|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Experiment 5: Brasenham Line Drawing Algorithm** |         **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 |

**An OpenGL program to draw a line using Brasenham line drawing algorithm**

**Code:**

#*include <GL/glut.h>*

#*include <stdio.h>*

int x1, y1, x2, y2;

void myInit()

{

    glClear(GL\_COLOR\_BUFFER\_BIT);

    glClearColor(0.0, 0.0, 0.0, 1.0);

    glMatrixMode(GL\_PROJECTION);

    gluOrtho2D(0, 500, 0, 500);

}

void draw\_pixel(int x, int y)

{

    glBegin(GL\_POINTS);

    glVertex2i(x, y);

    glEnd();

}

void draw\_line(int x1, int x2, int y1, int y2) {

    int dx, dy, i, e;

    int incx, incy, inc1, inc2;

    int x, y;

    dx = x2 - x1;

    dy = y2 - y1;

    if (dx < 0) dx = -dx;

    if (dy < 0) dy = -dy;

    incx = 1;

    if (x2 < x1) incx = -1;

    incy = 1;

    if (y2 < y1) incy = -1;

    x = x1; y = y1;

    if (dx > dy) {

        draw\_pixel(x, y);

        e = 2 \* dy - dx;

        inc1 = 2 \* (dy - dx);

        inc2 = 2 \* dy;

        for (i = 0; i < dx; i++)

        {

            if (e >= 0)

            {

                y += incy;

                e += inc1;

            }

            else

                e += inc2;

            x += incx;

            draw\_pixel(x, y);

        }

    }

    else

    {

        draw\_pixel(x, y);

        e = 2 \* dx - dy;

        inc1 = 2 \* (dx - dy);

        inc2 = 2 \* dx;

        for (i = 0; i < dy; i++)

        {

            if (e >= 0)

            {

                x += incx;

                e += inc1;

            }

            else

                e += inc2;

            y += incy;

            draw\_pixel(x, y);

        }

    }

}

void myDisplay()

{

    draw\_line(x1, x2, y1, y2);

    glFlush();

}

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

{

    printf("Enter (x1, y1, x2, y2)\n");

    scanf("%d %d %d %d", &x1, &y1, &x2, &y2);

    glutInit(&argc, argv);

    glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

    glutInitWindowSize(500, 500);

    glutInitWindowPosition(0, 0);

    glutCreateWindow("Bresenham's Line Drawing");

    myInit();

    glutDisplayFunc(myDisplay);

    glutMainLoop();

}

**Output**:

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

**Graph**:

**Discussion**:

The Bresenham method chooses the best pixel at each step to build a straight line between two endpoints. Instead of floating-point arithmetic, it considers line slope and makes judgments using integer values. The graph shows line drawing coordinates. The x-axis is horizontal and the y-axis vertical. Labels show line beginnings and ends. The implementation of Bresenham's line drawing algorithm shows its precision and low processing cost. Computer graphics requires knowledge and use of such techniques for rendering and picture processing.