CS352 Assignment-3

Krishanu Saini
190001029

Write a program to draw a line joining two endpoints given by the user by Implementing the following line drawing algorithms:

- Bresenham's Line drawing algorithm (for m>1 and m<1)
- Midpoint Line Drawing algorithm

Solution

Bresenham line drawing algorithm.
Calculate points using xi, yi, pi. P0 = 2*y - x. Then increment x = x+1 and y according to formula

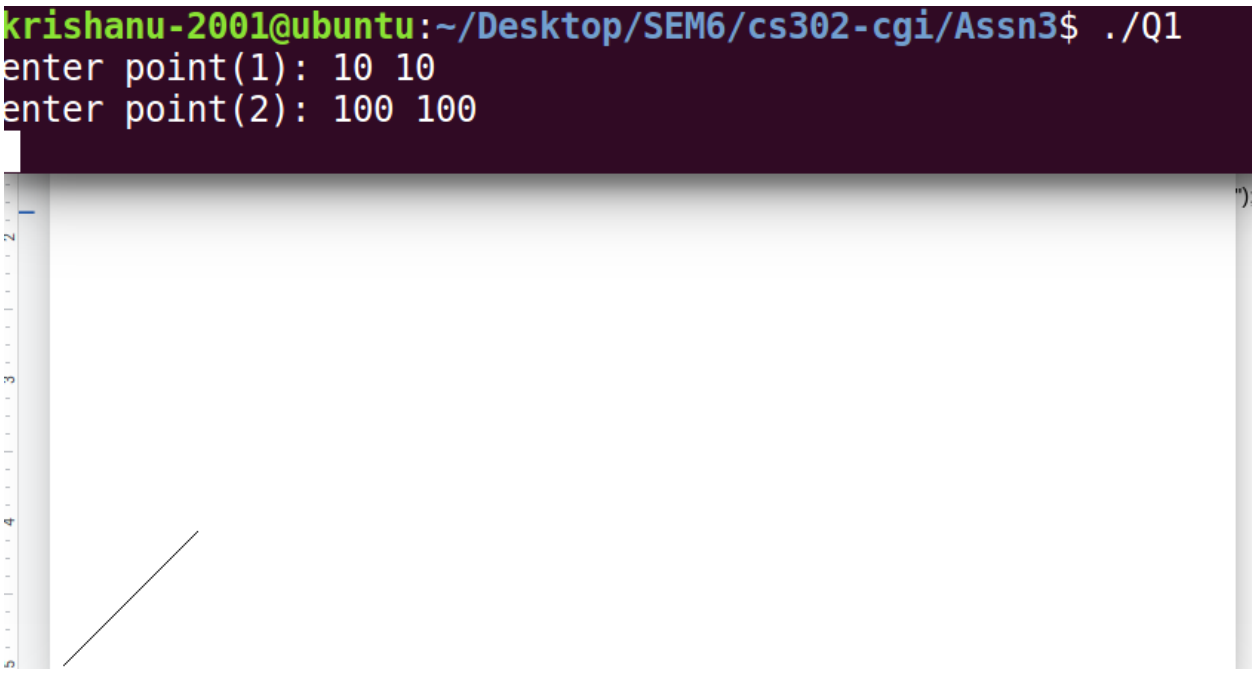
Code

```
#include <math.h>
#include <GL/glut.h>
#include <iostream>
#include <cassert>
#define debug(x,y) cout << #x << " " << x << " " << #y << " " << y << "\n";

using namespace std;

/*
 * Name: Krishanu Saini
 * Roll: 190001029
 * Ques: Problem 1) Bresenham
 * Date: 17/01/22
 */

float xcord1,ycord1;
float xcord2,ycord2;

void Bresenham(float x1, float y1, float x2, float y2) {
    glColor3f(0,0,0);

    float xi = x1;
    float yi = y1;

    float delx = (x2 - x1);
    float dely = (y2 -y1);

    int totalSteps = 0;

    totalSteps = delx;
    if(abs(delx) < abs(dely)) {
        totalSteps = dely;
    }
```

```
    glBegin(GL_POINTS);
    glVertex2i((int)xi, (int)yi);

    int pi = 2*dely - delx;

    for(int i=1;i<=totalSteps;i++) {
        if(abs(delx) > abs(dely)) {
            // m < 1
            // if pi < 0, normal case
            xi = xi + 1;
            pi = pi + 2*dely;
            if(pi >= 0) { // if pi > 0 then increase yi - case-1
                pi = pi - 2*delx;
                yi = yi + 1;
            }
        } else {
            // m > 1
            // if pi < 0, normal case
            yi = yi + 1;
            pi = pi + 2*delx;
            if(pi >= 0) { // if pi > 0 then increase xi - case-1
                pi = pi - 2*dely;
                xi = xi + 1;
            }
        }
        glVertex2i((int)round(xi), (int)round(yi));
    }


    glEnd();
}

void Draw() {
    glClear(GL_COLOR_BUFFER_BIT);
    glPointSize(1);

    Bresenham(xcord1,ycord1,xcord2,ycord2);

    glFlush();
}

int main(int argc, char *argv[]) {
    printf("enter point(1): ");
    cin >> xcord1 >> ycord1;
```

```
    printf("enter point(2): ");
    cin >> xcord2 >> ycord2;
    if(xcord1 > xcord2) swap(xcord1, xcord2), swap(ycord1, ycord2);
    glutInit(&argc, argv);
    glutInitWindowPosition(100, 100);
    glutInitWindowSize(800, 800);
    glutInitDisplayMode(GLUT_RGB);

    glutCreateWindow("Bresenham Model");
    gluOrtho2D(0, 800, 0, 800);
    glClearColor(1,1,1,0.0);
    glutDisplayFunc(Draw);
    glutMainLoop();
    return 0;
}
```

Output



Solution

Mid point algorithm is a way to draw lines. We choose 2 points East and N East, we check if the distance between center and line is +ve or not and calculate the answer.
We initialize d with a+b/2. If b is even, no floating calculation is needed.

Code

```
#include <math.h>
#include <GL/glut.h>
#include <iostream>
#include <cassert>
#define debug(x,y) cout << #x << " " << x << " " << #y << " " << y << "\n";

using namespace std;

/*
 * Name: Krishanu Saini
 * Roll: 190001029
 * Ques: Problem 2) DDA Method
 * Date: 17/01/22
 */

int xcord1,ycord1;
int xcord2,ycord2;

void MidPoint(int x1, int y1, int x2, int y2) {
    glColor3f(0,0,0);

    int xi = x1;
    int yi = y1;

    int delx = (x2 - x1);
    int dely = (y2 -y1);

    int totalSteps = 0;
    int d = 0;

    if(abs(delx) > abs(dely)) {
        totalSteps = delx;
        d = dely - delx / 2;
    } else {
```

```
        totalSteps = dely;
        d = delx - dely / 2;
    }

    glBegin(GL_POINTS);
    glVertex2i((int)xi, (int)yi);

    for(int i=1;i<=totalSteps;i++) {
        if(abs(delx) >= abs(dely)) {
            // m < 1, normal direction
            xi += 1;
            // East direction
            if (d < 0) {
                d = d + dely;
            } else { // North East direction
                d += dely - delx;
                yi++;
            }
        } else {
            yi += 1;
            // North direction
            if (d < 0) {
                d = d + delx;
            } else { // North East direction
                d += delx - dely;
                xi++;
            }
        }
        glVertex2i((int)round(xi), (int)round(yi));
    }

    glEnd();
}

void Draw() {
    glClear(GL_COLOR_BUFFER_BIT);
    glPointSize(1);

    MidPoint(xcord1,ycord1,xcord2,ycord2);

    glFlush();
}

int main(int argc, char *argv[]) {
```

```
    printf("enter point(1): ");
    cin >> xcord1 >> ycord1;
    printf("enter point(2): ");
    cin >> xcord2 >> ycord2;
    if(xcord1 > xcord2) swap(xcord1, xcord2), swap(ycord1, ycord2);
    glutInit(&argc, argv);
    glutInitWindowPosition(100, 100);
    glutInitWindowSize(800, 800);
    glutInitDisplayMode(GLUT_RGB);

    glutCreateWindow("DDA Model");
    gluOrtho2D(0, 800, 0, 800);
    glClearColor(1,1,1,0.0);
    glutDisplayFunc(Draw);
    glutMainLoop();
    return 0;
}
```

Output

```
krishanu-2001@ubuntu:~/Desktop/SEM6/cs302-cgi/Assn3$ ./Q2
enter point(1): 10 10
enter point(2): 100 110
```