**Pune Institute Of Computer Technology Dhankawadi,**

**Pune – 43.**

Assignment No. 2

Computer Graphics

**SE-IT-10 ACADEMIC YEAR :- 2020-2021**

**Name :- Diptesh Ravindra Varule Roll No :- 23277**

**Topic Name**:

Implement DDA and Bresenham line drawing algorithm to draw:

i) Simple Line ii) Dotted Line iii) Dashed Line iv) Solid line ;

using mouse interface Divide the screen in four quadrants with center as (0, 0). The line should work for all the slopes positive as well as negative.

**Bresenhem Algorithm :**

#include <GL/freeglut.h>

#include <GL/gl.h>

#include<math.h>

#include<cstdio>

float X1,X2,Y1,Y2,xmax,ymax;

int choice;

void init()

{

glClearColor(0.0,0.0,0,1.0);

glColor3f(1.0,1.0,0.0);

gluOrtho2D(0,1024,0,768);

}

void putpixel(int x, int y){

glPointSize(0.2f);

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

void SimpleLine(int x1,int y1,int x2,int y2){

glBegin(GL\_LINES);

glVertex2f(x1, y1);

glVertex2f(x2, y2);

glEnd();

glFlush();

}

int sign(int x ){

if (x > 0) return 1;

if (x < 0) return -1;

return 0;

}

void bres(int x1,int y1,int x2,int y2,int dotted=0,int solid=1,int dashed=0)

{

int interchange;

int x=x1+512;

int y=y1+384;

int X=abs(x2-x1);

int Y=abs(y2-y1);

int s1=sign(x2-x1);

int s2=sign(y2-y1);

if(Y>X){

int temp=X;

X=Y;

Y=temp;

interchange=1;

}

else interchange=0;

int e=2\*Y-X;

int i;

if( dotted==0 && solid==1 && dashed==0){

for(i=0;i<X;++i){

putpixel(x,y);

while(e>0){

if(interchange==1) x+=s1;

else y+=s2;

e=e-(2\*X);

}

if(interchange==1) y+=s2;

else x+=s1;

e=e+2\*Y;

}

}

else if(dotted==1 && solid==0 && dashed==0){

for(i=0;i<X;++i){

while(e>0){

if(interchange==1) x+=s1;

else y+=s2;

e=e-(2\*X);

}

if(interchange==1) y+=s2;

else x+=s1;

e=e+2\*Y;

if ((int)i%3==0)

{

putpixel(x,y);

}

}

}

else if(dotted==0 && solid==0 && dashed==1){

for(i=0;i<X;++i){

while(e>0){

if(interchange==1) x+=s1;

else y+=s2;

e=e-(2\*X);

}

if(interchange==1) y+=s2;

else x+=s1;

e=e+2\*Y;

if ((int)i%8){

putpixel(x,y);

}

}

}

glFlush();

}

void renderFunction()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

bres(-512,0,512,0,0,1,0); //bres(0,xmax,ymax/2,ymax/2);

bres(0,-384,0,384,0,1,0); // bres

if(choice==1)

bres(X1,Y1,X2,Y2,1,0,0);

else if(choice==2)

bres(X1,Y1,X2,Y2,0,1,0);

else if(choice==3)

bres(X1,Y1,X2,Y2,0,0,1);

else if(choice==4)

SimpleLine(X1+512,Y1+384,X2+512,Y2+384);

else

printf("Wrong input..!!");

glFlush();

}

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

{

printf("Enter the value of x1 : ");

scanf("%f",&X1);

printf("Enter the value of y1 : ");

scanf("%f",&Y1);

printf("Enter the value of x2 : ");

scanf("%f",&X2);

printf("Enter the value of y2 : ");

scanf("%f",&Y2);

printf("Press 1 : For Dotted Line\nPress 2 : For Solid Line \nPress 3 : For Dashed Line\nPress 4 : For Simple Line\n");

printf("Choice : ");

scanf("%d",&choice);

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowPosition(0,0);

glutInitWindowSize(1024,768);

glutCreateWindow("Bresenham Line pattern");

init();

glutDisplayFunc(renderFunction);

glutMainLoop();

return 0;

}

Output:

Text

Description automatically generatedA picture containing graphical user interface

Description automatically generatedText

Description automatically generatedA picture containing graphical user interface

Description automatically generated

Text

Description automatically generatedA picture containing graphical user interface

Description automatically generatedText

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

**DDA Algorithm :**

#include<GL/glut.h>

#include<stdio.h>

#include<stdlib.h>

float X1,X2,Y1,Y2,xmax,ymax;

int choice;

void init()

{

glClearColor(0.0,0.0,0.0,0.0);

glColor3f(1.0,1.0,0.0);

gluOrtho2D(0,1024,0,768);

}

void putpixel(float x, float y)

{

glPointSize(1.0f);

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

void SimpleLine(int x1,int y1,int x2,int y2){

glBegin(GL\_LINES);

glVertex2f(x1, y1);

glVertex2f(x2, y2);

glEnd();

glFlush();

}

void ddaLine(int x1,int y1,int x2,int y2,int dotted=0,int solid=1,int dashed=0)

{

float dy,dx,steps,x,y,k,Xinc,Yinc;

dx=x2-x1;

dy=y2-y1;

if(abs(dx)> abs(dy))

{

steps = abs(dx);

}

else

steps = abs(dy);

Xinc = dx/steps;

Yinc = dy/steps;

x= x1+512;

y=y1+384;

putpixel(x,y); //user defined function to plot point

if( dotted==0 && solid==1 && dashed==0){

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

{

x= x + Xinc;

y= y + Yinc;

putpixel(x,y);

}

}

else if(dotted==1 && solid==0 && dashed==0){

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

{

x= x + Xinc;

y= y + Yinc;

if((int)k%3){

putpixel(x,y);

}

}

}

else if(dotted==0 && solid==0 && dashed==1){

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

{

x= x + Xinc;

y= y + Yinc;

if((int)k%8){

putpixel(x,y);

}

}

}

glFlush();

}

void primitives(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

xmax=glutGet(GLUT\_WINDOW\_WIDTH);

ymax=glutGet(GLUT\_WINDOW\_HEIGHT);

ddaLine(-512,0,512,0,0,1,0); //ddaLine(0,xmax,ymax/2,ymax/2);

ddaLine(0,-384,0,384,0,1,0); // dda

if(choice==1)

ddaLine(X1,Y1,X2,Y2,1,0,0);

else if(choice==2)

ddaLine(X1,Y1,X2,Y2,0,1,0);

else if(choice==3)

ddaLine(X1,Y1,X2,Y2,0,0,1);

else if(choice==4)

SimpleLine(X1+512,Y1+384,X2+512,Y2+384);

else

printf("Wrong Input..... !!");

glFlush();

}

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

{

printf("Enter the value of x1 : ");

scanf("%f",&X1);

printf("Enter the value of y1 : ");

scanf("%f",&Y1);

printf("Enter the value of x2 : ");

scanf("%f",&X2);

printf("Enter the value of y2 : ");

scanf("%f",&Y2);

printf("Press 1 : For Dotted Line\nPress 2 : For Solid Line \nPress 3 : For Dashed Line\nPress 4 : For Simple Line\n");

printf("Choice : ");

scanf("%d",&choice);

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_SINGLE);

glutInitWindowPosition(0,0);

glutInitWindowSize(1024,768);

glutCreateWindow("DDA Line Pattern");

init();

glutDisplayFunc(primitives);

glutMainLoop();

}

**Output:**

Text

Description automatically generatedA picture containing graphical user interface

Description automatically generated

Text

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence

Text

Description automatically generatedA picture containing graphical user interface

Description automatically generatedText

Description automatically generatedA picture containing graphical user interface

Description automatically generated