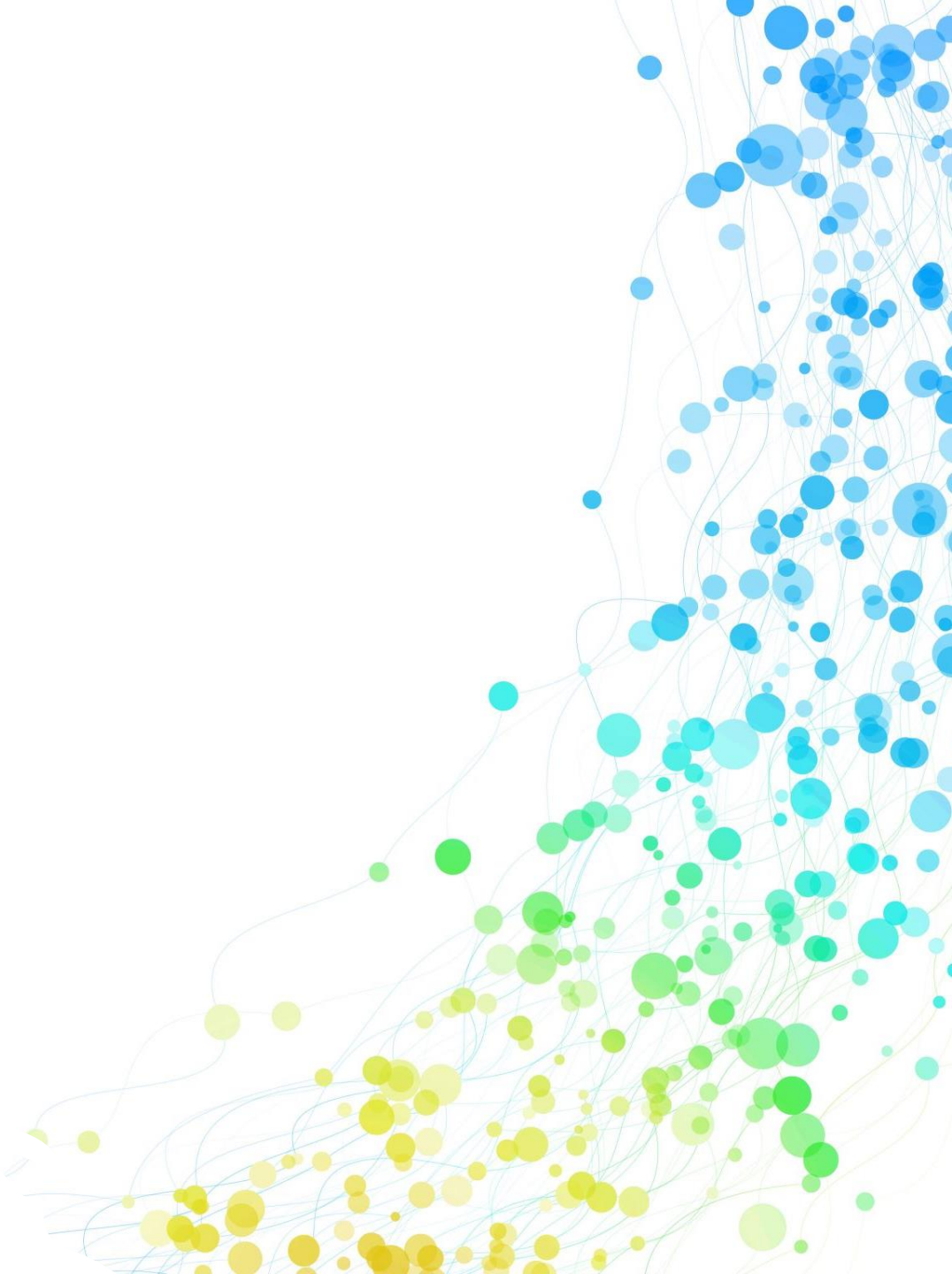
Blending

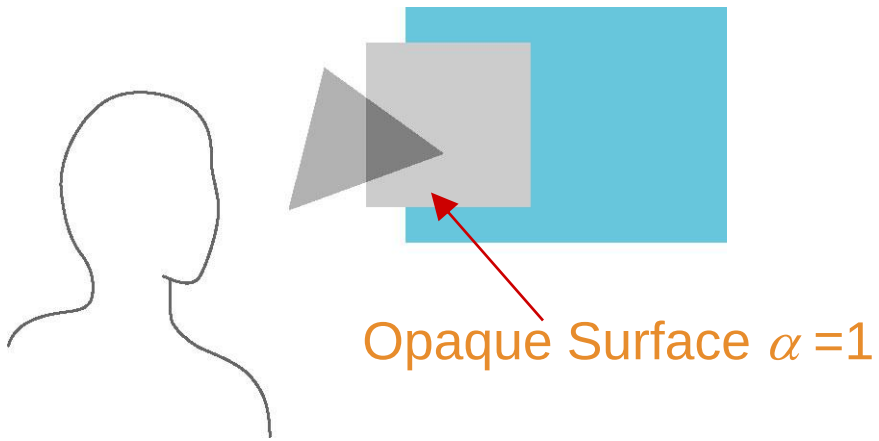
13TH WEEK, 2022



Opacity and Transparency

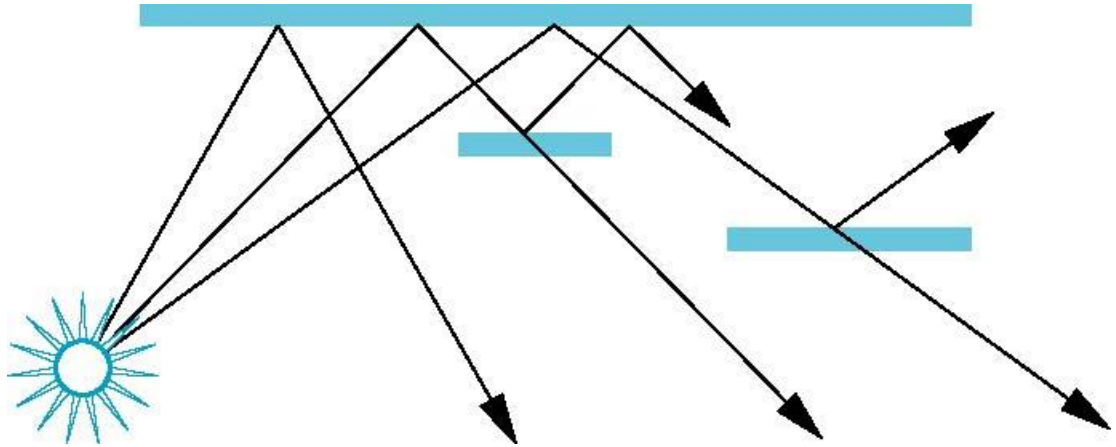
- Opaque surfaces permit no light to pass through
- Transparent surfaces permit all light to pass
- Translucent surfaces pass some light

Translucency = 1 – Opacity(α)



Physical Models

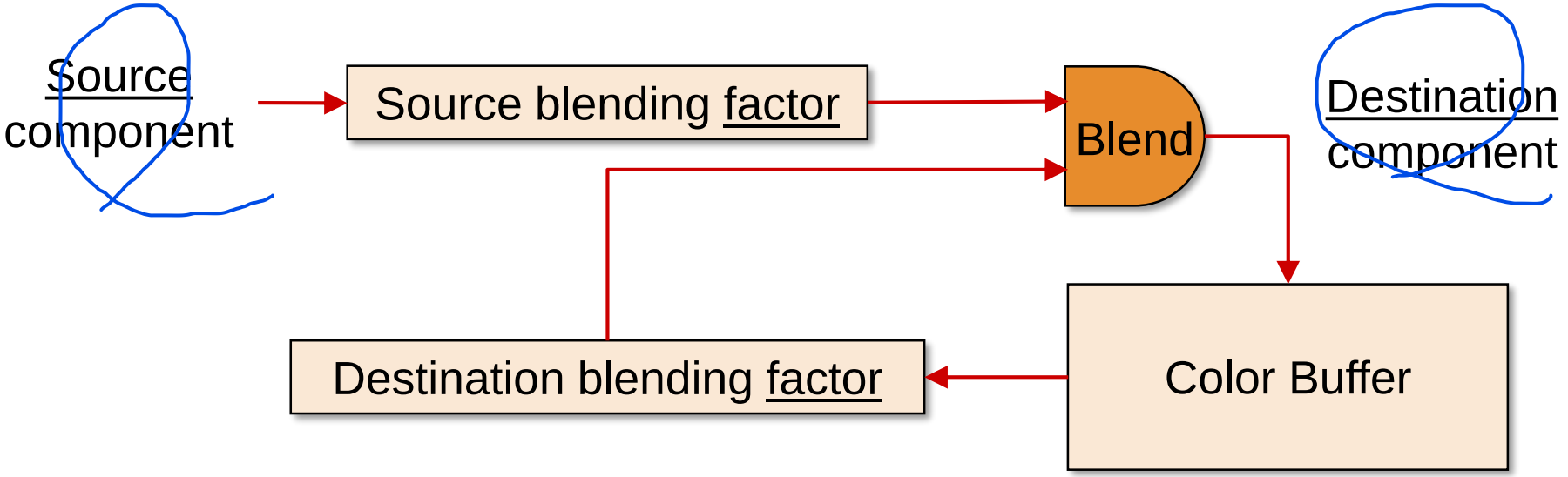
- Dealing with translucency in a physically correct manner is difficult due to
  - The complexity of the internal interactions of light and matter
  - Using a pipeline renderer



Scene with translucent objects

Writing Model for Blending

- Use A component of RGBA (or RGBα) color to store opacity
- During rendering we can expand our writing model to use RGBA values



Blending Equation

- We can define source and destination blending factors for each RGBA component

$$\mathbf{s} = [s_r, s_g, s_b, s_\alpha]$$

$$\mathbf{d} = [d_r, d_g, d_b, d_\alpha]$$

- Suppose that the source and destination colors are

$$\mathbf{b} = [b_r, b_g, b_b, b_\alpha]$$

$$\mathbf{c} = [c_r, c_g, c_b, c_\alpha]$$

- Blend as

$$\mathbf{c}' = [b_r s_r + c_r d_r, b_g s_g + c_g d_g, b_b s_b + c_b d_b, b_\alpha s_\alpha + c_\alpha d_\alpha]$$

WebGL Blending and Compositing

- Must enable blending and pick source and destination factors

```
gl.enable( gl.BLEND );
gl.blendFunc( source_factor, destination_factor );
```

- Only certain factors supported
  - `gl.ZERO`, `gl.ONE`
  - `gl.SRC_ALPHA`, `gl.ONE_MIMUS_SRC_ALPHA`
  - `gl.DST_ALPHA`, `gl.ONE_MIMUS_DST_ALPHA`
  - `gl.SRC_COLOR`, `gl.ONE_MIMUS_SRC_COLOR`
  - `gl.DST_COLOR`, `gl.ONE_MIMUS_DST_COLOR`
  - `gl.SRC_ALPHA_SATURATE`

Example: Blending

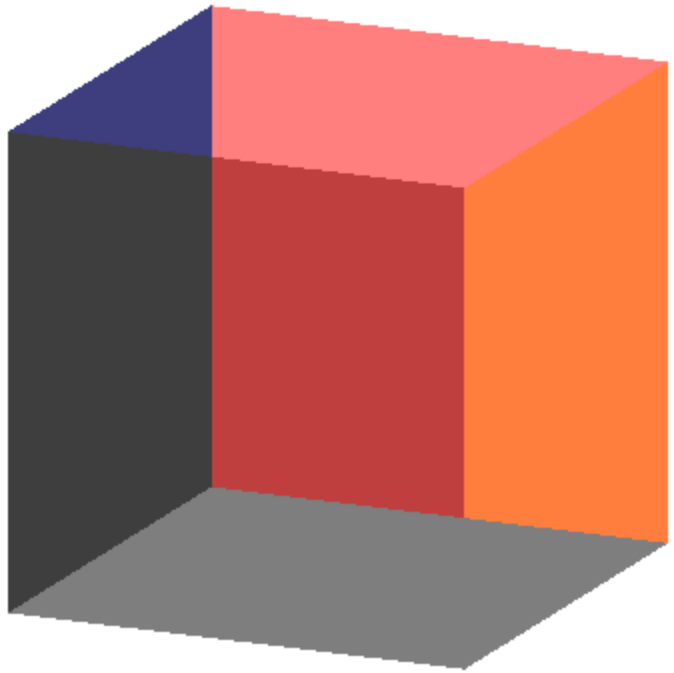
- Suppose that we start with the opaque background color $(R_0, G_0, B_0, 1)$
  - This color becomes the initial destination color
- We now want to blend in a translucent polygon with color $(R_1, G_1, B_1, \alpha_1)$
- Select `gl.SRC_ALPHA` and `gl.ONE_MINUS_SRC_ALPHA` as the source and destination blending factors

$$R'_1 = \alpha_1 R_1 + (1 - \alpha_1) R_0 \qquad G'_1 = \alpha_1 G_1 + (1 - \alpha_1) G_0 \qquad B'_1 = \alpha_1 B_1 + (1 - \alpha_1) B_0$$

  - Note that this formula is correct if polygon is either opaque or transparent

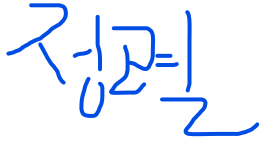
Order Dependency

- Is this image correct?
  - Probably not
  - Polygons are rendered in the order they pass down the pipeline
  - Blending functions are order dependent



Opaque and Translucent Polygons

- Suppose that we have a group of polygons some of which are opaque and some translucent
  - How do we use hidden-surface removal?
  - Opaque polygons block all polygons behind them and affect the depth buffer
  - Translucent polygons should not affect depth buffer
    - Render with `gl.depthMask(false)` which makes depth buffer read-only
- Sort polygons first to remove order dependency



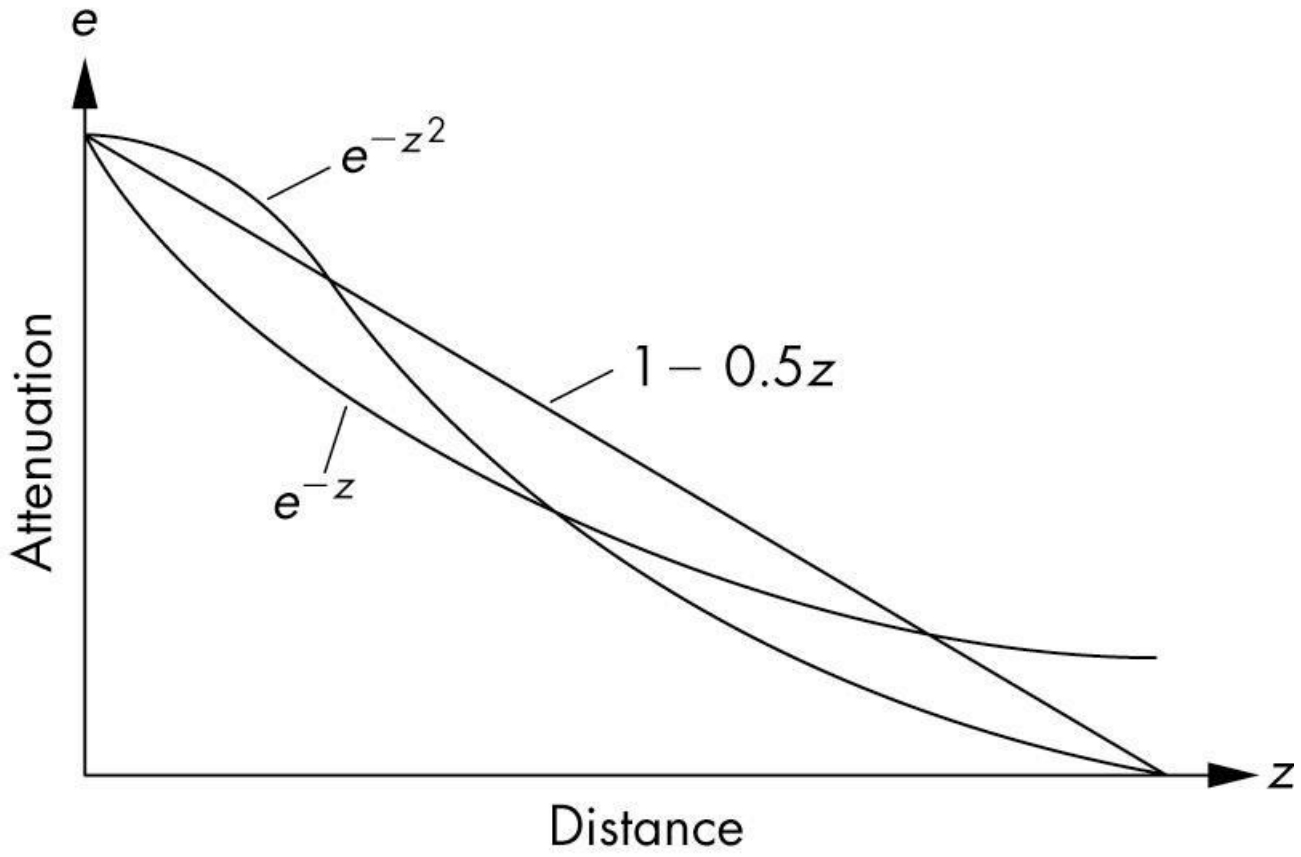
Fog

- We can composite with a fixed color and have the blending factors depend on depth
  - Simulates a fog effect
- Blend source color C_s and fog color C_f by

$$\mathbf{C}_s' = f\,\mathbf{C}_s + (1-f)\,\mathbf{C}_f$$

- f is the fog factor
  - Exponential
  - Gaussian
  - Linear (depth cueing)

Fog Functions

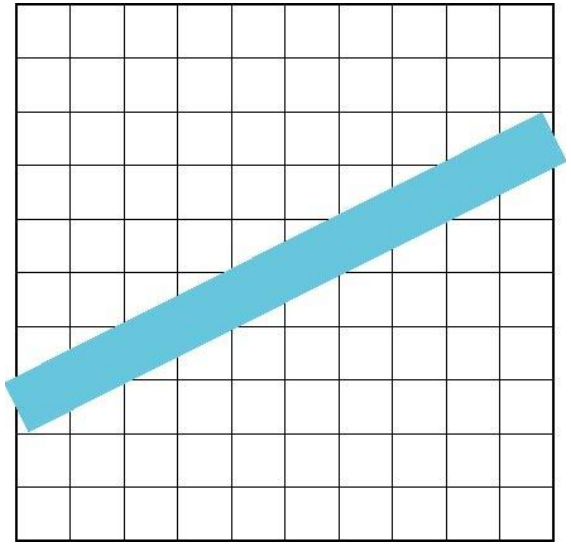


Compositing and HTML

- In desktop OpenGL, the A component has no effect unless blending is enabled
- In WebGL, an A other than 1.0 has an effect because WebGL works with the HTML5 Canvas element
- A = 0.5 will cut the RGB values by ½ when the pixel is displayed
- Allows other applications to be blended into the canvas along with the graphics

Line Aliasing

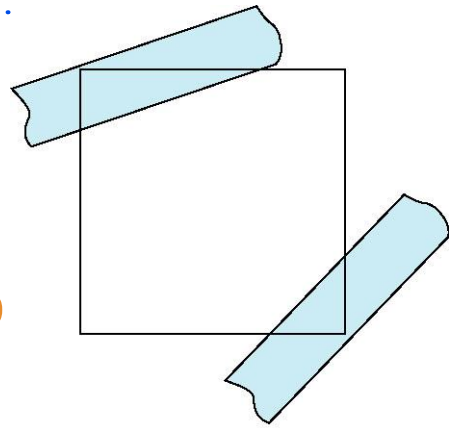
- Ideal raster line is one pixel wide
- All line segments, other than vertical and horizontal segments, partially cover pixels
- Simple algorithms color only whole pixels
- Lead to the “jaggies” or aliasing 계단현상
- Similar issue for polygons



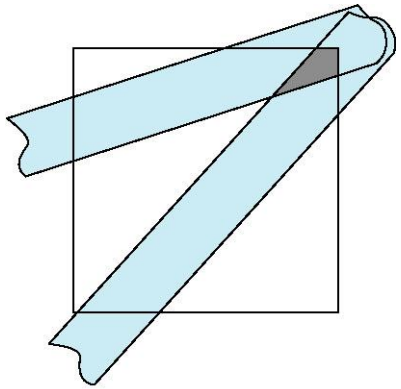
Antialiasing

조각

- Can try to color a pixel by adding a fraction of its color to the frame buffer
  - Fraction depends on percentage of pixel covered by fragment
    - Setting the alpha value for the corresponding pixel to be a number between 0 and 1 that is the amount of that pixel covered by the fragment
  - Fraction depends on whether there is overlap



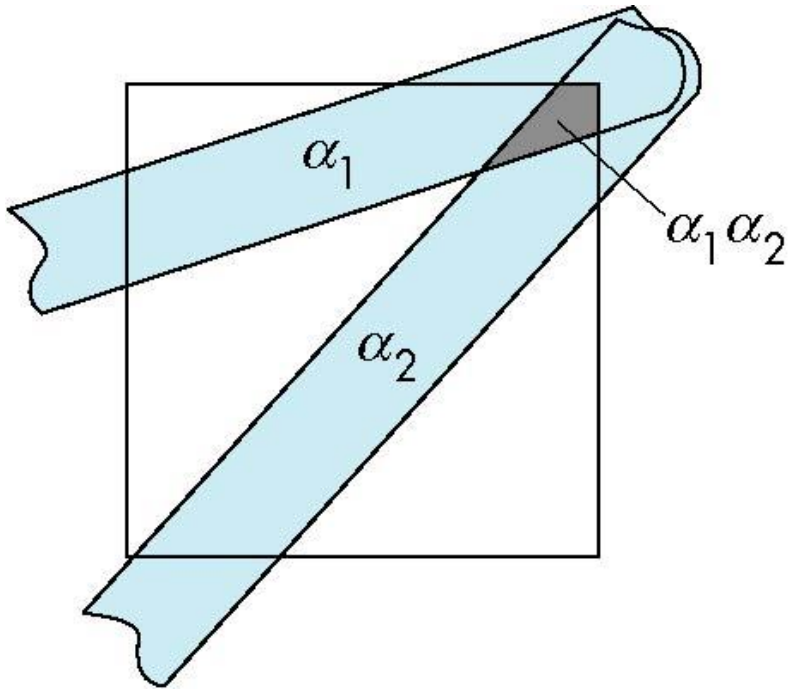
No Overlap



Overlap

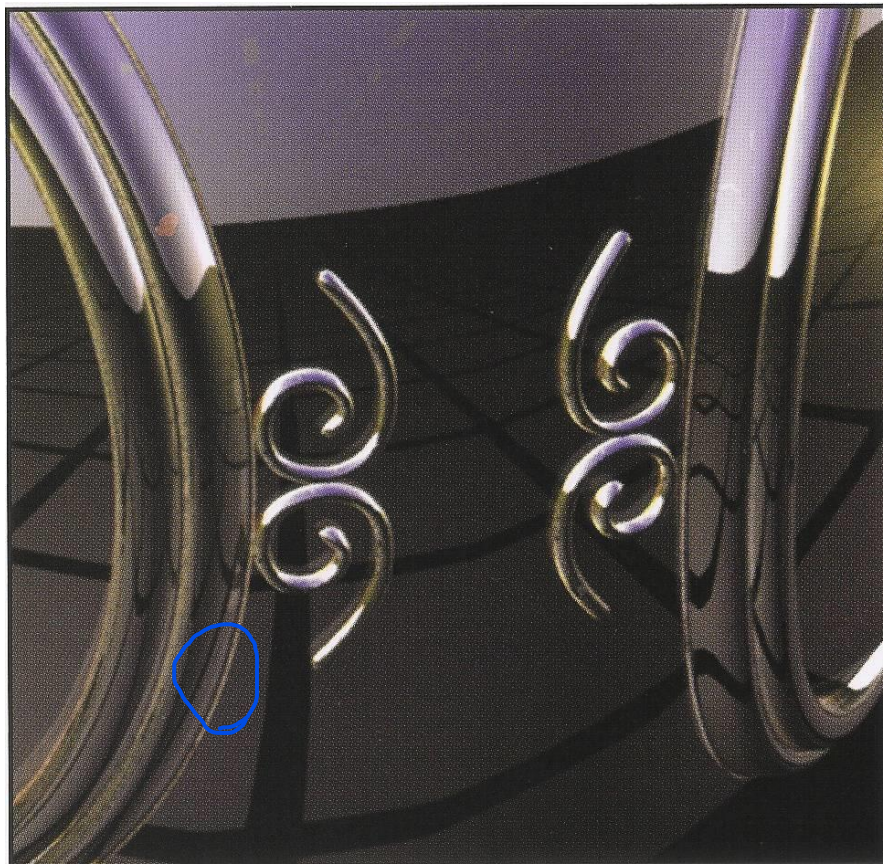
Area Averaging

- Use average area $\alpha_1+\alpha_2-\alpha_1\alpha_2$ as blending factor



Example: Antialiasing

Without Antialiasing



Antialiasing

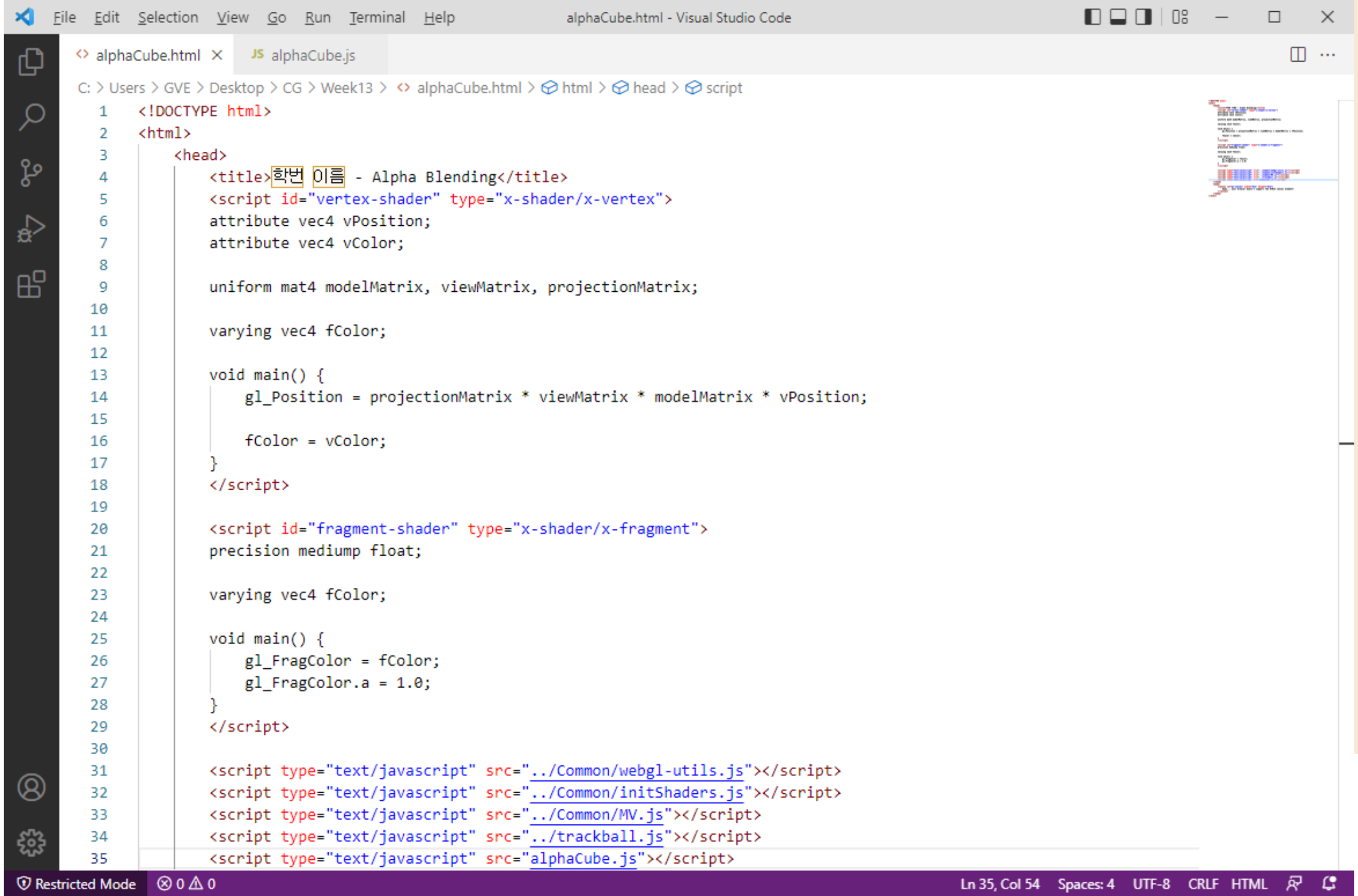
OpenGL Antialiasing

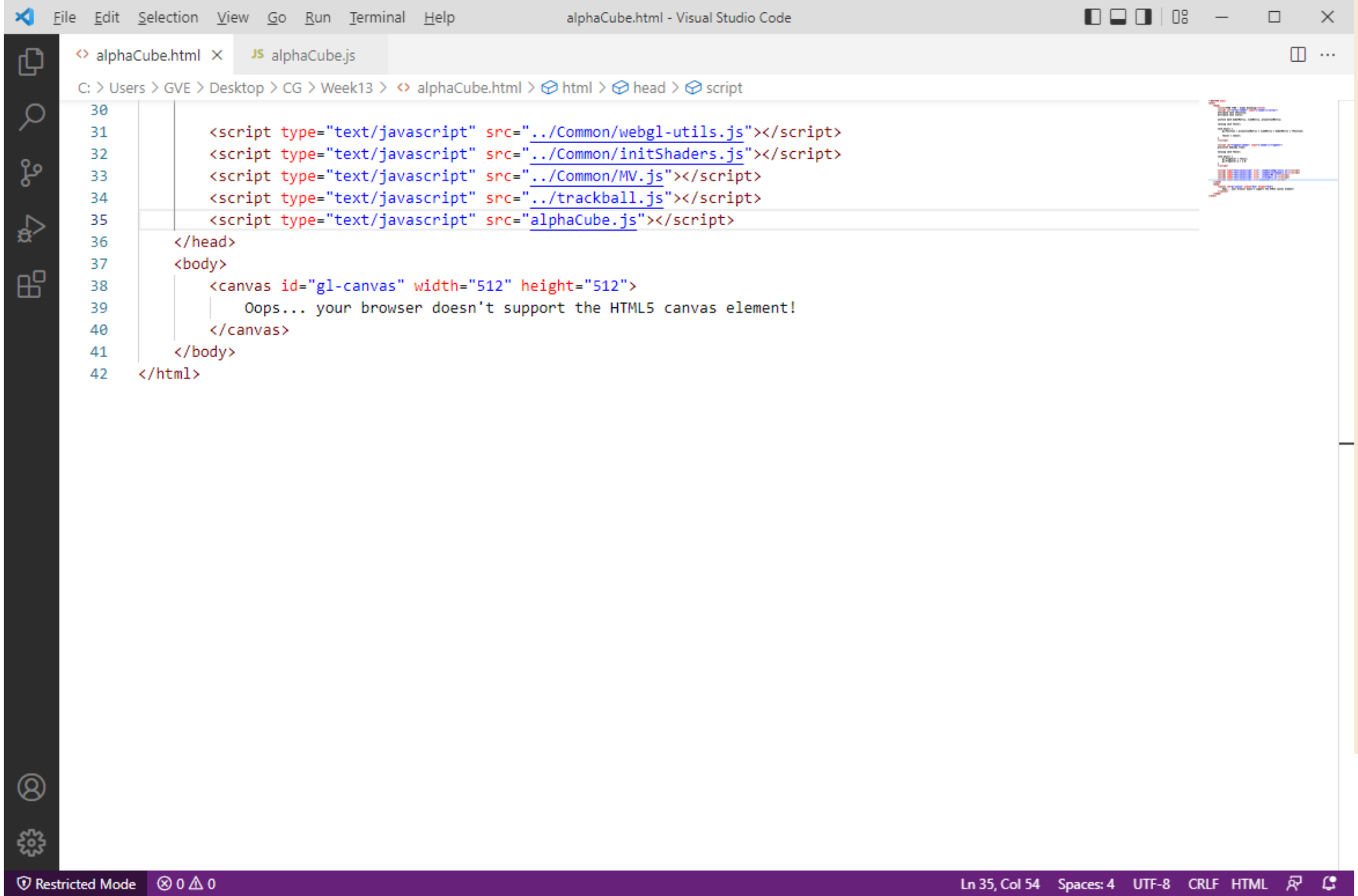
- Not (yet) supported in WebGL
- Can enable separately for points, lines, or polygons

```
glEnable( GL_POINT_SMOOTH );
glEnable( GL_LINE_SMOOTH );
glEnable( GL_POLYGON_SMOOTH );

glEnable( GL_BLEND );
glBlendFunc( GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA );
```

- Note most hardware will automatically antialias


File Edit Selection View Go Run Terminal Help
alphaCube.html - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.html > html > head > script
<!DOCTYPE html>
<html>
<head>
<title>학번 이름 - Alpha Blending</title>
<script id="vertex-shader" type="x-shader/x-vertex">
attribute vec4 vPosition;
attribute vec4 vColor;
uniform mat4 modelMatrix, viewMatrix, projectionMatrix;
varying vec4 fColor;
void main() {
gl_Position = projectionMatrix * viewMatrix * modelMatrix * vPosition;
fColor = vColor;
}
</script>
<script id="fragment-shader" type="x-shader/x-fragment">
precision mediump float;
varying vec4 fColor;
void main() {
gl_FragColor = fColor;
gl_FragColor.a = 1.0;
}
</script>
<script type="text/javascript" src="../Common/webgl-utils.js"></script>
<script type="text/javascript" src="../Common/initShaders.js"></script>
<script type="text/javascript" src="../Common/MV.js"></script>
<script type="text/javascript" src="../trackball.js"></script>
<script type="text/javascript" src="alphaCube.js"></script>
Restricted Mode
Ln 35, Col 54 Spaces: 4 UTF-8 CRLF HTML


File Edit Selection View Go Run Terminal Help
alphaCube.html - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.html > html > head > script
30
31 <script type="text/javascript" src="../Common/webgl-utils.js"></script>
32 <script type="text/javascript" src="../Common/initShaders.js"></script>
33 <script type="text/javascript" src="../Common/MV.js"></script>
34 <script type="text/javascript" src="../trackball.js"></script>
35 <script type="text/javascript" src="alphaCube.js"></script>
36 </head>
37 <body>
38 <canvas id="gl-canvas" width="512" height="512">
39 Oops... your browser doesn't support the HTML5 canvas element!
40 </canvas>
41 </body>
42 </html>
Restricted Mode
0 0
Ln 35, Col 54
Spaces: 4
UTF-8
CRLF
HTML

File Edit Selection View Go Run Terminal Help alphaCube.js - Visual Studio Code

alphaCube.html | alphaCube.js

C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > ...

```
var gl;
var points = [];
var colors = [];

//var modelMatrix;
var modelMatrixLoc;

var trballMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
var numVertCubeTri;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    generateColorCube();

    // virtual trackball
    var trball = trackball(canvas.width, canvas.height);
    var mouseDown = false;

    canvas.addEventListener("mousedown", function (event) {
        trball.start(event.clientX, event.clientY);

        mouseDown = true;
    });

    canvas.addEventListener("mouseup", function (event) {
        mouseDown = false;
    });

```

Restricted Mode ⊗ 0 ⚠ 0 Ln 174, Col 1 Spaces: 4 UTF-8 CRLF {} JavaScript

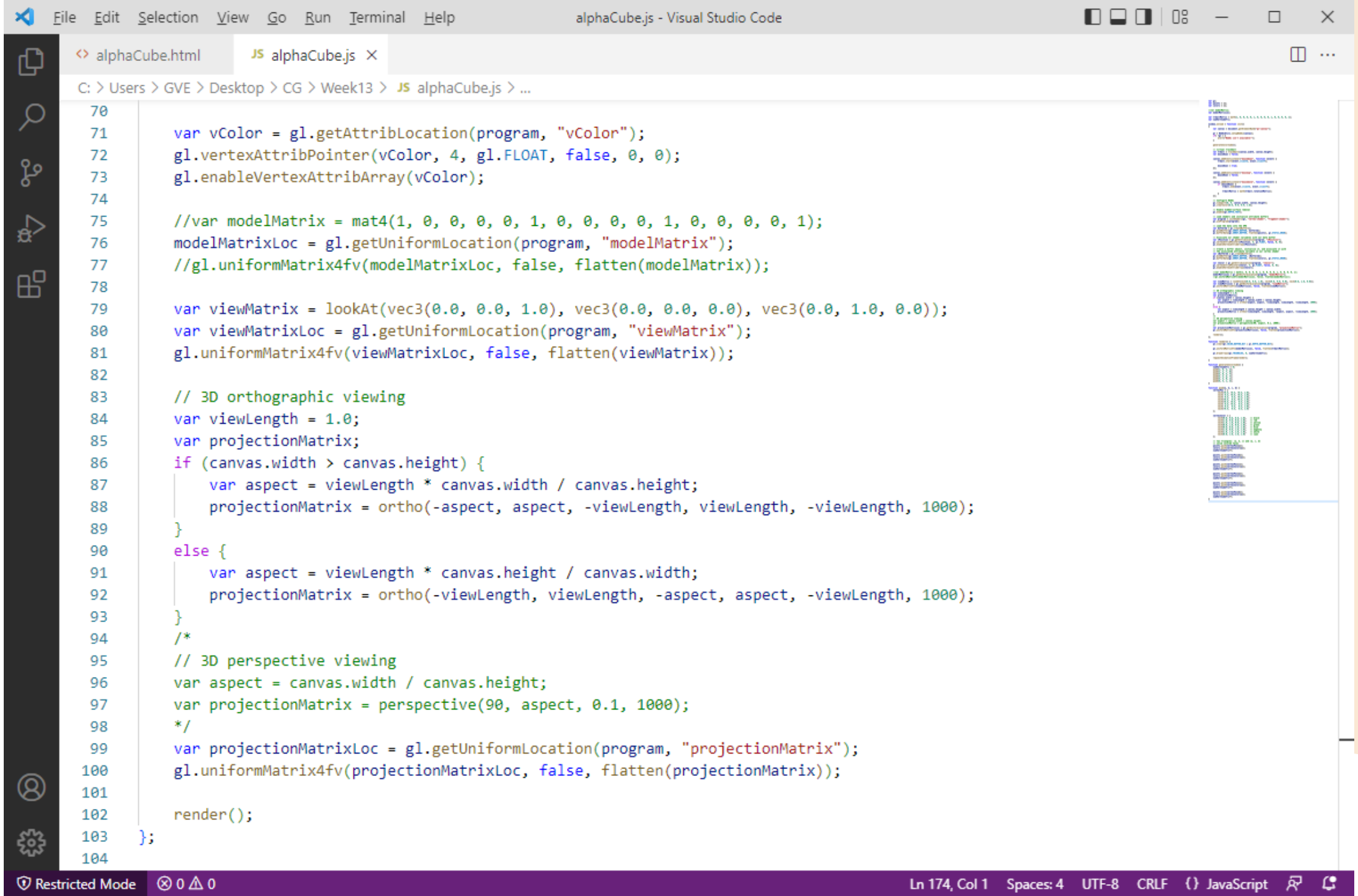
File Edit Selection View Go Run Terminal Help alphaCube.js - Visual Studio Code

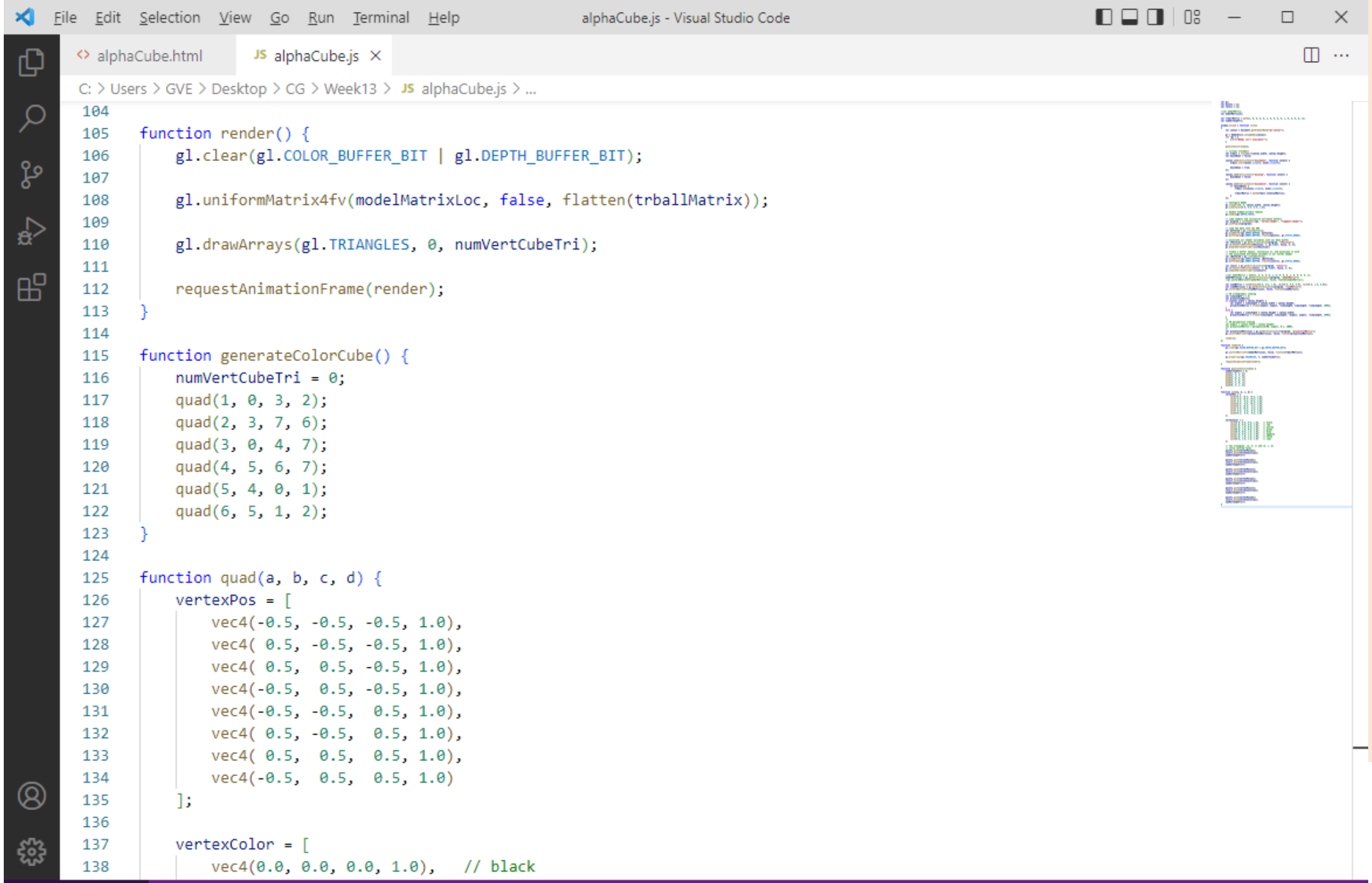
alphaCube.html | alphaCube.js

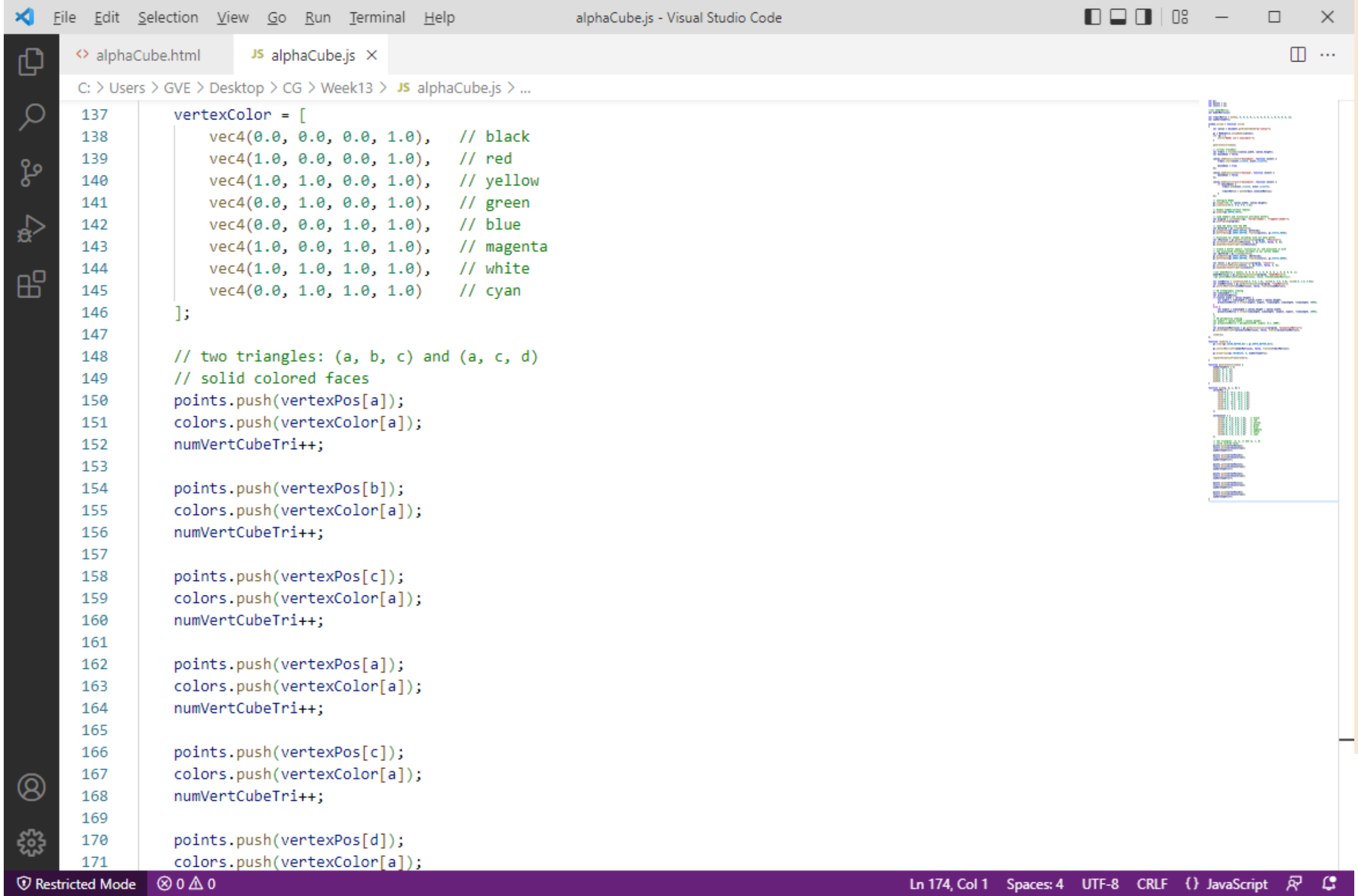
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > ...

```
35
36      canvas.addEventListener("mousemove", function (event) {
37          if (mouseDown) {
38              trball.end(event.clientX, event.clientY);
39
40              trballMatrix = mat4(trball.rotationMatrix);
41          }
42      });
43
44      // Configure WebGL
45      gl.viewport(0, 0, canvas.width, canvas.height);
46      gl.clearColor(0.9, 0.9, 0.9, 1.0);
47
48      // Enable hidden-surface removal
49      gl.enable(gl.DEPTH_TEST);
50
51      // Load shaders and initialize attribute buffers
52      var program = initShaders(gl, "vertex-shader", "fragment-shader");
53      gl.useProgram(program);
54
55      // Load the data into the GPU
56      var bufferId = gl.createBuffer();
57      gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
58      gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
59
60      // Associate our shader variables with our data buffer
61      var vPosition = gl.getAttribLocation(program, "vPosition");
62      gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
63      gl.enableVertexAttribArray(vPosition);
64
65      // Create a buffer object, initialize it, and associate it with
66      // the associated attribute variable in our vertex shader
67      var cBufferId = gl.createBuffer();
68      gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
69      gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
```

Restricted Mode ⊗ 0 ⚠ 0 Ln 174, Col 1 Spaces: 4 UTF-8 CRLF {} JavaScript


File Edit Selection View Go Run Terminal Help
alphaCube.js - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > ...
70
71 var vColor = gl.getAttribLocation(program, "vColor");
72 gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
73 gl.enableVertexAttribArray(vColor);
74
75 //var modelMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
76 modelMatrixLoc = gl.getUniformLocation(program, "modelMatrix");
77 //gl.uniformMatrix4fv(modelMatrixLoc, false, flatten(modelMatrix));
78
79 var viewMatrix = lookAt(vec3(0.0, 0.0, 1.0), vec3(0.0, 0.0, 0.0), vec3(0.0, 1.0, 0.0));
80 var viewMatrixLoc = gl.getUniformLocation(program, "viewMatrix");
81 gl.uniformMatrix4fv(viewMatrixLoc, false, flatten(viewMatrix));
82
83 // 3D orthographic viewing
84 var viewLength = 1.0;
85 var projectionMatrix;
86 if (canvas.width > canvas.height) {
87 var aspect = viewLength * canvas.width / canvas.height;
88 projectionMatrix = ortho(-aspect, aspect, -viewLength, viewLength, -viewLength, 1000);
89 }
90 else {
91 var aspect = viewLength * canvas.height / canvas.width;
92 projectionMatrix = ortho(-viewLength, viewLength, -aspect, aspect, -viewLength, 1000);
93 }
94 /*
95 // 3D perspective viewing
96 var aspect = canvas.width / canvas.height;
97 var projectionMatrix = perspective(90, aspect, 0.1, 1000);
98 */
99 var projectionMatrixLoc = gl.getUniformLocation(program, "projectionMatrix");
100 gl.uniformMatrix4fv(projectionMatrixLoc, false, flatten(projectionMatrix));
101
102 render();
103 };
104
Restricted Mode
0 0
Ln 174, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript

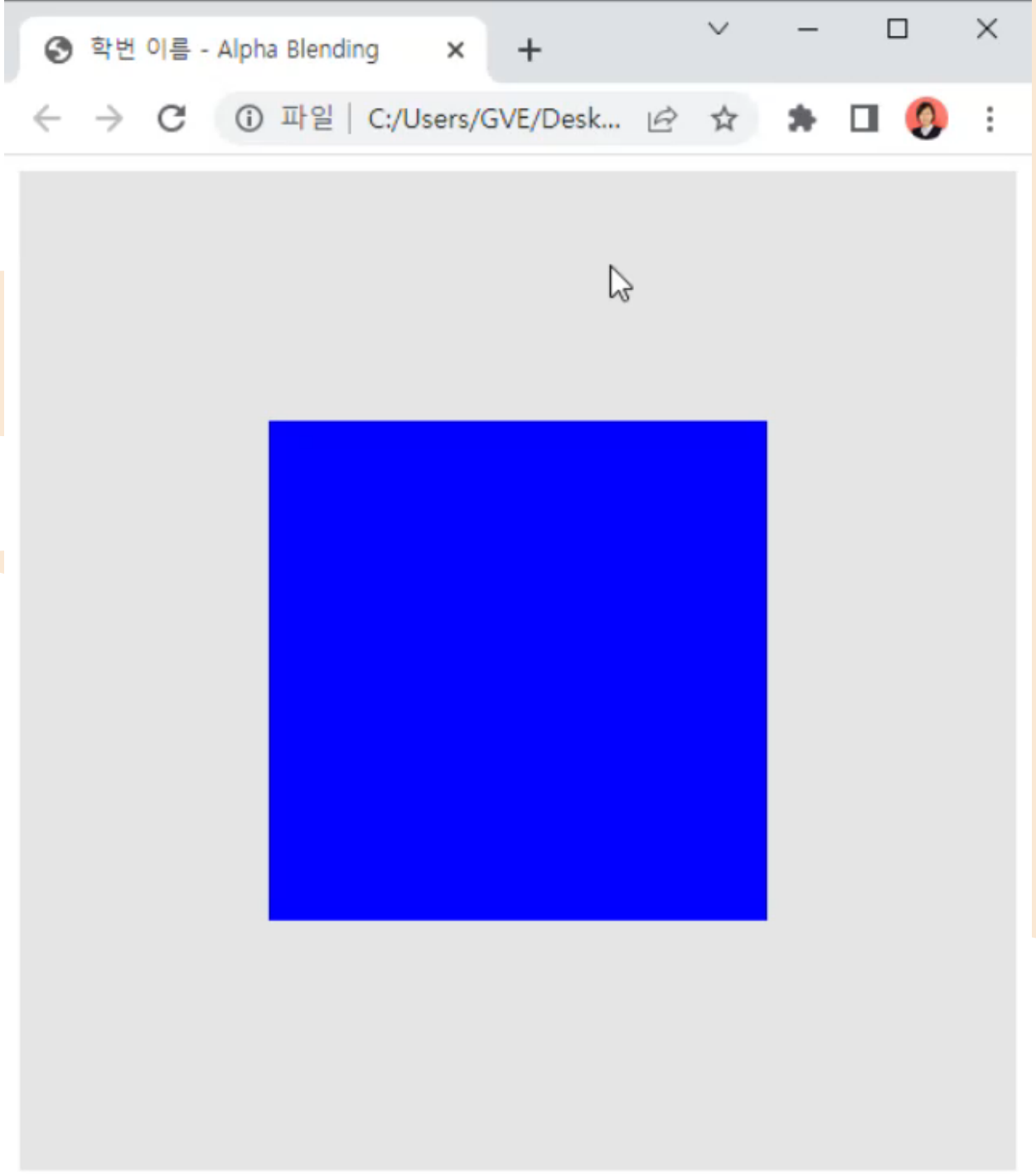

File Edit Selection View Go Run Terminal Help
alphaCube.js - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > ...
104
105 function render() {
106 gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
107
108 gl.uniformMatrix4fv(modelMatrixLoc, false, flatten(trballMatrix));
109
110 gl.drawArrays(gl.TRIANGLES, 0, numVertCubeTri);
111
112 requestAnimationFrame(render);
113 }
114
115 function generateColorCube() {
116 numVertCubeTri = 0;
117 quad(1, 0, 3, 2);
118 quad(2, 3, 7, 6);
119 quad(3, 0, 4, 7);
120 quad(4, 5, 6, 7);
121 quad(5, 4, 0, 1);
122 quad(6, 5, 1, 2);
123 }
124
125 function quad(a, b, c, d) {
126 vertexPos = [
127 vec4(-0.5, -0.5, -0.5, 1.0),
128 vec4(0.5, -0.5, -0.5, 1.0),
129 vec4(0.5, 0.5, -0.5, 1.0),
130 vec4(-0.5, 0.5, -0.5, 1.0),
131 vec4(-0.5, -0.5, 0.5, 1.0),
132 vec4(0.5, -0.5, 0.5, 1.0),
133 vec4(0.5, 0.5, 0.5, 1.0),
134 vec4(-0.5, 0.5, 0.5, 1.0)
135];
136
137 vertexColor = [
138 vec4(0.0, 0.0, 0.0, 1.0), // black
Restricted Mode
0 0
Ln 174, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript


File Edit Selection View Go Run Terminal Help
alphaCube.js - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > ...
137 vertexColor = [
138 vec4(0.0, 0.0, 0.0, 1.0), // black
139 vec4(1.0, 0.0, 0.0, 1.0), // red
140 vec4(1.0, 1.0, 0.0, 1.0), // yellow
141 vec4(0.0, 1.0, 0.0, 1.0), // green
142 vec4(0.0, 0.0, 1.0, 1.0), // blue
143 vec4(1.0, 0.0, 1.0, 1.0), // magenta
144 vec4(1.0, 1.0, 1.0, 1.0), // white
145 vec4(0.0, 1.0, 1.0, 1.0) // cyan
146];
147
148 // two triangles: (a, b, c) and (a, c, d)
149 // solid colored faces
150 points.push(vertexPos[a]);
151 colors.push(vertexColor[a]);
152 numVertCubeTri++;
153
154 points.push(vertexPos[b]);
155 colors.push(vertexColor[a]);
156 numVertCubeTri++;
157
158 points.push(vertexPos[c]);
159 colors.push(vertexColor[a]);
160 numVertCubeTri++;
161
162 points.push(vertexPos[a]);
163 colors.push(vertexColor[a]);
164 numVertCubeTri++;
165
166 points.push(vertexPos[c]);
167 colors.push(vertexColor[a]);
168 numVertCubeTri++;
169
170 points.push(vertexPos[d]);
171 colors.push(vertexColor[a]);
Restricted Mode
0 0
Ln 174, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript

alphaCube.html | alphaCube.js

C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > ...

```
148     // two triangles: (a, b, c) and (a, c, d)
149     // solid colored faces
150     points.push(vertexPos[a]);
151     colors.push(vertexColor[a]);
152     numVertCubeTri++;
153
154     points.push(vertexPos[b]);
155     colors.push(vertexColor[a]);
156     numVertCubeTri++;
157
158     points.push(vertexPos[c]);
159     colors.push(vertexColor[a]);
160     numVertCubeTri++;
161
162     points.push(vertexPos[a]);
163     colors.push(vertexColor[a]);
164     numVertCubeTri++;
165
166     points.push(vertexPos[c]);
167     colors.push(vertexColor[a]);
168     numVertCubeTri++;
169
170     points.push(vertexPos[d]);
171     colors.push(vertexColor[a]);
172     numVertCubeTri++;
173 }
174
```


학번 이름 - Alpha Blending
파일 | C:/Users/GVE/Desk...

File Edit Selection View Go Run Terminal Help alphaCube.html - Visual Studio Code

alphaCube.html alphaCube.js

C: > Users > GVE > Desktop > CG > Week13 > alphaCube.html > html > head > script#fragment-shader

```
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - Alpha Blending</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;

        uniform mat4 modelMatrix, viewMatrix, projectionMatrix;

        varying vec4 fColor;

        void main() {
            gl_Position = projectionMatrix * viewMatrix * modelMatrix * vPosition;

            fColor = vColor;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;

        varying vec4 fColor;

        void main() {
            gl_FragColor = fColor;
            //gl_FragColor.a = 1.0;
        }
        </script>

        <script type="text/javascript" src="../Common/webgl-utils.js"></script>
        <script type="text/javascript" src="../Common/initShaders.js"></script>
        <script type="text/javascript" src="../Common/MV.js"></script>
        <script type="text/javascript" src="../trackball.js"></script>
        <script type="text/javascript" src="alphaCube.js"></script>
```

Restricted Mode ⊗ 0 ⚠ 0 Ln 27, Col 15 Spaces: 4 UTF-8 CRLF HTML

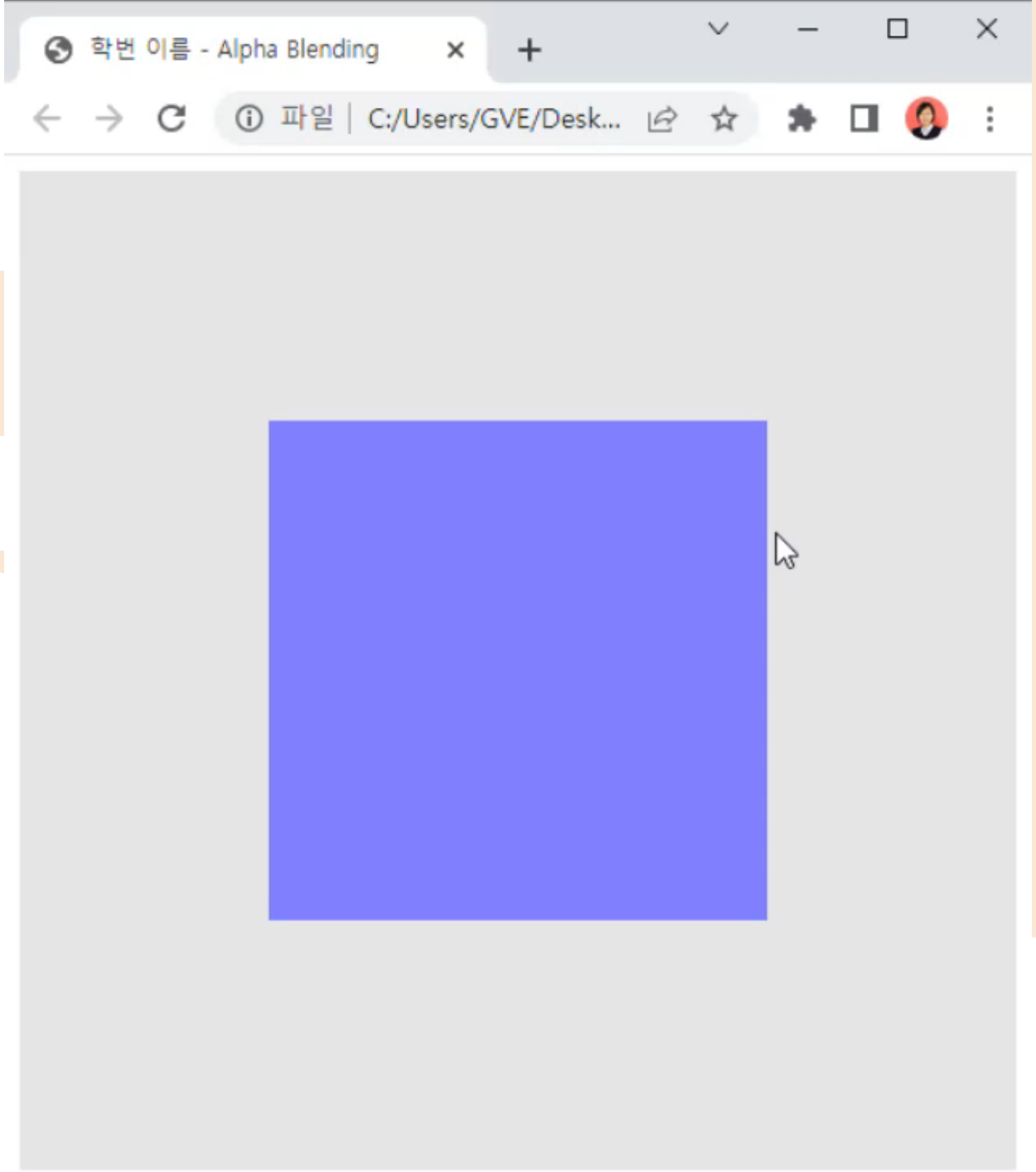
File Edit Selection View Go Run Terminal Help alphaCube.js - Visual Studio Code

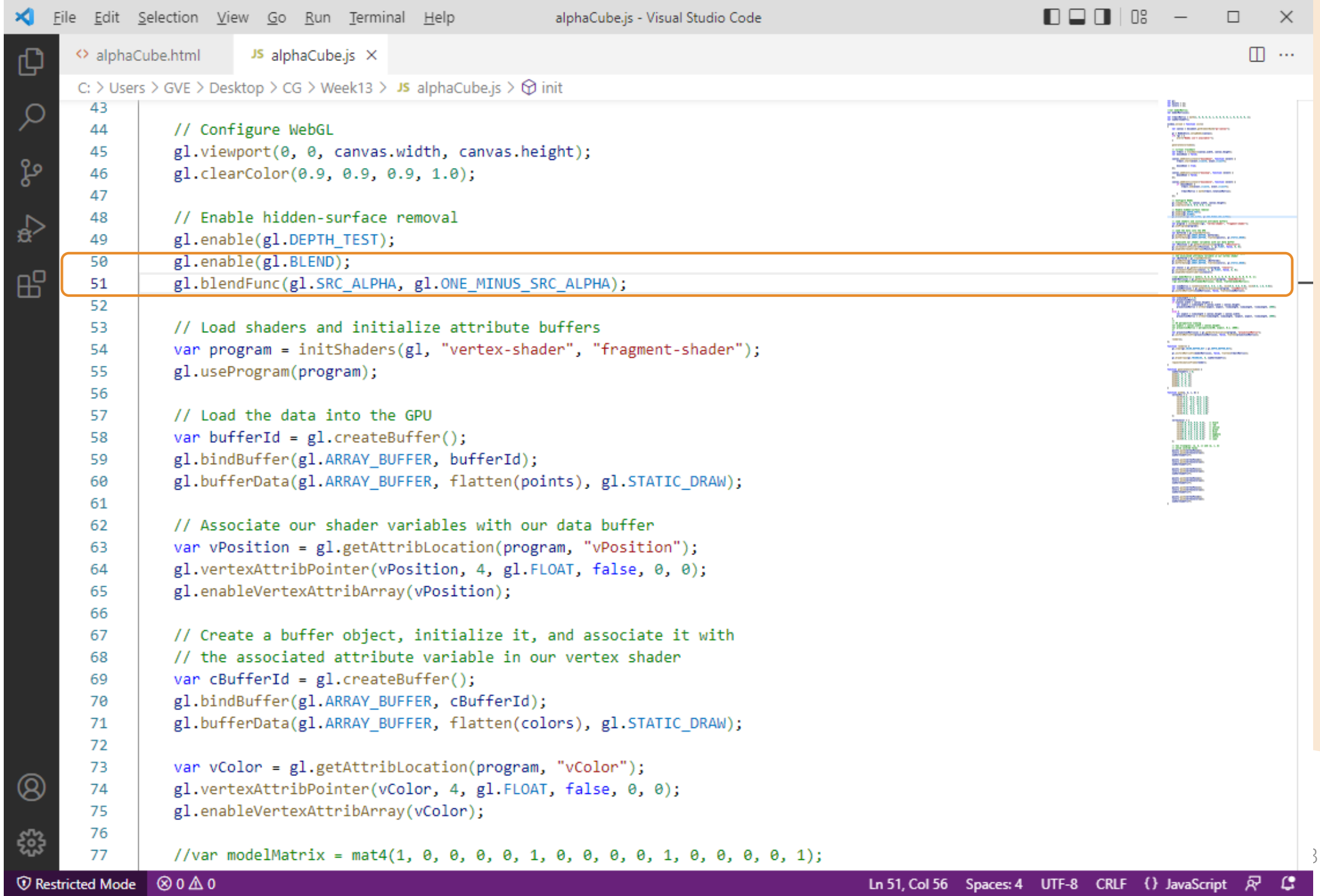
alphaCube.html | alphaCube.js

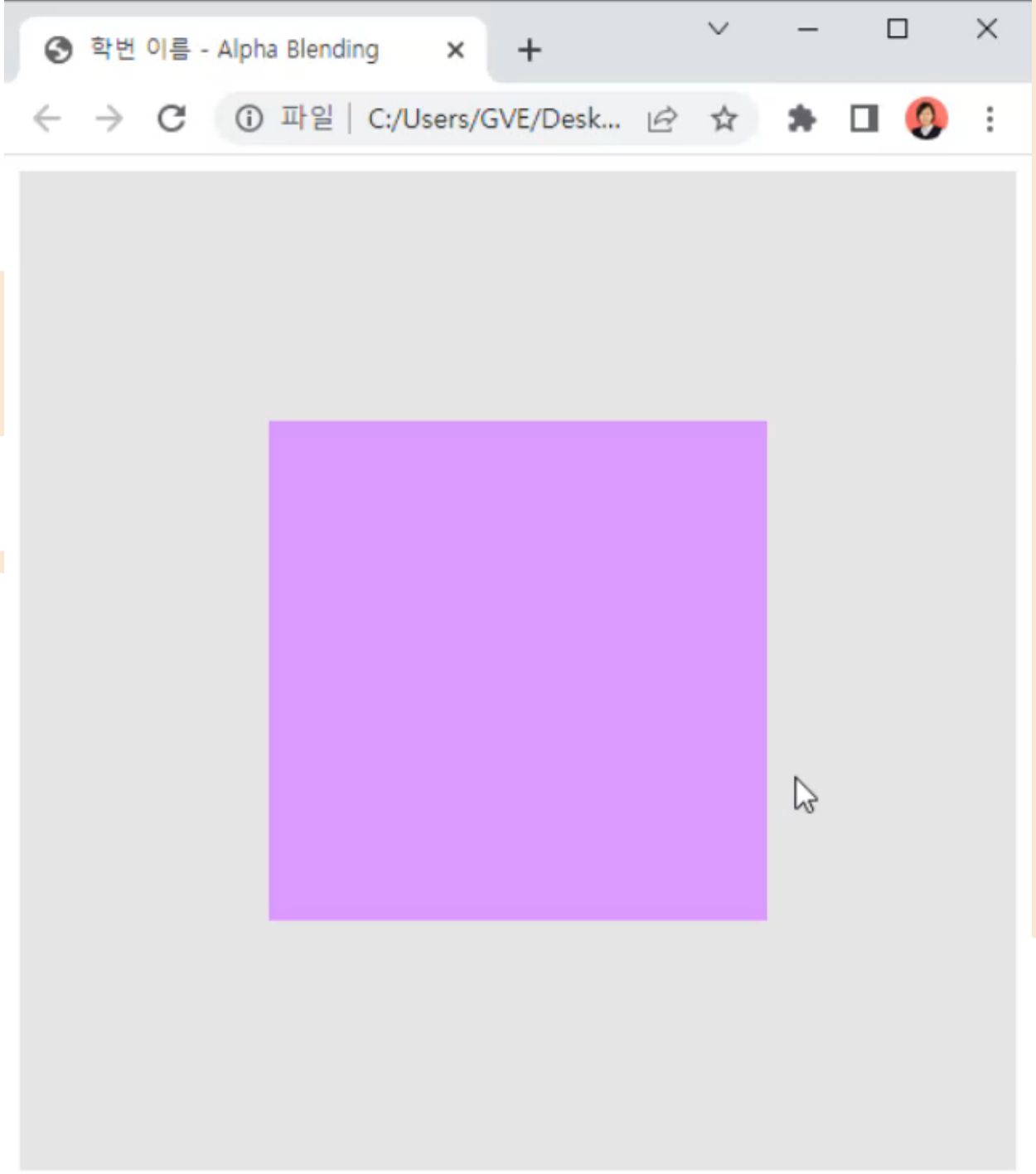
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > quad

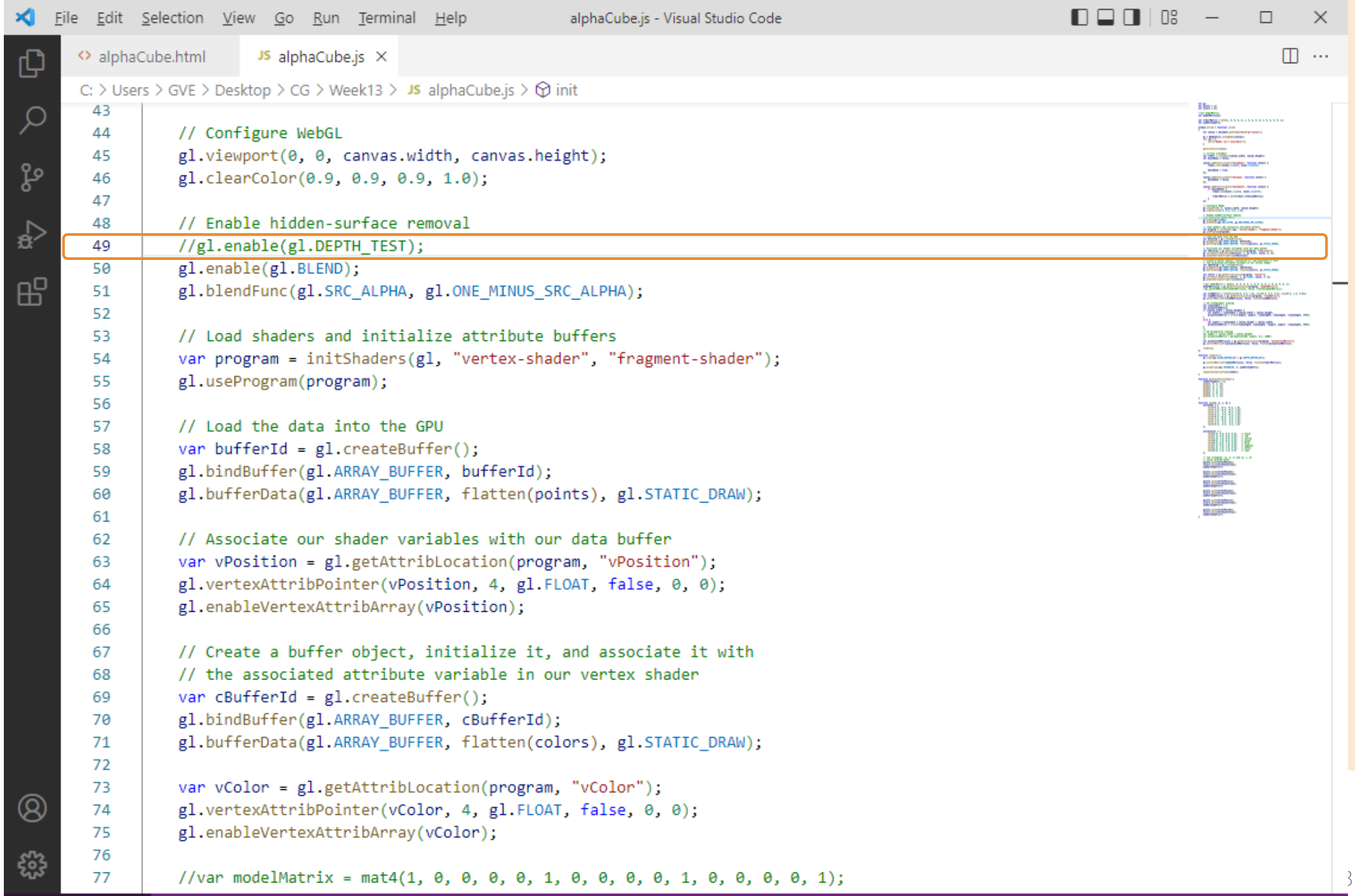
```
124
125 function quad(a, b, c, d) {
126     vertexPos = [
127         vec4(-0.5, -0.5, -0.5, 1.0),
128         vec4( 0.5, -0.5, -0.5, 1.0),
129         vec4( 0.5,  0.5, -0.5, 1.0),
130         vec4(-0.5,  0.5, -0.5, 1.0),
131         vec4(-0.5, -0.5,  0.5, 1.0),
132         vec4( 0.5, -0.5,  0.5, 1.0),
133         vec4( 0.5,  0.5,  0.5, 1.0),
134         vec4(-0.5,  0.5,  0.5, 1.0)
135     ];
136
137     vertexColor = [
138         vec4(0.0, 0.0, 0.0, 0.5),   // black
139         vec4(1.0, 0.0, 0.0, 0.5),   // red
140         vec4(1.0, 1.0, 0.0, 0.5),   // yellow
141         vec4(0.0, 1.0, 0.0, 0.5),   // green
142         vec4(0.0, 0.0, 1.0, 0.5),   // blue
143         vec4(1.0, 0.0, 1.0, 0.5),   // magenta
144         vec4(1.0, 1.0, 1.0, 0.5),   // white
145         vec4(0.0, 1.0, 1.0, 0.5)    // cyan
146     ];
147
148     // two triangles: (a, b, c) and (a, c, d)
149     // solid colored faces
150     points.push(vertexPos[a]);
151     colors.push(vertexColor[a]);
152     numVertCubeTri++;
153
154     points.push(vertexPos[b]);
155     colors.push(vertexColor[a]);
156     numVertCubeTri++;
157
158     points.push(vertexPos[c]);
```

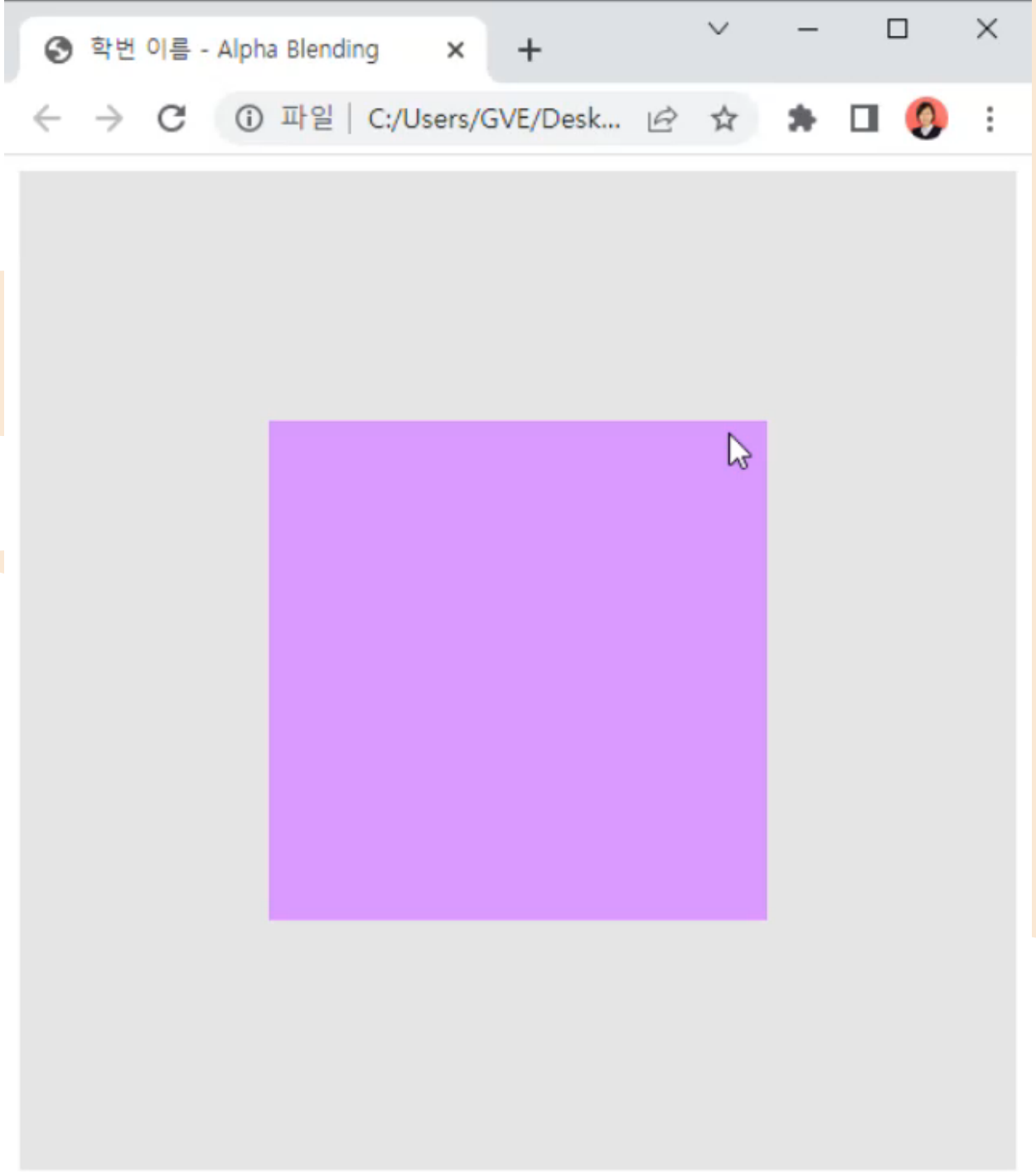
Restricted Mode ⊗ 0 ⚠ 0 Ln 145, Col 32 Spaces: 4 UTF-8 CRLF {} JavaScript


학번 이름 - Alpha Blending
파일 | C:/Users/GVE/Desk...


File Edit Selection View Go Run Terminal Help
alphaCube.js - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > init
43
44 // Configure WebGL
45 gl.viewport(0, 0, canvas.width, canvas.height);
46 gl.clearColor(0.9, 0.9, 0.9, 1.0);
47
48 // Enable hidden-surface removal
49 gl.enable(gl.DEPTH_TEST);
50 gl.enable(gl.BLEND);
51 gl.blendFunc(gl.SRC_ALPHA, gl.ONE_MINUS_SRC_ALPHA);
52
53 // Load shaders and initialize attribute buffers
54 var program = initShaders(gl, "vertex-shader", "fragment-shader");
55 gl.useProgram(program);
56
57 // Load the data into the GPU
58 var bufferId = gl.createBuffer();
59 gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
60 gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
61
62 // Associate our shader variables with our data buffer
63 var vPosition = gl.getAttribLocation(program, "vPosition");
64 gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
65 gl.enableVertexAttribArray(vPosition);
66
67 // Create a buffer object, initialize it, and associate it with
68 // the associated attribute variable in our vertex shader
69 var cBufferId = gl.createBuffer();
70 gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
71 gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
72
73 var vColor = gl.getAttribLocation(program, "vColor");
74 gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
75 gl.enableVertexAttribArray(vColor);
76
77 //var modelMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
Restricted Mode
0 0
Ln 51, Col 56
Spaces: 4
UTF-8
CRLF
JavaScript


학번 이름 - Alpha Blending
파일 | C:/Users/GVE/Desk...


File Edit Selection View Go Run Terminal Help
alphaCube.js - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > GVE > Desktop > CG > Week13 > alphaCube.js > init
43
44 // Configure WebGL
45 gl.viewport(0, 0, canvas.width, canvas.height);
46 gl.clearColor(0.9, 0.9, 0.9, 1.0);
47
48 // Enable hidden-surface removal
49 //gl.enable(gl.DEPTH_TEST);
50 gl.enable(gl.BLEND);
51 gl.blendFunc(gl.SRC_ALPHA, gl.ONE_MINUS_SRC_ALPHA);
52
53 // Load shaders and initialize attribute buffers
54 var program = initShaders(gl, "vertex-shader", "fragment-shader");
55 gl.useProgram(program);
56
57 // Load the data into the GPU
58 var bufferId = gl.createBuffer();
59 gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
60 gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
61
62 // Associate our shader variables with our data buffer
63 var vPosition = gl.getAttribLocation(program, "vPosition");
64 gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
65 gl.enableVertexAttribArray(vPosition);
66
67 // Create a buffer object, initialize it, and associate it with
68 // the associated attribute variable in our vertex shader
69 var cBufferId = gl.createBuffer();
70 gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
71 gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
72
73 var vColor = gl.getAttribLocation(program, "vColor");
74 gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
75 gl.enableVertexAttribArray(vColor);
76
77 //var modelMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
Restricted Mode
0 0
Ln 49, Col 7
Spaces: 4
UTF-8
CRLF
JavaScript


학번 이름 - Alpha Blending
파일 | C:/Users/GVE/Desk...