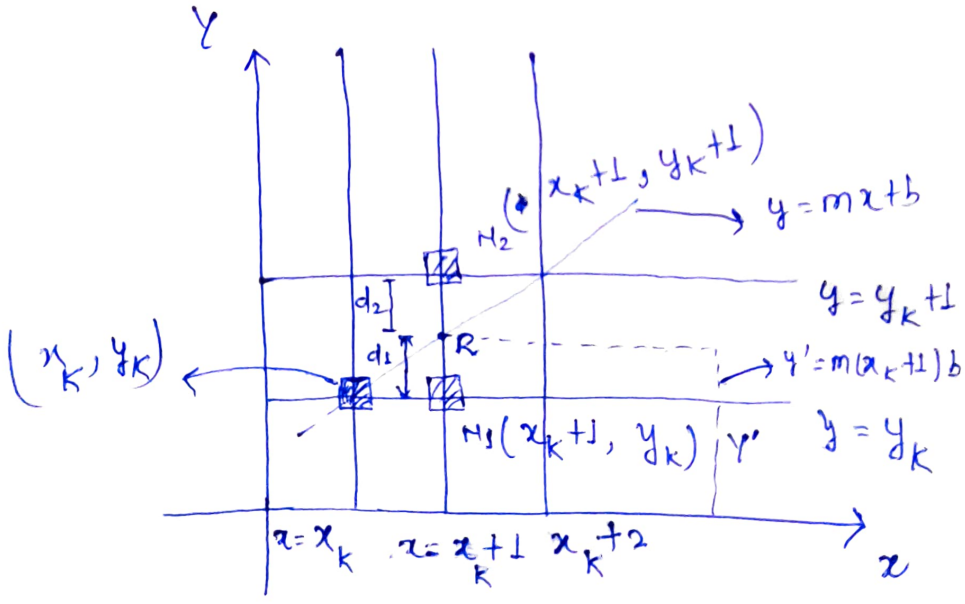
In Breshem's algo. the process of pixel determination is left-l → Right ($x_1 < x_2$) x_2 is strictly greater then x_1 since $m < 1$ so ΔY is less then Δx at every step of pixel determin to get the next pixel position we will give the maximum possible of 1 to x-co-ordinate.

Now we are only remaning with the problem for increment 1 in y co-ordinate.

To decicles wether y-co-orclinate will be increment by 1 or not, algorithm will set a recusive equtions of decision parameter whose sign will tell us about the increment n in y co-orclinate.

Let pixell (x_k, y_k) is to be displayed and the next pixell position is (x_{k+1}, y_{k+1}) then

$$x_{k+1} = x_k + 1 \text{ and } y_{k+1} = y_k \text{ or } y_k + 1$$



R = Real theorehical point.

$y = mx + b$

$R = (x_k + 1, m(x_k + 1) + b)$

from fig. R is the Real theoretical point with is intersechon of lines $y = mx+b$ and $x = x_k + 1$

from fig -

$$d_1 = R_{H_1} = y' - y_k = (m(x_k+1)+b-y_k) \quad -(1)$$

$$d_2 = R_{H_2} = y_k + 1 - y' = \{(y_k+1) - [m(x_k+1) + b]\} \quad -(2)$$

then

$$d_1 - d_2 = \{m(x_k+1)+b-y_k\} - \{(y_k+1) - [m(x_k+1)+b]\}$$

$$d_1 - d_2 = mx_k + m + b - y_k - y_k - 1 + mx_k + m + b$$

$$d_1 - d_2 = 2mx_k - 2y_k + \underbrace{2m + 2b - 1}_{\text{Constant}}$$

$$\Rightarrow d_1 - d_2 = 2mx_k - 2y_k + c \quad \{\text{where } c = 2m + 2b - 1\}$$

$$\because m = \frac{\Delta y}{\Delta x}$$

$$\Rightarrow d_1 - d_2 = 2\frac{\Delta y x_k}{\Delta x} - 2y_k + c \rightarrow (3)$$

Since $(\Delta x > 0)$ then the sign of $(d_1 - d_2)$ and $\Delta x(d_1 - d_2)$ will remain same.

$\Rightarrow \Delta x(d_1 - d_2) = 2\Delta y\, x_k - 2\Delta x.y_k + c.\Delta x$

if $(d_1-d_2) \geq 0 \Rightarrow \Delta x(d_1-d_2) \geq 0$.

$(d_1-d_2) < 0 \Rightarrow \Delta x.(d_1-d_2) < 0$.

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

$y_{k+1} = y_k$ or $y_k + 1$

from fig it is quit clear that if $\Delta x.(d_1-d_2) < 0$ then N_1 is close to R then $y_{k+1} = y_k$.

If $\Delta x.(d_1-d_2) \geq 0$ then N_2 will be close to R

$$y_{k+1} = y_k + 1$$

Taking

$\Delta x.(d_1-d_2) = P_k$ → Decision parameter for k^{th} step

$$P_k = 2\Delta y.x_k - 2\Delta x.y_k + c.\Delta x \quad -(4)$$

Similarly

$$P_{k+1} = 2\Delta y.x_{k+1} - 2\Delta x.y_{k+1} + c.\Delta x \quad -(5)$$

Subtracting eq (4) from (5) we have

$$P_{k+1} - P_k = 2\Delta y.(x_{k+1} - x_k) - 2\Delta x(y_{k+1} - y_k)$$

$$\Rightarrow P_{k+1} = P_k + 2\Delta y.(x_{k+1} - x_k) - 2\Delta x.(y_{k+1} - y_k)$$

$$P_{K+1} = P_K + 2\Delta y - 2\Delta x \cdot (y_{K+1} - y_K)$$

if $P_K < 0$, $y_{K+1} = y_K$

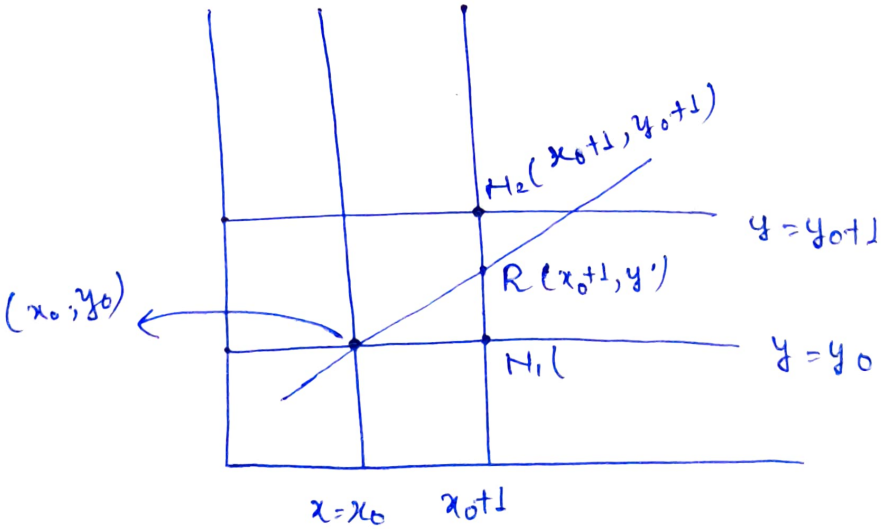
$$\boxed{P_{K+1} = P_K + 2\Delta y} \quad \text{—} \quad (6)$$

if $P_K \geq 0$

$$P_{K+1} = P_K + 2\Delta y - 2\Delta x\,(y_{K+1} - y_K)$$

$$\boxed{P_{K+1} = P_K + 2\Delta y - 2\Delta x} \quad \text{—} \quad (7)$$

The degree of recursive ~~rela~~ eq 7 is 1 so we need one initial value of the decision parameter that is P_0



$$P_0 = \Delta x \cdot (d_1 - d_2)$$

$$d_1 = R_{H_1} = m(x_0+1) + b - y_0$$

$$d_2 = R_{H_2}(y_0+1) - \{m(x_0+1)+b\}$$

$$P_0 = \Delta x \cdot (d_1 - d_2) = \Delta x \{ mx_0 + m + b - y_0 - y_0 - 1 \}$$

$$+ mx_0 + m + b$$

$$P_0 = \Delta x \cdot \{ 2mx_0 - 2y_0 + 2m + 2b - 1 \}$$

$$= \Delta x_0 \{ 2(mx_0 + b - y_0) + 2m - 1 \}$$

$$P_0 = \Delta x \{ 2\frac{\Delta y}{\Delta x} - 1 \}$$

$$\boxed{P_0 = 2\Delta y - \Delta x} \quad (8)$$

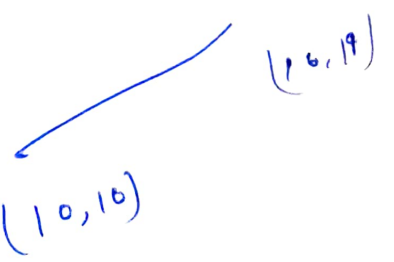
By using eq 6, 7, 8 Algo determines the pixell position all computer screen and this way digitized the line.

Q Digitze the line joing points (10, 10 to (16, 14) using B. Algo. and plot on cartesian grid.

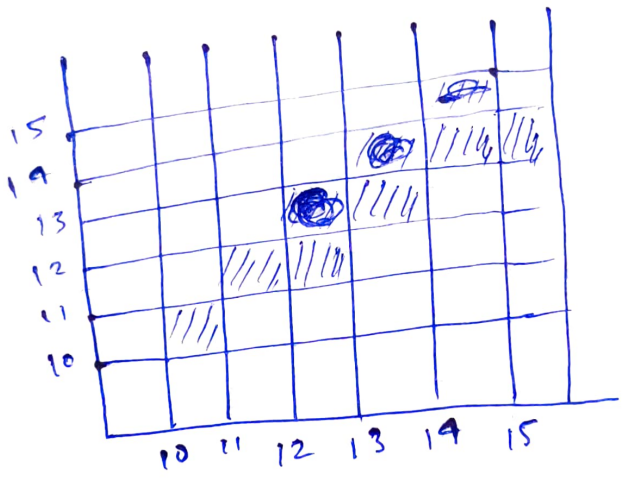
Sol - $\Delta x = 16 - 10 = 6$

$\Delta y = 14 - 10 = 4$

$m = \frac{4}{6} = \frac{2}{3} = .66$



C-P	P_K	N.P
(10,10)	$P_0 = 2 \cdot 4 - 6 = 2$	(11,11)
(11,11)	$P_1 = 2+8-12 = -2$	(12,11)
(12,11)	$P_2 = -2+8 = 6$	(13,12)
(13,12)	$P_3 = 6+8-12 = 2$	(14,13)
(14,13)	$P_4 = 2+8-12 = -2$	(15,13)
(15,13)	$P_5 = -2+8 = 6$	(16,14)



(10,10) (20,18)

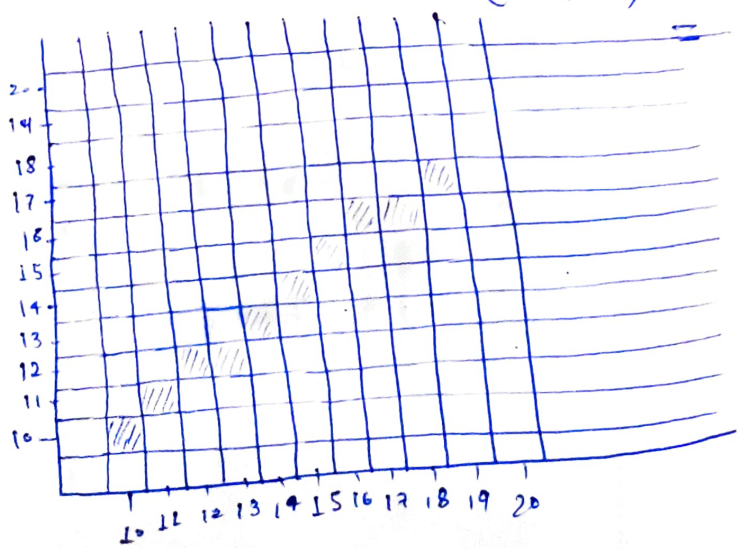
Q Digitize the line joing points (10,10) to (28,18)

Soln = $\Delta x = 20 - 10 = 10$

$\Delta y = 18 - 10 = 8$

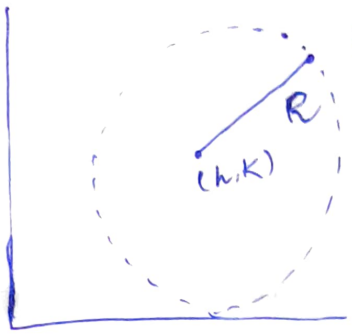
$m = \frac{\Delta y}{\Delta x} = \frac{8}{10} = 0.8$.

CP	P_k	NP
(10,10)	$P_0 = 2 \cdot 8 - 10 \Rightarrow 6$	(11,11)
(11,11)	$P_1 = 6 + 16 - 20 = 2$	(12,12)
(12,12)	$P_2 = 2 + 16 - 20 = -2$	(13,12)
(13,12)	$P_3 = -2 + 16 = 14$	(14,13)
(14,13)	$P_4 = 14 + 16 - 20 = 10$	(15,14)
(15,14)	$P_5 = 10 + 16 - 20 = 6$	(16,15)
(16,15)	$P_6 = -6 + 16 - 20 = 2$	(17,16)
(17,16)	$P_6 = 2 + 16 - 20 = -2$	(18,16)
(18,16)	$P_7 = -2 + 16 = 14$	(19,17)
(19,17)	$P_8 = 14 + 16 - 20 = 10$	(20,18)

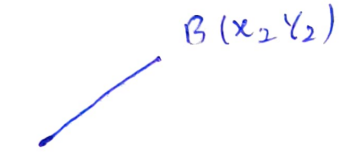


16-02-24

Circle is locas of a point which moves in such way that is distance from fixed point alway remain same.



$R = \sqrt{(x-h)^2 + (y-k)^2}$



$d(A,B) = \sqrt{(x_2-x_1)^2 + (Y_2-Y_1)^2}$

* $R = \sqrt{(x-h)^2 + (Y-k)^2}$

$R^2 = (x-h)^2 + (Y-k)^2$

$R^2 = x^2 - 2hx + h^2 + Y^2 - 2kY + k^2$

$x^2 + Y^2 - 2hk - 2kY + (h^2 + k^2 + h^2) = 0$

$(-h = g, \quad -k = f)$

$\boxed{x^2 + Y^2 + 2gx + 2fY + c = 0}$

general equation

circle with centre at O $(h=0, k=0)$.

eq

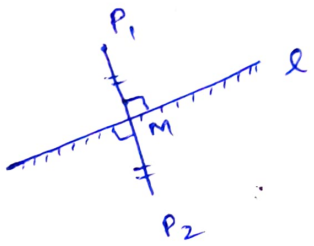
$R^2 = x^2 + y^2$

$x^2 + y^2 = R^2$ $((0,0))$

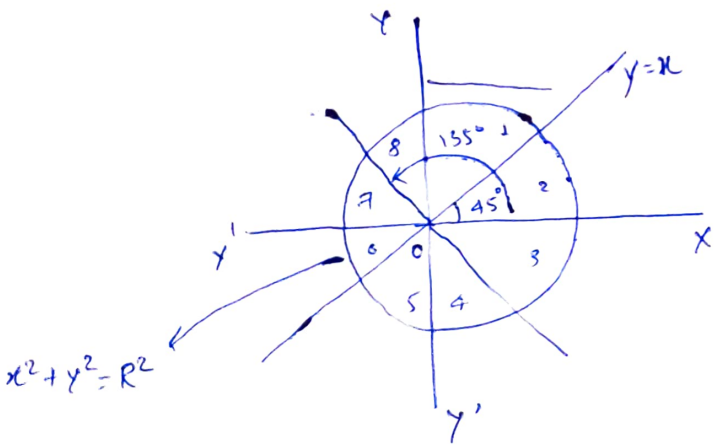
Radius R

Generation circle on computer screen.

Symmetry



P_1 and P_2 in fig. are the symmetry point about line ℓ. $\overline{P_1 P_2} \perp \ell$ and $P_1M = P_2M$



Line go on centre

$y = mx$

$m = \tan\theta$

$m = 1, \theta = 45°$

$y = x$

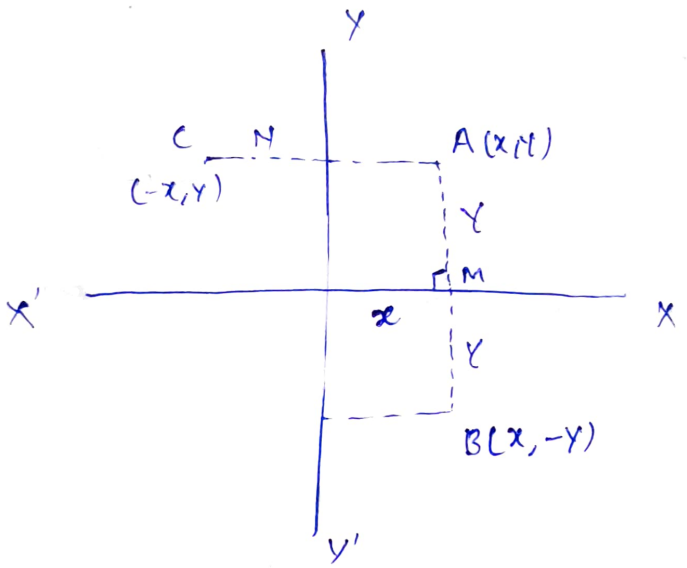
$y = -x$

The circle $x^2 + y^2 = R^2$ is symmetrical about x-axis, y-axis about the line $y = x$ and $y = -x$.

→ while drowing circle on computer screen the algorithm will determine the pixell only in one octant. $(45° \leq \theta \leq 90°)$

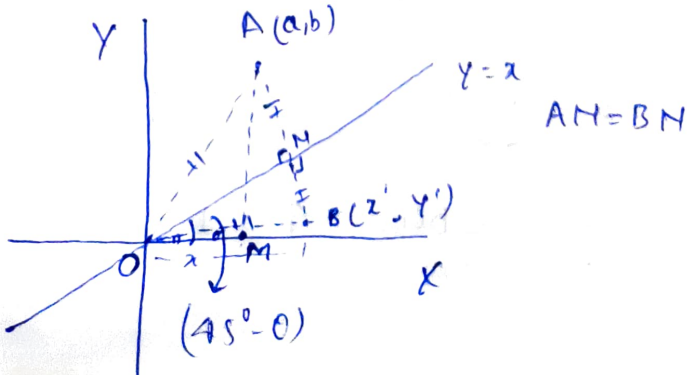
Remaning seven part of circle will be generated by taking reflextion of about ~~this pixell~~ x-axis, y-axis about $y=x$ and line about $y=-x$.

* Reflection about xaxis y-axis and line $y=x$



from fig point A (x,y) its about x-axis is given by B (x,-y) also point C (-x,y) the reflection of point A (x,y).

* Reflekion about the line $y=x$



from fig $A(a,b)$ is a point on x,y plane and $B(x',y')$ is reflection of point A about the line $y=x$.

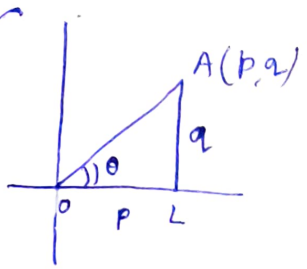
$$OA^2 = ON^2 + AN^2$$

$$= ON^2 + BN^2 = OB^2$$

$$OA = OB \quad (r \text{ says})$$

from fig $\Delta OHA \cong \Delta OBN$

$$\begin{cases} \cos(A \pm B) = \cos A \cos B \mp \sin A \sin B \\ \sin(A \pm B) = \sin A \cos B \pm \cos A \sin B \end{cases}$$



from fig - ΔOAM

$$OM = r\cos(45 + \varphi)$$

$$a = r\cos(45 + \varphi)$$

$$b = r\sin(45 + \varphi)$$

$$\Rightarrow a = r\{\cos 45 \cos\varphi - \sin 45 \sin\varphi\}$$

$$a = \frac{r}{\sqrt{2}}(\cos\varphi - \sin\varphi) \quad -(1)$$

$$b = r\{\sin 45 \cos\varphi + \cos 45 \sin\varphi\}$$

$$b = \frac{r}{\sqrt{2}}\{\cos\varphi + \sin\varphi\} \quad -(2)$$

again $\angle BON = 45 - \varphi$

In ΔOBN

$\cos\theta = \frac{B}{H}$

$\cos\theta = \frac{b}{OA}$

$b = OA\cos\theta$

$\sin\theta = \frac{a}{OA}$

$a = OA\sin\theta$

$$x' = r\cos(45 - \theta) = r\{\cos 45 \cos\theta + \sin 45 \sin\theta\}$$

$$= \frac{r}{\sqrt{2}}(\cos\theta + \sin\theta) = b$$

$$y' = r\sin(45 - \theta)$$

$$y' = r\{\sin 45^\circ \cos\theta - \cos 45^\circ \sin\theta\}$$

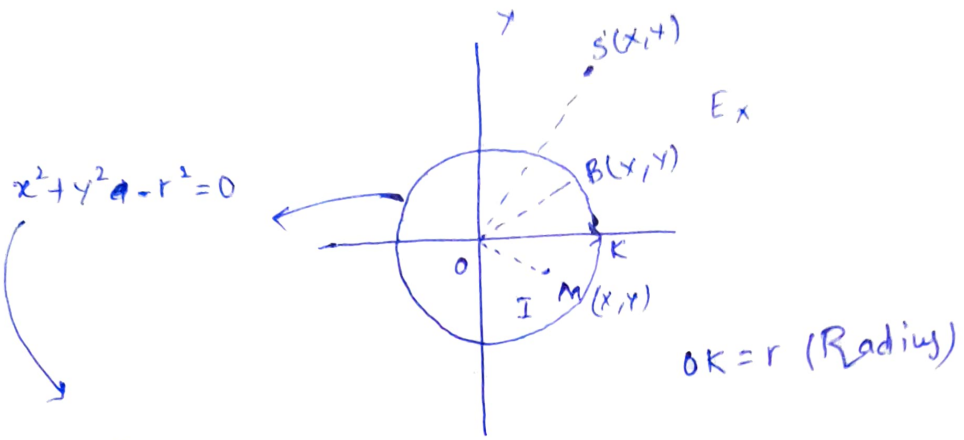
$$y' = \frac{r}{\sqrt{2}}(\cos\theta - \sin\theta) = a$$

$$x' = b$$

$$y' = a$$

19-02-24

Circle generation



$x^2+y^2-r^2=0$

$$f(x,y) = x^2+y^2-r^2$$

for interior point $M(X,Y)$

$OM = \sqrt{x^2+y^2}$

$OM < r$

$OM^2 < r^2$

$\Rightarrow x^2+y^2<r^2$

$\Rightarrow x^2+y^2-r^2<0$

$\Rightarrow f(x,y)<0$ for interior points

for exterior point $S(X,Y)$

$OS > r$

$OS^2 > r^2$

$x^2+y^2 > r^2$

$\Rightarrow x^2+y^2-r^2>0$

$f(x,y) > 0$ for exterior point

Boundary point

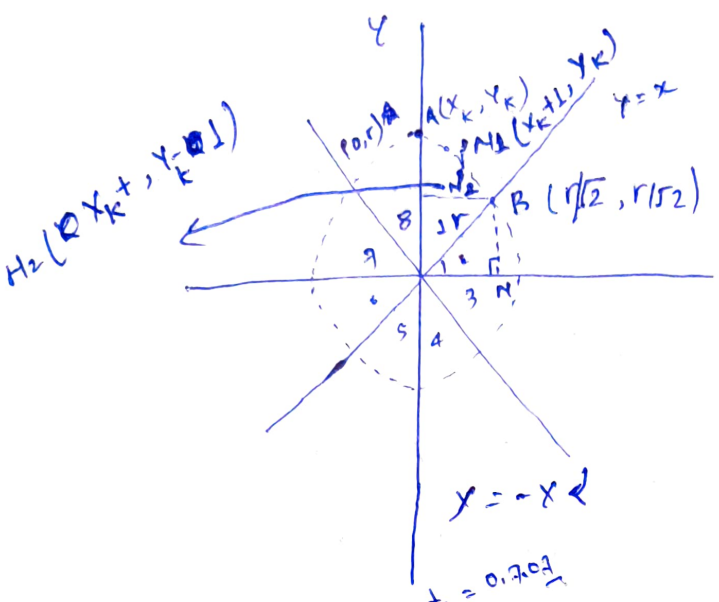
$OB = r$

$OB^2 = r^2$

$x^2+y^2-r^2=0$

⇒ for boundary points $f(x,y) = 0$

* The Bresenham's circle :-
Generation Algorithms



$45° \leq \theta \leq 90°$

$\sin 45° = \frac{b}{r}$

$b = \frac{r}{\sqrt{2}}$

$\frac{1}{\sqrt{2}} = 0.707$

$\Delta x = \frac{r}{\sqrt{2}} - 0 = \frac{r}{\sqrt{2}} = 0.707 \cdot r$

$\Delta y = \frac{r}{\sqrt{2}} - r = -\left(1 - \frac{1}{\sqrt{2}}\right) = .293r$

⇒ $|\Delta x| > |\Delta y|$

Since the total change in x value is larger than the change in y value while moving point A to B along the arc of circle so Algo. will give Maximum possible increment of 1. to x-value. while determine the next pixel position,

Now we are only remaining with the y-co-ordinate wether y-coordinate will decrement by 1 or not'

To decides this algorithm will develop a recursive eq. of parameter whose sign tell us y value will decrement by 1 or not.

Let pixell X_k, Y_k will display the next pixell poisition which contain (X_{K+1}, Y_{K+1}) where,

$$X_{k+1} = X_k + 1$$

and $Y_{K+1} = Y_k$ or $Y_k - 1$.

i.e- The posible pixell positions

$$H_1 = (X_k + 1, Y_k) \text{ or } H_2 (X_k + 1, Y_k - 1)$$

Out of ^two ~~these thing~~ pixell position algorithm will select that pixell which will be much close to the circumfrance.

How

$d_1 = f(x, y)$ at H_1

$d_1 = ((X_k + 1)^2 + Y_k^2 - r^2)$ ∴ H_1 is Exterior $d_1 > 0$

Again

$d_2 = f(x, y)$ at H_2

$d_2 = (X_k + 1)^2 + (Y_k - 1)^2 - r^2$ H_2 is interior $d_2 < 0$

then, setting

$$p_k = d_1 + d_2$$

, where p_k is the decision parameter for k^{th} step

hence

$p_k = d_1 + d_2$

if $|d_1| > |d_2|$

$p_k > 0$

if $|d_1| < |d_2|$

$p_k < 0$

if $p_k < 0$, the next pixell position is N_1 (and close to circumference).

i.e - $y_{k+1} = y_k$

if $p_k \geq 0$, the next pixell position is N_2

i.e- $y_{k+1} = y_k - 1$

$d_1 = (x_k + 1)^2 + y_k^2 - r^2$

$d_2 = (x_k + 1)^2 + (y_k - 1)^2 - r^2$

Now

$$p_k = d_1 + d_2 = 2(x_k + 1)^2 + y_k^2 + (y_k - 1)^2 - 2r^2 \quad -(1)$$

p_k is the decision parameter for k^{th} step so we can obtain the value of p_{k+1} by substituting $k+1$ for k in eq (1)

$$p_{k+1} = 2(x_{k+1} + 1)^2 + y_{k+1}^2 + (y_{k+1} - 1)^2 - 2r^2 \quad -(2)$$

Subtracting eq ① from ②

$$p_{k+1} - p_k = 2\{(x_{k+1}+1)^2 - (x_k+1)^2\} + (y_{k+1}^2 - y_k^2) + (y_{k+1}-1)^2 - (y_k-1)^2$$

$$= 2\{(x_k+2)^2 - (x_k+1)^2\} + \{y_{k+1}^2 - y_k^2\} + \{y_{k+1}^2 - 2y_{k+1}+1\} - \{y_k^2 - 2y_k + 1\}$$

$$p_{k+1} - p_k = 2\{x_k^2 + 4x_k + 4 - x_k^2 - 2x_k - 1\} + \{y_{k+1}^2 - y_k^2\} + \{y_{k+1}^2 - y_k^2\} - 2\{y_{k+1} - y_k\}$$

$$p_{k+1} - p_k = 4x_k + 6 + 2\{y_{k+1}^2 - y_k^2\} - 2\{y_{k+1} - y_k\} \quad — ③$$

If $p_k < 0$, $y_{k+1} = y_k$

$$p_{k+1} - p_k = 4x_k + 6$$

$$\boxed{p_{k+1} = p_k + 4x_k + 6} \quad — ④$$

if $p_k \geq 0$, $y_{k+1} = y_k - 1$

$$p_{k+1} = p_k + 4x_k + 6 + 2\{(y_k-1)^2 - y_k^2\} - 2\{y_k - 1 - y_k\}$$

$$p_{k+1} = p_k + 4x_k + 6 + 2\{y_k^2 - 2y_k + 1 - y_k^2\} + 2$$

$$\boxed{p_{k+1} = p_k + 4x_k - 4y_k + 10} \quad — ⑤$$

eq ④ & ⑤ are helpful to determine the value decision parameter for different steps. Since the degree of recursive eq is 1 so we need one initial value of decision parameter.

To get the initial value of decision parameter

$\because p_k$

the initial point is A(0,r) so to get p_0 putting $x_k = 0$, $y = r$ in eq - ①

$\because p_k = d_1 + d_2 = 2(x_k+1)^2 + (y_k-1)^2 + (y_k - 2r^2)$

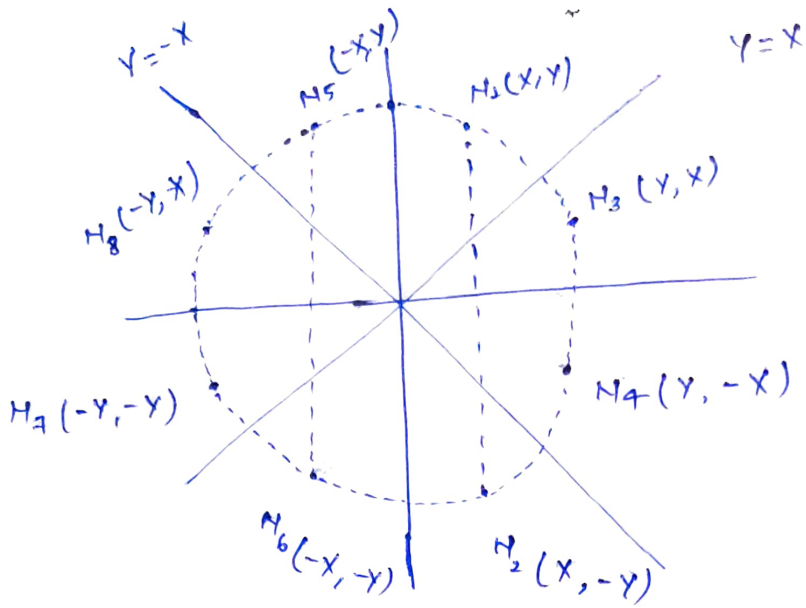
so, to get p_0, $x_k = 0$ & $y_k = r$ in eq ①

$p_0 = 2 + (r-1)^2 + r^2 - 2r^2$

$p_0 = 2 + r^2 - 2r + 1 + r^2 - 2r^2$

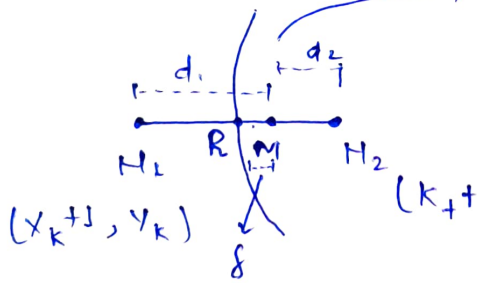
$\boxed{p_0 = 3 - 2r}$ — ⑥

By using eq ④, ⑤, ⑥ we can compute the decision parameter for all the step and thus in position to determine all the pixell. in the octant $45^\circ \leq 90^\circ$.



* Mid point circle Algorithms :-

$$f(x,y) = x^2 + y^2 - r^2$$



$$M \equiv (X_k + 1,\ Y_k - 1/2)$$

$$\frac{Y_k + Y_{k-1}}{2}$$

$$\frac{2Y_k - 1}{2} = Y_k - \frac{1}{2}$$

Q. Determine the pixel in an octant for circle $x^2 + y^2 = 100$ using mid point circle Algorithm & plot those pixels in cartesian grid graph.

Sol^n :- Eq^n of circle

$$x^2 + y^2 = 100$$

$$r = 10$$

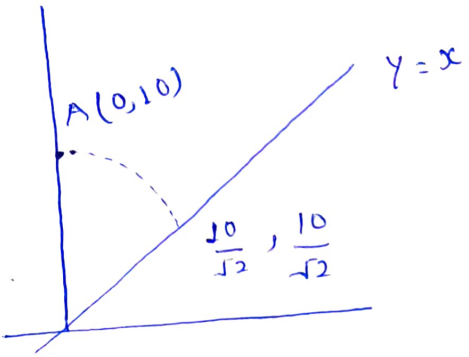
decision parameter

then d.P

$$p_0 = \frac{5}{4} - r$$

$$p_{k+1} = p_k + 2x_k + 3 \quad , p_k < 0$$

$$p_{k+1} = p_k + 2(x_k - 4_k) + 5 \quad , p_k \geq 0$$



Table

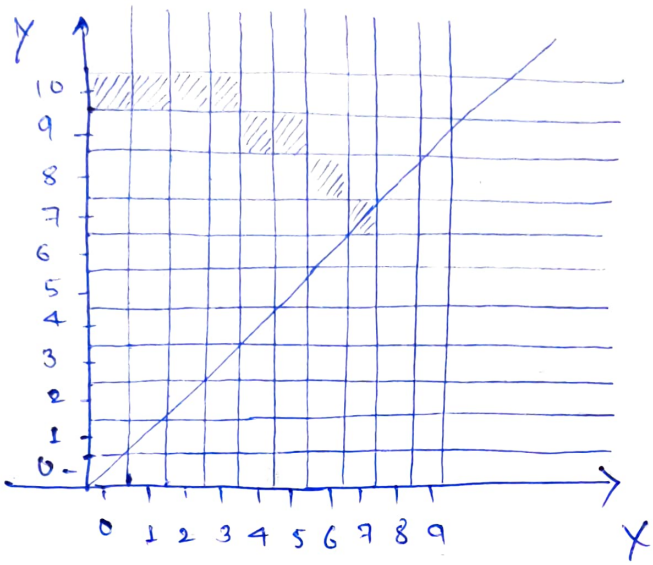
D.P	C.P	N.P
$p_0 = \frac{5}{4} - 10 = -\frac{35}{4}$	(0,10) (x_0, y_0)	(1,10)
$p_1 = -\frac{35}{4} + 2*0 + 3 = -23/4$	(1,10)	(2,10)
$p_2 = -\frac{23}{4} + 2*1 + 3 = -3/4$	(2,10)	(3,10)
$p_3 = -\frac{3}{4} + 2 \cdot 2 + 3 = \frac{25}{4}$	(3,10)	(4,9)

$p_4 = \frac{25}{4} + 2(3-10) + 5 = \frac{-11}{4}$ (4, 9) (5, 9)

$p_5 = \frac{-11}{4} + 11 = \frac{33}{4}$ (5, 9) (6, 8)

$p_6 = \frac{33}{4} + 2(-4) + 5$ (6, 8) (7, 7)

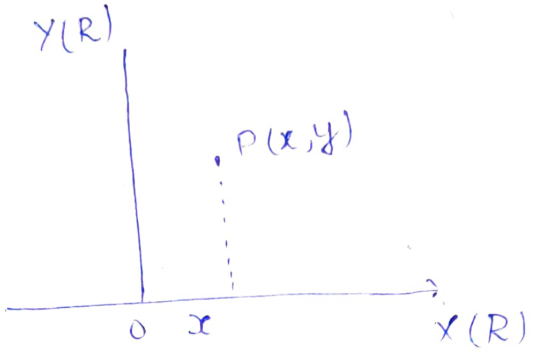
$= \frac{21}{4}$



26-02-24

2D transformation

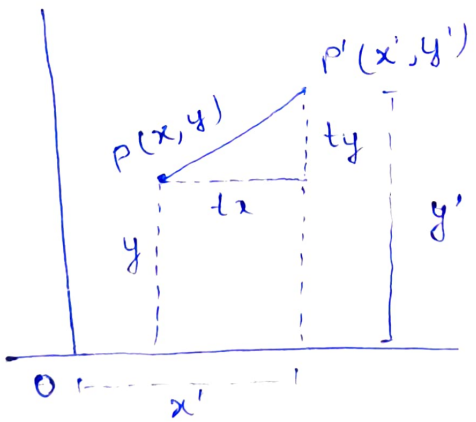
2-d transformation is general term which may include translation rotation, scaling reflection, shearing or few of them



$x \in X(R) \Rightarrow x \in R$
$y \in Y(R) \Rightarrow y \in R$

$(x,y) \in R^2 \rightarrow$ 2-D realplane

1. 2-d translation



from fig P(x,y) is point ~~comm~~ on the plane which is translated to point P'(x', y'), the translation component along x-axis tx and ~~co~~along y-axis is ty.

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

In Matrix form

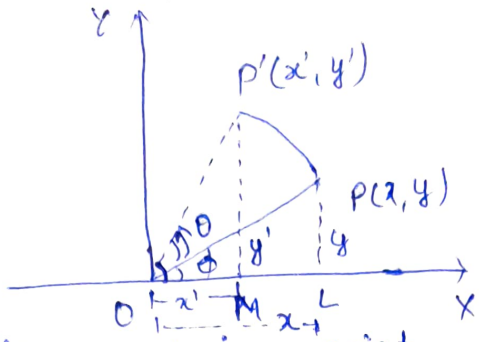
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} + \begin{bmatrix} x \\ y \end{bmatrix}$$

addition

RAT = Right angle triangle

θ = angular displacement

2. Rotation



from fig P(x,y) is a point which is rotated about origin through an angle θ in anticlockwise direction, P'(x', y') is the new position of point P after rotation.

clearly $OP = OP' = r$ (says)

in RAT OPL

$OL = x$, $PL = y$

$\angle POL = \phi$ (says)

$$\cos\phi = \frac{OL}{OP} = \frac{x}{r}$$

$$x = r\cos\phi$$

$$\sin\phi = \frac{PL}{OP} = \frac{y}{r}$$

$$y = r\sin\phi$$

Again from fig

in $\Delta OP'M$

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

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

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

$$x' = r\{\cos\theta\cos\phi - \sin\theta\sin\phi\}$$

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

$$x' = x\cos\theta - y\sin\theta \quad \text{—— (1)}$$

Again $\sin(\theta+\phi) = \frac{P'M}{OP'} = \frac{y'}{r}$

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

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

$$y' = r\{\sin\theta\cos\phi + \cos\theta\sin\phi\}$$

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

$$y' = x\sin\theta + y\cos\theta \quad \text{—— (2)}$$

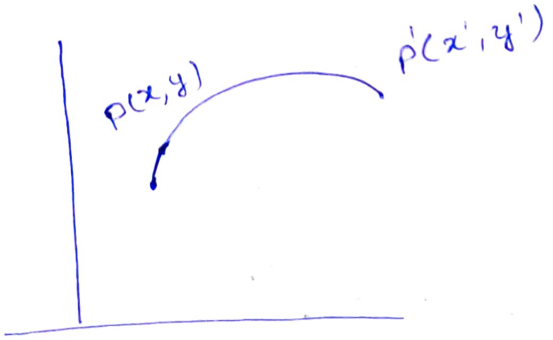
$$x' = x\cos\theta - y\sin\theta$$

In Matrix form

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

=

3. The scaling



P(x,y) is a point on 2-D plane and P'(x',y') is the position of point P after scaling about origin where scaling factors are S_x and S_y along OX and OY respectively.

$$x' = S_x \cdot x$$

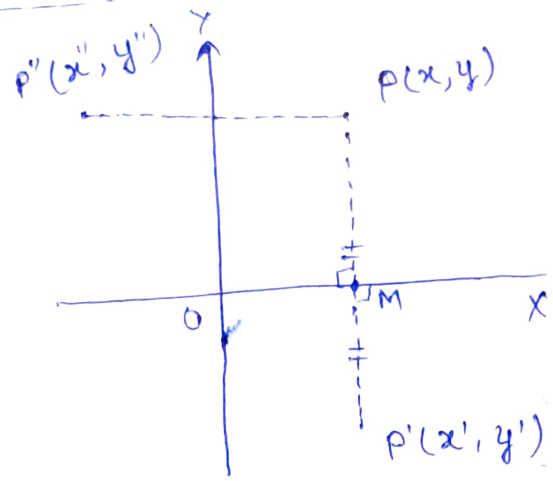
$$y' = S_y \cdot y$$

In matrix form

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

=

4. Reflection :



P(x,y) is a point on 2-D plane and its reflection about x-axis P'(x', y'). with the simple geometery point P and P' are symmetric points.

$$PM = P'M = y$$

Then $x' = x$ → Reflection about OX

$y' = -y$

In matrix form

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

Similarly Reflection OY

$x'' = -x$

$y'' = y$

In Matrix form

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

Transformation doesn't consist only the translation, reflection, scaling, rotation shearing, it may be the combination of few or all of these operation thus to determine the resultant operator is a difficult task because the Matrix operation it not uniform for all type of operation. and so, we develop a new co-ordinate system known as homogeneus co-ordinate system.

Homogeneus co-ordinate system is helpfull to bring uniformatly change matrix.

27-02-24

Homogeneous co-ordinate system :-

$(x,y) \in R^2$

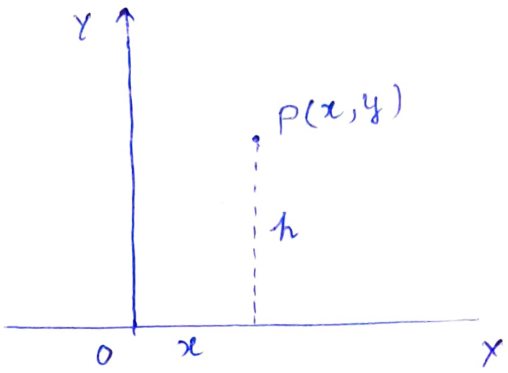
↓

Homogeneous cordinate system

(x_h, y_h, h)

↓

$h \rightarrow$ homogeneous factor



$x_h = x \cdot h$

$y_h = y \cdot h$

$h = 1$

$x = \frac{x_h}{h}$

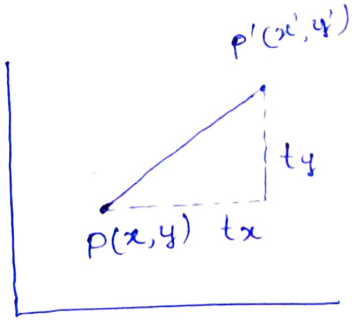
$y = \frac{y_h}{h}$

$(x, y, 1)$

1.) Translation

$x' = t_x + x$

$y' = t_y + y$



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

↳ In matrix form using homogeneous co-ordinate system.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}_{3\times3} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\boxed{P' = T_{(t_x, t_y)} \cdot P}$$

2.) Rotation about origin :

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

↳ In Homogenous system

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

↳ point form

$$P' = R(\theta) \cdot P$$

↳ Rotation operator

3.) scaling :-

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

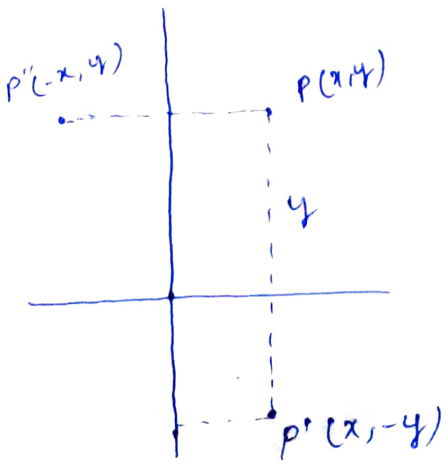
↳ In Homogenous system

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

↳ point form

$$P' = S_{(S_x, S_y)} \cdot P$$

4) Reflection



Reflection about OX (~~x-axis~~)

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

In Homogenous system

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

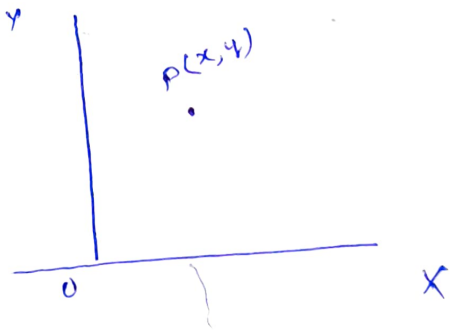
$$\boxed{P' = P_x \cdot P}$$

Reflection about OY

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

$$P' = P_y \cdot P$$

5) Shearing



Shx
Shy
Shearing co-efficient about OX and OY respectively

P(x,y) is point on 2-d space and P'(x',y') is the position of point P after shearing.

then,

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

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

In Matrix form

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left(\begin{bmatrix} 1 & Shx & 0 \\ Shy & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

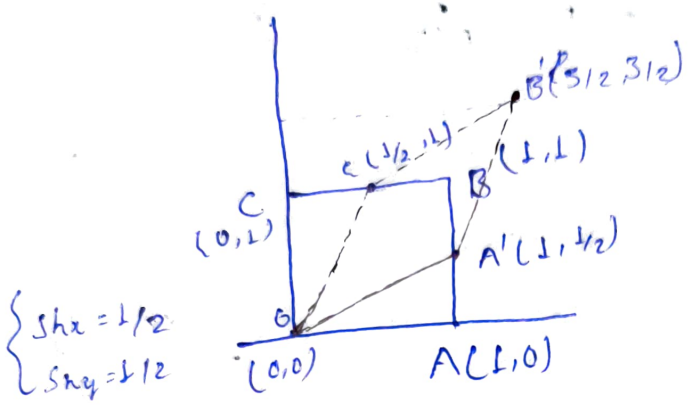
Shearing operator

↳ point form

$$P' = SH(sh_x, sh_y) \cdot P'$$

* Shearing with example (demonstrate)

- Before shearing
- After shearing.



Shearing on O

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

Shearing on A

$$A' = \begin{bmatrix} x' = 1 \\ y' = 1/2 \end{bmatrix}$$

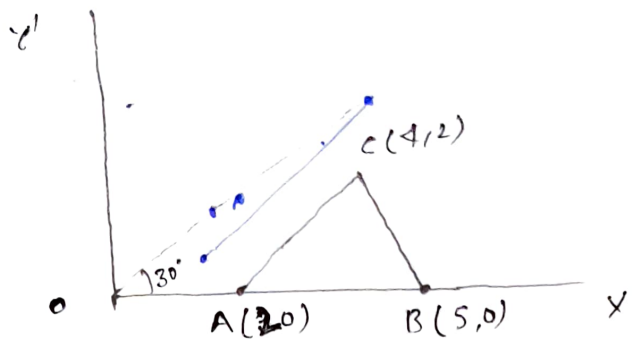
Shearing on B

$$B' = \begin{bmatrix} x' = 3/2 \\ y' = 3/2 \end{bmatrix}$$

Shearing on C

$$C' = \begin{bmatrix} x' = 1/2 \\ y' = 1 \end{bmatrix}$$

Q Rotate the given ΔABC about the origin through an angle 30° in Anticlock direction.



Sol^n

Angle of Rotation +30°

$\cos 30° = \sqrt{3}/2$

$\sin 30° = 1/2$

the rotation operator

$$R(30°) = \begin{bmatrix} \cos 30° & -\sin 30° & 0 \\ \sin 30° & \cos 30° & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 & -1/2 & 0 \\ 1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

After rotation ΔABC changed to $\Delta A'B'C'$

$$A' = R(30°)\,A$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 & -1/2 & 0 \\ 1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \sqrt{3} \\ 1 \\ 1 \end{bmatrix}_{3\times 1}$$

$$A' = (\sqrt{3}, 1) = (1.732, 1)$$

$$B' = R(30°)\,B$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 & -1/2 & 0 \\ 1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 \times 5 \\ 5/2 \\ 1 \end{bmatrix}$$

$$B' = (\sqrt{3}/2 \times 5,\ 5/2) = (4.33,\ 2.5)$$

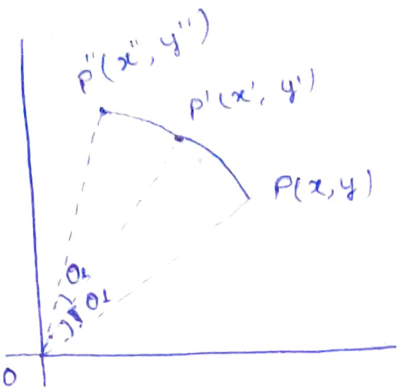
$$C' = R(30°)\,C$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \sqrt{3}/2 & -1/2 & 0 \\ 1/2 & \sqrt{3}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 2\sqrt{3}-1 \\ 2+\sqrt{3} \\ 1 \end{bmatrix}$$

$$C' = (2.464,\ 3.732)$$

Succesive Rotation and scaling:

XY plane
2D (R^2) plane



P(x,y) is a point on 2-d plane which is rotated through an angle θ_1 about origin p'(x',y') is the new position of point P after rotation. p'(x',y') perther roted an angle θ_2 and p''(x'',y'') is the new position of of point then we have.

$$P' = R(\theta_1) \cdot P \quad \text{— (1)}$$

Also $P'' = R(\theta_2) \cdot P'$ — (2)

from (1) & (2) we have

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

Associativity

$$P'' = \underbrace{(R(\theta_2) \cdot R(\theta_1))}_{\text{Operator}} \cdot P$$

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

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

$R(\theta_2) \cdot R(\theta_1) =$

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

θ

$\cos(\theta_1 \pm \theta_2) = \cos\theta_1 \cos\theta_2 \mp \sin\theta_1 \sin\theta_2$

$\sin(\theta_1 \pm \theta_2) = \sin\theta_1 \cos\theta_2 \pm \cos\theta_1 \sin\theta_2$

$-1 \leq \sin\theta \leq 1$

$-1 \leq \cos\theta \leq 1 \quad \forall\ \theta.$

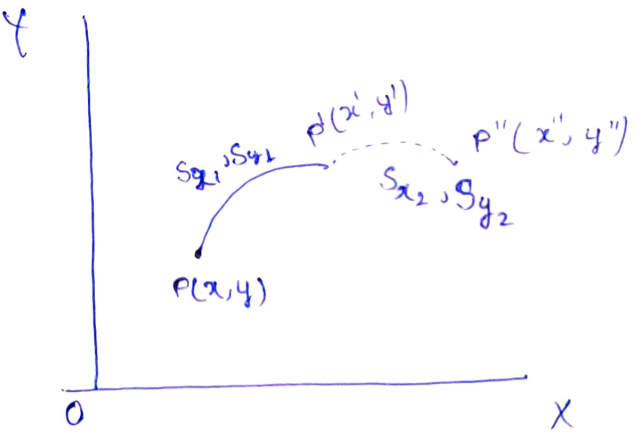
$= R(\theta_2+\theta_1) = R(\theta_1+\theta_2) = R(\theta_1) \cdot R(\theta_2)$

In general for n succesive rotation

$R(\theta_n) \cdot R(\theta_{n-1}) \cdots R(\theta_3) \cdot R(\theta_2) \cdot R(\theta_1) = R(\theta_1+\theta_2+\cdots\theta_n) =$

$R(\theta_1) \cdot R(\theta_2) \cdot R(\theta_n)$

* succesive scaling :-



Let $P(x,y)$ is a point on 2d the plane after scaling factor S_{x_1}, S_{y_1} $P'(x', y')$ is the position of point P, after doing ferther scaling with scaling factor S_{x_2}, S_{y_2} $P''(x'', y'')$ be the position of point P. then,

$$P' = S_{(S_{x_1}, S_{y_1})} \cdot P \quad \text{—— (1)}$$

$$P'' = S_{(S_{x_2}, S_{y_2})} \cdot P' \quad \text{—— (2)}$$

from ① and ② we have

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

$$P'' = \left\{ S_{(S_{x_2}, S_{y_2})} \cdot S_{(S_{x_1}, S_{y_1})} \right\} \cdot P \quad \text{— ③}$$

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

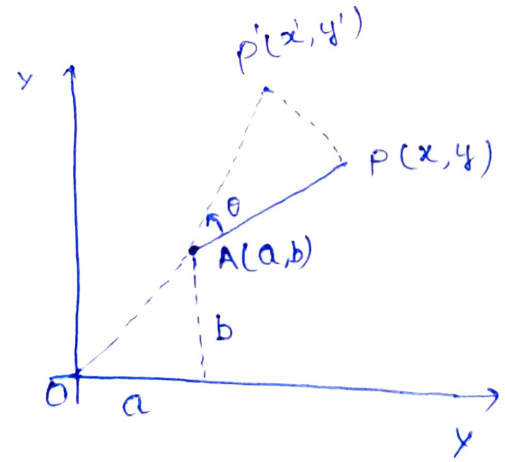
$$\begin{bmatrix} (S_{x_2} \cdot S_{x_1}) & 0 & 0 \\ 0 & (S_{y_2} \cdot S_{y_1}) & 0 \\ 0 & 0 & 1 \end{bmatrix} = S_{(S_{x_2} \cdot S_{x_1},\ S_{y_2} \cdot S_{y_1})} = S_{(S_{x_1} \cdot S_{x_2},\ S_{y_1} \cdot S_{y_2})}$$

In general, for n successive scaling.

$$S_{(S_{x_1}, S_{y_1})} \cdot S_{(S_{x_2} \cdot S_{y_2})} \cdots\cdots S_{(S_{x_n} \cdot S_{y_n})} = S_{(S_{x_1} \cdot S_{x_2} \cdots S_{x_n},\ S_{y_1} \cdot S_{y_2} \cdots S_{y_n})}$$

* Rotation of a 2D object about the point other then origin.

$\bar{OA} = a\hat{i} + b\hat{j}$



From fig point $P(x,y)$ is rotated about point @ $A(a,b)$ through an angle θ in anticlock wise direction & $P'(x',y')$ is a position on p after rotation.

step 1 : Shifting $A(a,b)$ to origin O.

$$T_{-\bar{V}} = \begin{bmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{bmatrix}$$

position vector

step 2 : Rotation about orign $R(\theta)$

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

step 3: shifting point $A(a,b)$ back to its initial position.

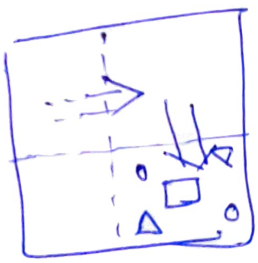
$$T_{\bar{V}} = \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix}$$

So, the resultant operator is

$$R_A(\theta) = T_{\bar{V}}\, R(\theta)\, T_{-\bar{V}}$$

```
#
#include <math.h>
#include <graphics.h>
void main()
{  float xinc, yin; x, y
   int gd = DETECT, gm, xa, ya, xb, yb, dx, dy, step, K
   initgraph("&gd, &gm",
   printf(" Enter first end point")
   scanf("%d, %d", &xa, &yb);
   printf("Enter second end point")
   scanf("
     dx = xb - xa;
     dy = yb - ya;

     if( abs(dx) > abs(dy) )
         step := abs(dx);
     else
         step := abs(dy);

         xin = float(dx/step);

         yin = float (dy/step);

C:\TurboC\     x = xa;
TC\BIN         y = ya;
     putpixel(x, y, RED);

     for( K=1; K<=step; K++)
     {
         x = x + xinc;
         y = y + yinc;
```

```
putpixel(floor(x+0.5), floor(y+0.5), RED);
}
getch();
}
```



Reflection

Q. find reflection of a ΔABC about line $y = x + 2$

Eq of line

$y = x + 2$

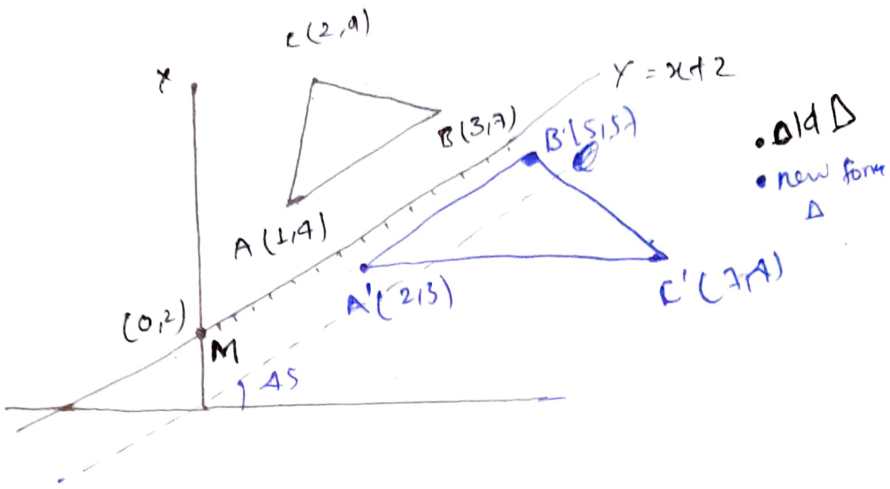
$y = mx + c$

$m = \tan\theta = 1$

$\theta = 45°$

~~$\theta =$~~

$x = 0, y = 2$



from fig we need to find the reflection of ΔABC about the line $y = x + 2$.

To find reflection of Δ about the given line we will follow these steps.

step1: shift point M to origin

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

step2: Rotation of 45° about origin in anticlockwise direction

$\{\cos 45° = \sin 45° = 1/\sqrt{2}\}$

$$R(45°) = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

line align on y-axis

Step:3. Now Reflection about y-axis

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

Step 4: Rotation of 45° in clockwise direction

$$R(-45^\circ) = \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ & 0 \\ -\sin 45^\circ & -\cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$\{\cos(-\theta) = \cos\theta$, $\sin(-\theta) = -\sin\theta\}$

Step 5: Shifting M back to initial position

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

So the resultant operator is

$$\text{Ref}_{y=x+2} \Rightarrow T_{\bar{v}} \cdot R(-45) \cdot \text{Ref}_y \cdot R(45) \cdot T_{-\bar{v}}$$

$$\Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

$$\text{Ref}_{y=x+2} = \begin{bmatrix} -1/\sqrt{2} & \frac{1}{\sqrt{2}} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 2/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} & -2/\sqrt{2} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & -2 \\ 1 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

Let $A'B'C'$ is the reflection of ΔABC about the line $y=x+2$ then

$$A' = \begin{bmatrix} 0 & 1 & -2 \\ 1 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \underbrace{\begin{bmatrix} 1 \\ 4 \\ 1 \end{bmatrix}}_{A} = \begin{bmatrix} 2 \\ 3 \\ 1 \end{bmatrix} = (2,3)$$

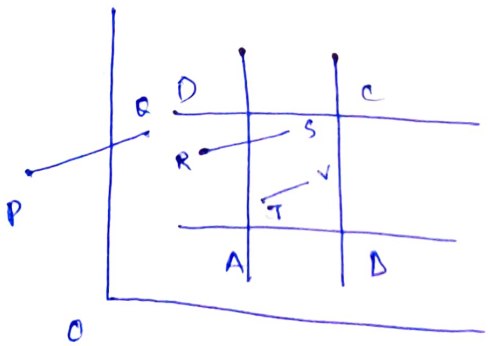
$$B' = \begin{bmatrix} 0 & 1 & -2 \\ 1 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3 \\ 7 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ 1 \end{bmatrix} = (5,5)$$

$$C' = \begin{bmatrix} 0 & 1 & -2 \\ 1 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 9 \\ 1 \end{bmatrix} = \begin{bmatrix} 7 \\ 4 \\ 1 \end{bmatrix} = (7,4)$$

Clipping clipping 2-D

In 2-D is a process to determine what portion of geometric object lies within a specifyed region. This specifyed region is known as clip window. In process of cliping we keep that portion of geometic object which lies inside the clip or on the clip windown and delete that portion is out of that cliping.

In general clip window is an recaglar shap.



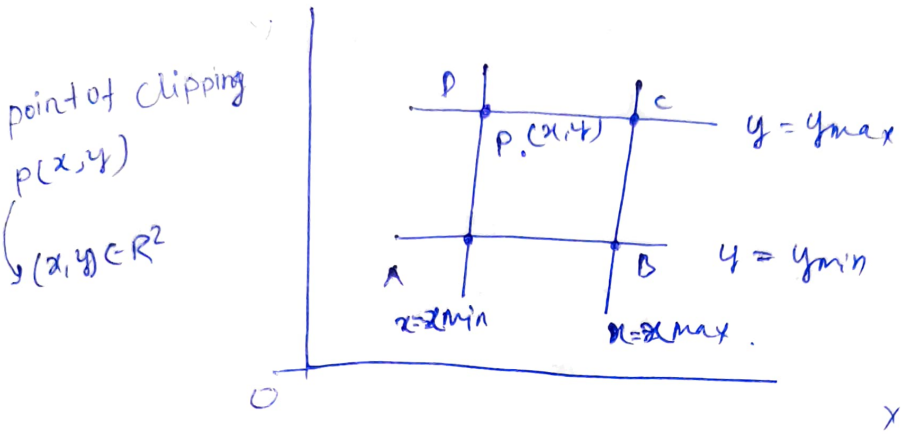
from ABCD is the clip window

Line cliping

In the fig line PQ is completelly outside of clip window line T completelly inside the clip window. and some portion of line RS is inside the clip window and some are outside.

point clipping

In point clipping a point $P(x,y)$ is given to us and we want to check the position of point P with respect to the clipp window ABCD.



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

point P(x,y) lies on 2-D plane and ABCD is clip window if point P satisfy inequality set {1} then point P is either inside of clip window.
Otherwise the point will be out of clipping.

* **The line clipping :-**

In line clipping we check position of line with respect to a given clip window.

* Coham suther land algorithms :-

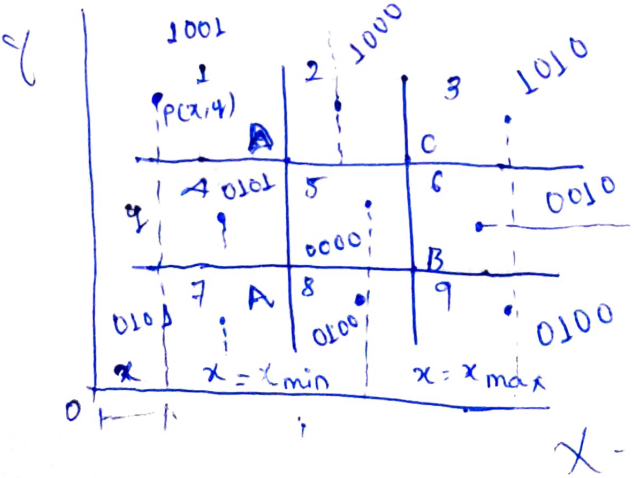
There are three category of the line segment with respect to a given clip window.

1) Line completly out of clip window. during the process of clipping we will discard the line.

2) Line segement compeletly inside the clip window. There will no change in line when we applied the clipping.

3) clipping candidate line -
If some portion of line is inside the clip window and some outside ~~some~~ such line are known as clipping candidate line.

- Coham suther algo. divides the 2-D plane in 9 region



and asign four bit code each of these region.

The bit asignment scheme.

B_1 : Algorithms compute $a=(y-y_{max})$ if $a>0$. $B_1 \equiv 1$ else $B_1 \equiv 0$

B_2 : Algorithms compute $a=(y_{min}-y)$ if $a>0$; $B_2 \equiv 1$ else $B_2 \equiv 0$.

B_3 : Algorithms compute $a=(x-x_{max})$ if $a>0$, $B_3 \equiv 1$ else $B_3 \equiv 0$.

B_4 : Algorithms compute $a=(x_{min}-x)$ if $a>0$ $B_4 \equiv 1$ else $B_4 \equiv 0$.

4 bit code for R_E (1)

$B_1 \equiv a=(y-y_{max}) > 0$, $B_1 \equiv 1$

$B_2 \equiv a=(y_{min}-y) < 0$, $B_2 \equiv 0$

$B_3 \equiv a=(x-x_{max}) < 0$, $B_3 \equiv 0$

$B_4 \equiv a=(x_{min}-x) > 0$ $B_4 \equiv 1$

4 bit code for R_E (5) (clip window)

$B_1 \equiv a=(y-y_{max}) \not> 0$, $B_1 = 0$

$B_2 \equiv a=(y_{min}-y) < 0$, $B_2 = 0$

$B_3 \equiv a=(x-x_{max}) < 0$, $B_3 = 0$

$B_4 \equiv a=(x_{min}-x) < 0$, $B_4 = 0$

4-bit code for (2)

$B_1 = a = (y - y_{max}) > 0, B_1 = 1$

$B_2 = a = (y_{min} - y) < 0, B_2 = 0$

$B_3 = a = (x - x_{max}) < 0, B_3 = 0$

$B_4 = a = (x_{min} - x) < 0, B_4 = 0$

4-bit code for (3)

$B_1 = a = (y - y_{max}) > 0, B_1 = 1$

$B_2 = a = (y_{min} - y) < 0, B_2 = 0$

$B_3 = a = (x - x_{max}) > 0, B_3 = 1$

$B_4 = a = (x_{min} - x) < 0, B_4 = 0$

4-bit code for (4)

$B_1 = a = (y - y_{max}) < 0, B_1 = 0$

$B_2 = a = (y_{min} - y) > 0, B_2 = 1$

$B_3 = a = (x - x_{max}) < 0, B = 0$

$B_4 = a = (x_{min} - x) > 0, B = 1$

4-bit code for (6)

$B_1 = a = (y - y_{max}) < 0, B_1 = 0$

$B_2 = a = (y_{min} - y) < 0, B_2 = 0$

$B_3 = a = (x_B - x_{max}) > 0, B_3 = 1$

$B_4 = a = (x_{min} - x) < 0, B_4 = 0$

4-bit code for (7)

$B_1 = a = (y - y_{max}) < 0, a = 0$

$B_2 = a = (y_{min} - y) > 0, a = 1$

$B_3 = a = (x - x_{max}) < 0, a = 0$

$B_4 = a = (x_{min} - x) > 0, a = 1$

4-bit code for (8)

$B_1 = a = (y - y_{max}) < 0, a = 0$

$B_2 = a = (y_{min} - y) > 0, a = 1$

$B_3 = a = (x - x_{min}) < 0, a = 0$

$B_4 = a = (x_{min} - x) < 0, a = 0$

4-bit code for (9)

$B_1 = a = (y - y_{max}) < 0, a = 0$

$B_2 = a = (y_{min} - y) > 0, a = 1$

$B_3 = a = (y_B - x_{max}) < 0, a = 0$

$B_4 = a = (x_{min} - x) < 0, a = 0$

After asigning the 4-bit code each of the 9 region algo. determines the category of line segment with respect to clip window by using the concept.

Case I

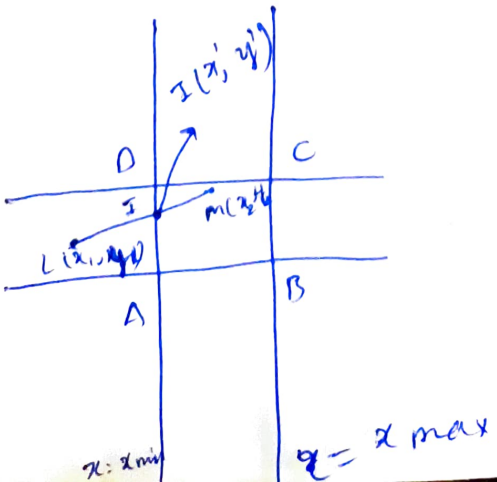
If 4-bit code for both end point of the line segment are 0000 then the line is completly inside the clip window.

case II

If 4
Both the end point of bit code are not 0000 then algorithms determines the bitwise logical AND of 4-bit codes of the end point of line segment. If it is 0000 then line is clipping cadidate.

$$\begin{array}{r} 1001 \\ 1110 \\ \hline 1000 \end{array}$$ x not zero

The 4 bit code of the end points of line will help us to determines with each boundary of clip window the line is intersecting.



If ~~capital~~ I is point of intersection of the line Lm of the left boundary of clip window then -

Eq of Lm

$$y - y_1 = m(x - x_1) \qquad m = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)$$

$$y - y_1 = \frac{(y_2 - y_1)}{(x_2 - x_1)} (x - x_1)$$

clearly $x' = x_{min}$

$$y' = y_1 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x' - x_1)$$

$$\boxed{y' = y_1 + \left(\frac{y_3 - y_1}{x_2 - x_1}\right)(x_{min} - x_1)}$$

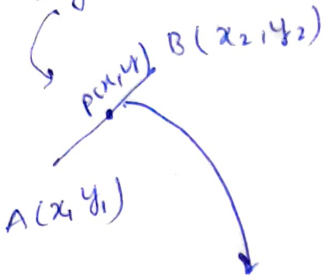
With help of these eq we can determine the point of intersection I, protecting the line from I to m and deleting the rest, will give us clip line.

* Liang Bansky line clipping algorithms

Eqⁿ of line

$ax + by + c = 0$

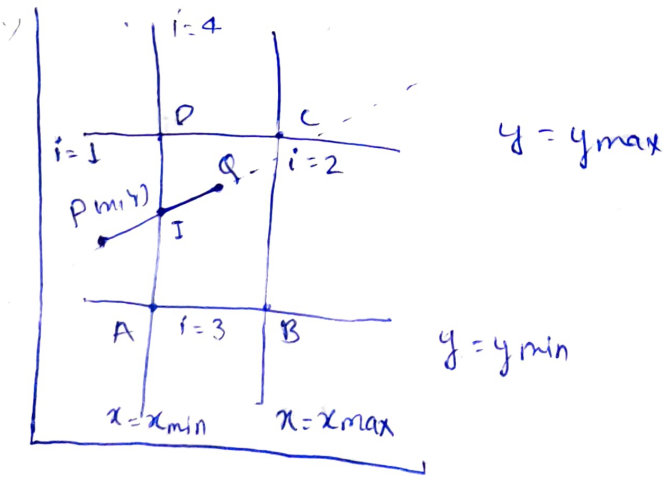
$y = mx + c$



parametic eq of line

$$x = x_1 + u(x_2 - x_1)$$
$$y = y_1 + u(y_2 - y_1) \quad \} \ (1)$$

where $0 \leq u \leq 1$



Linang Bansky Algo uses the concept of point clipping to clip a given line against the clip window. The point clipping is

$$x_{min} \leq x \leq x_{max}$$
$$y_{min} \leq y \leq y_{max} \quad \} \ (2)$$

from eq (1) and (2) we have

$$x_{min} \leq x_1 + u\Delta x \leq x_{max}$$

$$y_{min} \leq y_1 + u\Delta y \leq y_{max}$$

$$x_{min} \leq x_1 + u\Delta x$$

$$x_{min} - x_1 \leq u\Delta x$$

$$-u\Delta x \leq x_1 - x_{min} \quad \text{— (1)}$$

Again

$$x_1 + u\Delta x \leq x_{max}$$

$$u\Delta x \leq x_{max} - x_1 \quad \text{— (2)}$$

and

$$-u\Delta y \leq y_1 - y_{min} \quad \text{— (3)}$$

$$-u\Delta y \leq y_{max} - y_1 \quad \text{— (4)}$$

In eq 1, 2, 3, 4 in general form

$$u\,x\,d_i \leq q_i \qquad \text{for } i = 1, 2, 3, 4$$

$d_1 = -\Delta x$	$q_1 = x_1 - x_{min}$
$d_2 = \Delta x$	$q_2 = x_{max} - x_1$
$d_3 = -\Delta y$	$q_3 = y - y_{min}$
$d_4 = \Delta y$	$q_4 = y_{im} - y_1$

if $d_i = 0$, then the line will be parallel to ith boundary.

Again $q_i < 0$, then point $P(x_1, y_1)$ out of ith boundary.

if $q_i = 0$, then point $P(x, y)$ will lie on i^{th} boundary.

if $q_i \neq 0$, then point $P(x, y)$ is ~~inside~~ inside the i^{th} boundary.

To find the point of intersection of the line segement of boundary of clip window we need corresponding parametric value. we use this eq

$$u = \frac{q_i}{d_i} \quad \text{where } d_i \neq 0 \text{ for } i = 1, 2, 3 \dots N.$$

This eq will give us the value of u those value of u which are out of 0 to 1 interval.

11-03-24

To discard those value of u for which line intersect with the extended boundry lines of clip window, when u lies blw 0 or 1, ₹

To find those value we take Max of lower set of parametric value(u_1) or Min of upper set of parametric value. (u_2).

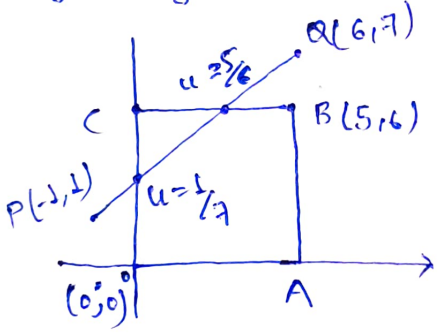
→ if $(u_1 > u_2)$ then the line is out of clip window

→ if $u_1 = 0.0$ and $u_2 = 1.0$ then the line is inside the clip window.

→ if $(u_1 < u_2)$ then we compute the point of intersection for $u = u_1$ and for $u = u_2$ and protecting the line b/w the point of intersection.

In this way clipping is achive.

Q Clip the given line using L-B linecliping Algorithm.



Sol - The parametric eq of $\overline{PQ}$

$x_1 = -1, \quad x_2 = 6 \quad \Delta x = x_2 - x_1 = 6 - (-1) = 7$

$y_1 = 1 \quad y_2 = 7 \quad \Delta y = y_2 - y_1 = 7 - 1 = 6$

$d_1 = -\Delta x = -7 \quad q_1 = x_1 - x_{min} = -1 - 0 = -1$

$d_2 = \Delta x = 7 \quad q_2 = x_{max} - x_1 = 5 - (-1) = 6$

$d_3 = -\Delta y = -6 \quad q_3 = y_1 - y_{min} = 1 - 0 = 1$

$d_4 = \Delta y = 6 \quad q_4 = y_{max} - y_1 = 6 - 1 = 5$

not value is 0 so line is not parallel.

$$u = \frac{q_1}{d_1} = \frac{-1}{-7} = \frac{1}{7}$$

$$u = \frac{q_2}{d_2} = \frac{6}{7}$$

$$u = \frac{q_3}{d_3} = \frac{1}{-6} = \frac{-1}{6}$$ (discard the value of b/c less than 0.)

$$u = \frac{q_4}{d_4} = \frac{5}{6} = \frac{5}{6}$$

note

lowerset means d is negative.

(d_1, d_3)

upper set mans d is maximus.

(d_2, d_4)

$$\Rightarrow u_1 = \max\left\{0, \frac{1}{7}\right\} = \frac{1}{7}$$

$$u_2 = \min\left\{1, \frac{6}{7}, \frac{5}{6}\right\} = \frac{5}{6}$$

$$u_1 = \frac{1}{7} \quad u_2 = \frac{5}{6}$$

The parametic eq of line PQ.

$$x = x_1 + u\Delta x$$

$$x = -1 + u\,7 = -1 + 7u$$

$$y = y_1 + u\Delta y = 1 + u6$$

for point I_1 $(u = \frac{1}{7})$

$$x = -1 + 7 \times \frac{1}{7} = -1 + 1 = 0$$

$$y = 1 + 6 \times \frac{1}{7} = 1 + \frac{6}{7} = \frac{13}{7} = 1.857$$

$$I_1 \equiv (0, 1.857)$$

for point I_2 $(u = \frac{5}{6})$

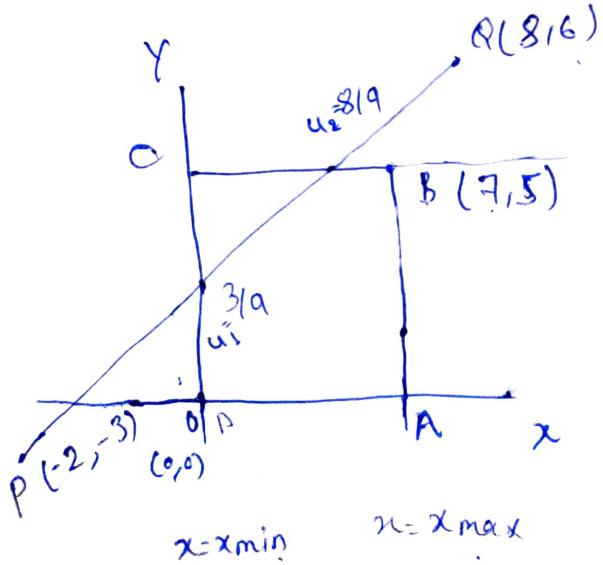
$$x = -1 + 7 \times \frac{5}{6} = \frac{29}{6} = 4.83$$

$$y = 1 + 6 \times \frac{5}{6} = 6$$

$$I_2 \equiv (4.83, 6)$$

protecting the line b/w the point of intersection I_1 and I_2 and deleting the remaning part of the line gives us the clip line.

Q.



8 The parametric eq of line PQ

$x_1 = -2$ $\quad x_2 = 8$ $\quad \Delta x \Rightarrow x_2 - x_1 = 8-(-2) = 10$

$y_1 = -3$ $\quad y_2 = 6$ $\quad \Delta y \Rightarrow y_2 - y_1 = 6-(-3) = 9$

$d_1 = -\Delta x \Rightarrow -10$ $\quad q_1 = x_1 - x_{min} \Rightarrow -2-0 = -2$

$d_2 = \Delta x \Rightarrow 10$ $\quad q_2 = x_{max} - x_1 \Rightarrow 7-(-2) = 9$

$d_3 = -\Delta y \Rightarrow -9$ $\quad q_3 = y_1 - y_{min} \Rightarrow -3-0 = -3$

$d_4 = \Delta y \Rightarrow 9$ $\quad q_4 = y_{max} - y_1 \Rightarrow 5-(-3) = 8$

$u \Rightarrow \frac{q_1}{d_1} = \frac{-2}{-10} = \frac{2}{10}$

$u \Rightarrow \frac{q_2}{d_2} = \frac{9}{10} = \frac{9}{10}$

$u = \frac{q_3}{d_3} = \frac{-3}{-9} = \frac{3}{9}$.

$u = \frac{q_4}{d_4} = \frac{8}{9} = \frac{8}{9}$

---

max lower parametire (d_1, d_3)

min upper parametric (d_2, d_4).

$\Rightarrow u_1 = \max(0, \frac{2}{10}, \frac{3}{9}) \Rightarrow \frac{3}{9}$

$\Rightarrow u_2 = \min(1, 9/10, 8/9) \Rightarrow 8/9$

$u_1 = 3/9, \quad u_2 = 8/9$

The parametric eq of line PQ

$x = x_1 + u\Delta x$

$x = -2 + u\,10 \Rightarrow -2+10u$

$y = y_1 + u\Delta y$

$-3 + u\,9 \Rightarrow -3+9u$

for point I_1 $(u = 3/9)$

$x = -2 + 10 \times \frac{3}{9} \Rightarrow -2 + \frac{30}{9} = 1.33$

$y = -3 + 9u$

$-3 + 9 \times \frac{3}{9} = 0$

$I_1 \equiv (1.33, 0)$

$\bar{x} \quad \bar{y}$

for point I_2 $(u = 8/9)$

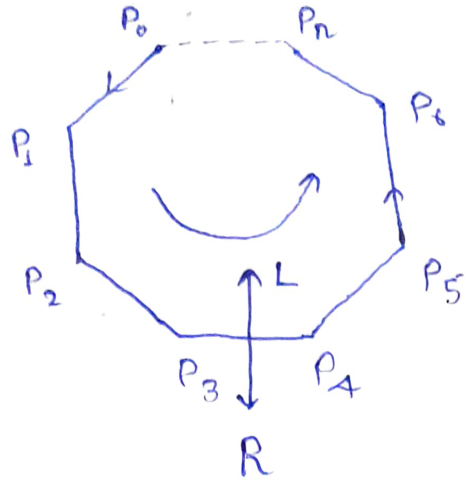
$x = -2 + 10 \cdot \frac{8}{9} \Rightarrow -2 + \frac{80}{9} \Rightarrow -2 + 8.88 = 6.88$

$y = -3 + 9 \times \frac{8}{9} = 5$

$I_2 \equiv (6.88, 5)$

11-03-24

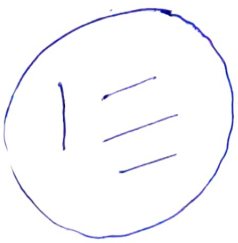
* polygon clipping :-



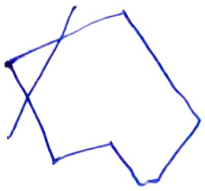
⇒ positively orientation polygon.

⇒ When we traverse vertices of polygon in anti-clock direction (positively orientation) the left away indicate inside the polygon and right indicate outside.

⇒ Convex polygon



convex polygon



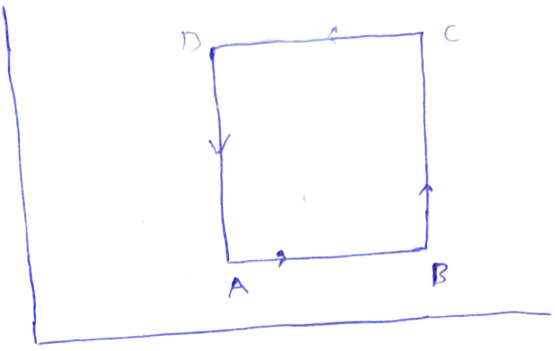
Non convex

A polygon said to convex if we take any two discitnt point in a polygon and the line joing those point which lies inside the polygon, otherwise the polygon is non-convex polygon.

12-03-24

Sutherland Hodgeman polygon

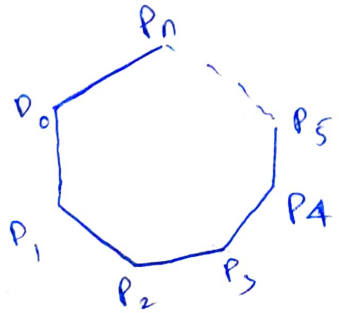
clipping :-



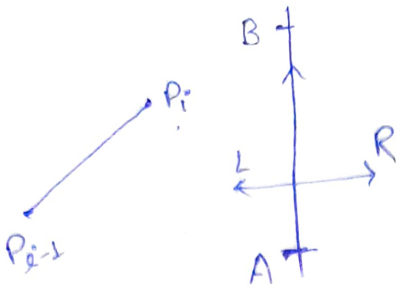
Sutherland Hodgeman polygon clipping is based on concept of line clipping algo. pass the polygon to each edge of positively oriented clip window. after processing the polygon edges with respect to a particular edge of clip window, the partially cliped polygon is passed to the next edge of the clip window.

When the processing with respect to all edges of the clip window is over we get the cliped ~~path~~ polygon.

The processing scheme :-



case-I

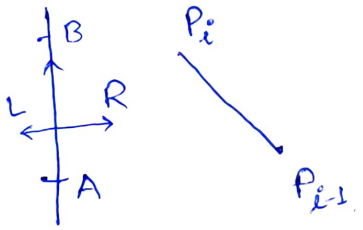


Vet
V.O.L

While processing the polygon edges with respect to a particular edge of clip window four case may arise.

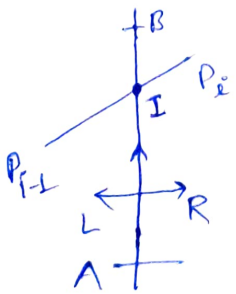
case-I :- when P_{i-1} P_i both end points lies in the left of edges AB then we place P_i in vertex output list.

case-II



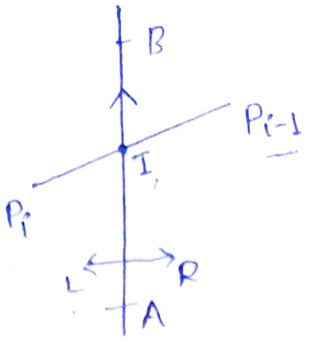
case II When P_{i-1} P_i both are in Right of edge AB then nothing will placed in vertex output list.

case III



When P_{i-1} is in left and P_i in the right. then the point of intersection I will be placed in vertex output list.

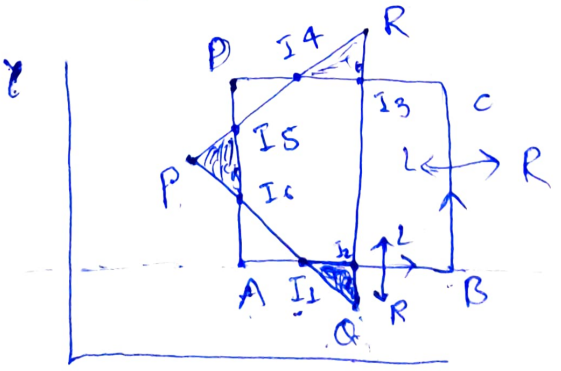
case-IV



When P_i is left and P_{i-1} is right then P_i and I both will placed in vertex output list.

Then the partially cliped polygon is pass to the next consecutive (Adjecent) edge of clip window. When processing with respect to all four edges of the clip window is completed then we get clip polygon.

Q clip the given ploygon using S.H. polygon clipping.



ABCD is the clip window and PQR is the polygon which we want to clip. The initial vertex input list.

[P Q R]

V. O

Taking Edge $\overline{AB}$ of clip window

Edge	$\overline{AB}$	vertex output point
P Q	case - III	I_1
Q R	case IV	I_2, R
R P	case I	P

V.O.L = $[I_1, I_2, R, P]$

* $[I_1, I_2, R, P]$ input list

Again for Edge $\overline{BC}$

Edge	BC	vertex output
$I_1 I_2$	case - I	I_2
I_2, R	case - I	R
R, P	case - I	P
P I_1	case - I	I_1.

V.O.L = $[I_2, R, P, I_1]$

* $[I_2, R, P, I_1]$

Again for edge $\overline{CD}$

Edge	CD	vertex output
I_2 R	case - III	I_3
R P	case IV	I_4, P
P I_1	case I	I_1
$I_1 I_2$	case I	I_2

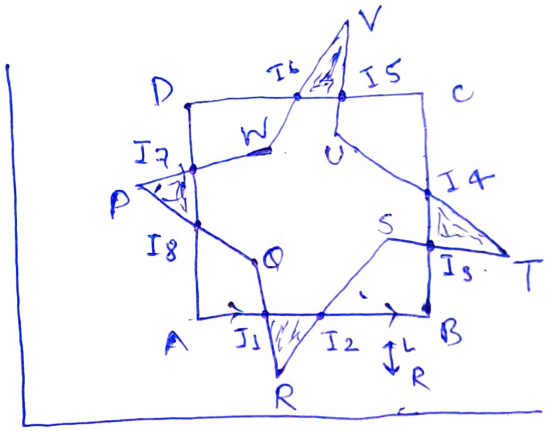
V.O.L. $[I_3, I_4, P, I_1, I_2]$

finally Edge $\overline{DA}$

Edge	$\overline{DA}$	vertex output
$I_3\ I_4$	case I	I_4
$I_4\ P$	case- III	I_5
$P\ I_1$	case- IV	I_6, I_1
$I_1\ I_2$	Case I	I_2
$I_2\ I_3$	Case I	I_3

V·O·L = $[I_4, I_5, I_6, I_1, I_2, I_3]$

Q



ABCD is the clip window and PQRSTUVW is the polygon which want to clip, the initial vertex.

[PQRSTUVW]

Taking Edge $\overline{AB}$ of clip window.

Edge	$\overline{AB}$	Vertex output
PQ	case I	Q
QR	case- III	I_1
RS	case IV	I_2, S
ST	case I	T
TU	case I	U
UV	case I	V
VW	case I	W
WP	case I	P

V.O.L = [Q, I_1, I_2, S, T U V W P]

Again taking Edge $\overline{BC}$ of clip window

Edge	Case	AB	vertex output
Q I_1		I	I_1
I_1 I_2	))	I	I_2
I_2 S	))	I	S
S T	))	III	I_3
T U	))	IV	I_4, U
U V	))	I	V
V W	))	I	W
W P	))	I	P
P Q	))	I	Q

V.O.L [I_1 I_2 S I_3 I_4 U V W P Q]

Again taking Edge $\overline{CD}$ of clip window

Edge	case	$\overline{CD}$	vertex output
I_1 I_2		I	I_2
I_2 S	))	I	S
S I_3	))	I	I_3
I_3 I_4	))	I	I_4
I_4 U	))	I	U
U V	))	III	I_5
V W	))	IV	I_6 W
W P	))	I	P
P Q	))	I	Q
Q I_1	))	I	I_1

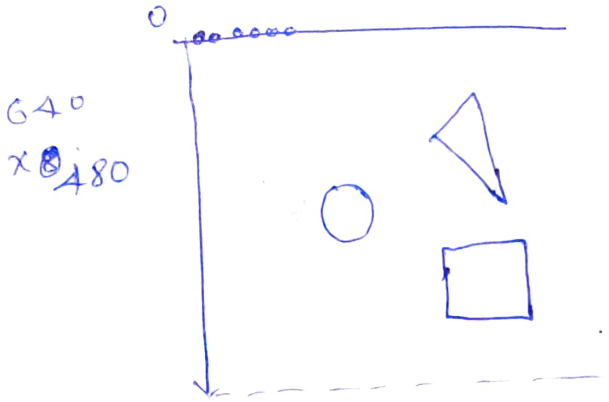
V.O.L = [I_2 S I_3 I_4 U I_5 I_6 W P Q I_1]

finally we take Edge $\overline{DA}$

Edge		DA	vertex Output
I_2 S	case.	I	S
S I_3	))	I	I_3
I_3 I_4	))	I	I_4
I_4 U	))	I	U
U I_5	))	I	I_5
I_5 I_6	))	I	I_6
I_6 W	))	I	W
W P	))	III	I_7
P Q	))	IV	I_8, Q
Q I_1	))	I	I_1
I_1 I_2	))	I	I_2

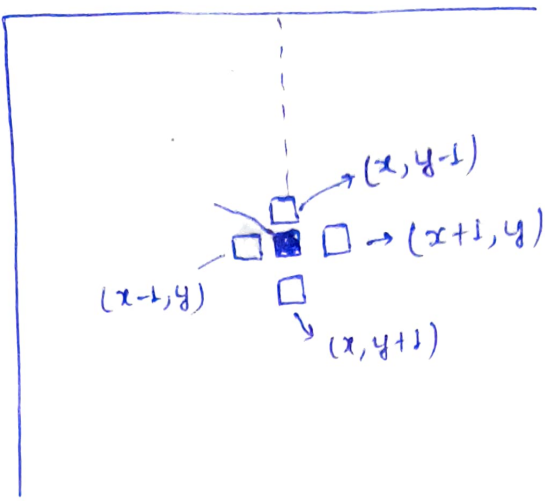
V.O.L = [S I_3 I_4 U I_5 I_6 W I_7 I_8, Q I_1 I_2]

Filling of polygon (Area filling algorithms)



* We have three popular algorithm for polygon filling algo.-
* Boundry filling : algorith and flood fill algorithms. are simple and less complex so can be use easily with complex polygon edges, The scan-line filling algorithms is computanally expensive, requires to compute the point of intersection of scan line with the edges of polygon, scan-line method is suitable for less complex polygon boundry.

* Boundry fill Algorithms :-



Boundry fill algo. is aplicble

Algo. starts from interior pixeel and by using four connected approach

step 1 :→ Read the interior point (x,y).
step 2 :→ Read the boundary color & fill-color.
step 3 :→ If the current color pixeel is in fill color or in boundary color then goto step9.
step 4 :→ paint pixel (x,y) with fill color.
step 5 :→ goto step:2 with pixel patter (x+1, y)
step 6 :- " " (x-1, y)
step 7 : " " (x, y+1)
step 8 " " (x, y-1)
step 9 Stop the proces.

* flood fill- Algorithm:

flood fill algo. is suitable when the boundry of color is multicolor. This algorithm start with interior points and paint outword using four connected approach. It paints pixel which exist in old color and continue the process till all the pixels in old color has not been not painted.

flood fill Algo.

Step 1: Read the interior point (x,y).

Step 2: Read the old color and fill-color.

Step 3: If the current pixel is in fill color then goto step 9.

Step 4: paint pixel (x,y) with fill color.

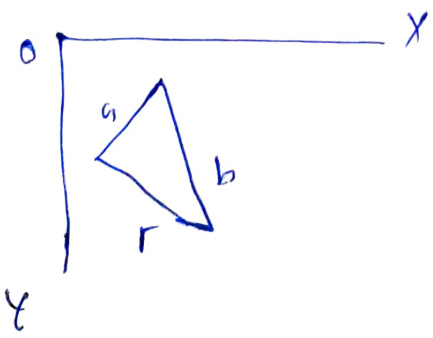
Step 5: goto step 2 with pixel position (x+1, y)

Step 6 " " (x-1, y)

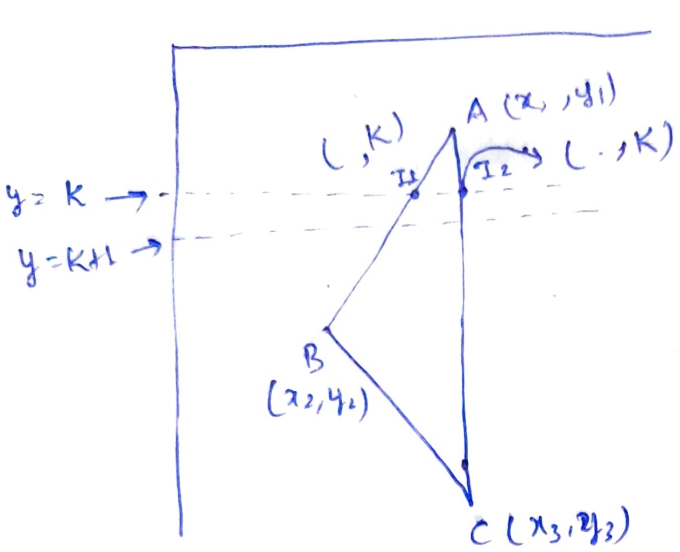
Step 7 " " (x, y+1)

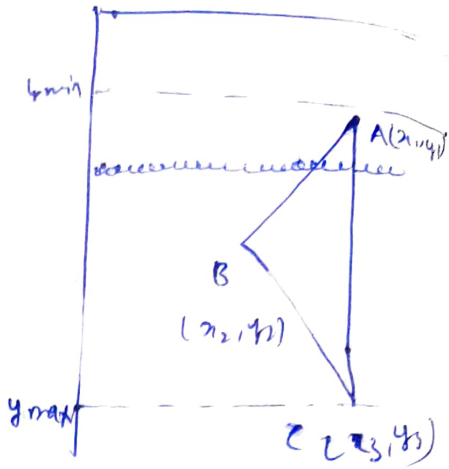
Step 8 " " (x, y-1)

Step 9 Stop process.



* Scan-line Algorithms :-





$$\text{Slop of AB} = \left(\frac{y_2 - y_1}{x_2 - x_1}\right) = m$$

slope of AB = slop of $I_1, I_2 = m$

$$\Delta x = (x_{K+1} - x_K)$$

$$\Delta y = K+1 - K = 1$$

$$m = \frac{1}{x_{K+1} - x_K} = x_{K+1} - x_K = 1/m$$

$$\left[x_{K+1} = x_K + 1/m\right]$$

If initial value x_0

$$x_1 = x_0 + 1/m$$

$$x_2 = x_1 + 1/m$$

$$x_2 = x_0 + 2/m$$

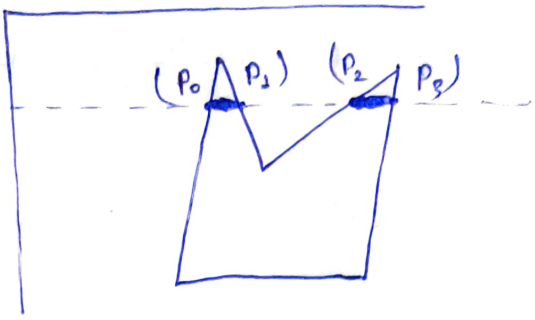
$$\vdots$$

$$x_K = x_0 + K/m$$

$$x_{K+1} = x_0 + K+1/m$$

* scan-line fill algorithms.

1. Intersection of scan line with polygon edges.
2. Fill b/w pair of intersection.



3. for $y = y_{min}, y_{max}$
   i) Intersect scan line y with each edge of polygon.
   ii) Sort point of intersection in increasing x [P_0 P_1 P_2 P_3]
   iii) fill pair wise
   $P_0 - P_1$, $P_2 - P_3$

- However we need to handle some special cases.

- Special Handle :-

a) Make sure we only fill the interior pixels.

Define interior :-

for a given pair of intersection point.

(x_i, y) (x_j, y)

→ Fill ceiling (x_i) to floor of (x_j).

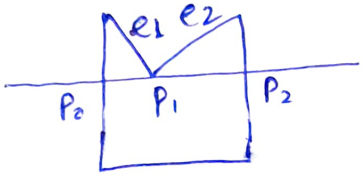
Important when we have polygon adjecent to each other.

b) ~~If~~ a intersection has integer x-co-ordinate.

(i) If x_i integer define interior (point)

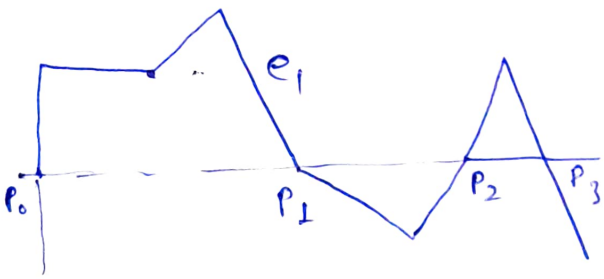
ii) If x_j is integer define it ~~exterior~~ (not exterior. fill)

c) Intersection is an edge^end point, then intersection point are $(P_0 P_1 P_2)$



→ then $(P_0 P_1 P_1 P_2)$ Filling $(P_0 \to P_1, P_1 \to P_2)$

In fact we compute the intersection of ~~Co~~ scan-line Sparately we get intersection point ~~P~~ towise keep point $P_1 P_2$.



Intersection is an ~~end~~ edge and point however in this case we donot want to count P_1 twice.