Lab Report

Week 3

Jitendra Kumar , 1401CS19

05/01/2017

Title

- Draw a circle in OpenGL and Matlab using :
  1). Bresenham's mid-point circle Drawing Algorithm

Procedure

OpenGL

1). Draw a circle using Bresenhams mid point Algorithm.

- Create a C file and name it as *circleBresenhams.c*.
- Define global variables to store coordinates of center and radius of a circle .
- Following is the Bresenham Algorithm to draw circle in 1st octant :

```
while x < y
        if (d < 0)
           d += 4*x + 6;
        else
           d += 4*(x-y) + 10;
           y--;
        end
        x++;
end \\


extend this to other octants. please see code
```

- Following is the final code for Bresenhams circle drawing algorithms :

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

int centre_x = 0 ; int centre_y=0 ; int radius =0 ;
int n = 0 ;
int x_coordinate[1000];
int y_coordinate[1000];

void displayCircle(void)
{
        glClear(GL_COLOR_BUFFER_BIT);

        int d = 3-2*radius;
        int x = 0, y = radius;
        putPixels(centre_x, centre_y, x, y);

        while (x < y)
        {
                if (d < 0)
                {
                   d += 4*x + 6;
                }
                else
                {
                   d += 4*(x-y) + 10;
                   y--;
                }
                x++;
                putPixels(centre_x, centre_y, x, y);
        }
        int i;
        for (i = 0; i < n; i++ )
        {
                glBegin(GL_POINTS);
                glColor3f(1.0, 1.0, 1.0);
                //printf("x : %d y : %d\n", x_coordinate[i], y_coordinate[i]);
                glVertex2f(x_coordinate[i]/100.0, y_coordinate[i]/100.0);
                glEnd();
        }
        glFlush();
}

void putPixels(int X, int Y, int P, int Q )
{
        x_coordinate[n] = X + P;
        y_coordinate[n++] = Y + Q;

        x_coordinate[n] = X - P;
        y_coordinate[n++] = Y + Q;

        x_coordinate[n] = X + P;
        y_coordinate[n++] = Y - Q;

        x_coordinate[n] = X - P;
        y_coordinate[n++] = Y - Q;

        x_coordinate[n] = X + Q;
        y_coordinate[n++] = Y + P;

        x_coordinate[n] = X - Q;
        y_coordinate[n++] = Y + P;

        x_coordinate[n] = X + Q;
        y_coordinate[n++] = Y - P;

        x_coordinate[n] = X - Q;
        y_coordinate[n++] = Y - P;
}
int main(int argc, char const *argv[])
{
        printf("centre coordinates : ");
        scanf("%d  %d", &centre_x , &centre_y);
        printf("radius of circle   : ");
        scanf("%d",&radius);

        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_RGB);
        glutInitWindowSize(640,480);
        glutCreateWindow("Bresenham Circle Drawing Algorithm");
        glutInitWindowPosition(100,100);
        glutDisplayFunc(displayCircle);
        glutMainLoop();
        return 0;
}
```

- Compile and run the executable file in terminal by typing in the following commands :
  (a) *gcc circleBresenhams.c -lGL -lGLU -lglut -lm*
  (b) *./a.out*

MatLab

1). Draw a circle using Bresenham's Line Drawing Algorithm :

- Open a new Script and contruct a function circle(). The script prompts user for inputs center and radius coordinates.
- Following is the Matlab Script Code for Bresenham's circle Drawing Algorithm :

```
function [] = circle()
        x_centre = input("enter the x coordinate of circle : ");
        y_centre = input("enter the x coordinate of circle : ");
        radius = input("enter the radius of circle : ");
        d = 3-2*radius;
        x = 0, y = radius;
        px = [x];
        py = [y];

        while x < y
                if (d < 0)
                   d += 4*x + 6;
                else
                   d += 4*(x-y) + 10;
                   y--;
                end
                x++;

                px = cat(1,px,round(x_centre+x));
                py = cat(1,py,round(y_centre+y));

                px = cat(1,px,round(x_centre-x));
                py = cat(1,py,round(y_centre+y));

                px = cat(1,px,round(x_centre+x));
                py = cat(1,py,round(y_centre-y));

                px = cat(1,px,round(x_centre-x));
                py = cat(1,py,round(y_centre-y));

                px = cat(1,px,round(x_centre+y));
                py = cat(1,py,round(y_centre+x));

                px = cat(1,px,round(x_centre-y));
                py = cat(1,py,round(y_centre+x));

                px = cat(1,px,round(x_centre+y));
                py = cat(1,py,round(y_centre-x));

                px = cat(1,px,round(x_centre-y));
                py = cat(1,py,round(y_centre-x));

        end
        plot(px,py,'-*');
```

Output

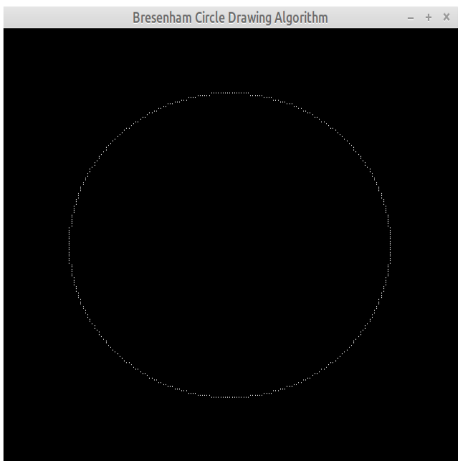

FIGURE 1 – Draw circle using bresenhams mid point algo in OpenGL

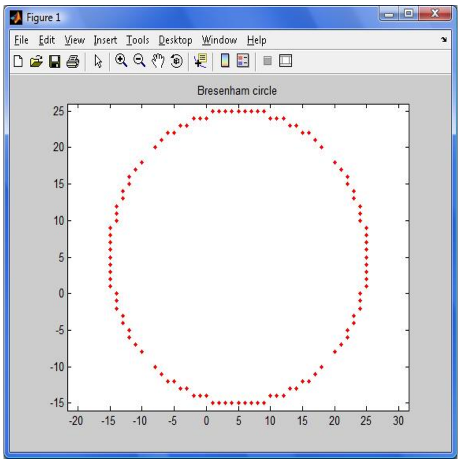

Figure 2 – Draw circle using bresenhams mid point algo in matlab