CGPractcal4

#include <GL/glut.h>

#include <vector>

using namespace std;

int fillMethod = 0;

float fillColor[3] = {1.0, 0.0, 0.0}; // Red fill color

float boundaryColor[3] = {0.0, 0.0, 0.0}; // Black boundary

vector<pair<int, int>> polygon;

void setPixel(int x, int y, float color[3]) {

glColor3fv(color);

glBegin(GL\_POINTS);

glVertex2i(x, y);

glEnd();

glFlush();

}

void getPixelColor(int x, int y, float color[3]) {

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

}

void floodFill(int x, int y, float oldColor[3], float newColor[3]) {

float pixelColor[3];

getPixelColor(x, y, pixelColor);

if (pixelColor[0] == oldColor[0] &&

pixelColor[1] == oldColor[1] &&

pixelColor[2] == oldColor[2]) {

setPixel(x, y, newColor);

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

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

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

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

}

}

void boundaryFill(int x, int y, float boundary[3], float newColor[3]) {

float pixelColor[3];

getPixelColor(x, y, pixelColor);

if ((pixelColor[0] != boundary[0] || pixelColor[1] != boundary[1] || pixelColor[2] != boundary[2]) &&

(pixelColor[0] != newColor[0] || pixelColor[1] != newColor[1] || pixelColor[2] != newColor[2])) {

setPixel(x, y, newColor);

boundaryFill(x + 1, y, boundary, newColor);

boundaryFill(x - 1, y, boundary, newColor);

boundaryFill(x, y + 1, boundary, newColor);

boundaryFill(x, y - 1, boundary, newColor);

}

}

void mouse(int button, int state, int x, int y) {

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

y = 500 - y; // Flip Y coordinate

if (fillMethod == 1) {

float oldColor[3] = {1.0, 1.0, 1.0}; // White background

floodFill(x, y, oldColor, fillColor);

} else if (fillMethod == 2) {

boundaryFill(x, y, boundaryColor, fillColor);

}

}

}

void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3fv(boundaryColor);

glBegin(GL\_LINE\_LOOP);

for (auto &point : polygon)

glVertex2i(point.first, point.second);

glEnd();

glFlush();

}

void keyboard(unsigned char key, int x, int y) {

if (key == 'f') fillMethod = 1;

if (key == 'b') fillMethod = 2;

if (key == 'r') polygon.clear(); // Reset polygon

glutPostRedisplay();

}

void menu(int option) {

fillMethod = option;

}

void init() {

glClearColor(1.0, 1.0, 1.0, 1.0); // White background

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(0, 500, 0, 500);

}

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

polygon.push\_back({100, 100});

polygon.push\_back({400, 100});

polygon.push\_back({400, 400});

polygon.push\_back({100, 400});

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(500, 500);

glutCreateWindow("Polygon Fill - Flood & Boundary");

init();

glutDisplayFunc(display);

glutMouseFunc(mouse);

glutKeyboardFunc(keyboard);

glutCreateMenu(menu);

glutAddMenuEntry("Flood Fill", 1);

glutAddMenuEntry("Boundary Fill", 2);

glutAttachMenu(GLUT\_RIGHT\_BUTTON);

glutMainLoop();

return 0;

}