Name:Harshada Gopal Rayate
Roll No:19 SEDA
Subject :CGAVR

Polygon Filling - Boundary Fill Algorithm

```
#include <GL/glut.h>
#include <stdio.h>

#define WINDOW_WIDTH 500
#define WINDOW_HEIGHT 500

float fillColor[3] = {0.0f, 1.0f, 0.0f}; // Fill color (green)
float boundaryColor[3] = {1.0f, 0.0f, 0.0f}; // Boundary color (red)

// Function to plot a pixel
void plotPixel(int x, int y, float* color) {
glColor3f(color[0], color[1], color[2]);
glBegin(GL_POINTS);
glVertex2i(x, y);
glEnd();
glFlush();
}

// Function to check the color of a pixel
void getPixelColor(int x, int y, float* color) {
glReadPixels(x, y, 1, 1, GL_RGB, GL_FLOAT, color);
}

// Boundary Fill Algorithm
void boundaryFill(int x, int y) {
float currentColor[3];
getPixelColor(x, y, currentColor);

// Check if the current pixel is not the boundary and not already filled
if ((currentColor[0] != boundaryColor[0] || currentColor[1] != boundaryColor[1] || currentColor[2]
!= boundaryColor[2]) &&
(currentColor[0] != fillColor[0] || currentColor[1] != fillColor[1] || currentColor[2] != fillColor[2])) {
plotPixel(x, y, fillColor); // Fill the pixel

// Recursively fill neighboring pixels
boundaryFill(x + 1, y);
boundaryFill(x - 1, y);
boundaryFill(x, y + 1);
boundaryFill(x, y - 1);
}
}

// Function to draw the polygon
```

```
void drawPolygon() {
glBegin(GL_LINE_LOOP);
glVertex2i(100, 100);
glVertex2i(200, 300);
glVertex2i(300, 300);
glVertex2i(400, 100);
glVertex2i(300, 200);
glVertex2i(200, 200);
glEnd();
}

// Display function
void display() {
glClear(GL_COLOR_BUFFER_BIT);

// Draw the polygon
drawPolygon();

// Set the boundary color for filling
glColor3f(boundaryColor[0], boundaryColor[1], boundaryColor[2]);
// Draw the polygon outline
glFlush();
}

// Mouse click event handler
void mouse(int button, int state, int x, int y) {
if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN) {
// Convert mouse coordinates to OpenGL coordinates
int viewportY = WINDOW_HEIGHT - y; // Flip Y coordinate

// Start filling from the clicked position
boundaryFill(x, viewportY);
}
}

// Initialization function
void init() {
glClearColor(1.0f, 1.0f, 1.0f, 1.0f); // Set background color to white
gluOrtho2D(0.0f, WINDOW_WIDTH, 0.0f, WINDOW_HEIGHT); // Set up orthographic projection
}

int main(int argc, char** argv) {
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(WINDOW_WIDTH, WINDOW_HEIGHT); // Window size
glutCreateWindow("Boundary Fill Algorithm");

init();
glutDisplayFunc(display); // Register display callback function
glutMouseFunc(mouse); // Register mouse callback function
```

```
glutMainLoop(); // Enter the GLUT main loop

return 0;
}
```