OpenGl

Ex: Boundaryfill square

#include <math.h>

#include <gl/glut.h>

struct Point {

   GLint x;

   GLint y;

};

struct Color {

   GLfloat r;

   GLfloat g;

   GLfloat b;

};

void init() {

   glClearColor(1.0, 1.0, 1.0, 0.0);

   glColor3f(0.0, 0.0, 0.0);

   glPointSize(1.0);

   glMatrixMode(GL\_PROJECTION);

   glLoadIdentity();

   gluOrtho2D(0, 640, 0, 480);

}

Color getPixelColor(GLint x, GLint y) {

   Color color;

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

   return color;

}

void setPixelColor(GLint x, GLint y, Color color) {

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

   glBegin(GL\_POINTS);

   glVertex2i(x, y);

   glEnd();

   glFlush();

}

void BoundaryFill(int x, int y, Color fillColor, Color boundaryColor) {

   Color currentColor = getPixelColor(x, y);

   if(currentColor.r != boundaryColor.r && currentColor.g != boundaryColor.g && currentColor.b != boundaryColor.b) {

      setPixelColor(x, y, fillColor);

      BoundaryFill(x+1, y, fillColor, boundaryColor);

      BoundaryFill(x-1, y, fillColor, boundaryColor);

      BoundaryFill(x, y+1, fillColor, boundaryColor);

      BoundaryFill(x, y-1, fillColor, boundaryColor);

   }

}

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

{

   Color fillColor = {1.0f, 0.0f, 0.0f};     // red color will be filled

   Color boundaryColor = {0.0f, 0.0f, 0.0f}; // black- boundary

   Point p = {321, 241}; // a point inside the square

   BoundaryFill(p.x, p.y, fillColor, boundaryColor);

}

void draw\_dda(Point p1, Point p2) {

   GLfloat dx = p2.x - p1.x;

   GLfloat dy = p2.y - p1.y;

   GLfloat x1 = p1.x;

   GLfloat y1 = p1.y;

   GLfloat step = 0;

   if(abs(dx) > abs(dy)) {

      step = abs(dx);

   } else {

      step = abs(dy);

   }

   GLfloat xInc = dx/step;

   GLfloat yInc = dy/step;

   for(float i = 1; i <= step; i++) {

      glVertex2i(x1, y1);

      x1 += xInc;

      y1 += yInc;

   }

}

void draw\_square(Point a, GLint length) {

   Point b = {a.x + length, a.y},

      c = {b.x,   b.y+length},

      d = {c.x-length, c.y};

   draw\_dda(a, b);

   draw\_dda(b, c);

   draw\_dda(c, d);

   draw\_dda(d, a);

}

void display(void) {

   Point pt = {320, 240};

   GLfloat length = 50;

   glClear(GL\_COLOR\_BUFFER\_BIT);

   glBegin(GL\_POINTS);

      draw\_square(pt, length);

   glEnd();

   glFlush();

}

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

{

   glutInit(&argc, argv);

   glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB);

   glutInitWindowSize(640, 480);

   glutInitWindowPosition(200, 200);

   glutCreateWindow("Open GL");

   init();

   glutDisplayFunc(display);

   glutMouseFunc(onMouseClick);

   glutMainLoop();

   return 0;

}

OutPut:

