Chapter-2

A picture can be drawn on the Screen with the help of geometric structure such as point, line, circles etc. To draw these output primitives, graphics programming packages provides variety of functions. some of O/P Primitives

1. Point
2. Line
3. Circle
4. Polygons
5. Splines curves

Scan Conversion → The picture defn is stored in the refresh buffer of display device. This picture is read by the video controller of computer and acc to voltage applied to control grid and deflection plate is varied to get on the desired pixel. This process of plotting the objects on screen is called scan conversion

Scan converting the point →

1. (x, y) are real numbers within an image area, needs to be scan converted to a pixel at location x', y'.

x' → integer part of x

y' → integer part of y

For eg →

(1.7, 0.6) ⇒ (1, 0)

(2.2, 1.3) ⇒ (2, 1)

(2.8, 1.7) ⇒ (2, 1)

Pixel

② $(x, y) \Rightarrow x' = \text{floor}(x + 0.5)$

$y' = \text{floor}(y + 0.5)$

All points satisfy $x' - 0.5 < x < x' + 0.5$ &

$y' - 0.5 < y < y' + 0.5$ are mapped to pixel (x, y).

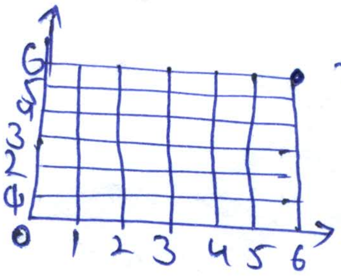
$(2.2, 1.3) \Rightarrow x' = (2.2 + 0.5) = 2.7$

$y' = (1.3 + 0.5) = 1.8$

$\underline{1.7 < 2.2 < 2.7}$

$(2.8, 1.9) \Rightarrow x' = 2.8 + 0.5 = 3.3$

$y' = 1.9 + 0.5 = 2.4$



x_{max}, y_{max}

Resolution = 6 × 6

Scan converting the straight line →

General equation of line

$$(y - y_1)/(x - x_1) = (y_2 - y_1)/(x_2 - x_1)$$

$$y = mx + b \quad \text{—— (I)}$$

$m \rightarrow$ slope $= \dfrac{y_2 - y_1}{x_2 - x_1} = \dfrac{\Delta y}{\Delta x}$

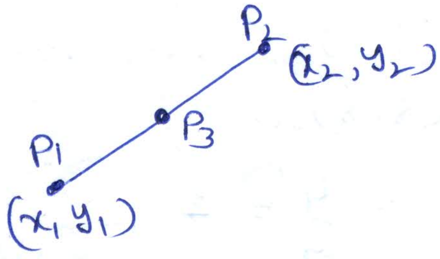
$b \rightarrow$ y intercept

$$m = \frac{\Delta y}{\Delta x} \quad \text{—— (II)}$$

$$\Delta y = m \cdot \Delta x \quad \text{—— (III)}$$

for any given x interval along a line we can compute corresponding y interval Δy.

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



$y = mx + b$

$P_3 \rightarrow (x_3, y_3)$ lies on the segment if

$y_3 = mx_3 + b$

Method of line drawing →

(I) DDA (Digital Differential Analyzer)

(II) Bresenham line Algorithm

(I) DDA → In line drawing algorithm, the line is started with start point, then an increment is added with the previous ~~stat~~ value, this process continues until the line of required length is obtained.

$$y = mx + c \qquad m = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1}$$

$$c = y_1 - mx_1$$

for given Δx interval, we can compute

$$\Delta y = m \cdot \Delta x$$

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

Once the values of interval are knowns,

(1) if m is positive & $m < 1$ or $m = 1$ then $\Delta x = 1$ we will compute Δy

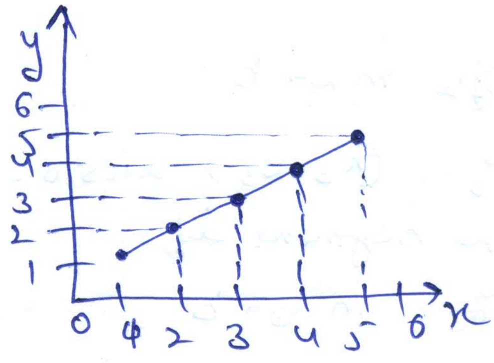
(2) If m is positive & $m > 1$ the $\Delta y = 1$ then we will calculate Δx.

Both these assumption are made when lines to be processed from left end point to right end point.

$(x_1, y_1) = (1, 1)$

$(x_2, y_2) = (5, 5)$

$m = \dfrac{5-1}{5-1} = \dfrac{4}{4} = 1$



$m = \dfrac{\Delta y}{\Delta x}$

$\Delta y = m \cdot \Delta x$

when $\Delta x = 1$, $\Delta y = 1 \cdot 1$ | $\Delta x = 2$, $\Delta y = 2$ | $\Delta x = 3$, $\Delta y = 3$

Program
~~Algorithm~~ →

```
#include "device.h"
#define Round(a) ((int)(a+0.5))

void lineDDA(int xa, int ya, int xb, int yb)
{
  int dx= xb-xa;
  int dy= yb-ya;
  int setps, k;
  float xincr, yincr, x=xa, y=ya;
  if(abs(dx)>abs(dy))
    steps= abs(dx);
  else
    seteps =abs(dy);
  xincr= dx/(float)steps;
  yincr= dy/(float) steps;
  setpixel (Round(x), Round(y));
  for (k=0; k<steps; k++)
  {
     x= x+ xincr;
     y= y+ yincr;
   setpixel (Round(x), Round(y));
  }
}
```

Algorithm →

1. Input two endpoint co-ordinates (x_1, y_1) & (x_2, y_2)
2. Calculate the value of dx and dy

$$dx = x_2 - x_1$$
$$dy = y_2 - y_1$$

3. set the running coordinates (x, y) to the starting value (x_1, y_1)
4. Determine the length of a line

   if abs(dx) ≥ abs(dy) then
      set length = abs(dx)
   else
      set length = abs(dy)

5. calculate the increamental values in x and y directions as

$$\Delta x = \frac{dx}{length}$$

$$\Delta y = \frac{dy}{length}$$

6. Plot the point at current (x, y) location
7. Calculate the Δx and Δy for subsequent points as

```
i = 1
while (i ≤ length)
{
  Plot (integer(x), integer(y))
  x = x + Δx;
  y = y + Δy;
  i = i + 1;
}
```

DDA

~~Bresenham's~~ Line Algorithm →

The endpoints of a line are (0,0) and (4,4). use DDA algorithm to rasterize the line.

$x_1 = 0 \quad x_2 = 4$

$y_1 = 0 \quad y_2 = 4$

$dx = 4 - 0 = 4$

$dy = 4 - 0 = 4$

$dx = dy$ so length $= dx = 4$

$y = mx + c$

$0 = 1.0 + c$

calculating $\Delta x = \frac{dx}{length} = \frac{4}{4} = 1$

$\Delta y = \frac{dy}{length} = \frac{4}{4} = 1$

$x = x_1 + 0.5 = 0.5$

$y = y_1 + 0.5 = 0.5$

Iteration	x value	y value	Point Plotted
	0.5	0.5	(0,0)
1	1.5	1.5	(1, 1)
2	2.5	2.5	(2,2)
3	3.5	3.5	(3,3)
4	4.5	4.5	(4,4)

Advantage of DDA Algorithm →

1. It is faster than direct use of line eqn as it calculate points on line without any floating point multiplication

Disadvantages → 1. It is time consuming as it deals with rounding off operation & floating point arithmetic.

2. It is orientation dependent, because of this the endpoint accuracy is poor.

3. The cumulative error due to limited precision in floating point representation may cause calculated points to drift away from their true position when the line is relatively long.

Bresenham's Line Algorithm →

1. It is highly efficent incremental method for scan converting lines.

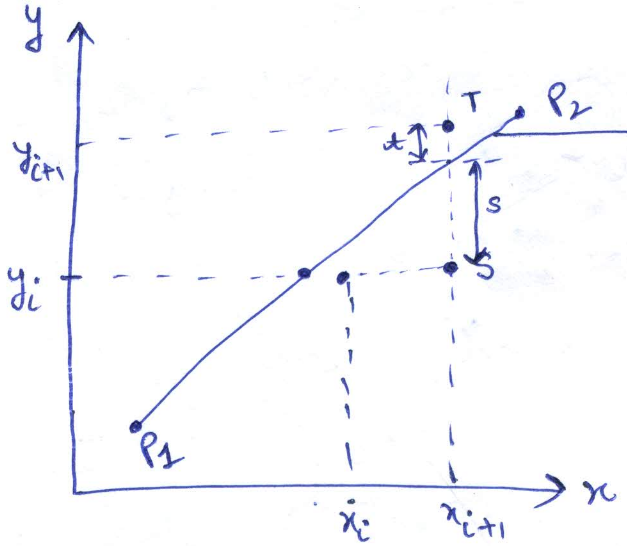
2. This was developed by Jack Elton Bresenham in 1962 at IBM

3. It produces results using only increamental integer calculations. It avoids the round off function and use only integer Add^n, Sub^n & multiplication by 2.

4. It is based on finding those pixel locations that lie closet to the true line path.

5. Depending upon the slope m of line, Bresenham increaments either x value and y value by one unit. It then finds the other value (x or y) on the basis of distance b/w the actual line location and nearest pixel. This distance is called decision variable or decision parameter.

For example →



(i) find s, t

(ii) find s−t

(iii) find decision variable $d = \Delta x(s-t)$

(iv) $d_i < 0$
$d_i > 0$

Pixel S, $x_{i+1} = x_i + 1$ and $y_{i+1} = y_i$

Pixel T, $x_{i+1} = x_i + 1$ and $y_{i+1} = y_i + 1$

Algorithm to draw a line using Bresenham's Method where $|m| < 1$

1. Input endpoint co-ordinates (x_1, y_1) and (x_2, y_2)
2. Calculate the various initial values:

$dx = x_2 - x_1$

$dy = y_2 - y_1$

$d_1 = 2 * dy - dx$

$ds = 2 * dy$

$dt = 2 * (dy - dx)$

d is decision variable

dt is the decision parameter when pixel T is chosen i.e $(d \geq 0)$

ds is decision parameter when pixel S is chosen $(d < 0)$

3. Set (x, y) equal to the lower left hand endpoint and xend equal to the largest value of x.

if $dx < 0$ i.e $x_2 - x_1 < 0$, i.e $x_2 < x_1$

then $x = x_2$, $y = y_2$ and $xend = x_1$

if $dx > 0$ i.e $x_2 - x_1 > 0$ i.e $x_2 > x_1$

then $x = x_1$, $y = y_1$ and $xend = x_2$

4. Plot the point at current (x, y) location
5. Test to see whether the entire line has been drawn
if $x = xend$, stop
6. Calculate the co-ordinate value of next pixel is
if $d < 0$ then $d = d + ds$
if $d \geq 0$ then $d = d + dt$ & increament y such that $y = y + 1$;
7. increament value of x such that $x = x + 1$
8. Plot the point at the current (x, y) location
9. Go to step 5.

for eg → find out raster location by Bresenham algorithm for end points of straight line (1,1) to (8,5).

$x_1 = 1$, $x_2 = 8$

$y_1 = 1$, $y_2 = 5$

$dx = 8-1 = 7$

$dy = 5-1 = 4$

$m = \frac{4}{7} < 1$

Also $ds = 2 * dy = 2 * 4 = 8$

$dt = 2(dy - dx) = 2 * (4-7) = -6$

$d = 2 * dy - dx = 2 * 4 - 7 = 1$

decision parameter (d)	x value	y-value	Point plotted
1 (>0) d = d+dt	1+1 = 2	1+1 = 2	2, 2
1+(-6) = -5 (<0) d = d+ds	2+1 = 3	2	3, 2
-5+8 = 3 (>0) d = d+dt	3+1 = 4	2+1 = 3	4, 3
3+(-6) = -3 (<0) d = d+ds	4+1 = 5	3	5, 3
-3+8 = 5 (>0) d = d+dt	5+1 = 6	3+1 = 4	6, 4
5+(-6) = -1 (<0) d = d+ds	6+1 = 7	4	7, 4
-1+8 = 7 (>0) d = d+dt	7+1 = 8	4+1 = 5	8, 5

Note → when |m| > 1 then exchange value of dx & dy

$m = \frac{7}{6} > 1$ then

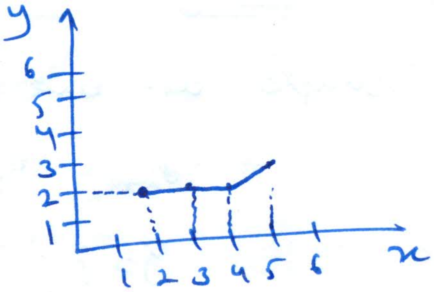
$dx = 6 \rightarrow dx = 7$

$dy = 7 \rightarrow dy = 6$

P. Singh

Line intercept

(8)



if slop $m < 1$

for eg → $(2,2)$ $(5,3)$

$m = \frac{3-2}{5-2} = \frac{1}{3} = 0.33$

$x_1 = 2$
$y_1 = 2$

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

$\Delta y = m \Delta x$

$y_1 = m x_1 + c$

$c = 2 - 0.33 \times 2$

$c = 2 - 0.66$

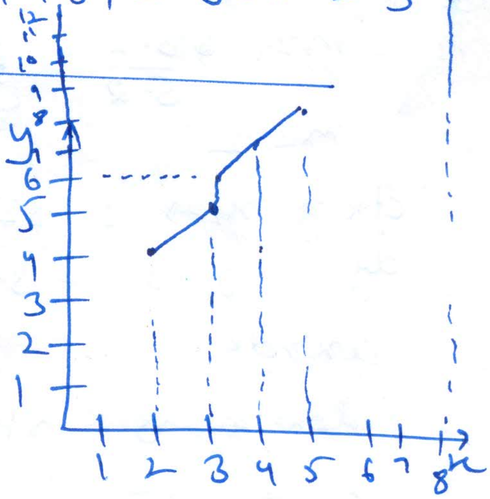
$c = 1.34$

2,2
(3,2)
(4,2)
(5,3)

$x=3$ $y = mx + c \Rightarrow \frac{1}{3} \times 3 + 1.34 = 2.34 = 2$

$x=4$ $\Rightarrow \frac{1}{3} \times 4 + 1.34 = 1.33 + 1.34 = 1.69 = 2$

$x=5$ $\Rightarrow \frac{1}{3} \times 5 + 1.34 = 1.66 + 1.34 = 3.00 = 3$



if slop $m > 1$

$(2,4)$ $(8,12)$

$m = \frac{12-4}{8-2} = \frac{8}{6} = 1.3$

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

$y=5$ $y_1 = m x_1 + c$

$4 = \frac{8}{6_3} \times 2 + c \Rightarrow 4 - \frac{8}{3} = c$

$\Rightarrow c = \frac{4}{3} = 1.33$

$y=5$ $y = mx + c \Rightarrow x = \frac{y-c}{m}$ $\frac{5-1.33}{1.3} = \frac{3.67}{1.3} = 2.82 = 3$

$y=6$ $x = \frac{6-1.33}{1.3} = 3.59 = 3$

$y=7$ $x = \frac{7-1.33}{1.3} = 4.36 = 4$

$y=8$ $x = \frac{8-1.33}{1.3} = 5.13 = 5$

DDA → Case 1 → $m \leq 1$ & positive

we sample at unit x interval ($\Delta x = 1$)

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

$x=1$ $y=?$
$x=2$ $y=?$
$x=3$ $y=?$

Case 1 negtive

($\Delta x = -1$)

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

Case 2 → $m > 1$ & positive ($\Delta y = 1$)

$$x_{k+1} = x_k + \frac{1}{m}$$

$y=1$ $x=?$
$y=2$ $x=?$
$y=3$ $x=?$

($\Delta y = -1$)

$$x_{k+1} = x_k - 1/m$$

For eg → $(2,2)$ & $(5,3)$

$m = \frac{3-2}{5-2} = \frac{1}{3} = 0.33$

$m < 1$

$dx = x_2 - x_1 = 5-2 = 3$

$dy = y_2 - y_1 = 3-2 = 1$

$dx > dy$

length $= 3 =$ step

$\Delta x = \frac{dx}{3} = \frac{3}{3} = 1$

$\Delta y = \frac{1}{3} = .33$

$x=3$ $y = 2 + .33 = 2.33 = 2$

$x=4$ $y = 2.33 + .33 = 2.66 = 3$

$(2,2)$ $x = x + \Delta x = 2+1 = 3$ $(3,2)$

$y = y + \Delta y = 2 + .33 = 2.33$

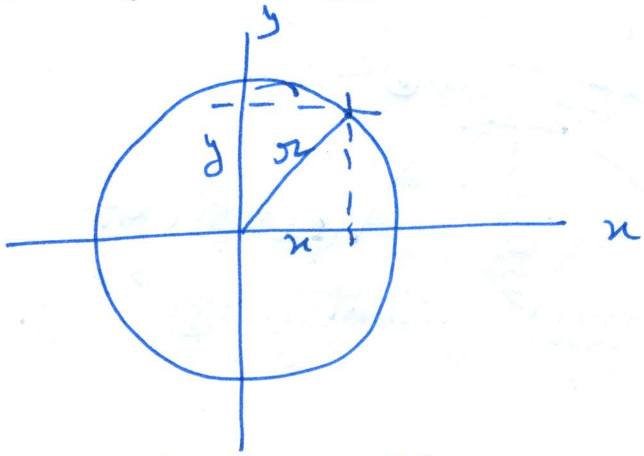
$x = 3+1 = 4$ $(4,2)$

$y = 2.33 + .33 = 2.33$

$x = 4+1 = 5$ $(5,2)$

$y = 2 + .33 = 2$

Circle generating algorithm →



There are two method to define the circle

① Polynomial eqn of circle →

$$y^2 + x^2 = r^2 \quad | \quad x^2 + y^2 = r^2$$

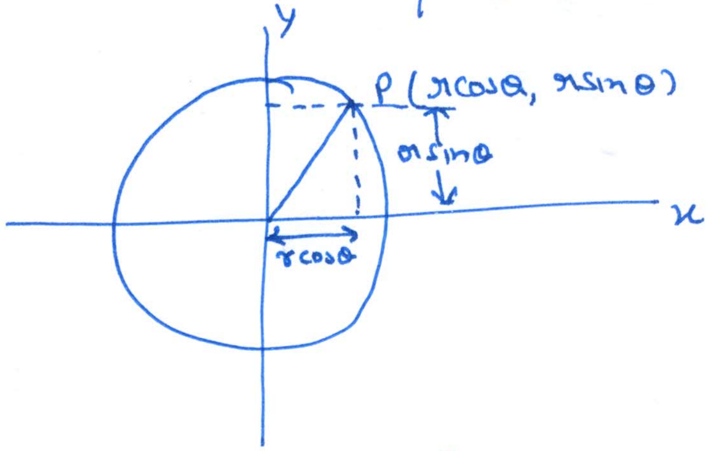
$$y = \sqrt{r^2 - x^2}$$

This eqn is used to calculate the position of points on circle circumference by stepping along the x-axis by unit distance and calculating corresponding y values at each position. But this method is not best method to draw circle due to unequal spacing b/w plotted pixel position.

② Tringometric eqn of circle / using polar co-ordinates

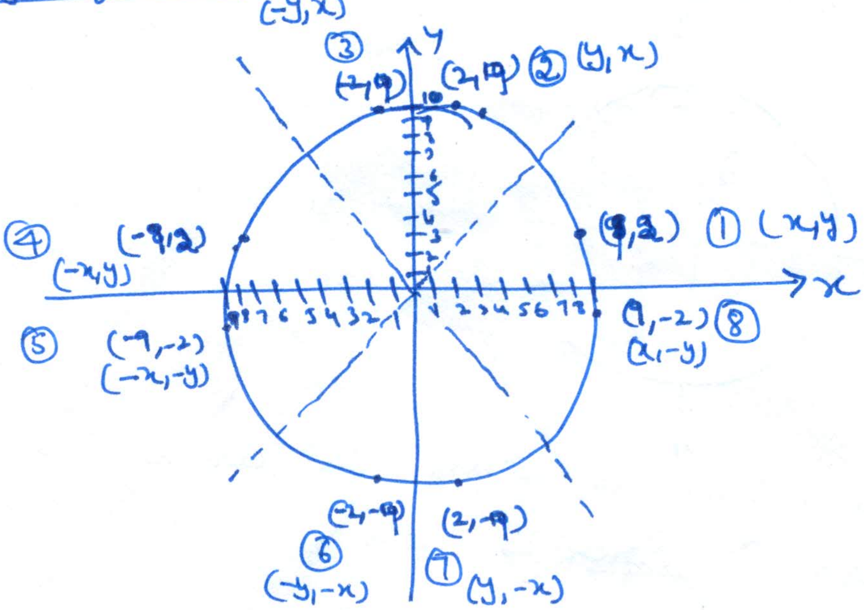
$$x = r\cos\theta \quad | \quad x = x_c + r\cos\theta$$

$$y = r\sin\theta \quad | \quad y = y_c + r\cos\theta$$



By this method θ is stepped from θ to π/4 and each value of x and y is calculated.

Symmetry of circle →



8 way symmetry of circle about 45°

for eg if point 1 is calculated then seven more points on ~~circle~~ circle can be found by reflection. The reflection is accomplished by reversing x, y co-ordinates as in point 2

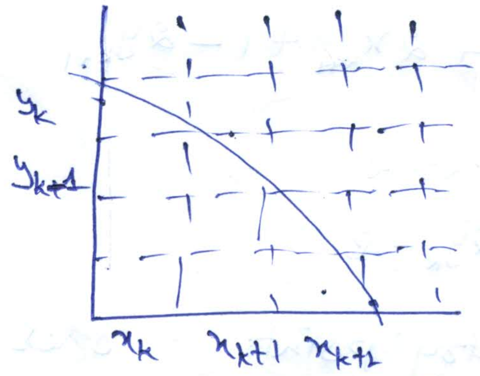
Mid point circle generating algorithm →

1. we can first set up our algorithm to calculate pixel positions around the circle path centered at co-ordinate origin (0,0)
2. It uses the following function to test the position of point wheather it lies on circle path or not.

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

3.

$$f(x,y) = x^2+y^2-r^2 \begin{cases} <0, & \text{if } (x,y) \text{ is inside the circle} \\ =0, & \text{if } (x,y) \text{ on the circle} \\ >0, & \text{if } (x,y) \text{ is outside the circle} \end{cases}$$



These function tests are performed for midpoints b/w pixel near the circle path at each sampling step. Thus the circle function is decision parameter in the midpoint algorithm. Asuming we have just plotted the pixel at (x_k, y_k), we next to determine whether pixel at position (x_{k+1}, y_k) or (x_{k+1}, y_{k-1}) is closer to the circle. so our decision parameter is

$$P_k = f_{circle}(x_k+1, y_k-1/2)$$

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

$$P_0 = \frac{5}{4} - r \quad (x=0, y=r) \text{ origin}$$

① Input radius r and circle center (x_c, y_c) and Obtain the first point on the circumference of circle.

$$x_0 = 0$$
$$y_0 = r$$

② Calculate the initial value of decision parameter

$$P_0 = \frac{5}{4} - r$$

③ At each x_k position, starting $k = 0$, perform the following test: if $P_k < 0$ the next point along the circle centered on $(0,0)$ is (x_{k+1}, y_k)

$$P_{k+1} = P_k + 2x_{k+1} + 1$$

Otherwise the next point along the circle is $(x_k + 1, y_k - 1)$

$$P_{k+1} = P_k + 2x_{k+1} + 1 - 2y_{k+1}$$

$$2x_{k+1} = 2x_k + 2$$
$$2y_{k+1} = 2y_k - 2$$

④ determine symmetry points in other seven octants

⑤ Move each calculate pixel position (x, y) onto the circular path centered (x_c, y_c) & plot the co-ordinate

$$x = x + x_c, \quad y = y + y_c$$

⑥ Repeat step 3 through 5 until $x \geq y$

for eg Plot the various points by using mid points algorithm centered at origin with radius 10 :

① $(x_0, y_0) = (0, r)$
$\Rightarrow (0, 10)$

② $P_0 = \frac{5}{4} - r = 1 - 10 = -9$
$(x_{k+1}, y_{k+1}) = (1, 10)$

③ $k=0$ $P_0 < 0$ then next point is (x_{k+1}, y_k) (x_1, y_1)

$x_{k+1} = x_k + 1 = 0 + 1$
$x_{k+1} = 1$
$y_k = y_0 = 10$
$(1, 10)$

$P_{k+1} = P_k + 2x_{k+1} + 1$
$P_1 = P_0 + 2(x_k + 1) + 1$
$= -9 + 2(0+1) + 1$
$= -9 + 3$
$P_1 = -6$

$k=1$ $P_1 < 0$ then next point is (x_{k+1}, y_k)

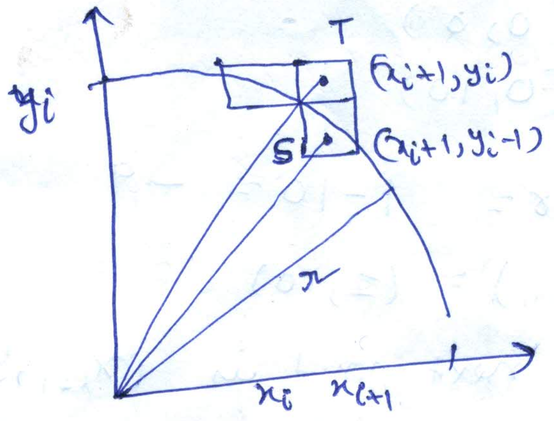
$x_{k+1} = x_k + 1 = x_1 + 1$
$x_{k+1} = 2$
$y_k = y_1 = 10$
$(2, 10)$

$P_{k+1} = P_k + 2(x_k + 1) + 1$
$P_2 = -6 + 2(x_1 + 1) + 1$
$= -6 + 2(1+1) + 1$
$= -6 + 4 + 1$
$P_2 = -1$

K	P_K	(x_{k+1}, y_{k+1})
0	-9	(1,10)
1	-6	(2,10)
2	-1	(3,10)
3	6	(4,9)
4	-3	(5,9)
5	8	(6,8)
6	5	(7,7)

Bresenham circle generating algorithm

① It uses eight way symmetry of circle and generate the points for 1/8 part (for 45°) of circle. The rest 7 parts will be copied using these pixel values



① $D(T) = (x_i+1)^2 + y_i^2 - r^2$ (will be +ve)

$D(S) = (x_i+1)^2 + (y_i-1)^2 - r^2$ (will be -ve)

② $d_i = D(T) + D(S)$

$d_i < 0$ then pixel T is chosen

$d_i \geq 0$ then pixel S is chosen

③ decision variable for first quadrant (90 to 45°)

$d_1 = 3 - 2r$

$d_{i+1} = d_i + 4x_i + 6$ if $d_i < 0$ $(x = x+1)$

$d_{i+1} = d_i + 4(x_i - y_i) + 10$ if $d_i \geq 0$ $\begin{pmatrix} x = x+1 \ \& \\ y = y-1 \end{pmatrix}$

for eg The radius of circle is 10. Calculate the various pixel location that would be plotted on circle if origin is (30, 30) using Bresenham algorithm. (10)

(i) $R = 10$, $(x_c, y_c) = (30, 30)$

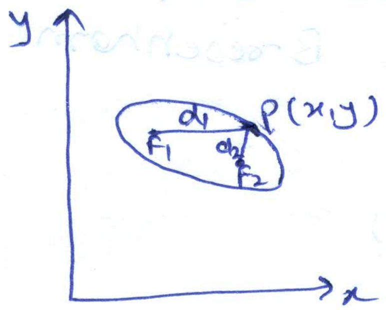
$x_0 = 0$, $y_0 = r = 10$

(ii) $d = 3 - 2r = 3 - 2 \times 10 = -17$

(iii)

decision parameter (d)	x value	y value	point plotted $x + x_c, y + y_c$
	0	10	(30, 40)
$-17 (<0)$ $d = d + 4x + 6 = -17 + 4(1) + 6 = -17 + 10 = -7$	$0 + 1 = 1$	10	$1 + 30 = 31$, $10 + 30 = 40$ (31, 40)
$-7 (<0)$ $d = d + 4x + 6 = -7 + 4(2) + 6 = 7$	$1 + 1 = 2$	10	(32, 40)
$7 (>0)$ $d = d + 4(x - y) + 10 = 7 - 24 + 10 = -7$	$2 + 1 = 3$	$10 - 1 = 9$	33, 30+9 (33, 39)
$-7 (<0)$ $d = d + 4x + 6$ $d = 13$	$3 + 1 = 4$	9	(34, 39)
$15 (>0)$ $d = d + 4(x - y) + 10 = 13$	$4 + 1 = 5$	$9 - 1 = 8$	(35, 38)
$13 (>0)$	$5 + 1 = 6$	$8 - 1 = 7$	(36, 37)

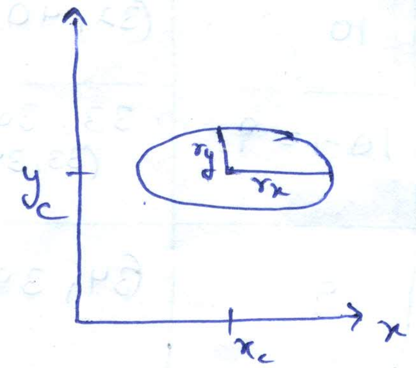
ELLIPSE Generating algorithm →



if the distances to the two foci from any point $P = (x, y)$ on the ellipse are labeled d_1 and d_2 then the general equation of an ellipse can be stated as

$$d_1 + d_2 = \text{constant}$$

$$\sqrt{(x - x_1)^2 + (y - y_1)^2} + \sqrt{(x - x_2)^2 + (y - y_2)^2} = \text{Constant}$$

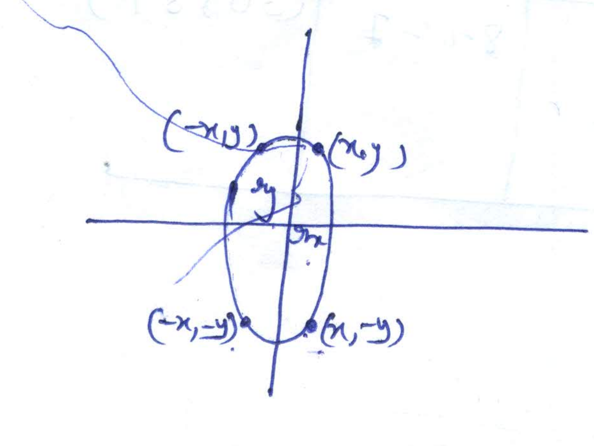


$$\left(\frac{x - x_c}{r_x}\right)^2 + \left(\frac{y - y_c}{r_y}\right)^2 = 1$$

$$x = x_c + r_x \cos\theta$$

$$y = y_c + r_y \sin\theta$$

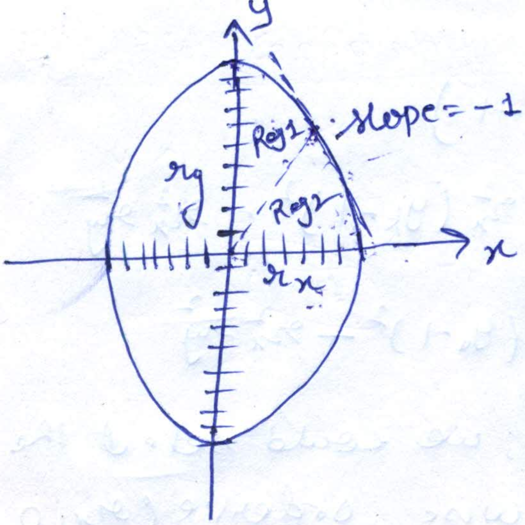
Mid point Ellipse algorithm →



$$f_{ellipse}(x, y) = r_y^2 x^2 + r_x^2 y^2 - r_x^2 r_y^2$$

$$\boxed{\begin{gathered} \left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 = 1 \\ r_x = a, \quad r_y = b \\ x_c = 0, \quad y_c = 0 \end{gathered}}$$

11



first quadrant is divided into two regions.

(I) we can start at position $(0, r_y)$ & step clockwise along the elliptical path in first quadrant.

(II) we can start at position $(r_x, 0)$ & step counterclockwise along the elliptical path.

$$f_{ellipse}(x,y) = r_y^2 x^2 + r_x^2 y^2 - r_x^2 r_y^2$$

$$f(x,y) \begin{cases} <0 & \text{if } (x,y) \text{ is inside} \\ =0 & \text{if } (x,y) \text{ on ellipse} \\ >0 & \text{if } (x,y) \text{ outside} \end{cases}$$

Now starting at $(0, r_y)$, we take unit steps in the x direction until we reach the boundary b/w region 1 & region 2.

slop of ellipse $\Rightarrow \dfrac{dy}{dx} = -\dfrac{2r_y^2 x}{2r_x^2 y}$

at boundary $2r_y^2 x = 2r_x^2 y$

so $\dfrac{dy}{dx} = -1$

for region 1 $\Rightarrow$ $p1_k = f_{ellipse}(x_k+1, y_k - \frac{1}{2})$

$$p1_k = r_y^2 (x_k+1)^2 + r_x^2 (y_k - \tfrac{1}{2})^2 - r_x^2 r_y^2$$

at $(x_0, y_0) \Rightarrow (0, r_y)$

$$p1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$$

over region 2 $\quad P2_k = f_{ellipse}(x_k+\frac{1}{2}, y_k-1)$

$$P2_k = r_y^2(x_k+\frac{1}{2})^2 + r_x^2(y_k-1)^2 - r_x^2 r_y^2$$

at (x_0, y_0) $\quad P2_k = r_y^2(x_0+\frac{1}{2})^2 + r_x^2(y_0-1)^2 - r_x^2 r_y^2$

To simplify the calculation of $P2_0$, we could select the pixel positions in counterclockwise order i.e $(r_x, 0)$

Mid point ellipse algorithm

① Input r_x, r_y and ellipse center (x_c, y_c) and obtain the first point on an ellipse centered on origin as

$$(x_0, y_0) = (0, r_y)$$

② Calculate the initial value of the decision parameter in region 1 as

$$P1_0 = r_y^2 - r_x^2 r_y^2 + \frac{1}{4} r_x^2$$

③ At each x_k position in region 1 starting at $k=0$, perform the following test if $P1_k < 0$, the next point along the ellipse centered on $(0,0)$ is (x_{k+1}, y_k) and

$$P1_{k+1} = P1_k + 2r_y^2 x_{k+1} + r_y^2$$

otherwise

$$P1_{k+1} = P1_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_y^2$$

with

$$2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2 \text{ and } 2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$$

and continue until $2r_y^2 x \geq 2r_x^2 y$

④ Calculate initial value of decision parameter in region 2

$$P2_0 = r_y^2 (x_0 + 1/2)^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$$

⑤ At each y_k position in region 2, starting at $k=0$, Perform the following test: if $P2_k > 0$, the next point along the ellipse centered on $(0,0)$ is $(x_k, y_k - 1)$ &

$$P2_{k+1} = P2_k - 2r_x^2 y_{k+1} + r_x^2$$

otherwise

$$P2_{k+1} = P2_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_x^2$$

6. Determine symmetry points in other three quadrants

7. Move each calculated pixel position (x, y) onto the elliptical path centered on (x_c, y_c) and plot the co-ordinates values

$$x = x + x_c$$
$$y = y + y_c$$

8. Repeat the steps for region 1 until $2r_y^2 x \geq 2r_x^2 y$