|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Experiment 6: Mid Point Circle Drawing** |         **SUBMITTED BY**   |  |  |  | | --- | --- | --- | | Name | ID | Section | | Kabid Yeiad | 202-15-14440 | 57\_A |   **SUBMITTED TO**  **Deawan Rakin Ahamed Remal,**  **Lecturer**  **Dept. of CSE**  **Daffodil International University**   |  | | --- | |  |   Submitted on November 1, 2023 |

**Mid-Point Circle Drawing Algorithm**

**Code:**

#*include <GL/glut.h>*

#*include <stdio.h>*

#*include <math.h>*

int centerX, centerY; // Center coordinates of the circle

int radius;

void drawPixel(int x, int y) {

    glBegin(GL\_POINTS);

    glVertex2i(x, y);

    glEnd();

}

void drawCircle() {

    int x = 0;

    int y = radius;

    int d = 1 - radius;

    while (x <= y) {

        drawPixel(centerX + x, centerY + y);

        drawPixel(centerX + y, centerY + x);

        drawPixel(centerX + y, centerY - x);

        drawPixel(centerX + x, centerY - y);

        drawPixel(centerX - x, centerY - y);

        drawPixel(centerX - y, centerY - x);

        drawPixel(centerX - y, centerY + x);

        drawPixel(centerX - x, centerY + y);

        if (d < 0) {

            d += 2 \* x + 3;

        } else {

            d += 2 \* (x - y) + 5;

            y--;

        }

        x++;

    }

}

void display() {

    glClear(GL\_COLOR\_BUFFER\_BIT);

    glColor3f(1.0f, 1.0f, 1.0f); // Set color to white

    glPointSize(1.0);

    drawCircle();

    glFlush();

}

void init() {

    glClearColor(0.0f, 0.0f, 0.0f, 1.0f); // Set background color to black

    glMatrixMode(GL\_PROJECTION);

    glLoadIdentity();

    gluOrtho2D(0.0, 500.0, 0.0, 500.0); // Set the coordinate system

}

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

    printf("Enter the center coordinates (x y): ");

    scanf("%d %d", &centerX, &centerY);

    printf("Enter the radius: ");

    scanf("%d", &radius);

    glutInit(&argc, argv);

    glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

    glutInitWindowSize(500, 500);

    glutCreateWindow("Mid-point Circle Algorithm");

    init();

    glutDisplayFunc(display);

    glutMainLoop();

    return 0;

}

**Output**:

|  |  |
| --- | --- |
|  |  |

**Graph**:

**Discussion**:

The mid-point circle algorithm chooses the optimal pixels for each step to construct a circle. It rasterizes circles efficiently using their symmetry. This approach uses integer arithmetic, eliminating floating-point calculations. The program iteratively selects pixels to approximate a circle and meet the desired circumference. The circle drawing coordinates are shown in the graph. The x-axis is horizontal and the y-axis vertical. Labels show the circle's beginning, middle, and end.This graph shows how the Mid-point technique optimizes rasterization by intelligently selecting pixels to make a circle. This implementation shows that the Mid-point circle algorithm can generate exact circles with low computational overhead.