Vidyavardhini’s College of Engineering & Technology



Department of Computer Engineering

**NAME :- TANMAY PATIL**

**ROLL NO :- 60/ DIV 2**

**EXPERIMENT NO 3**

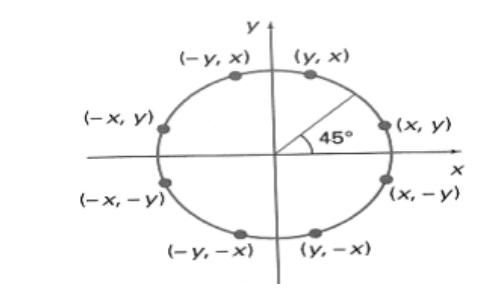
**Aim**: To implement midpoint circle algorithm.

**Objective:**

Draw a circle using mid-point circle drawing algorithm by determining the points needed for rasterizing a circle. The mid-point algorithm to calculate all the perimeter points of the circle in the first octant and then print them along with their mirror points in the other octants.

**Theory:**

The shape of the circle is similar in each quadrant. We can generate the points in one section and the points in other sections can be obtained by considering the symmetry about x-axis and y-axis.



The equation of circle with center at origin is x 2 + y 2 = r 2

Let the circle function is f circle (x, y) -

is &lt; 0, if (x, y) is inside circle boundary,

is = 0, if (x, y) is on circle boundary,

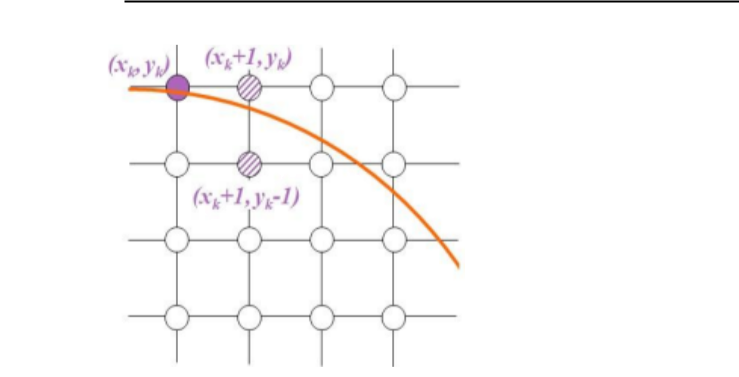
is &gt; 0, if (x, y) is outside circle boundary.

Consider the pixel at (xk, yk) is plotted,

Vidyavardhini’s College of Engineering & Technology



Department of Computer Engineering



Now the next pixel along the circumference of the circle will be either (xk + 1, yk) or (xk + 1, yk – 1) whichever is closer the circle boundary.

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

If pk &lt; 0, the midpoint is inside the circle and the pixel at yk is closer to the circle boundary. Otherwise, the midpoint is outside or on the circle boundary and the pixel at yk – 1 is closer to the circle boundary.

**Algorithm :**

1. Start the graphics mode using `initgraph`.

2. Declare variables to store circle parameters and initial values.

3. Prompt the user to enter the center coordinates (xc, yc) and the radius r.

4. Initialize the center point of the circle at x=0 and y=r .

5. Calculate the initial value of the decision parameter p using p = 1 - r.

6. Start a do-while loop until y becomes less than x.

7. Inside the loop, plot the eight symmetric points of the circle using `putpixel`

8. Check if p is less than 0. If it is update p using p = p + (2 \* x) + 3.

9. If p is greater than or equal to 0, decrement y by 1 and update p using p = p + (2 \* x) - (2 \* y) + 5.

10. Increment x by 1.

11. Repeat steps 7 to 10 until y is greater than or equal to x.

12. Exit the do-while loop.

13. Display the circle on the screen using getch.

14. Close the graphics mode using closegraph.

**Program :**

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main(){

int x,y, r,p,xc, yc ;

int gd=DETECT , gm ;

initgraph(&gd , &gm ,"C:[\\TURBOC3\\BGI](../../..//BGI)");

printf("Enter xc, yc:");

scanf("%d %d",&xc, &yc);

printf("Enter r :");

scanf("%d", &r);

x=0;

y=r;

p=1-r;

do{

putpixel(xc+x,yc+y,RED);

putpixel(xc+y,yc+x,RED);

putpixel(xc+x,yc-y,RED);

putpixel(xc+y, yc-x,RED);

putpixel(xc-y, yc-x,RED);

putpixel(xc-x, yc-y,RED);

putpixel(xc-x, yc+y,RED);

putpixel(xc-y, yc+x,RED);

if(p <0){

p = p + (2\*x)+3;

}

else

{

y= y-1;

p=p + (2\*x) - (2\*y) + 5;

}

x = x + 1 ;

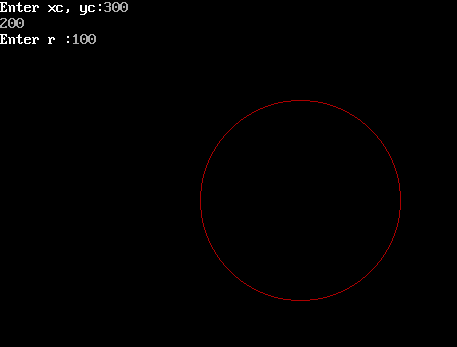
}while(y>=x);

getch();

closegraph();

}

**output :**



Vidyavardhini’s College of Engineering & Technology



Department of Computer Engineering

**Conclusion: Comment on**

1. **Fast or slow:** The Midpoint circle Algorithm is fast due to its integer arithmetic and minimal computations.

**2.Draw one arc only and repeat the process in 8 quadrants:** It draws only one-eight of the circle and mirrors the points in the other seven octants, minimizing calculations.

**3.Difference with line drawing method:**

Unlike line-drawing algorithm , it directly calculates points on the circle, avoiding inaccuracies and requiring fewer operations.

CSL402: Computer Graphics Lab