ICT 4203

Computer Graphics and Animation

Lecture 04

Bresenham Circle Drawing Algorithm

Md. Mahmudur Rahman
Lecturer
Institute of Information Technology

Contents

- **Circle Equation:**
  - Polar Form
  - Cartesian Form
- **Circle Drawing Algorithms:**
  - Bresenham's Circle Algorithm

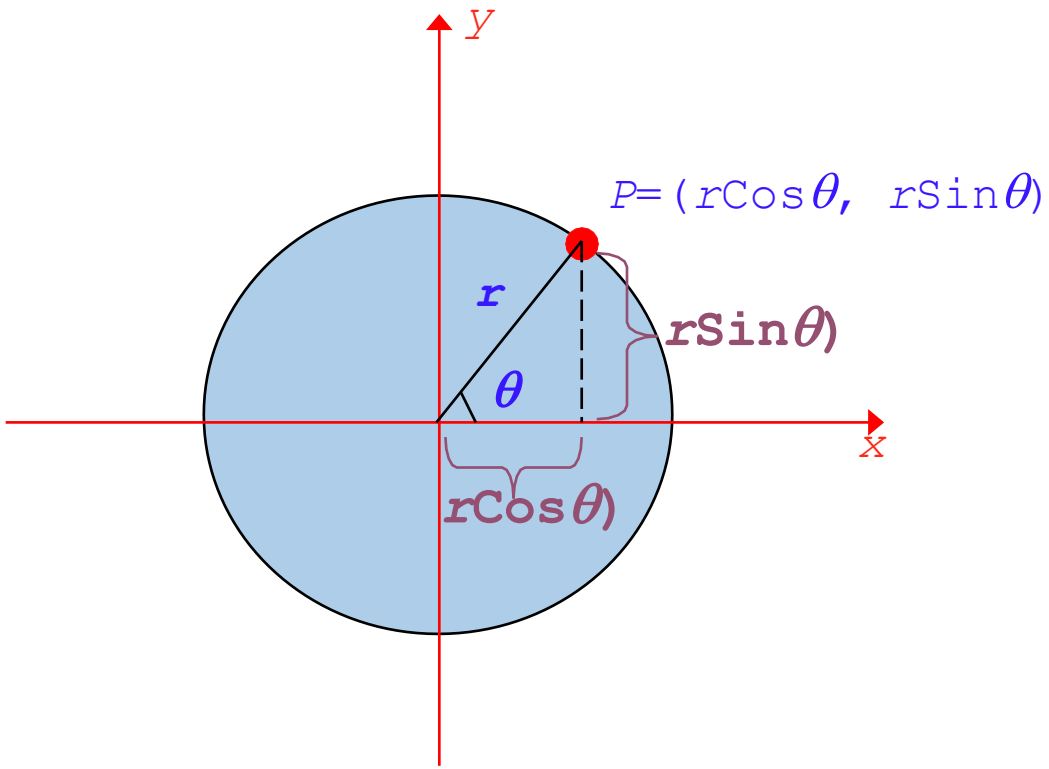
Circle Equations

- Polar form

  $x = r\text{Cos}\theta$

  $y = r\text{Sin}\theta$

(r = radius of circle)



Drawing a Circle

```
θ = 0°
while (θ < 360°)
        x = rCosθ
        y = rSinθ
        setPixel(x,y)
        θ = θ + 1°
end while
```

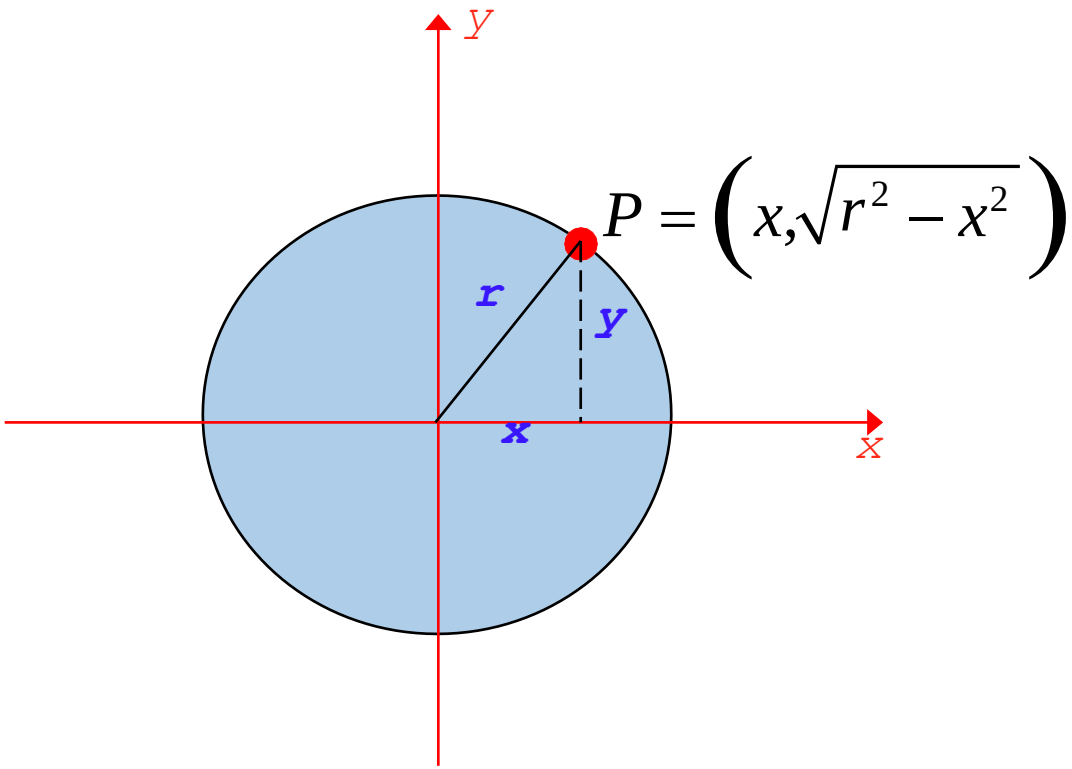
Disadvantages

- To find a complete circle, θ varies from 0° to 360°
- The calculation of trigonometric functions is very slow.

Cartesian Form

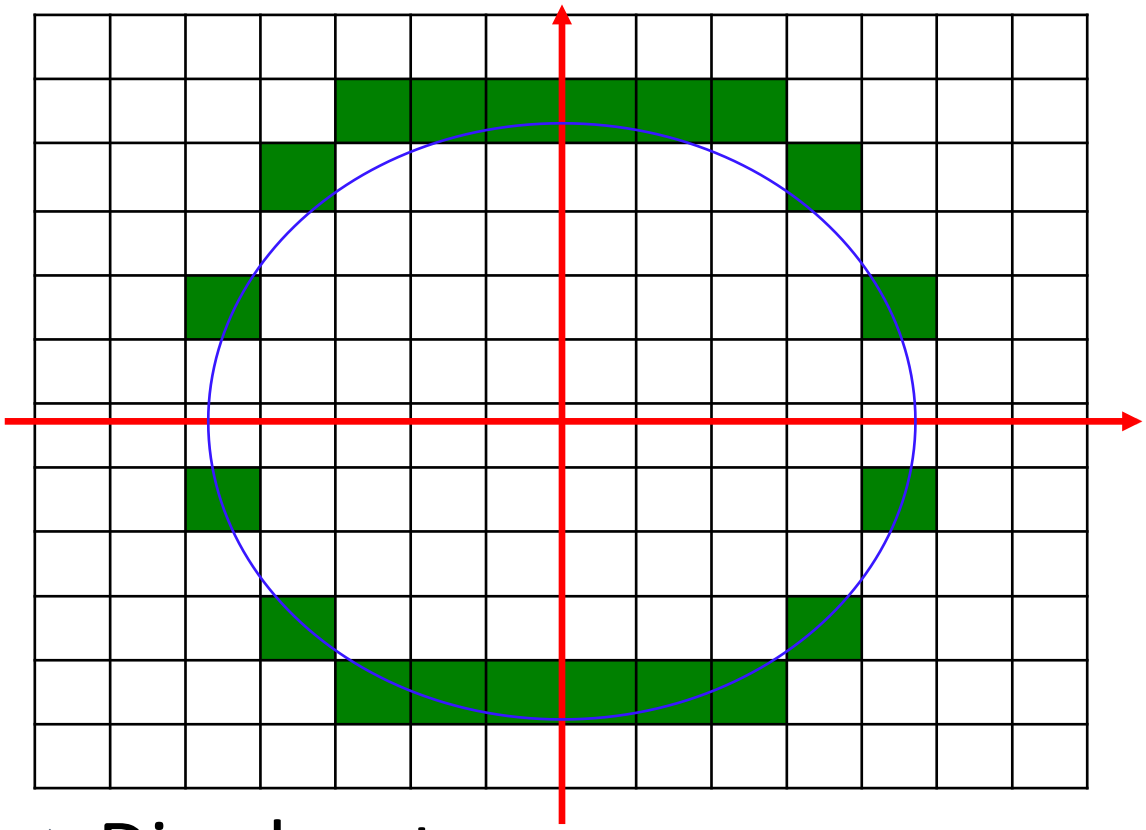
- Use Pythagoras theorem

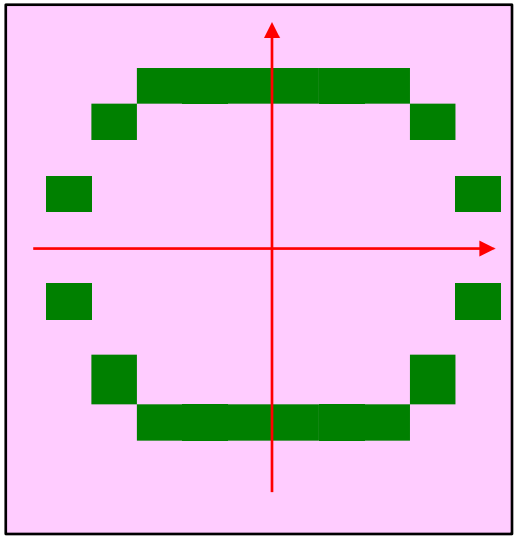
$x^2 + y^2 = r^2$



Drawing a Circle

- Step through *x*-axis to determine *y*-values

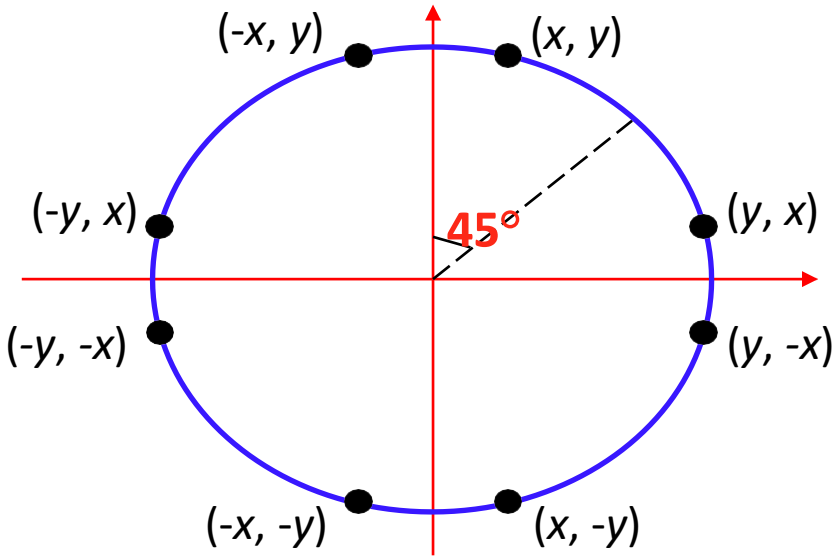




- Disadvantages:
  - Not all pixel filled in
  - Square root function is very slow

Circle

- Circle is an eight-way symmetric figure.
- The shape of circle is the same in all quadrants.
- In each quadrant, there are two octants.
- If the calculation of the point of one octant is done, then the other seven points can be calculated easily by using the concept of eight-way symmetry.

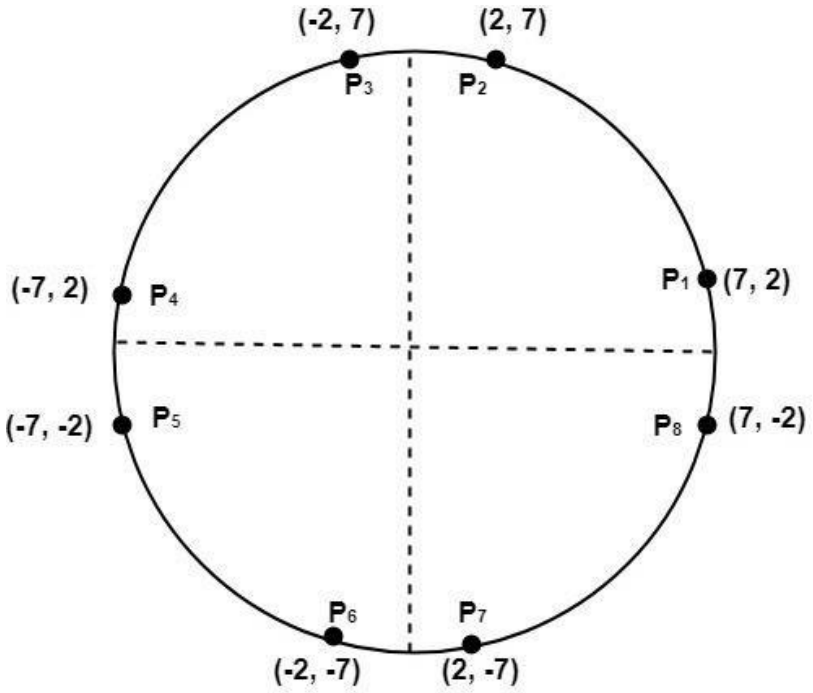


putpixel (x, y, color)
putpixel (x, -y, color)
putpixel (-x, y, color)
putpixel (-x, -y, color)
putpixel (y, x, color)
putpixel (y, -x, color)
putpixel (-y, x, color)
putpixel (-y, -x, color)

- If we want to display circle on screen then the putpixel() function is used for eight points as shown:

Continue…

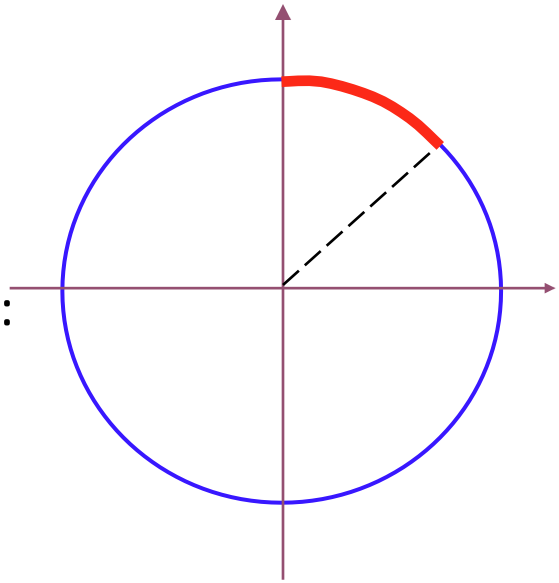
- Let we determine a point (2, 7) of the circle then other points will be (2, -7), (-2, -7), (-2, 7), (7, 2), (-7, 2), (-7, -2), (7, -2).
- These seven points are calculated by using the property of reflection. The reflection is accomplished by reversing x, y co-ordinates.



Eight way symmetry of a Circle

Bresenham's Circle Algorithm

- Scan-Converting a circle using Bresenham's algorithm works as follows:
- Points are generated from 90° to 45°, moves will be made only in the +x & -y directions as shown in fig:



- We want to generate the points from 90° to 45°. Assume that the last scan-converted pixel is P_1. Each new point closest to the true circle can be found by taking either of two actions:
- Move in the x-direction one unit or
- Move in the x- direction one unit & move in the negative y-direction one unit.

Bresenham's Circle Algorithm

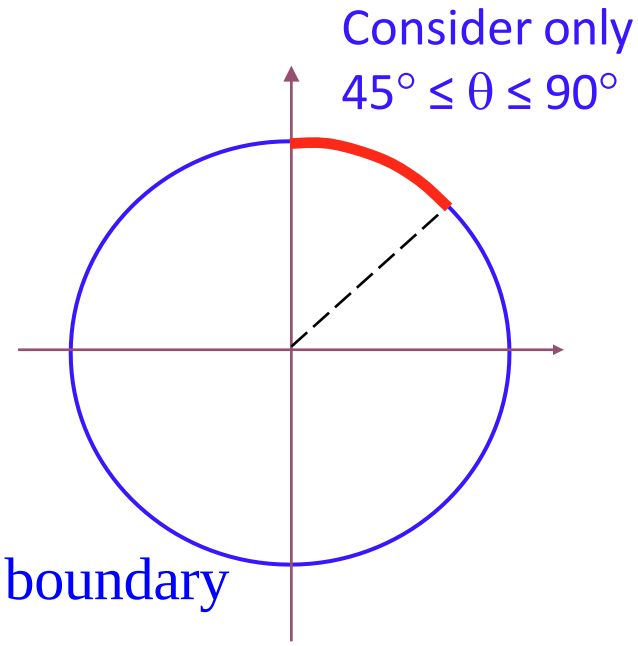
General Principle

- The circle function:

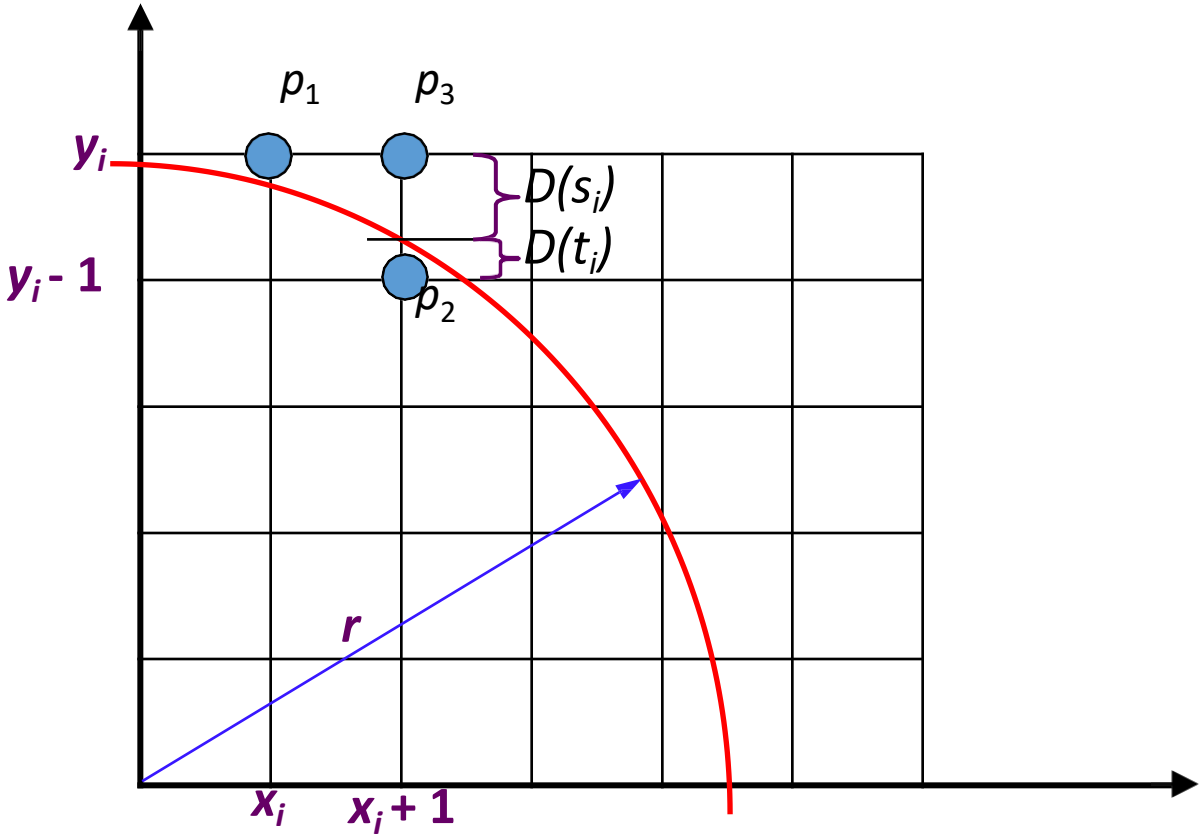
$$f_{circle}(x, y) = x_2 + y_2 - r_2$$

and

$$f_{circle}(x, y) = \begin{cases} < 0 & \text{if (x,y) is inside the circle boundary} \\ = 0 & \text{if (x,y) is on the circle boundary} \\ > 0 & \text{if (x,y) is outside the circle boundary} \end{cases}$$



Bresenham's Circle Algorithm



After point p_1, do we choose p_2 or p_3?

Bresenham's Circle Algorithm

Define: $D(s_i)$ = distance of p_3 from circle

$D(t_i)$ = distance of p_2 from circle

i.e. $D(s_i) = (x_i + 1)^2 + y_i^2 - r^2$ [always +ve]

$D(t_i) = (x_i + 1)^2 + (y_i - 1)^2 - r^2$ [always -ve]

- Decision Parameter $d_i = D(s_i) + D(t_i)$

so if $d_i < 0$ then the circle is closer to p_3 (point above)

if $d_i \geq 0$ then the circle is closer to p_2 (point below)

- From this equation, we can drive initial values of d_i.
- If it is assumed that the circle is centered at the origin, then at the first step x = 0 & y = r.
- Therefore,

$$d_i=(0+1)^2+r^2 -r^2+(0+1)^2+(r-1)^2-r^2$$
$$=1+1+r^2-2r+1-r^2$$
$$= 3 - 2r$$

The Algorithm

$$
\begin{cases}
\text{if } d_i < 0 \text{ then} \\
\quad y_{i+1} = y_i \\
\quad d_{i+1} = d_i + 4x_i + 6 \\
\\
\text{else if } d_i \geq 0 \text{ then} \\
\quad y_{i+1} = y_i - 1 \\
\quad d_{i+1} = d_i + 4(x_i - y_i) + 10
\end{cases}
\qquad x_{i+1} = x_i + 1
$$

- Stop when $x_i \geq y_i$ and determine symmetry points in the other octants.

The Algorithm

- **Step1:** Start Algorithm
- **Step2:** Declare p, q, x, y, r, d variables
  p, q are coordinates of the center of the circle
  r is the radius of the circle
- **Step3:** Enter the value of r
- **Step4:** Calculate d = 3 - 2r
- **Step5:** Initialize x=0
  &nbsy= r
- **Step6:** Check if the whole circle is scan converted
  If x > = y
  Stop
- **Step7:** Plot eight points by using concepts of eight-way symmetry. The center is at (p, q). Current active pixel is (x, y).
  putpixel (x+p, y+q)
  putpixel (y+p, x+q)
  putpixel (-y+p, x+q)
  putpixel (-x+p, y+q)
  putpixel (-x+p, -y+q)
  putpixel (-y+p, -x+q)
  putpixel (y+p, -x+q)
  putpixel (x+p, -y-q)
- **Step8:** Find location of next pixels to be scanned
  If: d < 0
  then d = d + 4x + 6
  increment x = x + 1
  Else If: d ≥ 0
  then d = d + 4 (x - y) + 10
  increment x = x + 1
  decrement y = y - 1
- **Step9:** Go to step 6
- **Step10:** Stop Algorithm

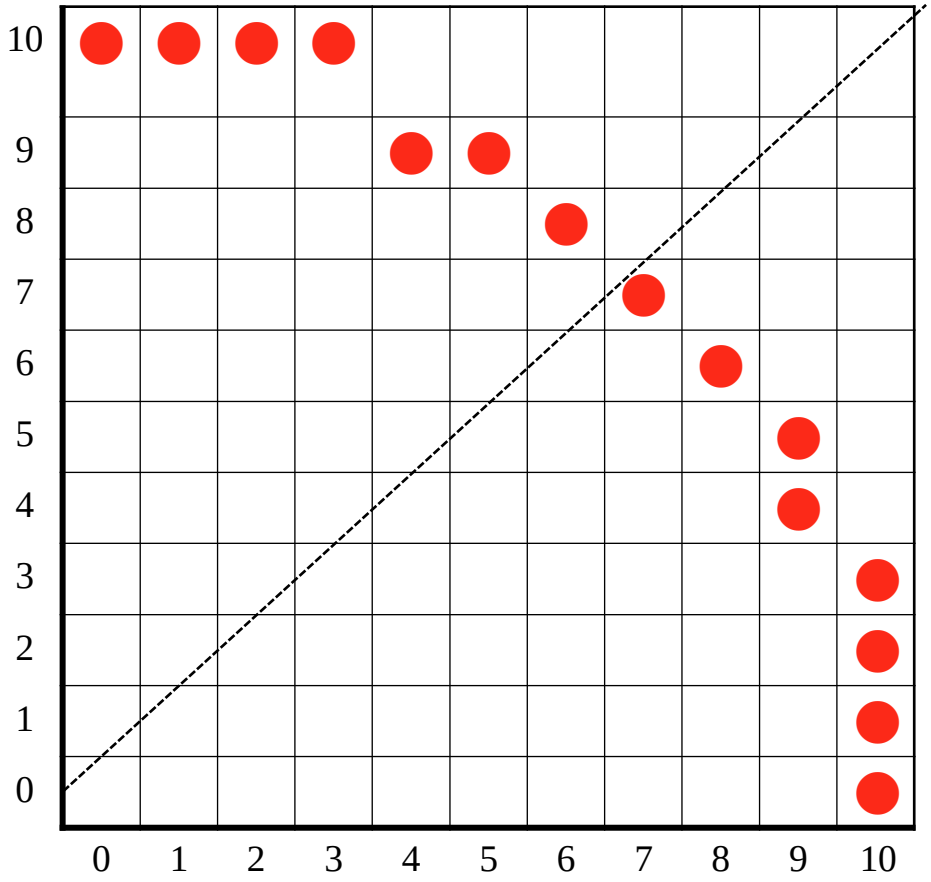
Example-1

$r = 10$

$d_0 = 3 - 2r = -17$

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

i	d_i	(x_i, y_i)
0	-17	(0, 10)
1	-11	(1, 10)
2	-1	(2, 10)
3	13	(3, 10)
4	-5	(4, 9)
5	15	(5, 9)
6	9	(6, 8)
7		(7,7)



Example-2

Plot 6 points of circle using Bresenham Algorithm, when radius of the circle is 10 units. The circle has centre (50, 50).

Example-2

Plot 6 points of circle using Bresenham Algorithm, when radius of the circle is 10 units. The circle has centre (50, 50).

Solution: Let r = 10 (Given)

Step1: Take initial point (0, 10)

d = 3 - 2r

d = 3 - 2 * 10 = -17

d < 0

d = d + 4x + 6

= -17 + 4 (0) + 6

= -11

Step2: Plot (1, 10)

d = d + 4x + 6 (d < 0)

= -11 + 4 (1) + 6

= -1

Step3: Plot (2, 10)

d = d + 4x + 6 (d < 0)

= -1 + 4 x 2 + 6

= 13

Step4: Plot (3, 9)

d > 0

x = x + 1, y = y - 1

d = d + 4 (x-y) + 10 (d > 0)

= 13 + 4 (3-9) + 10

= 13 + 4 (-6) + 10

= 23-24=-1

Step5: Plot (4, 9)

d = -1 + 4x + 6 (d < 0)

= -1 + 4 (4) + 6

= 21

Step6: Plot (5, 8)

d = d + 4 (x-y) + 10 (d > 0)

= 21 + 4 (5-8) + 10

= 21-12 + 10 = 19

Answer:

P_1 (0,0)⟹(50,50)
P_2 (1,10)⟹(51,60)
P_3 (2,10)⟹(52,60)
P_4 (3,9)⟹(53,59)
P_5 (4,9)⟹(54,59)
P_6 (5,8)⟹(55,58)