**SSN COLLEGE OF ENGINEERING, KALAVAKKAM**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**UCS1712 – GRAPHICS AND MULTIMEDIA LAB**

**------------------------------------------------------------------------------------------------------------**

**Lab Exercise 2 : DDA Line Drawing Algorithm in C++ using OpenGL**

2) To plot points that make up the line with endpoints (x0,y0) and (xn,yn) using DDA line drawing algorithm.

Case 1: +ve slope Left to Right line

Case 2: +ve slope Right to Left line

Case 3: -ve slope Left to Right line

Case 4: -ve slope Right to Left line

Each case has two subdivisions

1. |m|<= 1
2. (ii) |m|>1

Note that all four cases of line drawing must be given as test cases.

**CODE:**

#include<bits/stdc++.h>

#include <GL/glut.h>

using namespace std;

double X1, Y1, X2, Y2;

double arrx1[4], arry1[4], arrx2[4], arry2[4];

float round\_value(float v)

{

return floor(v + 0.5);

}

void myInit() {

glClearColor(1.0, 1.0, 1.0, 0.0);

glColor3f(0.0f, 0.0f, 0.0f);

glPointSize(4);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(0.0,1000.0, 0.0, 1000.0);

}

void LineDDA(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

int j = 0;

while (j < 4)

{

X1 = arrx1[j];

Y1 = arry1[j];

X2 = arrx2[j];

Y2 = arry2[j];

double dx = (X2 - X1);

double dy = (Y2 - Y1);

double steps;

float xInc, yInc, x = X1, y = Y1;

steps = (fabs(dx) > fabs(dy)) ? (fabs(dx)) : (fabs(dy));

float m = dy / dx;

xInc = dx / (float)steps;

yInc = dy / (float)steps;

glBegin(GL\_POINTS);

glVertex2d(x, y);

int k;

for (k = 0; k < steps; k++)

{

x += xInc;

y += yInc;

glVertex2d(round\_value(x), round\_value(y));

}

glEnd();

j++;

glFlush();

}

}

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

{

int i = 0;

while (i < 4)

{

cout<<"Enter two end points of the line to be drawn:"<<endl;

cout<<endl<<"Case "<<i + 1<<":";

cout<<endl<<"Enter Point1( X1 , Y1):"<<endl;

cin >> X1;

cin >> Y1;

arrx1[i] = X1;

arry1[i] = Y1;

cout << endl;

cout << endl << "Case " << i + 1 << ":";

cout << endl << "Enter Point2( X2 , Y2):" << endl;

cin >> X2;

cin >> Y2;

arrx2[i] = X2;

arry2[i] = Y2;

i++;

}

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(1000, 1000);

glutCreateWindow("Ex2 DDA Line Drawing");

glutDisplayFunc(LineDDA);

myInit();

glutMainLoop();

return 1;

}

**OUTPUT:**



