#include <GL/glut.h>

#include <iostream>

using namespace std;

#define FILL\_MODE 1 // 0 = Flood Fill, 1 = Boundary Fill

const int windowSize = 512;

const int cellSize = 64; // 8x8 grid

const int rows = windowSize / cellSize;

const int cols = windowSize / cellSize;

// Fill color and boundary color

GLubyte fillColor[3] = {255, 0, 0}; // Red

GLubyte boundaryColor[3] = {0, 0, 0}; // Black

GLubyte originalColor[3];

void getPixelColor(int x, int y, GLubyte\* color) {

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

}

bool isSameColor(GLubyte\* a, GLubyte\* b) {

return a[0] == b[0] && a[1] == b[1] && a[2] == b[2];

}

void setPixelColor(int x, int y, GLubyte\* color) {

glColor3ub(color[0], color[1], color[2]);

glBegin(GL\_POINTS);

glVertex2i(x, y);

glEnd();

glFlush();

}

// Recursive Flood Fill

void floodFill(int x, int y, GLubyte\* targetColor) {

GLubyte current[3];

getPixelColor(x, y, current);

if (isSameColor(current, targetColor)) {

setPixelColor(x, y, fillColor);

floodFill(x + 1, y, targetColor);

floodFill(x - 1, y, targetColor);

floodFill(x, y + 1, targetColor);

floodFill(x, y - 1, targetColor);

}

}

// Recursive Boundary Fill

void boundaryFill(int x, int y) {

GLubyte current[3];

getPixelColor(x, y, current);

if (!isSameColor(current, boundaryColor) && !isSameColor(current, fillColor)) {

setPixelColor(x, y, fillColor);

boundaryFill(x + 1, y);

boundaryFill(x - 1, y);

boundaryFill(x, y + 1);

boundaryFill(x, y - 1);

}

}

void drawChessBoard() {

bool white = true;

for (int i = 0; i < rows; ++i) {

white = (i % 2 == 0);

for (int j = 0; j < cols; ++j) {

if (white)

glColor3f(1, 1, 1);

else

glColor3f(0, 0, 0);

white = !white;

int x = j \* cellSize;

int y = i \* cellSize;

glBegin(GL\_POLYGON);

glVertex2i(x, y);

glVertex2i(x + cellSize, y);

glVertex2i(x + cellSize, y + cellSize);

glVertex2i(x, y + cellSize);

glEnd();

}

}

glFlush();

}

void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

drawChessBoard();

}

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

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

y = windowSize - y; // OpenGL Y flip

GLubyte target[3];

getPixelColor(x, y, target);

if (FILL\_MODE == 0)

floodFill(x, y, target);

else

boundaryFill(x, y);

}

}

void init() {

glClearColor(1, 1, 1, 1);

gluOrtho2D(0, windowSize, 0, windowSize);

glPointSize(1.0);

}

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

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(windowSize, windowSize);

glutCreateWindow("Flood/Boundary Fill on Chessboard");

init();

glutDisplayFunc(display);

glutMouseFunc(mouse);

glutMainLoop();

return 0;

}