**Flood fill:**

#include<string.h>

#include<GL/glut.h>

#define ROUND(x)((int)(x+0.5))

struct Point

{

GLint x;

GLint y;

};

struct Color

{

GLfloat r;

GLfloat g;

GLfloat b;

};

Color getPixelColor(GLint x, GLint y) //Fun for getting color of pixel

{

Color color;

glReadPixels(x,y,1,1,GL\_RGB, GL\_FLOAT, &color);

return color;

}

void setPixelColor(GLint x, GLint y, Color color) //set color of pixel

{

glColor3f(color.r, color.g, color.b);

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

glFlush();

}

void floodFill(GLint x, GLint y, Color oldColor, Color newColor)//seed fill algorithm

{

Color color;

color= getPixelColor(x,y);

if(color.r == oldColor.r && color.g == oldColor.g && color.b == oldColor.b)

{

setPixelColor(x,y, newColor);

floodFill(x+1,y, oldColor, newColor);

floodFill(x,y+1, oldColor, newColor);

floodFill(x-1,y, oldColor, newColor);

floodFill(x,y-1, oldColor, newColor);

}

}

int Height =650, Width=650;

int startX,startY;

static Point vertex [1];

static int pt=0;

Color fillcolor;

void myMouse(int button, int state, int x, int y);

void drawline(double X1,double Y1, double X2, double Y2) //line drawing using dda

{

float x,y,dx,dy,length;

int i;

dx=abs(X2-X1);

dy=abs(Y2-Y1);

if(dx>=dy)

length=dx;

else

length=dy;

dx=(X2-X1)/length;

dy=(Y2-Y1)/length;

x=X1;

y=Y1;

i=1;

while(i<=length)

{

glColor3f(1.0,1.0,0.0);

glBegin(GL\_POINTS);

glVertex2i(ROUND(x),ROUND(y));

glEnd();

glFlush();

x=x+dx;

y=y+dy;

i=i+1;

}

glFlush();

}

void display(void)

{

char string[]="Step 1: Draw Polygon and press Enter key to draw final edge";

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f(1.0,1.0,1.0);

glRasterPos2f(10,600);

int len,i;

len=(int)strlen(string);

for(i=0;i<len;i++)

{

glutBitmapCharacter(GLUT\_BITMAP\_9\_BY\_15,string[i]);

}

glColor3f(1.0,0.0,0.0);

glRecti(10,30,60,10);

glColor3f(0.0,1.0,0.0);

glRecti(90,30,140,10);

glColor3f(0.0,0.0,1.0);

glRecti(170,30,220,10);

glFlush();//send all output to display

}

void myinit()

{

glClearColor(0.0,0.0,0.0,1.0); //set background as black

glColor3f(1.0,1.0,0.0); // Draw in yellow

glMatrixMode(GL\_PROJECTION);//Establish coordinate system

glLoadIdentity();

gluOrtho2D(0.0,650.0,0.0,650.0);

}

void myKeyboard(unsigned char key, int mouseX, int mouseY)

{

char string[]="Step 2:pick clicking on desired color rectangle";

switch(key)

{

case 13: // If Enter key is pressed

glColor3f(1.0,1.0,1.0);

glRasterPos2f(10,580);

int len,i;

len=(int)strlen(string);

for(i=0;i<len;i++)

{

glutBitmapCharacter(GLUT\_BITMAP\_9\_BY\_15,string[i]);

}

drawline(vertex[0].x, vertex[0].y,startX,startY);//draw final edge of poly

pt=2;

break;

case 27: //If Esc key pressed

exit(0);

}

}

void myMouse(int button,int state,int x,int y)

{

if(button==GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN)

{

if(pt==0) // to accept first point

{

vertex[pt].x=x;

vertex[pt].y=Height-y;

startX=x;

startY=Height-y;

pt++;

}

else if(pt==1)//to draw edges of poly

{

drawline(vertex[0].x,vertex[0].y,x,Height-y);

vertex[0].x=x;

vertex[0].y=Height-y;

}

else if(pt==2) //accept desired color

{

fillcolor=getPixelColor(x,Height-y);

char string[]="Step 3:Click inside the polygon to fill color";

glColor3f(1.0,1.0,1.0);

glRasterPos2f(10,560);

int len,i;

len=(int)strlen(string);

for(i=0;i<len;i++)

{

glutBitmapCharacter(GLUT\_BITMAP\_9\_BY\_15,string[i]);

}

pt=3;

}

else if(pt==3) //fill polygon

{

Color newColor={fillcolor.r,fillcolor.g,fillcolor.b};

Color oldColor= { 0.0f,0.0f,0.0f};

floodFill(x,Height-y,oldColor,newColor);

pt=4;

}

}

glFlush();

}

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

{

glutInit(&argc,argv); //Initialize toolkit

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB); //set display mode

glutInitWindowSize(650,650); //Set window size

glutInitWindowPosition(0,0); //set window position on screen

//Open screen window

glutCreateWindow("Draw Polygon using Interaction using OpenGL");

glutDisplayFunc(display); //Register redraw func

glutKeyboardFunc(myKeyboard); //Register mouse fun

glutMouseFunc(myMouse);

myinit();

glutMainLoop();

return 0;

}