Experiment-03

Course Code: CSE422

Course Title: Computer Graphics Lab

Submitted By:

Md Raihan Hasan

201-15 -3098

PC-H2

Submitted To:

Mr. Md. Mubtasim Fuad

Department of CSE

Experiment Name: An OpenGL Program to draw a Circle using Mid-Point Circle Algorithm

Source code:

```
#include <stdio.h>
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>

int r, x_center, y_center;
double p, x, y;


void display(void)
{
glClear (GL_COLOR_BUFFER_BIT);
 glColor3f (0.0, 1.0, 0.0);

glPointSize(3);
```

```
glBegin(GL_POINTS);
   p=1-r;     while(x <= y)
{

if(p<0)
{
x=x+0.05;
    y=y;
printf("%0.2f %0.2f\n",x,y);
       p=p+(2*x)+1;
}

else
{
x=x+0.05;
 y=y-0.05;
printf("%0.2f %0.2f\n",x,y);
p=p+(2*x)+1-(2*y);
}
glVertex2f(x+x_center, y+y_center);
      glVertex2f(y+x_center, x+y_center);
     glVertex2f(-x+x_center, y+y_center);
       glVertex2f(-y+x_center, x+y_center);
      glVertex2f(-x+x_center, -y+y_center);
      glVertex2f(-y+x_center, -x+y_center);
    glVertex2f(x+x_center, -y+y_center);
    glVertex2f(y+x_center, -x+y_center);

};
glEnd();
  glFlush(); }
```

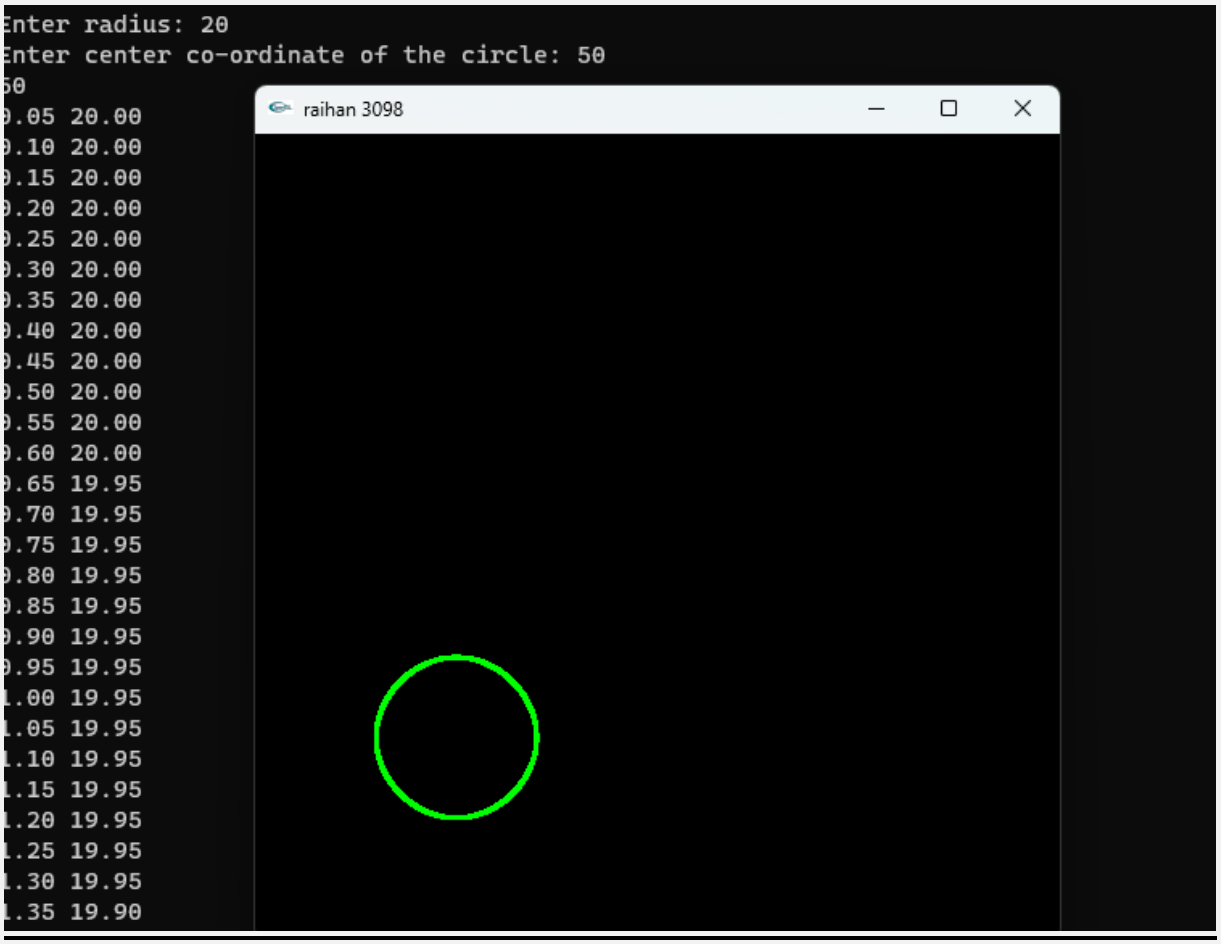
```
void init (void)
{
glClearColor (0.0, 0.0, 0.0, 0.0);
  glMatrixMode(GL_PROJECTION);
gluOrtho2D(0.0, 200.0, 0.0, 200.0);
}

int main(int argc, char** argv)
{

printf("Enter radius: ");
 scanf("%d",&r);
printf("Enter center co-ordinate of the circle: ");
   scanf("%d %d", &x_center, &y_center);
   x=0;
y=r;
glutInit(&argc, argv);
glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize (500, 500);
  glutInitWindowPosition (100, 100);
glutCreateWindow ("Mid-Point Circle Algorithm");
 init ();
glutDisplayFunc(display);
 glutMainLoop();

return 0;
}
```

Output:



Discussion

An OpenGL Program to draw a Circle using Mid-Point Circle Algorithm.The circle's radius and center should be selected first. Make x and y's initial values the circle's equatorial coordinates. The decision parameter p should start out with a value of 1 - r. As long as x is larger than or equal to y, create a loop that keeps going.The radius of the circle that I am using in this case is 20, and its x- and y-coordinates are 50 and 50 respectively.Draw the pixel at coordinates (x, y) inside the function along with its eight symmetrical duplicates (y, x), (-y, x), (-x, y), (-y, x), (y, x), and (x, -y).Decrease y and if the

decision parameter p is larger than or equal to 0. y and change p's value to p = p - 2*y + 1. Update the value of p as follows: p = p + 2*x + 1. Increase x. Up until x exceeds y, repeat the cycle. The Mid-Point Circle Algorithm draws the circle using the circle's radius and center coordinates as input. The algorithm draws every pixel of the circle and its eight symmetrical copies, giving the impression that the curve is continuous.