Computer Graphics

Module - 2

Output Primitives

CSC305

By Prof. Sunil Katkar

Department of Computer Engineering, VCET

Module -2 Output Primitives

Objective

To emphasize on implementation aspect of Computer Graphics Algorithms

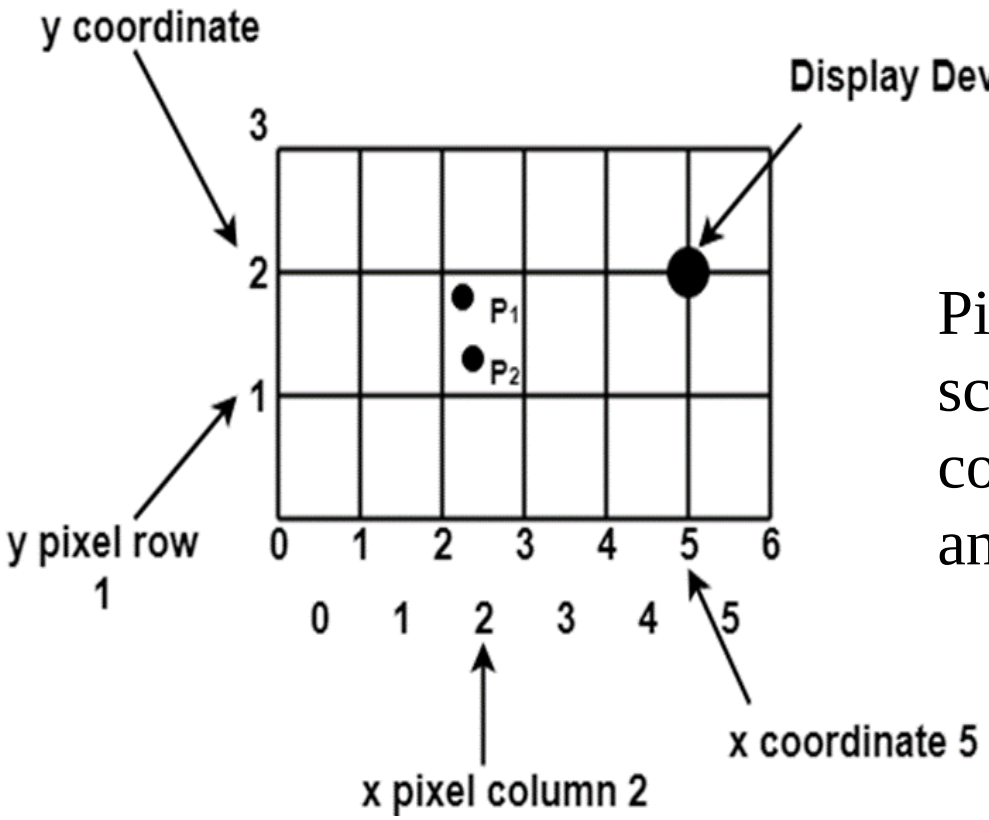
.

Outcome

At the end of the course student will be able to:

apply scan conversions algorithms to draw point, line, circle, ellipse and compare flood fill, boundary fill algorithms

Scan conversions of point



Pixel is the smallest addressable screen element which we can control by setting the intensity and colour of the pixel.

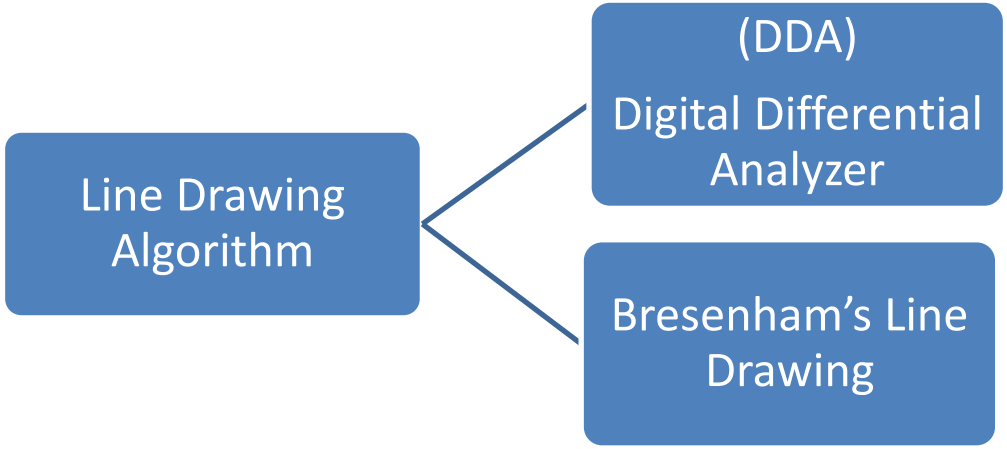
The process of determining appropriate pixels for representing graphic object is known as **Rasterization**.

The process of representing continuous graphic object as a collection of discrete pixels is called **scan conversion**.

Line Drawing Algorithm

Properties of Good Line Drawing -

1. Line should appear as a straight line and it should start and end accurately.
2. The line should be displayed with constant brightness along its length independent of its length and orientation.
3. The line should be drawn rapidly.



Vector generation/Digital Differential Analyzer (DDA)

DDA is a scan conversion line drawing algorithm.

It takes unit step with one coordinate and calculate the corresponding integer value for the other coordinate.

Slope of straight line is

$m = \Delta y/\Delta x = y2-y1/x2-x1$ ----------1

$\Delta y = (y2-y1/x2-x1) * \Delta x$ ----------2

$\Delta x = (x2-x1/y2-y1) * \Delta y$ ----------3

Now the values for next x and next y on the straight line can be calculated as

$$x_{i+1} = x_i + \Delta x$$
$$= x_i + (x2\text{-}x1/y2\text{-}y1) * \Delta y \text{ ----------4}$$

$$y_{i+1} = y_i + \Delta y$$
$$= y_i + (y2\text{-}y1/x2\text{-}x1) * \Delta x \text{ ----------5}$$

The equations 4 and 5 represents a recursion relation for successive values of x and y along the required line.

For DDA either Δx or Δy whichever is larger is chosen as one raster unit.

If $| \Delta x | \geq | \Delta y |$, then

$\Delta x = 1$

else

$\Delta y = 1$

With this simplification

If $\Delta x = 1$, then we have

$$y_{i+1} = y_i + (y2-y1/x2-x1)$$

and $x_{i+1} = x_i + 1$

If $\Delta y = 1$, then we have

$$y_{i+1} = y_i + 1$$

and $x_{i+1} = x_i + (x2-x1/y2-y1)$

DDA Algorithm

1. Read the end point co-ordinates of the line segment such that they are not equal i.e. A(x1,y1) and B(x2,y2).

   If they are equal then plot that point and exit.

2. Calculate

   $\Delta x = | x2 - x1|$

   $\Delta y = | y2 - y1|$

3. If $| \Delta x | \geq | \Delta y |$, then

   step = Δx

   else

   step = Δy

   end if

4. $\Delta x = (x2 - x1) / step$

   $\Delta y = (y2 - y1) / step$

DDA Algorithm

5. $x = x1 + 0.5 * sign(\Delta x)$

   $y = y1 + 0.5 * sign(\Delta y)$

6. $i = 1$

   while ($i \leq step$)

   {

   $x = x + \Delta x$

   $y = y + \Delta y$

   plot (int x, int y)

   $i = i + 1$

   }

7. Stop

Ex1- Consider the line from (0,0) to (4,6). Use the simple DDA algorithm to rasterize this line.

# x1 = 0, y1 = 0
  x2 = 4, y2 = 6

$\Delta x = | x2 - x1| = | 4 - 0 | = 4$
  $\Delta y = | y2 - y1| = | 6 - 0 | = 4$

# If $| \Delta x | \geq | \Delta y |$, then
        step = Δx
  else
        step = Δy
  As $| \Delta y | \geq | \Delta x |$ i.e. 6 > 4
$\therefore$ step = Δy = 6

$\Delta x = (x2 - x1) / step$

$= 4/6 = 0.667$

$\Delta y = (y2 - y1) / step$

$= 6/6 = 1$

Now the initial values are

$x = x1 + 0.5 * sign(\Delta x)$

$= 0 + 0.5 * sign(0.667)$

$= 0 + 0.5 * (+1)$

$= 0.5$

$y = y1 + 0.5 * sign(\Delta y)$

$= 0 + 0.5 * sign(1)$

$= 0 + 0.5 * (+1)$

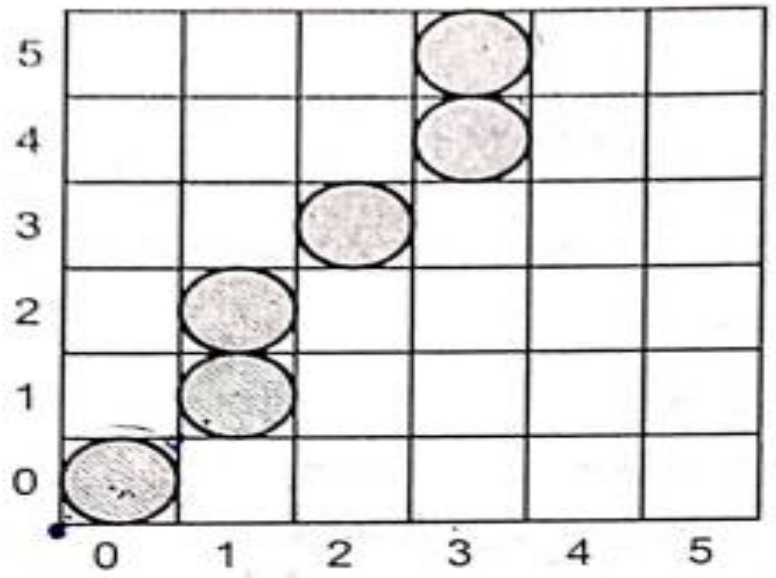
$= 0.5$

```
#   i = 1
      while (i ≤ step)
            {
            x = x + Δx
            y = y + Δy
      plot (int x, int y)
            i = i + 1
            }
```

i.e. i = 1

```
while (i ≤ 6)
{           x = x + 0.667
            y = y + 1
      plot (int x, int y)
            i = i + 1
            }
```

i	x	y	Putpixel (int x, int y)
	0.5	0.5	(0, 0)
1	1.167	1.5	(1,1)
2	1.833	2.5	(1, 2)
3	2.5	3.5	(2, 3)
4	3.167	4.5	(3, 4)
5	3.833	5.5	(3, 5)
6	4.499	6.5	(4, 6)



Ex2- Consider the line from (0,0) to (- 6, -6). Use the simple DDA algorithm to rasterize this line.

x1 = 0, y1 = 0

x2 = - 6 , y2 = - 6

$\Delta x = | x2 - x1| = | - 6 - 0 | = 6$

$\Delta y = | y2 - y1| = | - 6 - 0 | = 6$

If $| \Delta x | \geq | \Delta y |$, then

step = Δx

else

step = Δy

As $| \Delta x | = | \Delta y |$ i.e. 6 = 6

$\therefore$ step = Δx = 6

$\Delta x = (x2 - x1) / step$

$= - 6 / 6 = - 1$

$\Delta y = (y2 - y1) / step$

$= - 6/6 = - 1$

Now the initial values are

$x = x1 + 0.5 * sign(\Delta x)$

$= 0 + 0.5 * sign(-1)$

$= 0 + 0.5 * (-1)$

$= - 0.5$

$y = y1 + 0.5 * sign(\Delta y)$

$= 0 + 0.5 * sign(-1)$

$= 0 + 0.5 * (-1)$

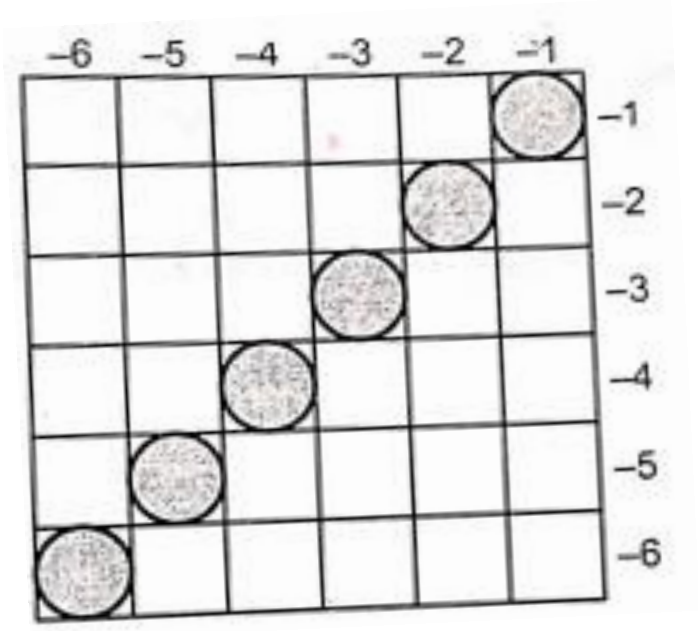
$= - 0.5$

```
#   i = 1
      while (i ≤ step)
            {
            x = x + Δx
            y = y + Δy
putpixel(int x, int y, Colour)
            i = i + 1
            }
```

i.e. i = 1

```
while (i ≤ 6)
{           x = x - 1
            y = y - 1
      putpixel (int x, int y, Colour)
            i = i + 1
            }
```

i	x	y	Putpixel (int x, int y)
	- 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)
5	- 5.5	- 6.5	(-5, -5)
6	- 6.5	- 6.5	(-6, -6)



Limitations of DDA algorithm

Advantages

- ✓ Simple algorithm
- ✓ Does not required special skills for implementation
- ✓ Faster method for calculating pixel positions than direct use of equation $y = mx + c$.
- ✓ It eliminates the multiplication in the equation by making use of raster characteristics, so that appropriate increments are applied in the x and y direction to find the pixel positions along the line path.

Disadvantages

- ✓ Floating point arithmetic in DDA is still time-consuming.
- ✓ The algorithm is orientation dependent and hence end point accuracy is poor.

Bresenham's Line Drawing Algorithm

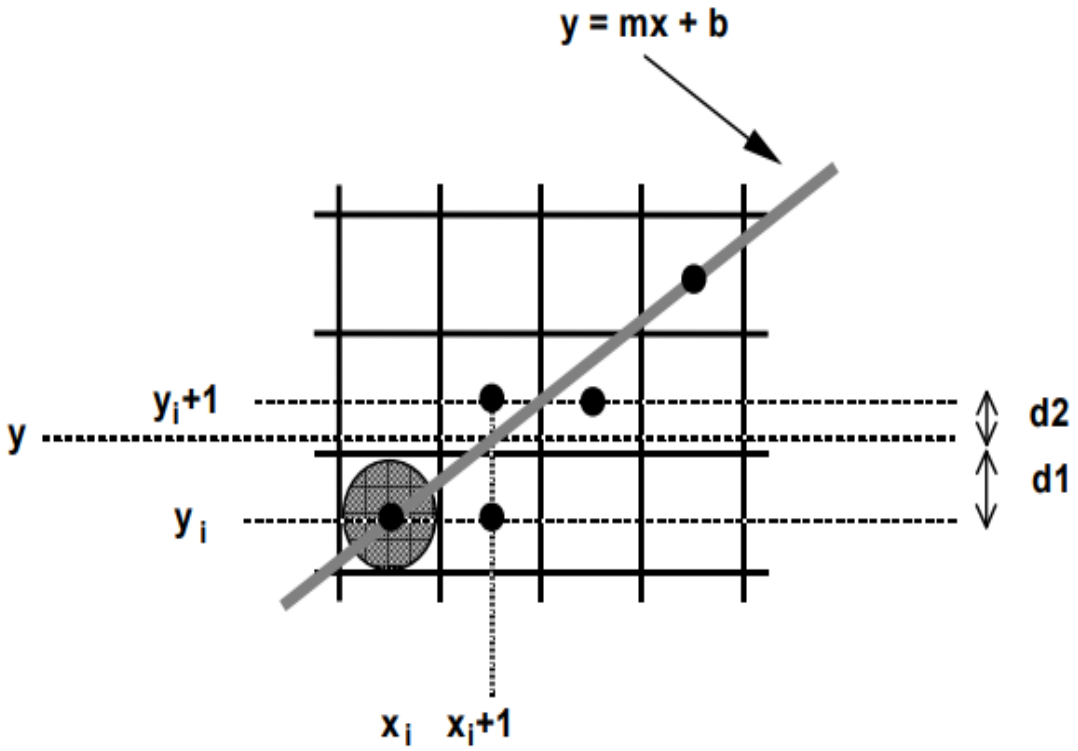
Select the optimum raster locations to represent a straight line.

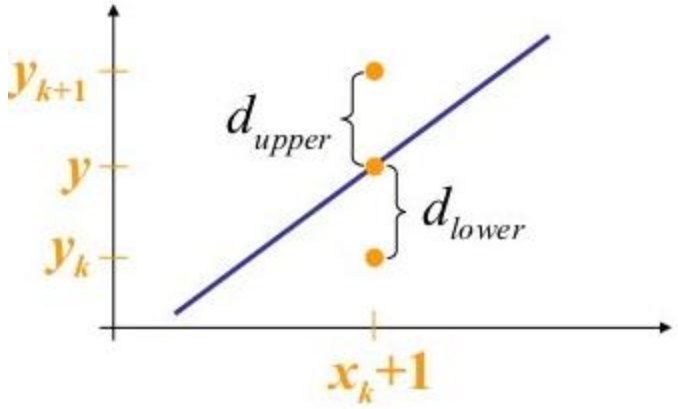
Algorithm always increments either x or y by one unit depending on the slope of line.

The increment in the other variable is determined by examining the distance between the actual line location and the nearest pixel.

This distance is called decision variable/parameter or the error.

Derivation of Bresenham's Algorithm (m < 1)





Let x_k, y_k be the selected positions along the line path.

The next positions (x_{next} , y_{next}) will be

$$x_{next} = x_{k+1} = x_k + 1$$

$$y_{next} = y_{k+1} = y_k \text{ or } y_k + 1$$

The equation of line is

$$y = mx + b$$

at $x = x_k + 1$

$$y = m(x_k + 1) + b$$

If d1 < d2 then y_k is closer to the actual line path and $y_{k+1} = y_k$

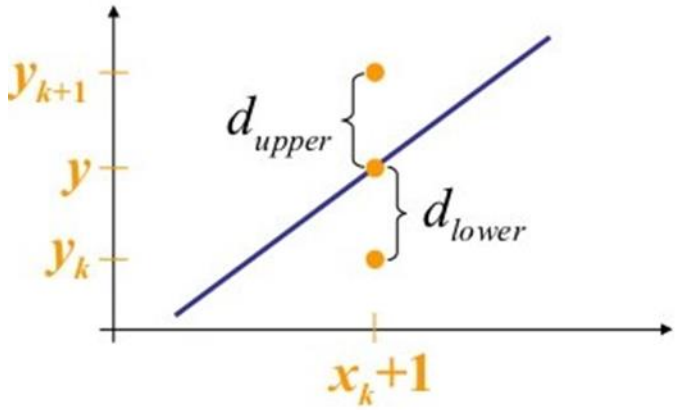
However if d2 < d1 then y_{k+1} is closer to the line path and $y_{k+1} = y_k + 1$

Now

$$d2 = y_{k+1} - y$$
$$= y_k + 1 - y$$
$$= y_k + 1 - [m(x_k + 1) + b]$$
$$= y_k + 1 - m(x_k + 1) - b$$

and

$$d1 = y - y_k$$
$$= [m(x_k + 1) + b] - y_k$$
$$= m(x_k + 1) + b - y_k$$



Now $d1 - d2 = m(x_k + 1) + b - y_k - [y_k + 1 - m(x_k + 1) - b]$

$= m(x_k + 1) + b - y_k - y_k - 1 + m(x_k + 1) + b$

$= 2m(x_k + 1) - 2y_k + 2b - 1$

$= 2\,\Delta y/\Delta x\,(x_k + 1) - 2y_k + 2b - 1$

$\Delta x\,(d1 - d2) = 2\,\Delta y\,(x_k + 1) - 2\,\Delta x\,y_k + 2\,\Delta x\,b - \Delta x$

$= 2\,\Delta y\,x_k + 2\Delta y - 2\,\Delta x\,y_k + 2\,\Delta x\,b - \Delta x$

Let the decision parameter is p_k

$p_k = \Delta x\,(d1 - d2)$

$\mathbf{p_k = 2\,\Delta y\,x_k + 2\Delta y - 2\,\Delta x\,y_k + 2\,\Delta x\,b - \Delta x}$

$\therefore\ p_{k+1} = 2\,\Delta y\,x_{k+1} + 2\Delta y - 2\,\Delta x\,y_{k+1} + 2\,\Delta x\,b - \Delta x$

$p_{k+1} - p_k = 2\,\Delta y\,x_{k+1} + 2\Delta y - 2\,\Delta x\,y_{k+1} + 2\,\Delta x\,b - \Delta x$

$- [2\,\Delta y\,x_k + 2\Delta y - 2\,\Delta x\,y_k + 2\,\Delta x\,b - \Delta x]$

$= 2\,\Delta y\,x_{k+1} + 2\Delta y - 2\,\Delta x\,y_{k+1} + 2\,\Delta x\,b - \Delta x$

$- 2\,\Delta y\,x_k - 2\Delta y + 2\,\Delta x\,y_k - 2\,\Delta x\,b + \Delta x$

$= 2\Delta y\,[x_{k+1} - x_k] - 2\,\Delta x\,[y_{k+1} - y_k]$

$\therefore$ $p_{k+1} = p_k + 2\Delta y\,[x_{k+1} - x_k] - 2\,\Delta x\,[y_{k+1} - y_k]$

Now initial value of decision parameter (p_0)

The equation of line is

$$y = m\,x + b$$

at k = 0, $(x_k, y_k) = (x_0, y_0)$

$$y_0 = m\,x_0 + b$$

$$b = y_0 - m\,x_0$$

$$b = y_0 - (\Delta y/\Delta x)\,x_0$$

Now we have decision parameter p_k as

$$p_k = 2\,\Delta y\,x_k + 2\Delta y - 2\,\Delta x\,y_k + 2\,\Delta x\,b - \Delta x$$

$\therefore$ $p_0 = 2\,\Delta y\,x_0 + 2\Delta y - 2\,\Delta x\,y_0 + 2\,\Delta x\,b - \Delta x$

Substituting the value of b

$$p_0 = 2\,\Delta y\,x_0 + 2\Delta y - 2\,\Delta x\,y_0 + 2\,\Delta x\,[y_0 - (\Delta y/\Delta x)\,x_0\,] - \Delta x$$

$$p_0 = 2\,\Delta y\,x_0 + 2\Delta y - 2\,\Delta x\,y_0 + 2\,[y_0\,\Delta x - \Delta y\,x_0\,] - \Delta x$$

$$p_0 = 2\,\Delta y\,x_0 + 2\Delta y - 2\,\Delta x\,y_0 + 2\,y_0\,\Delta x - 2\,\Delta y\,x_0 - \Delta x$$

$p_0 = 2\,\Delta y\, x_0 + 2\Delta y - 2\,\Delta x\, y_0 + 2\, y_0\, \Delta x - 2\,\Delta y\, x_0 - \Delta x$

$\mathbf{p_0 = 2\Delta y - \Delta x}$

If $p_k < 0$, then

$\mathbf{x_{next} = x_{k+1} = x_k + 1}$

$\mathbf{y_{next} = y_{k+1} = y_k}$

$\mathbf{p_{k+1} = p_k + 2\Delta y\,[x_{k+1} - x_k] - 2\,\Delta x\,[y_{k+1} - y_k]}$

$p_{k+1} = p_k + 2\Delta y\,[x_k + 1 - x_k] - 2\,\Delta x\,[y_k - y_k]$

$\mathbf{p_{k+1} = p_k + 2\Delta y}$

If $p_k \geq 0$, then

$\mathbf{x_{next} = x_{k+1} = x_k + 1}$

$\mathbf{y_{next} = y_{k+1} = y_k + 1}$

$\mathbf{p_{k+1} = p_k + 2\Delta y\,[x_{k+1} - x_k] - 2\,\Delta x\,[y_{k+1} - y_k]}$

$p_{k+1} = p_k + 2\Delta y\,[x_k + 1 - x_k] - 2\,\Delta x\,[y_k + 1 - y_k]$

$\mathbf{p_{k+1} = p_k + 2\Delta y - 2\,\Delta x}$

Bresenham's Algorithm for m < 1

1. Read the end point co-ordinates of the line segment such that they are not equal i.e. A(x1,y1) and B(x2,y2).

   If they are equal then plot that point and exit.

2. Calculate

   $\Delta x = x2 - x1$

   $\Delta y = y2 - y1$

3. Initialize starting point

   $x_{next} = x_{k+1} = x1$

   $y_{next} = y_{k+1} = y1$

4. Calculate the initial value of decision parameter, p_0 as

   $p_0 = 2\Delta y - \Delta x$

5. Initialize counter, k = 0

6. Plot $(x_{next}, y_{next}) = (x_{k+1}, y_{k+1})$

5. At each x_k position starting at k = 0 perform the following test -

If $p_k < 0$, then

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1} = p_k + 2\Delta y$$

If $p_k \geq 0$, then

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1} = p_k + 2\Delta y - 2\,\Delta x$$

6. k = k + 1

7. Plot $(x_{next}, y_{next}) = (x_{k+1}, y_{k+1})$

8. Repeat the steps 5 and 7 until

[$x_{k+1} = x_2$ or/and $y_{k+1} = y_2$]

9. Stop

Derivation of Bresenham's Algorithm (m ≥ 1)

Let x_k, y_k be the selected positions along the line path.

The next positions (x_{next} , y_{next}) will be

$x_{next} = x_{k+1} = x_k$ or $x_k + 1$

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

If d1 < d2 then x_k is closer to the actual line path and $x_{k+1} = x_k$

However if d2 < d1 then x_{k+1} is closer to the line path & $x_{k+1} = x_k + 1$

Now

$d2 = x_{k+1} - x$

$= x_k + 1 - x$

and

$d1 = x - x_k$

Initial value of decision parameter (p_0) is

$$\mathbf{p_0 = 2\Delta x - \Delta y}$$

If $p_k < 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k}$$
$$\mathbf{y_{next} = y_{k+1} = y_k + 1}$$
$$\mathbf{p_{k+1} = p_k + 2\Delta x}$$

If $p_k \geq 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k + 1}$$
$$\mathbf{y_{next} = y_{k+1} = y_k + 1}$$
$$\mathbf{p_{k+1} = p_k + 2\Delta x - 2\Delta y}$$

Bresenham's Algorithm for $m \geq 1$

1. Read the end point co-ordinates of the line segment such that they are not equal i.e. A(x1,y1) and B(x2,y2).

   If they are equal then plot that point and exit.

2. Calculate

   $\Delta x = x2 - x1$

   $\Delta y = y2 - y1$

3. Calculate the initial value of decision parameter, p_0 as

   $p_0 = 2\Delta x - \Delta y$

4. At each x_k position starting at k = 0 perform the following test -

If $p_k < 0$, then

$$x_{next} = x_{k+1} = x_k$$

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

$$p_{k+1} = p_k + 2\Delta x$$

If $p_k \geq 0$, then

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1} = p_k + 2\Delta x - 2\Delta y$$

5. Display $(x_{next}, y_{next}) = (x_{k+1}, y_{k+1})$
6. Repeat the steps 4 and 5 until

$$x_{k+1} = x_2$$

$$y_{k+1} = y_2$$

7. Stop

Ex1- Consider the line from (5,5) to (13,9).
Use the Bresenham's algorithm to rasterize this line.

$x1 = 5, y1 = 5$

$x2 = 13, y2 = 9$

$\Delta x = | x2 - x1| = | 13 - 5 | = 8$

$\Delta y = | y2 - y1| = | 9 - 5 | = 4$

Slope, $m = \Delta y/\Delta x = 4/8 = 0.5 < 1$

Initialize starting point

$$x_{next} = x_{k+1} = x1 = 5$$

$$y_{next} = y_{k+1} = y1 = 5$$

The initial value of decision parameter, p_0 is

$$p_0 = 2\Delta y - \Delta x$$

$$= 2*4 - 8$$

$$= 8 - 8$$

$$\mathbf{p_0 = 0}$$

At each x_k position starting at k = 0 perform the following test -

If $p_k < 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k + 1}$$

$$\mathbf{y_{next} = y_{k+1} = y_k}$$

$$p_{k+1} = p_k + 2\Delta y$$

$$p_{k+1} = p_k + 2\Delta y$$

$$p_{k+1} = p_k + 2(4)$$

$$\mathbf{p_{k+1} = p_k + 8}$$

If $p_k \geq 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k + 1}$$

$$\mathbf{y_{next} = y_{k+1} = y_k + 1}$$

$$p_{k+1} = p_k + 2\Delta y - 2\,\Delta x$$

$$p_{k+1} = p_k + 2(4) - 2\,(8)$$

$$p_{k+1} = p_k + 8 - 16$$

$$\mathbf{p_{k+1} = p_k - 8}$$

$p_0 = 0$

If $p_k < 0$, then

$x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k + 8$

If $p_k \geq 0$, then

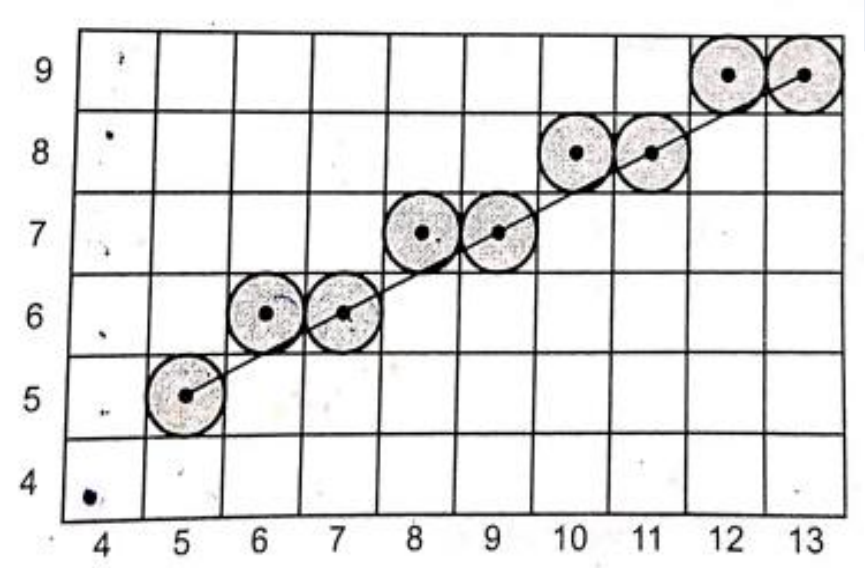
$x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k - 8$

k	p_k	$x_{next} = x_{k+1}$	$y_{next} = y_{k+1}$	p_{k+1}	Putpixel (int x, int y)
	0	5	5	--	(5, 5)
1	0	6	6	- 8	(6, 6)
2	- 8	7	6	0	(7, 6)
3	0	8	7	- 8	(8, 7)
4	- 8	9	7	0	(9, 7)
5	0	10	8	- 8	(10, 8)
6	- 8	11	8	0	(11, 8)
7	0	12	9	- 8	(12, 9)
8	- 8	13	9	0	(13, 9)

k	p_k	$x_{next} = x_{k+1}$	$y_{next} = y_{k+1}$	p_{k+1}	Putpixel (int x, int y)
	0	5	5	--	(5, 5)
1	0	6	6	- 8	(6, 6)
2	- 8	7	6	0	(7, 6)
3	0	8	7	- 8	(8, 7)
4	- 8	9	7	0	(9, 7)
5	0	10	8	- 8	(10, 8)
6	- 8	11	8	0	(11, 8)
7	0	12	9	- 8	(12, 9)
8	- 8	13	9	0	(13, 9)



Ex2- Consider the line from (10,20) to (18,30).
Use the Bresenham's algorithm to rasterize this line.

x1 = 10, y1 = 20

x2 = 18, y2 = 30

$\Delta x = | x2 - x1| = | 18 - 10 | = 8$

$\Delta y = | y2 - y1| = | 30 - 20 | = 10$

Slope, m $= \Delta y/\Delta x = 10/8 = 1.25 >$ **1**

Initialize starting point

$x_{next} = x_{k+1} = x1 = 10$

$y_{next} = y_{k+1} = y1 = 20$

The initial value of decision parameter, p_0 is

$p_0 = 2\Delta x - \Delta y$

$= 2*8 - 10$

$= 16 - 10$

$\mathbf{p_0 = 6}$

At each x_k position starting at k = 0 perform the following test -

If $p_k < 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k}$$

$$\mathbf{y_{next} = y_{k+1} = y_k + 1}$$

$$p_{k+1} = p_k + 2\Delta x$$

$$p_{k+1} = p_k + 2(8)$$

$$\mathbf{p_{k+1} = p_k + 16}$$

If $p_k \geq 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k + 1}$$

$$\mathbf{y_{next} = y_{k+1} = y_k + 1}$$

$$p_{k+1} = p_k + 2\Delta x - 2\,\Delta y$$

$$p_{k+1} = p_k + 2(8) - 2\,(10)$$

$$p_{k+1} = p_k + 16 - 20$$

$$\mathbf{p_{k+1} = p_k - 4}$$

k	p_k	$x_{next} = x_{k+1}$	$y_{next} = y_{k+1}$	p_{k+1}	Putpixel (int x, int y)
		10	20		(10, 20)
1	6	11	21	2	(11, 21)
2	2	12	22	- 2	(12, 22)
3	- 2	12	23	14	(12, 23)
4	14	13	24	10	(13, 24)
5	10	14	25	6	(14, 25)
6	6	15	26	2	(15, 26)
7	2	16	27	- 2	(16, 27)
8	- 2	16	28	14	(16, 28)
9	14	17	29	10	(17, 29)
10	10	18	30	6	(18, 30)
$11 > \Delta y$					

Circle Drawing

Any circle generating algorithm uses Eight-way symmetry of a circle to plot the eight points by calculating the coordinates of any one point.

putpixel ($x + x_c$, $y + y_c$,WHITE)

putpixel ($x + x_c$, $- y + y_c$)

putpixel ($- x + x_c$, $y + y_c$)

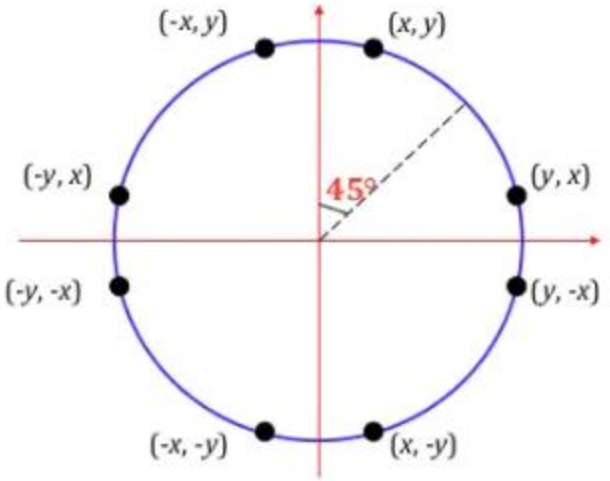
putpixel ($- x + x_c$, $- y + y_c$)

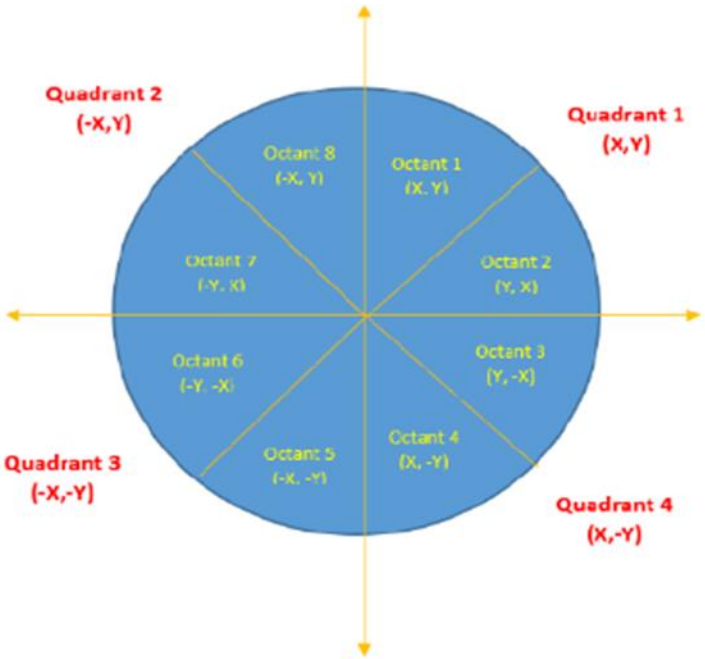
putpixel ($y + x_c$, $x + y_c$)

putpixel ($y + x_c$, $- x + y_c$)

putpixel ($- y + x_c$, $x + y_c$)

putpixel ($- y + x_c$, $- x + y_c$)

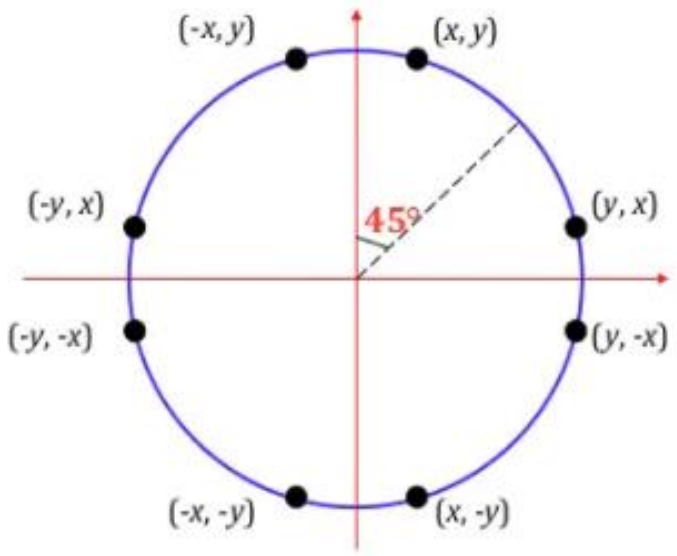




Midpoint Circle Drawing

It uses the eight-way symmetry of the circle.

It plots 1/8 part of the circle i.e. from 90^0 to 45^0, as shown



x moves in the positive direction

y moves in the negative direction

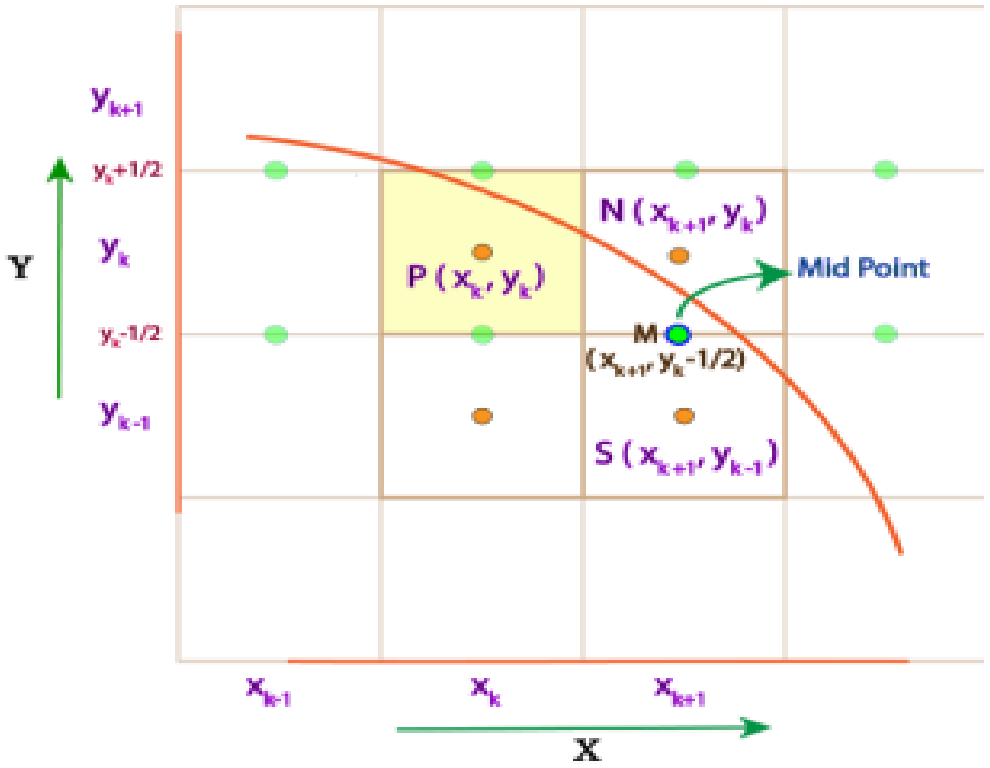
To draw 1/8 part of the circle we take unit step in the positive x-direction and make use of decision parameter to determine which of the two possible y-positions is closer to the circle path at each step.

$$f_{circle}(x, y) = x^2 + y^2 - r^2 = 0$$

$$f_{circle}(x, y) = < 0$$

$$= 0$$

$$= > 0$$



Consider the pixel at (x_k, y_k).

The next pixel (x_{next}, y_{next}) along the circumference of a circle will be either (x_{k+1}, y_k) OR (x_{k+1}, y_{k-1}) whichever is closer to the circle path at each step.

Let the decision parameter p_k is equal to the circle function evaluated at the mid-point between two pixels.

$$p_k = f_{circle}(x, y)$$

$$= f_{circle}(x_{k+1}, y_k - 1/2)$$

$$p_k = f_{circle}(x, y)$$
$$= f_{circle}(x_{k+1}, y_k - 1/2)$$
$$= f_{circle}(x_k + 1, y_k - 1/2)$$
$$= (x_{k+1})^2 + (y_k - 1/2)^2 - r^2$$
$$= (x_k + 1)^2 + (y_k - 1/2)^2 - r^2$$
$$\mathbf{p_k = (x_k)^2 + (2\,x_k) + 1 + (y_k)^2 - (y_k) + (1/4) - r^2}$$

Now

$$p_k = f_{circle}(x_k + 1, y_k - 1/2)$$
$$\therefore\ p_{k+1} = f_{circle}(x_{k+1} + 1, y_{k+1} - 1/2)$$
$$\therefore\ p_{k+1} = (x_{k+1} + 1)^2 + (y_{k+1} - 1/2)^2 - r^2$$
$$= (x_k + 1 + 1)^2 + (y_{k+1} - 1/2)^2 - r^2$$
$$= (x_k + 2)^2 + (y_{k+1} - 1/2)^2 - r^2$$
$$= x_k^2 + 4x_k + 4 + y_{k+1}^2 - y_{k+1} + 1/4 - r^2$$

Now, $p_{k+1} - p_k = x_k^2 + 4x_k + 4 + y_{k+1}^2 - y_{k+1} + 1/4 - r^2$
$$- [x_k^2 + 2x_k + 1 + y_k^2 - y_k + 1/4 - r^2\ ]$$

Now, $p_{k+1} - p_k = x_k^2 + 4x_k + 4 + y_{k+1}^2 - y_{k+1} + 1/4 - r^2$

$$- [x_k^2 + 2x_k + 1 + y_k^2 - y_k + 1/4 - r^2]$$

$p_{k+1} - p_k = x_k^2 + 4x_k + 4 + y_{k+1}^2 - y_{k+1} + 1/4 - r^2$

$$- x_k^2 - 2x_k - 1 - y_k^2 + y_k - 1/4 + r^2]$$

$p_{k+1} - p_k = 2x_k + 3 + [y_{k+1}^2 - y_k^2] - [y_{k+1} - y_k]$

$\therefore$ **$p_{k+1} = p_k + 2x_k + 3 + [y_{k+1}^2 - y_k^2] - [y_{k+1} - y_k]$**

If $p_k < 0$, then

mid-point is INSIDE the circle and the pixel at y_k is closer to the circle boundary.

Otherwise

mid-point is OUTSIDE or ON the circle boundary and the pixel at y_{k-1} is closer to the circle boundary.

If $p_k < 0$, then

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1} = p_k + 2x_k + 3 + [y_{k+1}^2 - y_k^2] - [y_{k+1} - y_k]$$

$$p_{k+1} = p_k + 2x_k + 3 + [y_k^2 - y_k^2] - [y_k - y_k]$$

$$p_{k+1} = p_k + 2x_k + 3$$

If $p_k \geq 0$, then

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1} = p_k + 2x_k + 3 + [y_{k+1}^2 - y_k^2] - [y_{k+1} - y_k]$$

$$p_{k+1} = p_k + 2x_k + 3 + [(y_k - 1)^2 - y_k^2] - [y_k - 1 - y_k]$$

$$p_{k+1} = p_k + 2x_k + 3 + [(y_k^2 - 2\,y_k + 1 - y_k^2] - [y_k - 1 - y_k]$$

$$p_{k+1} = p_k + 2x_k + 3 - 2\,y_k + 1 + 1$$

$$p_{k+1} = p_k + 2x_k - 2\,y_k + 5$$

The starting point is at (0, r)

The initial value of decision parameter is obtained by evaluating the circle function at the start position.

Now we have

$$p_k = f_{circle}(x, y)$$
$$= f_{circle}(x_{k+1}, y_k - 1/2)$$
$$= f_{circle}(x_k + 1, y_k - 1/2)$$
$$\therefore \; p_0 = f_{circle}(x_0 + 1, y_0 - 1/2)$$

Let $(x_0, y_0) = (0, r)$

$$\therefore \; p_0 = f_{circle}(1, r - 1/2)$$
$$p_0 = (1)^2 + (r - 1/2)^2 - r^2$$
$$p_0 = 1^2 + r^2 - r + 1/4 - r^2$$
$$\mathbf{p_0 = 5/4 - r}$$

Midpoint Circle Drawing Algorithm

1. Accept the radius ‘r’ and center of the circle (x_c, y_c)

2. Initialize the starting position as $(x_0, y_0) = (0, r)$

3. Calculate the initial value of decision parameter, p_0 as

$$p_0 = 5/4 - r = 1.25 - r$$

4. At each x_k position starting at k = 0 perform the following test

do

{ putpixel (x, y, COLOR)

If $p_k < 0$

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k + 2x_k + 3$

}

else

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k + 2x_k - 2\,y_k + 5$

}

}

while (x < y)

5. Determine the symmetry points in the other seven octants.

6. Translate each calculated pixel position in eight octant by (x_c, y_c) and plot the pixels.

$x = x_{k+1} + x_c$

$y = y_{k+1} + y_c$

putpixel (x, y, COLOUR)

7. Repeat the steps 4 to 6 until $x \geq y$.

8. Stop

Ex1- Calculate the pixel positions along the circle path with radius r = 8 centered at the origin using mid-point circle drawing algorithm.

Radius of the circle, r = 8 and center of the circle, $(x_c, y_c) = (0, 0)$.

Starting position is $(x_0, y_0) = (0, 8)$

Initial value of decision parameter, p_0 is

$$p_0 = 5/4 - 8 = 1.25 - 8 = -6.75$$

At each x_k position starting at k = 0 perform the following test

do

{ putpixel (x, y, COLOR)

If $p_k < 0$

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k + 2x_k + 3$

}

else

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k + 2x_k - 2\,y_k + 5$

}

}

while (x < y)

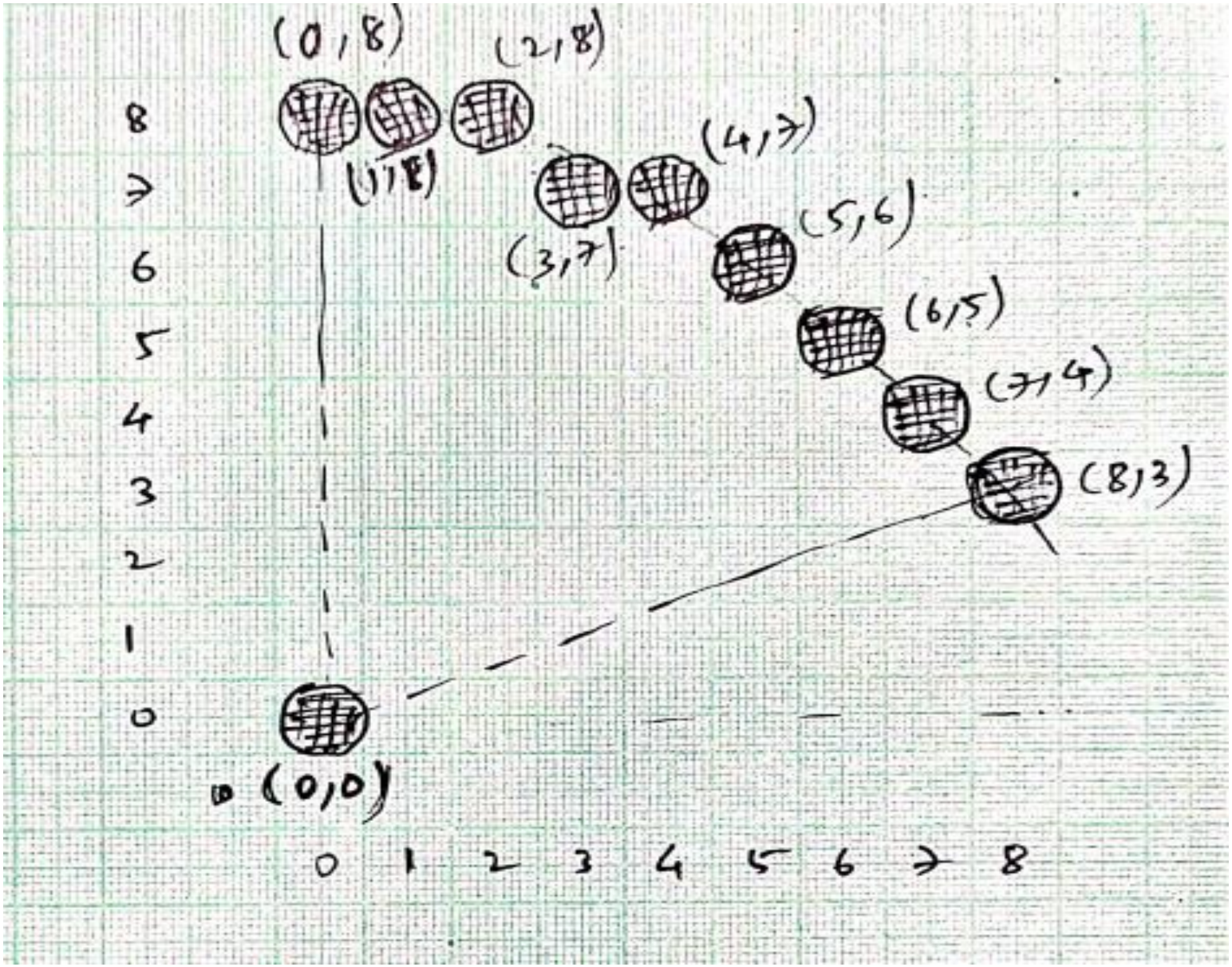
$(x_0, y_0) = (0, 8)$

$p_0 = 5/4 - 8 = 1.25 - 8 = -6.75$

$p_k = (x_k)^2 + (2\,x_k) + 1 + (y_k)^2 - (y_k) - 63.75$

$p_{k+1} = p_k + 2x_k + 3 + [y_{k+1}^2 - y_k^2] - [y_{k+1} - y_k]$

k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)
0	(0, 8)	- 6.75	(1, 8)	**- 3.75**	(1, 8)
1	(1, 8)	- 3.75	(2, 8)	1.25	(2, 8)
2	(2, 8)	1.25	(3, 7)	- 5.75	(3, 7)
3	(3, 7)	- 5.75	(4, 7)	3.25	(4, 7)
4	(4, 7)	3.25	(5, 6)	2.25	(5, 6)
5	(5, 6)	2.25	(6, 5)	5.25	(6, 5)
6	(6, 5)	5.25	(7, 4)	12.25	(7, 4)
7	(7,4)	12.25	(8,3)	37.25	(8, 3)


(0,8)
(1,8)
(2,8)
(3,7)
(4,7)
(5,6)
(6,5)
(7,4)
(8,3)
(0,0)
8
7
6
5
4
3
2
1
0
0 1 2 3 4 5 6 7 8

Ex2- Calculate the pixel positions along the circle path with radius r = 10 centered at the origin using mid-point circle drawing algorithm up to x = y.

Radius of the circle, r = 10 and center of the circle, $(x_c, y_c) = (0, 0)$.

Starting position is $(x_0, y_0) = (0, 10)$

Initial value of decision parameter, p_0 is

$$p_0 = 5/4 - 10 = 1.25 - 10 = -8.75$$

At each x_k position starting at k = 0 perform the following test

do

{ putpixel (x, y, COLOR)

If $p_k < 0$

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k + 2x_k + 3$

}

else

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1} = p_k + 2x_k - 2\,y_k + 5$

}

}

while (x < y)

$(x_0, y_0) = (0, 10)$

$p_0 = 5/4 - 8 = 1.25 - 8 = -8.75$

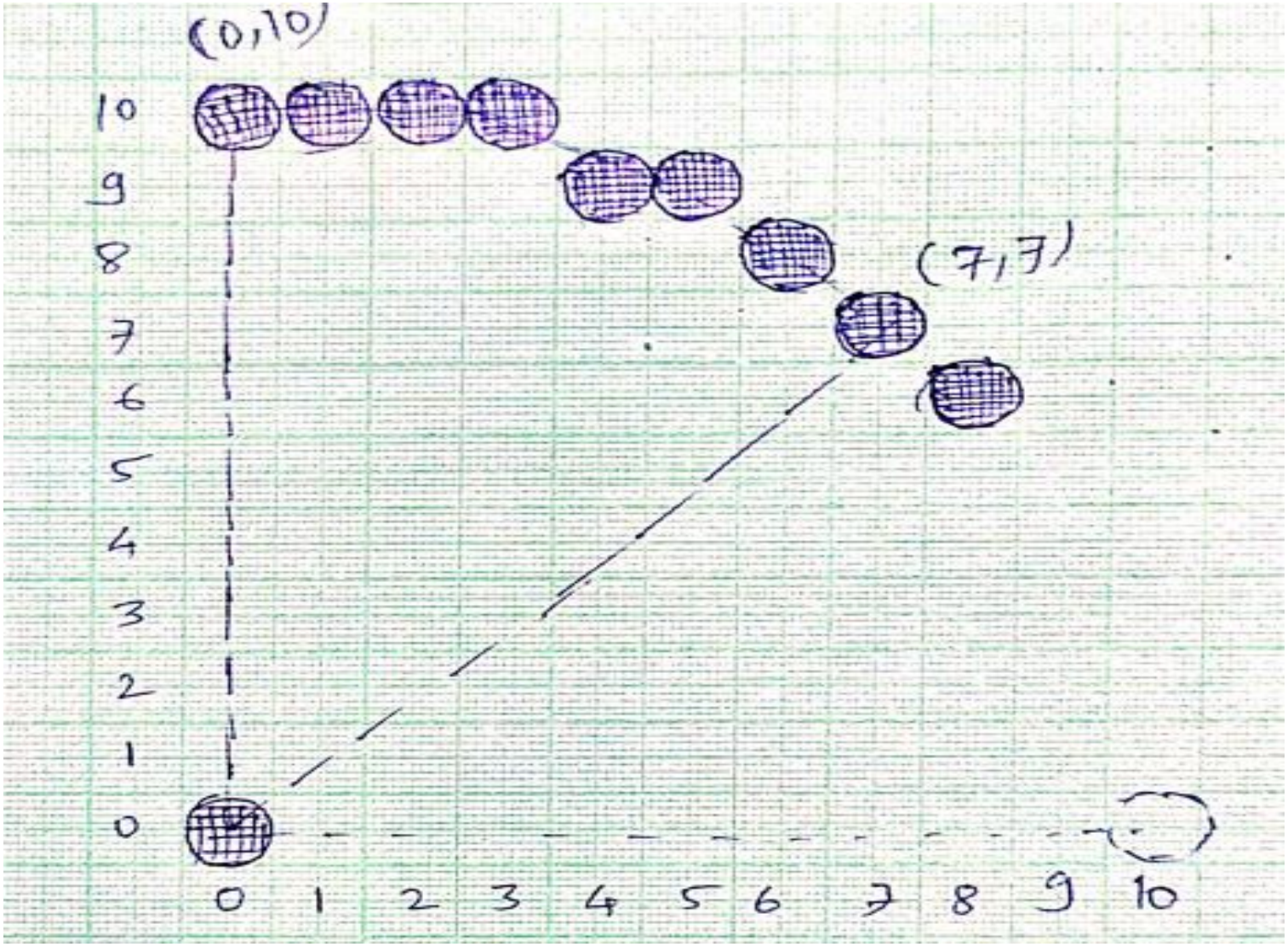
$p_k = (x_k)^2 + (2\,x_k) + 1 + (y_k)^2 - (y_k) + (1/4) - (10)^2$

$p_{k+1} = p_k + 2x_k + 3 + [y_{k+1}^2 - y_k^2] - [y_{k+1} - y_k]$

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

If $p_k \geq 0$, $p_{k+1} = p_k + 2x_k - 2\,y_k + 5$

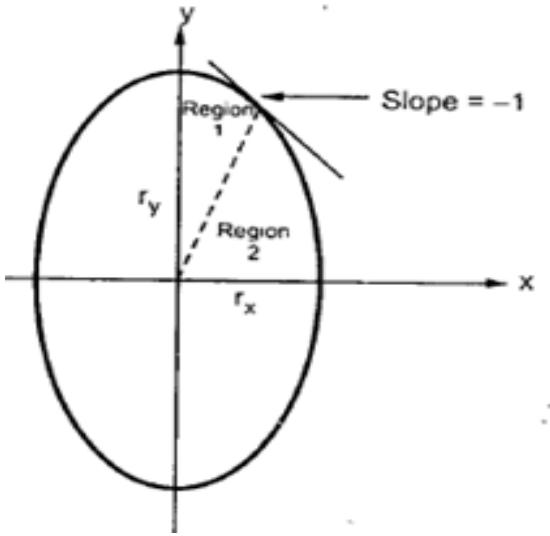
k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)
0	(0, 10)	- 8.75	(1, 10)	**- 5.75**	(1, 10)
1	(1, 10)	- 5.75	(2, 10)	- 0.75	(2, 10)
2	(2, 10)	- 0.75	(3, 10)	6.25	(3, 10)
3	(3, 10)	6.25	(4, 9)	- 2.75	(4, 9)
4	(4, 9)	- 2.75	(5, 9)	8.25	(5, 9)
5	(5, 9)	8.25	(6, 8)	5.25	(6, 8)
6	(6, 8)	5.25	(7, 7)	6.25	(7, 7)
7	(7,7)	6.25	(8,6)	--	--

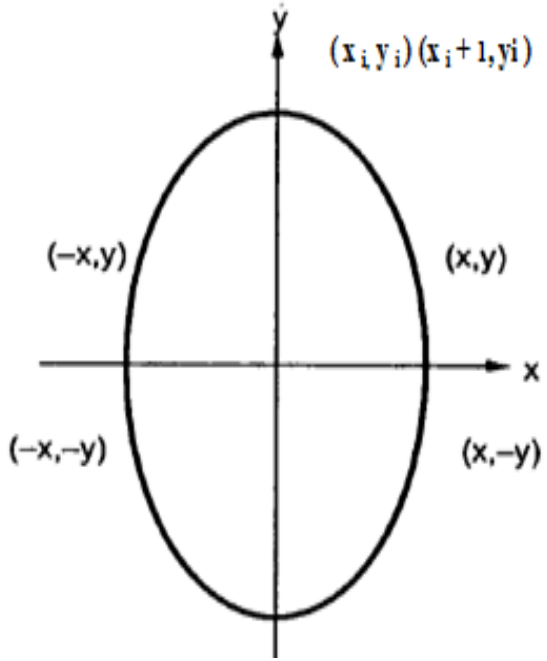

(0,10)
(7,7)
10
9
8
7
6
5
4
3
2
1
0
0 1 2 3 4 5 6 7 8 9 10

Midpoint Ellipse Drawing

It uses the four-way symmetry of the ellipse.

It plots 1/4th part of the ellipse i.e. from 90^0 to 0^0, as shown





x moves in the positive direction

y moves in the negative direction and it passes through two regions.

While processing first quadrant, take steps in the x-direction where the slope of the curve has a magnitude less than -1(for region 1).

and take steps in the y-direction where the slope has a magnitude greater than -1 (for region 2).

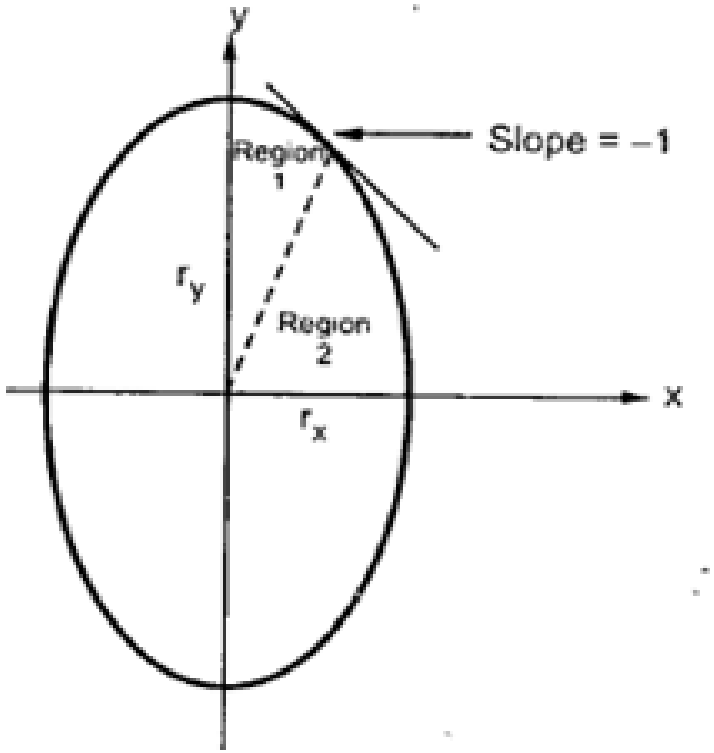
The equation of ellipse with center at origin is

$(x^2/r_x^2) + (y^2/r_y^2) = 1$

$x^2 r_y^2 + y^2 r_x^2 = r_x^2 . r_y^2$

$x^2 r_y^2 + y^2 r_x^2 - r_x^2 . r_y^2 = 0$

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



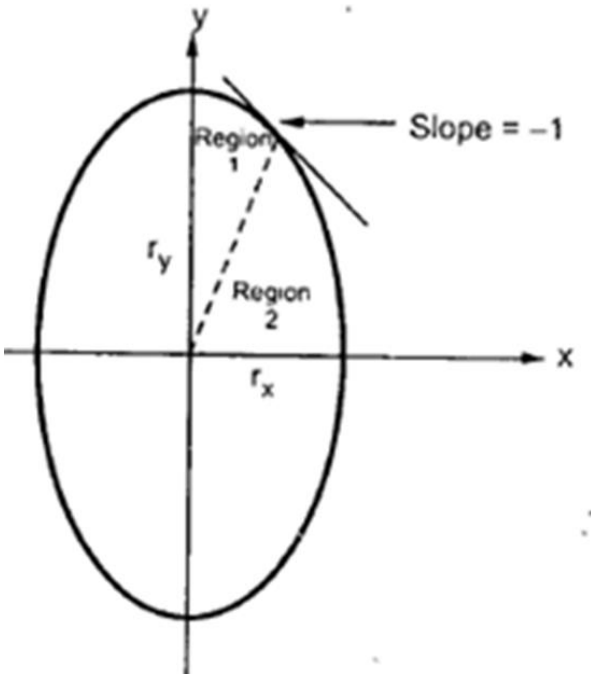
The ellipse function serves as the decision parameter.

At each sampling position, the next pixel (x_{next}, y_{next}) along the ellipse path is selected according to the sign of the ellipse function evaluated at mid-point between the pixels (x_{k+1}, y_k) OR (x_{k+1}, y_{k-1}) for region-1 and (x_k, y_{k-1}) OR (x_{k+1}, y_{k-1}) for region-2.

Starting at (0, r) take unit steps in the x-direction until the boundary between region-1 and region-2 reach.

Then take unit steps in the y-direction over the remainder of the curve in the first quadrant.

To check for boundary point between region-1 and region-2, test the value of the slope of the curve at each step.



The slope of the ellipse at each step is given as

$dy/dx = - (2 r_y^2 x / 2 r_x^2 y)$

At the boundary point between region-1 and region-2, slope is -1

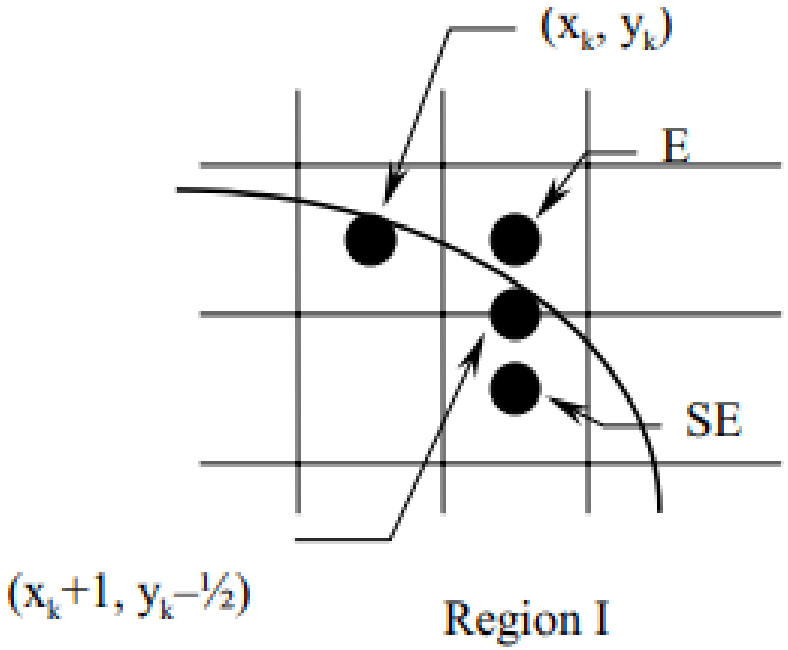
$\therefore\ 2 r_y^2 x = 2 r_x^2 y$

$\therefore$ when $2 r_y^2 x \geq 2 r_x^2 y$, take unit steps in the y-direction.

For Region-1 (Slope < 1)

Consider the pixel at (x_k, y_k).

The next pixel (x_{next}, y_{next}) along the ellipse path will be either (x_{k+1}, y_k) OR (x_{k+1}, y_{k-1}) whichever is closer to the ellipse path at each step.



Let the decision parameter p_k' is equal to the ellipse function evaluated at the mid-point between two pixels.

$$\therefore \quad p_k' = f_{ellipse}(x_{k+1}, y_k - 1/2)$$

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

$$= (x_k + 1)^2 r_y^2 + (y_k - 1/2)^2 r_x^2 - r_x^2 . r_y^2$$

$$p_k' = f_{ellipse}(x_{k+1}, y_k - 1/2)$$
$$= f_{circle}(x_k + 1, y_k - 1/2)$$
$$\mathbf{p_k' = (x_k + 1)^2 r_y^2 + (y_k - 1/2)^2 r_x^2 - r_x^2 . r_y^2}$$

Now

$$p_{k+1}' = (x_{k+1} + 1)^2 r_y^2 + (y_{k+1} - 1/2)^2 r_x^2 - r_x^2 . r_y^2$$
$$= (x_k + 1 + 1)^2 r_y^2 + (y_{k+1} - 1/2)^2 r_x^2 - r_x^2 . r_y^2$$
$$= (x_k + 2)^2 r_y^2 + (y_{k+1} - 1/2)^2 r_x^2 - r_x^2 . r_y^2$$

Now

$$p_{k+1}' - p_k' = (x_k + 2)^2 r_y^2 + (y_{k+1} - 1/2)^2 r_x^2 - r_x^2 . r_y^2$$
$$- [(x_k + 1)^2 r_y^2 + (y_k - 1/2)^2 r_x^2 - r_x^2 . r_y^2]$$
$$= (x_k + 2)^2 r_y^2 + (y_{k+1} - 1/2)^2 r_x^2 - r_x^2 . r_y^2$$
$$- (x_k + 1)^2 r_y^2 - (y_k - 1/2)^2 r_x^2 + r_x^2 . r_y^2$$
$$= r_y^2 (x_k^2 + 4x_k + 4 - x_k^2 - 2x_k - 1) + r_x^2 [(y_{k+1} - 1/2)^2 - (y_k - 1/2)^2]$$
$$= r_y^2 (2x_k + 2 + 1) + r_x^2 [(y_{k+1} - 1/2)^2 - (y_k - 1/2)^2]$$

$$p_{k+1}' - p_k' = r_y^2 (2x_k + 2 + 1) + r_x^2 [(y_{k+1} - 1/2)^2 - (y_k - 1/2)^2]$$

$$= 2 r_y^2 (x_k + 1) + r_y^2 + r_x^2 [(y_{k+1} - 1/2)^2 - (y_k - 1/2)^2]$$

$$\mathbf{p_{k+1}' = p_k' + 2 r_y^2 (x_k + 1) + r_y^2 + r_x^2 [(y_{k+1} - 1/2)^2 - (y_k - 1/2)^2]}$$

If $p_k' < 0$, then

mid-point is INSIDE the ellipse boundary and the pixel at y_k is closer to the ellipse boundary.

Otherwise

mid-point is OUTSIDE or ON the ellipse boundary and the pixel at y_{k-1} is closer to the ellipse boundary.

If $p_k' < 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k + 1}$$

$$\mathbf{y_{next} = y_{k+1} = (y_{k-1}) = y_k}$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_{k+1} - 1/2)^2 - (y_k - 1/2)^2]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_k - 1/2)^2 - (y_k - 1/2)^2]$$

$$\mathbf{p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2}$$

If $p_k' \geq 0$, then

$$\mathbf{x_{next} = x_{k+1} = x_k + 1}$$

$$\mathbf{y_{next} = y_{k+1} = (y_{k-1}) = y_k - 1}$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_{k+1} - 1/2)^2 - (y_k - 1/2)^2]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_{k+1}^2 - y_{k+1} + 1/4) - (y_k^2 - y_k + 1/4)]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[y_{k+1}^2 - y_{k+1} + 1/4 - y_k^2 + y_k - 1/4]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k)]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_k - 1)^2 - y_k^2) - (y_k - 1 - y_k)]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[y_k^2 - 2y_k + 1 - y_k^2 + 1]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k)]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[(y_k - 1)^2 - y_k^2) - (y_k - 1 - y_k)]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[y_k^2 - 2y_k + 1 - y_k^2 + 1]$$

$$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 + r_x^2\,[-2y_k + 1 + 1]$$

$$\mathbf{p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 - 2\,r_x^2\,(y_k - 1)}$$

Now, Starting point in region-1 is $(x_0, y_0) = (0, r_y)$

$\therefore$ Initial value of decision parameter is

$$\mathbf{p_k' = (x_k + 1)^2\,r_y^2 + (y_k - 1/2)^2\,r_x^2 - r_x^2.\,r_y^2}$$

$$p_0' = (x_0 + 1)^2\,r_y^2 + (y_0 - 1/2)^2\,r_x^2 - r_x^2.\,r_y^2$$

$$p_0' = (0 + 1)^2\,r_y^2 + (r_y - 1/2)^2\,r_x^2 - r_x^2.\,r_y^2$$

$$p_0' = r_y^2 + (r_y^2 - r_y + 1/4)\,r_x^2 - r_x^2.\,r_y^2$$

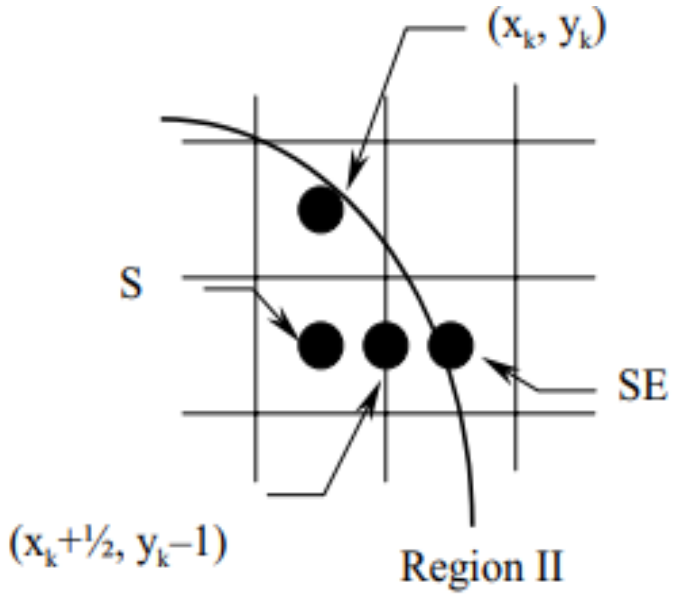
$$p_0' = r_y^2 + r_x^2\,(r_y^2 - r_y + 1/4 - r_y^2)$$

$$p_0' = r_y^2 + r_x^2\,(-r_y + 1/4)$$

$$\therefore\ \mathbf{p_0' = r_y^2 - r_x^2\,r_y + 1/4\,r_x^2}$$

For Region-2 (Slope > 1)

Consider the pixel at (x_k, y_k).



The next pixel (x_{next}, y_{next}) along the ellipse path will be either (x_k, y_{k-1}) OR (x_{k+1}, y_{k-1}) whichever is closer to the ellipse path at each step.

Let the decision parameter p_k" is equal to the ellipse function evaluated at the mid-point between two pixels.

$$\therefore \quad p_k'' = f_{ellipse}(x_{k+1}, y_{k-1})$$

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

$$= (x_k + 1/2)^2 r_y^2 + (y_k - 1)^2 r_x^2 - r_x^2 . r_y^2$$

$\therefore \quad p_k'' = f_{ellipse}(x_{k+1}, y_{k-1})$

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

$\therefore \quad \mathbf{p_k'' = (x_k + 1/2)^2 r_y^2 + (y_k - 1)^2 r_x^2 - r_x^2 . r_y^2}$

Now

$\therefore \quad p_{k+1}'' = f_{ellipse}(x_{k+1} + 1/2, y_{k+1} - 1)$

$= (x_{k+1} + 1/2)^2 r_y^2 + (y_k - 1 - 1)^2 r_x^2 - r_x^2 . r_y^2$

$= (x_{k+1} + 1/2)^2 r_y^2 + (y_k - 2)^2 r_x^2 - r_x^2 . r_y^2$

Now

$p_{k+1}'' - p_k'' = (x_{k+1} + 1/2)^2 r_y^2 + (y_k - 2)^2 r_x^2 - r_x^2 . r_y^2$

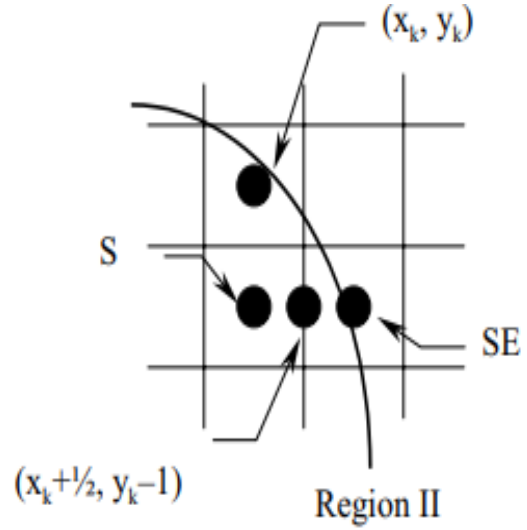
$- [(x_k + 1/2)^2 r_y^2 + (y_k - 1)^2 r_x^2 - r_x^2 . r_y^2]$

$p_{k+1}'' - p_k'' = (x_{k+1} + 1/2)^2 r_y^2 + (y_k - 2)^2 r_x^2 - r_x^2 . r_y^2$

$- (x_k + 1/2)^2 r_y^2 - (y_k - 1)^2 r_x^2 + r_x^2 . r_y^2]$

$p_{k+1}'' - p_k'' = r_y^2 [(x_{k+1} + 1/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$

$p_{k+1}'' = p_k'' + r_y^2 [(x_{k+1} + 1/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$



$$p_{k+1}'' = p_k'' + r_y^2 [(x_{k+1} + 1/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$$

Now

If $p_k'' > 0$, then

$$x_{next} = x_{k+1} = x_k$$

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

$$p_{k+1}'' = p_k'' + r_y^2 [(x_{k+1} + 1/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$$

$$p_{k+1}'' = p_k'' + r_y^2 [(x_k + 1/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$$

$$p_{k+1}'' = p_k'' + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$$

$$= p_k'' + r_x^2[(y_k^2 - 4y_k + 4) - (y_k^2 - 2y_k + 1)]$$

$$= p_k'' + r_x^2[y_k^2 - 4y_k + 4 - y_k^2 + 2y_k - 1]$$

$$= p_k'' + r_x^2[- 4y_k + 4 + 2y_k - 1]$$

$$= p_k'' + r_x^2[- 2y_k + 3]$$

$$p_{k+1}'' = p_k'' - 2 r_x^2 y_k + 3 r_x^2$$

Now

If $p_k' \leq 0$, then

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1}'' = p_k'' + r_y^2 [(x_{k+1} + 1/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$$

$$p_{k+1}'' = p_k'' + r_y^2 [(x_k + 1 + 1/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$$

$$= p_k'' + r_y^2 [(x_k + 3/2)^2 - (x_k + 1/2)^2] + r_x^2[(y_k - 2)^2 - (y_k - 1)^2]$$

$$= p_k'' + r_y^2 [(x_k^2 + 3x_k + 9/4) - (x_k^2 + x_k + 1/4)] + r_x^2[- 2y_k + 3]$$

$$= p_k'' + r_y^2 [x_k^2 + 3x_k + 9/4 - x_k^2 - x_k - 1/4] + r_x^2[- 2y_k + 3]$$

$$= p_k'' + r_y^2 [2x_k + 8/4] + r_x^2[- 2y_k + 3]$$

$$= p_k'' + r_y^2 [2x_k + 2] + r_x^2[- 2y_k + 3]$$

$$p_{k+1}'' = p_k'' + 2r_y^2 [x_k + 1] - 2 r_x^2 y_k + 3 r_x^2$$

Now,

Starting point in region-2 is (x_0, y_0) = last position in region-1

$\therefore$ Initial value of decision parameter for region-2 is

$$p_k'' = f_{ellipse}(x_{k+1}, y_{k-1})$$

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

$$p_0'' = f_{ellipse}(x_0 + 1/2, y_0 - 1)$$

$$\mathbf{p_0'' = (x_0 + 1/2)^2 r_y^2 + (y_0 - 1)^2 r_x^2 - r_x^2 . r_y^2}$$

Midpoint Ellipse Drawing Algorithm

1. Accept the radii of the ellipse 'r_x' , 'r_y' and center of the ellipse (x_c, y_c)

2. Initialize the starting position as $(x, y) = (0, r_y)$

3. Calculate the initial value of decision parameter for region-1 , p_0' as

$$p_0' = r_y^2 - r_x^2 r_y + 1/4\, r_x^2$$

4. At each x_k position in region-1 starting at k = 0 perform the following test

do

{ putpixel (x, y, COLOR)

If $p_k' < 0$

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1}' = p_k' + 2\, r_y^2\, (x_k + 1) + r_y^2$

}

else

{ $x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1}' = p_k' + 2\, r_y^2\, (x_k + 1) + r_y^2 - 2\, r_x^2\, (y_k - 1)$

}

} while $(2\, r_y^2\, x < 2\, r_x^2 y)$

5. Calculate the initial value of decision parameter for region-2 as

$$p_0'' = (x_0 + 1/2)^2 r_y^2 + (y_0 - 1)^2 r_x^2 - r_x^2 . r_y^2$$

6. At each x_k position in region-2 starting at k = 0 perform the following test

do { putpixel (x, y, COLOR)

If $p_k'' > 0$ {

$x_{next} = x_{k+1} = x_k$

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

$p_{k+1}'' = p_k'' - 2 r_x^2 y_k + 3 r_x^2$

}

else {

$x_{next} = x_{k+1} = x_k + 1$

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

$p_{k+1}'' = p_k'' + 2r_y^2 [x_k + 1] - 2 r_x^2 y_k + 3 r_x^2$

}

} while (y > 0)

7. Determine the symmetry points in the other three quadrant.

8. Translate each pixel position by (x_c, y_c)

$$x = x_{k+1} + x_c$$

$$y = y_{k+1} + y_c$$

putpixel (x, y, COLOR)

9. Stop

Ex1- Calculate the pixel positions along the ellipse path with radii $r_x = 4$, $r_y = 3$ and centered at the origin using mid-point ellipse drawing algorithm.

Radii of the ellipse , $r_x = 4$, $r_y = 3$

and center of the circle, $(x_c, y_c) = (0, 0)$.

Initialize the starting position as $(x, y) = (0, r_y) = (0, 3)$

Initial value of decision parameter for region-1

$$p_0' = r_y^2 - r_x^2 r_y + 1/4\, r_x^2$$

$$= 3^2 - 4^2 . 3 + 1/4\, 4^2$$

$$= 9 - 48 + 4$$

$$\mathbf{p_0' = -35}$$

At each x_k position in region-1 starting at k = 0 perform the following test

do { putpixel (x, y, COLOR)

If $p_k' < 0$ {

$\mathbf{x_{next} = x_{k+1} = x_k + 1}$

$\mathbf{y_{next} = y_{k+1} = y_k}$

$p_{k+1}' = p_k' + 2\,(3)^2\,(x_k + 1) + (3)^2$

$\mathbf{p_{k+1}' = p_k' + 18\,(x_k + 1) + 9}$

}

else {

$\mathbf{x_{next} = x_{k+1} = x_k + 1}$

$\mathbf{y_{next} = y_{k+1} = y_k - 1}$

$p_{k+1}' = p_k' + 2\,r_y^2\,(x_k + 1) + r_y^2 - 2\,r_x^2\,(y_k + 1)$

$p_{k+1}' = p_k' + 2\,(3)^2\,(x_k + 1) + (3)^2 - 2\,(4)^2\,(y_k + 1)$

$\mathbf{p_{k+1}' = p_k' + 18\,(x_k + 1) + 9 - 32\,(y_k - 1)}$

}

} while $(2\,(3)^2\,x < 2\,(4)^2 y)$ i.e while $(18x < 32y)$

If $p_k' < 0$,

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1}' = p_k' + 18\,(x_k + 1) + 9$$

else,

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1}' = p_k' + 18\,(x_k + 1) + 9 - 32\,(y_k - 1)$$

k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)	18x < 32y
0	(0, 3)	- 35	(1, 3)	**- 8**	(1, 3)	18 < 96
1	(1, 3)	- 8	(2, 3)	37	(2, 3)	36 < 96
2	(2, 3)	37	(3, 2)	36	(3, 2)	54 < 64
3	(3, 2)	36	(4, 1)	85	(4, 1)	**72 < 32**

Now

Starting point in region-2 is (x_0, y_0) = last position in region-1

$\therefore$ $(x_0, y_0) = (4, 1)$

Initial value of decision parameter for region-2 is

$$p_0'' = (x_0 + 1/2)^2 r_y^2 + (y_0 - 1)^2 r_x^2 - r_x^2 . r_y^2$$

$$p_0'' = (4 + 1/2)^2 (3)^2 + (1 - 1)^2 (4)^2 - (4)^2 . (3)^2$$

$$\mathbf{p_0'' = 38.25}$$

do { putpixel (x, y, COLOR)

If p_k” > 0 {

$x_{next} = x_{k+1} = x_k$

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

p_{k+1}” = p_k” - 2 r_x^2 y_k + 3 r_x^2

= p_k” - 2 $(4)^2$ y_k + 3 $(4)^2$

= p_k” - 32y_k + 48

}

else {

$x_{next} = x_{k+1} = x_k + 1$

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

p_{k+1}” = p_k” + 2r_y^2 [x_k + 1] - 2 r_x^2 y_k + 3 r_x^2

p_{k+1}” = p_k” + 2$(3)^2$ [x_k + 1] - 2 $(4)^2$ y_k + 3$(4)^2$

p_{k+1}” = p_k” + 18(x_k + 1) - 32y_k + 48

}

} while (y > 0)

If $p_k'' > 0$,

$x_{next} = x_{k+1} = x_k$

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

$p_{k+1}'' = p_k'' - 32y_k + 48$

else

$x_{next} = x_{k+1} = x_k + 1$

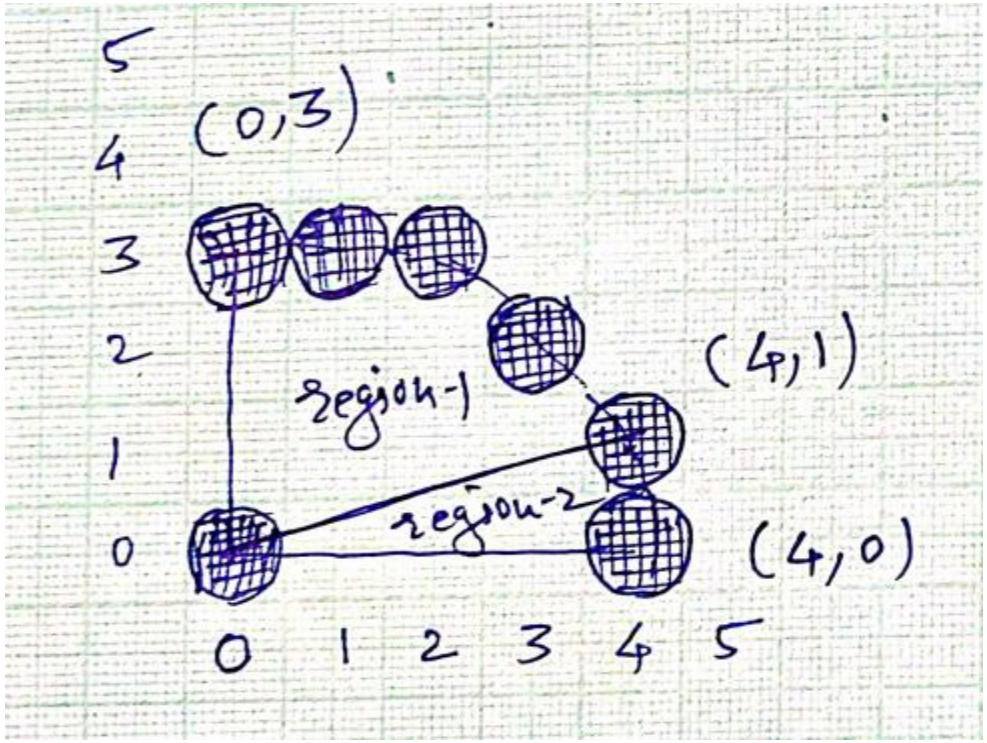
$y_{next} = y_{k+1} = y_{k-1} = y_k - 1$

$p_{k+1}'' = p_{k+1}'' = p_k'' + 18(x_k + 1) - 32y_k + 48$

k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)	y > 0
0	(4, 1)	38.25	(4, 0)		(4, 0)	0 > 0

k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)	18x < 32y
0	(0, 3)	- 35	(1, 3)	**- 8**	(1, 3)	18 < 96
1	(1, 3)	- 8	(2, 3)	37	(2, 3)	36 < 96
2	(2, 3)	37	(3, 2)	36	(3, 2)	54 < 64
3	(3, 2)	36	(4, 1)	85	(4, 1)	**72 < 32**

k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)	y > 0
0	(4, 1)	38.25	(4, 0)		(4, 0)	0 > 0



Ex2- Calculate the pixel positions along the ellipse path with radii $r_x = 8$, $r_y = 6$ and centered at the origin using mid-point ellipse drawing algorithm.

Radii of the ellipse , $r_x = 8$, $r_y = 6$

and center of the circle, $(x_c, y_c) = (0, 0)$.

Initialize the starting position as $(x, y) = (0, r_y) = (0, 6)$

Initial value of decision parameter for region-1

$$p_0' = r_y^2 - r_x^2 r_y + 1/4\, r_x^2$$

$$= 6^2 - 8^2 \cdot 6 + 1/4\, 8^2$$

$$= 36 - 384 + 16$$

$$\mathbf{p_0' = -332}$$

do {

putpixel (x, y, COLOR)

If $p_k' < 0$ {

$\mathbf{x_{next} = x_{k+1} = x_k + 1}$

$\mathbf{y_{next} = y_{k+1} = y_k}$

$\mathbf{p_{k+1}' = p_k' + 2\, r_y^2\, (x_k + 1) + r_y^2}$

$p_{k+1}' = p_k' + 2\,(6)^2\,(x_k + 1) + (6)^2$

$\mathbf{p_{k+1}' = p_k' + 72\,(x_k + 1) + 36}$

}

else {

$\mathbf{x_{next} = x_{k+1} = x_k + 1}$

$\mathbf{y_{next} = y_{k+1} = y_k - 1}$

$p_{k+1}' = p_k' + 2\, r_y^2\, (x_k + 1) + r_y^2 - 2\, r_x^2\, (y_k + 1)$

$p_{k+1}' = p_k' + 2\,(6)^2\,(x_k + 1) + (6)^2 - 2\,(8)^2\,(y_k + 1)$

$\mathbf{p_{k+1}' = p_k' + 72\,(x_k + 1) + 36 - 128\,(y_k - 1)}$

}

} while $(2\,(6)^2 x < 2\,(8)^2 y)$ i.e while $(72x < 128y)$

If $p_k' < 0$,

$x_{next} = x_{k+1} = x_k + 1$

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

$$\mathbf{p_{k+1}' = p_k' + 72\,(x_k + 1) + 36}$$

Else,

$x_{next} = x_{k+1} = x_k + 1$

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

$$\mathbf{p_{k+1}' = p_k' + 72\,(x_k + 1) + 36 - 128\,(y_k - 1)}$$

k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)	72x < 128y
0	(0, 6)	- 332	(1, 6)	-224	(1, 6)	72 < 768
1	(1, 6)	-224	(2, 6)	-44	(2, 6)	144 < 768
2	(2, 6)	-44	(3, 6)	208	(3, 6)	216 < 768
3	(3, 6)	208	(4, 5)	-108	(4, 5)	288 < 640
4	(4, 5)	-108	(5, 5)	288	(5, 5)	360 < 640
5	(5, 5)	288	(6, 4)	244	(6, 4)	432 < 512
6	(6, 4)	244	(7, 3)	400	(7, 3)	504 < 384

Now

Starting point in region-2 is (x_0, y_0) = last position in region-1

$\therefore$ $(x_0, y_0) = (7, 3)$

Initial value of decision parameter for region-2 is

$$p_0'' = (x_0 + 1/2)^2 r_y^2 + (y_0 - 1)^2 r_x^2 - r_x^2 . r_y^2$$

$$p_0'' = (7 + 1/2)^2 (6)^2 + (3 - 1)^2 (8)^2 - (8)^2 . (6)^2$$

$$\mathbf{p_0'' = - 23}$$

do { putpixel (x, y, COLOR)

If p_k” > 0 {

$x_{next} = x_{k+1} = x_k$

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

p_{k+1}” = p_k” - 2 r_x^2 y_k + 3 r_x^2

= p_k” - 2 $(8)^2$ y_k + 3 $(8)^2$

= p_k” - 128y_k + 192

}

else {

$x_{next} = x_{k+1} = x_k + 1$

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

p_{k+1}” = p_k” + 2r_y^2 [x_k + 1] - 2 r_x^2 y_k + 3 r_x^2

p_{k+1}” = p_k” + 2$(6)^2$ [x_k + 1] - 2 $(8)^2$ y_k + 3$(8)^2$

p_{k+1}” = p_k” + 72(x_k + 1) - 128y_k + 192

}

} while (y > 0)

If $p_k'' > 0$,

$$x_{next} = x_{k+1} = x_k$$

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

$$p_{k+1}'' = p_k'' - 128y_k + 192$$

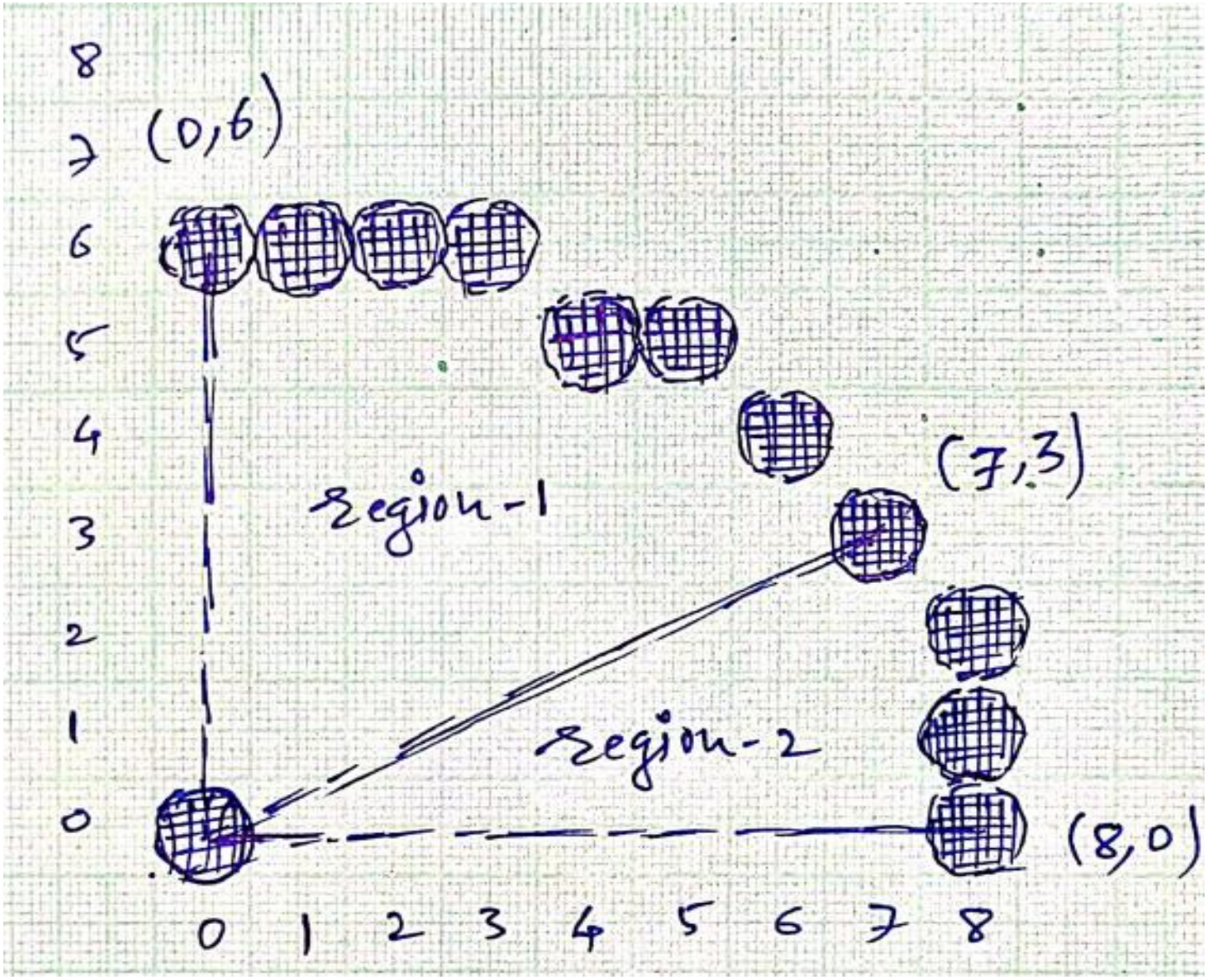
else

$$x_{next} = x_{k+1} = x_k + 1$$

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

$$p_{k+1}'' = p_k'' + 72(x_k + 1) - 128y_k + 192$$

k	(x_k, y_k)	p_k	(x_{k+1}, y_{k-1})	p_{k+1}	Putpixel (int x, int y)	y > 0
0	(7, 3)	-23	(8, 2)	**361**	(8, 2)	2 > 0
1	(8, 2)	361	(8, 1)	**297**	(8, 1)	1 > 0
2	(8, 1)	297	(8, 0)	**361**	(8, 0)	**0 > 0**


8
7
6
5
4
3
2
1
0
(0,6)
(7,3)
(8,0)
Region-1
Region-2
0 1 2 3 4 5 6 7 8