Lab Report No: 03

Problem Name:

Implementation of Midpoint Line Drawing Algorithm.

Description:

The Midpoint Line Drawing Algorithm is a fundamental algorithm in computer graphics used to efficiently rasterize a straight line between two points using only integer calculations. It is an improvement over the Digital Differential Analyzer (DDA) algorithm, eliminating the need for floating-point arithmetic and making it more optimized for raster displays. The algorithm determines the closest pixel to the ideal line path by evaluating a decision parameter based on the midpoint between two possible pixel choices.

Algorithm:

Given-

Starting coordinates = (X0, Y0)

Ending coordinates = (Xn, Yn)

The points generation using Mid-Point Line Drawing Algorithm involves the following steps-

Step-01:

Calculate ΔX and ΔY from the given input.

These parameters are calculated as-

DX = Xn – X0

DY =Yn – Y0

Step-02:

Calculate the value of initial decision parameter and ΔD.

These parameters are calculated as-

Dinitial = 2DY – DX

delD = 2(DY – DX)

Step-03:

The decision whether to increment X or Y coordinate depends upon the flowing values of Dinitial.

Follow the below two cases-

Case-1: If D<0

- X = X+1
- Dnew = D+2DY

Case-2: If D>=0

- X = X+1
- Y= Y+1
- Dnew = D + delD

Step-04:

Keep repeating Step-03 until the end point is reached.

For each Dnew value, follow the above cases to find the next coordinates.

Code:

```
#include <graphics.h>

#include <iostream>

using namespace std;


void drawMidPointLine(int X0, int Y0, int Xn, int Yn) {


    int DX = Xn - X0;

    int DY = Yn - Y0;

    int D = 2 * DY - DX;

    int delD=2 * (DY - DX);

    int Dnew;

    int x = X0, y = Y0;

    putpixel(x, y, BLACK);

    while (x < Xn) {

        if (D < 0) {

            Dnew = D + 2 * DY;

            x++;
```

```
        } else {
            Dnew = D + delD;
            x++;
            y++;
        }
        D = Dnew;
putpixel(x, y, BLACK);
    }
}

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");
    setbkcolor(WHITE); // Set background color
    cleardevice();
    int X0 = 200, Y0 = 200, Xn = 300, Yn = 400;
    drawMidPointLine( X0, Y0,  Xn,  Yn);
    getch();
    closegraph();
    return 0;
}
```

Output:

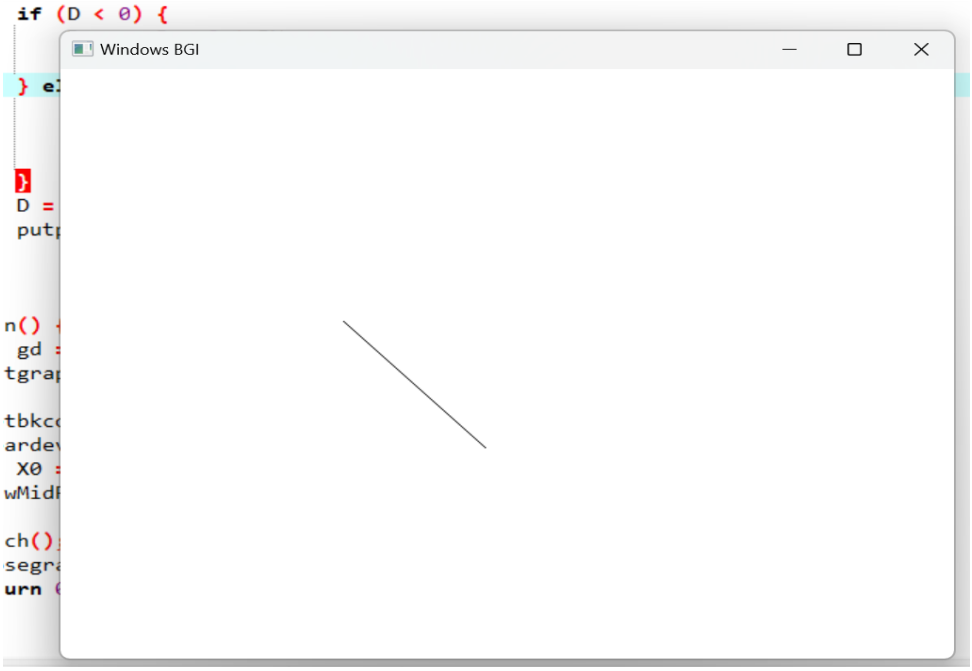

Figure Name: System Output.

Conclusion:

The Midpoint Line Drawing Algorithm is an efficient and optimized method for rasterizing straight lines in computer graphics. By using only integer calculations, it eliminates the need for floating-point arithmetic, making it faster and more suitable for real-time rendering on raster displays.

Remarks: Using App:

DEV C++