Experiment Name: Brasenham Line in Open GL

Introduction: This project I can see a picture that uses Open GL to draw the fundamental shapes of a Brasenham Line in Open GL. Through Open GL, I will learn how to draw shapes.

Code:

#include <stdio.h>

#include <GL/gl.h>

#include <GL/glut.h>

float x1,y1,x2,y2,m,i,j,p;

int dx=0,dy=0;

void display(void)

{

glClear (GL\_COLOR\_BUFFER\_BIT);

glEnd();

glColor3f (0.0, 1.0, 0.0);

glBegin(GL\_POINTS);

p=(2\*dy)-dx;

for(i=x1,j=y1;i<=x2,j<=y2; ){

if(p>=0){

i=i+1;

j=j+1;

if((i>x2)||(j>y2)){

break;

}

printf("%0.2f %0.2f\n",i,j);

glVertex3f ((i/100), (j/100), 0.0);

p=p+(2\*dy)-(2\*dx);

}

else if(p<0){

i=i+1;

if((i>x2)||(j>y2)){

break;

}

printf("%0.2f %0.2f\n",i,j);

glVertex3f ((i/100), (j/100), 0.0);

p=p+(2\*dy);

}

}

glEnd();

glFlush ();

}

void init (void)

{

/\* select clearing (background) color \*/

glClearColor (0.0, 0.0, 0.0, 0.0);

/\* initialize viewing values \*/

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);

}

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

{

printf("Enter first point: ");

scanf("%f %f",&x1,&y1);

printf("Enter second point: ");

scanf("%f %f",&x2,&y2);

dx=x2-x1;

dy=y2-y1;

glutInit(&argc, argv);

glutInitDisplayMode (GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize (500, 500);

glutInitWindowPosition (100, 100);

glutCreateWindow ("hello");

init ();

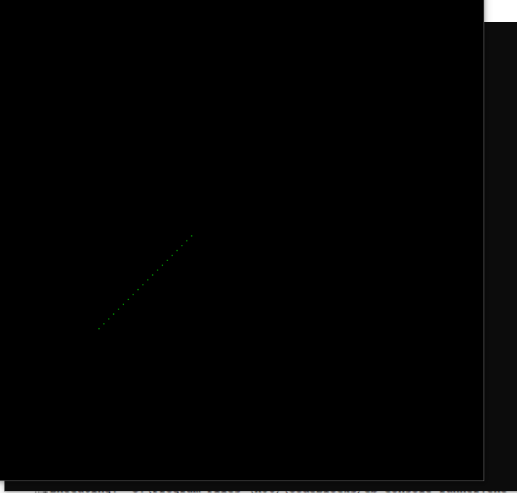
glutDisplayFunc(display);

glutMainLoop();

return 0; /\* ISO C requires main to return int. \*/

}

Output:



Conculation: Here I use some function  glutInit (&argc, argv).the window size using the function glutInitWindowSize(500, 500).the starting position for the window using the function glutInitWindowPosition (100, 100).Initialize the window and set the title using the function glutCreateWindow(“hello”).Initialize the myInit() function and perform the following steps:Set the background color to orange using the function glClearColor(0.0, 1.0, 0.0). Initialize the myDisplay() function and perform the following steps:Clear the screen using the function glClear(GL\_COLOR\_BUFFER\_BIT). glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0); initialize viewing values glMatrixMode(GL\_PROJECTION).