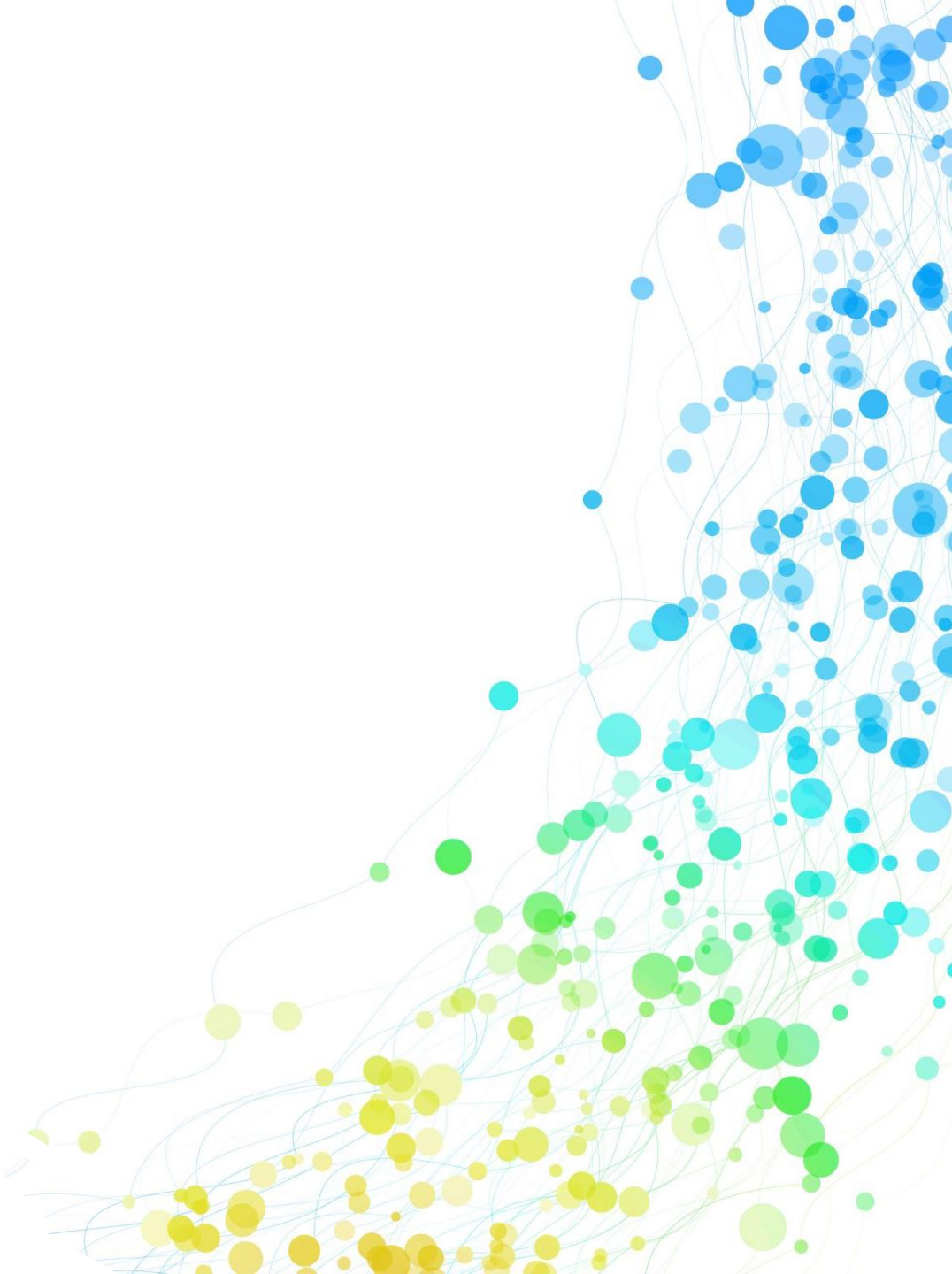
Blending

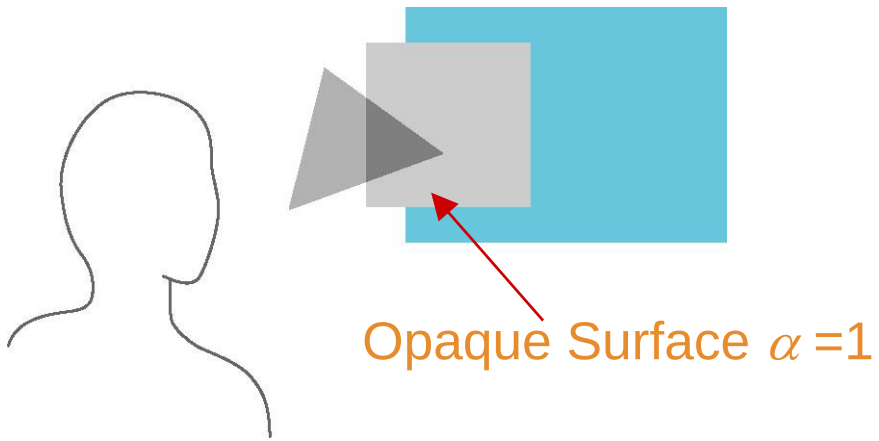
13TH WEEK, 2021



Opacity and Transparency

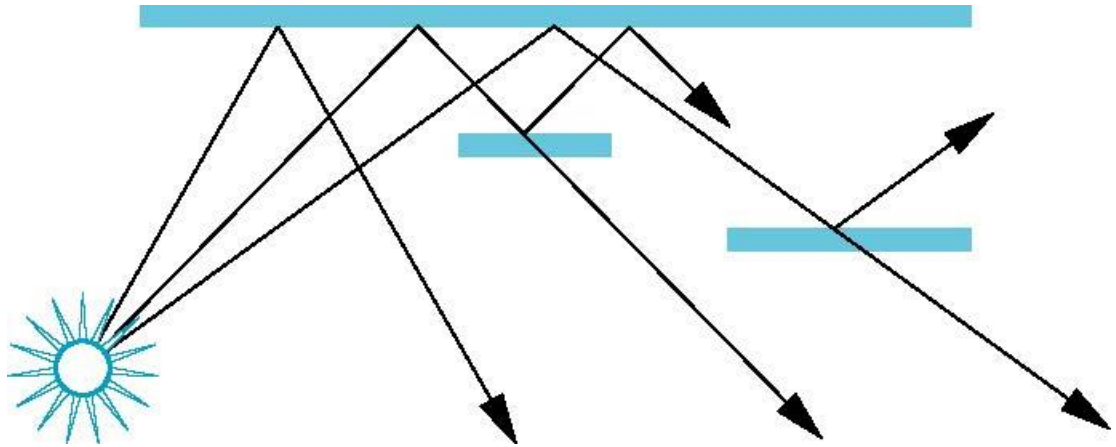
- Opaque surfaces permit no light to pass through
- Transparent surfaces permit all light to pass
- Translucent surfaces pass some light

Translucency = 1 – Opacity(α)



Physical Models

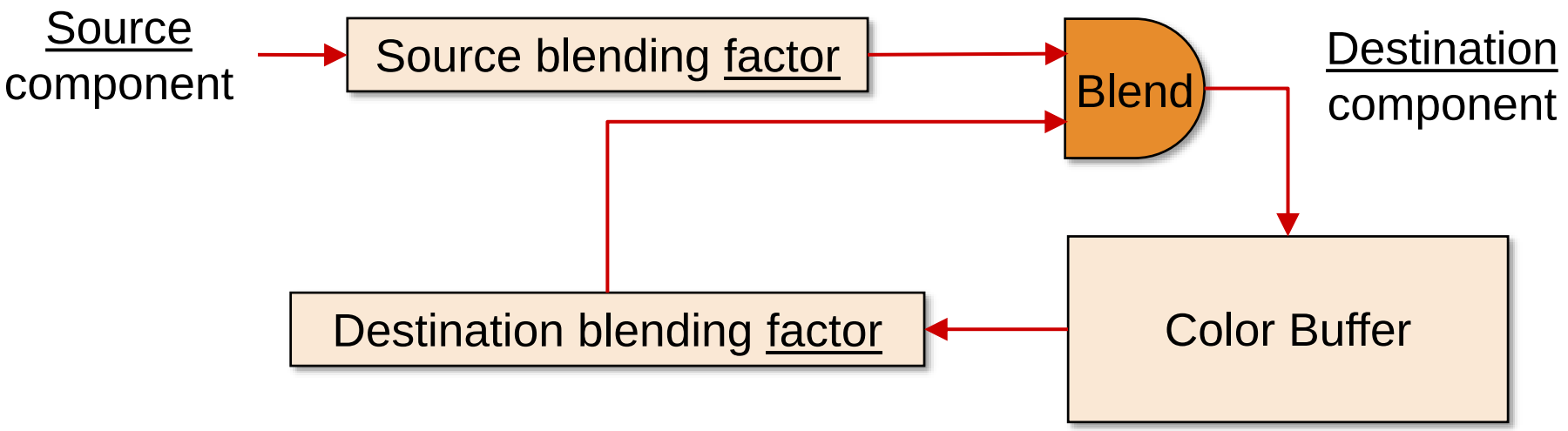
- Dealing with translucency in a physically correct manner is difficult due to
  - The complexity of the internal interactions of light and matter
  - Using a pipeline renderer



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

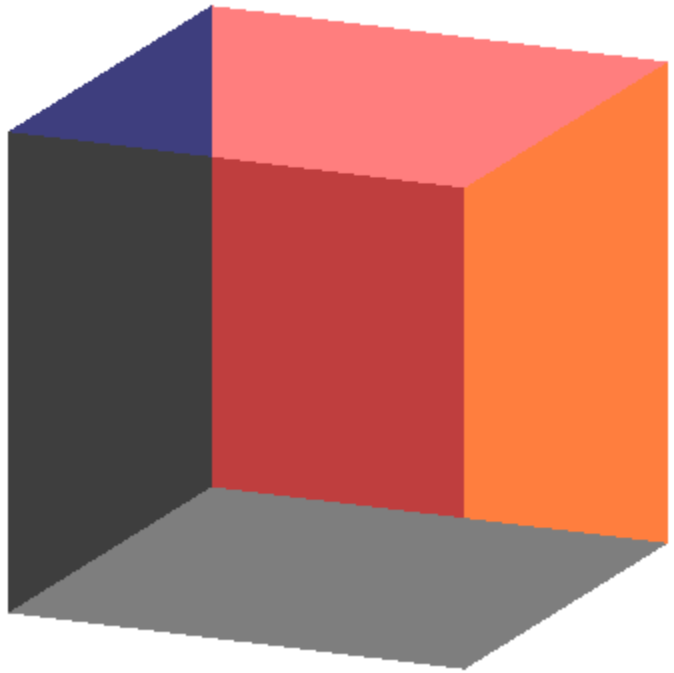
- Suppose that we start with the opaque background color $(R_0, G_0, B_0, 1)$
  - This color becomes the initial destination color
- We now want to blend in a translucent polygon with color $(R_1, G_1, B_1, \alpha_1)$
- Select `gl.SRC_ALPHA` and `gl.ONE_MINUS_SRC_ALPHA` as the source and destination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

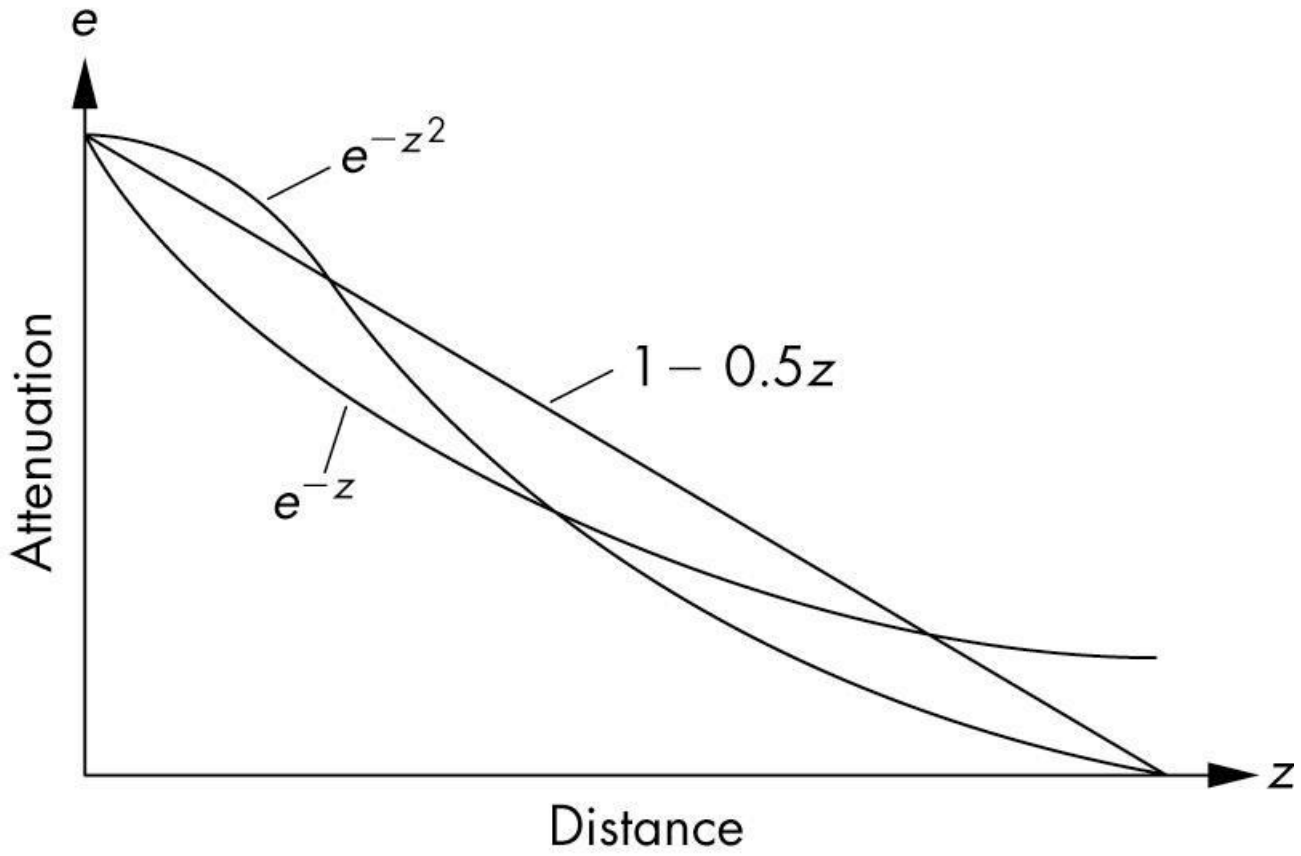
Fog

- 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

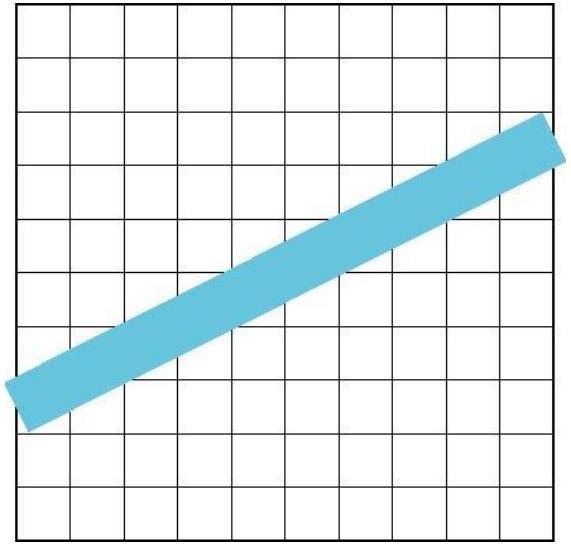


Compositing and HTML

- 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

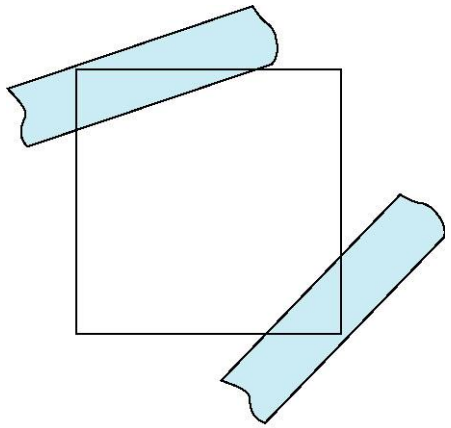
- Ideal raster line is one pixel wide
- All line segments, other than vertical and horizontal segments, partially cover pixels
- Simple algorithms color only whole pixels
- Lead to the “jaggies” or aliasing
- Similar issue for polygons

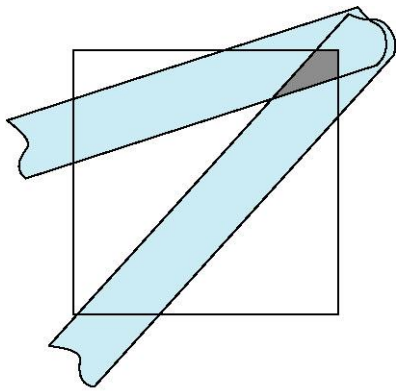


Antialiasing

- 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

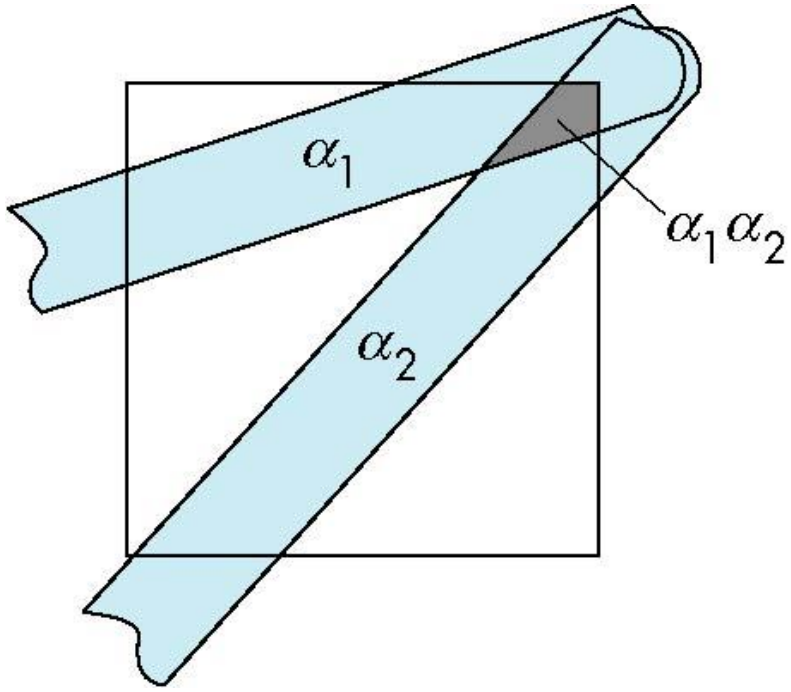




Overlap

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

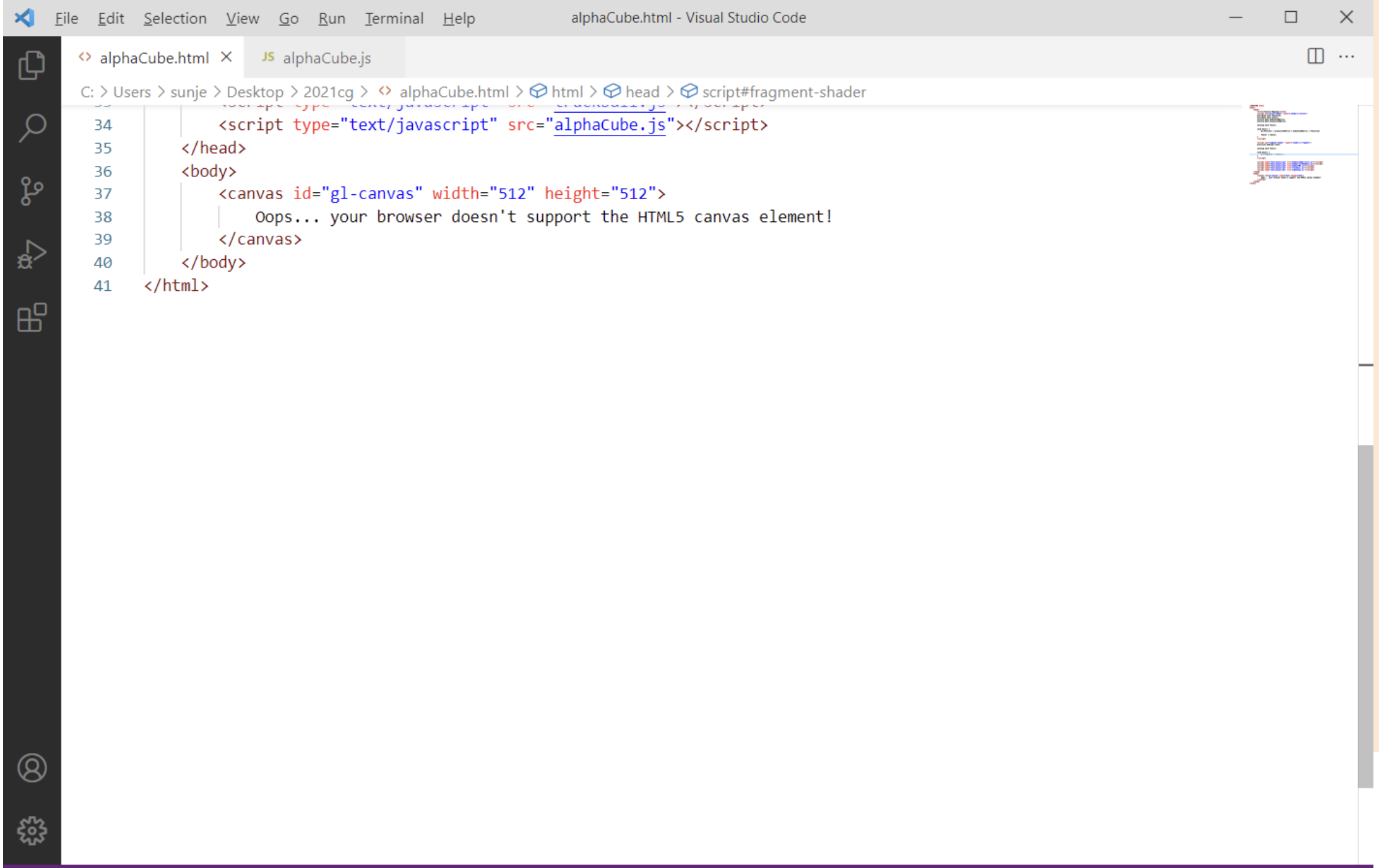
File Edit Selection View Go Run Terminal Help alphaCube.html - Visual Studio Code

alphaCube.html alphaCube.js

C: > Users > sunje > Desktop > 2021cg > alphaCube.html > html > head > script#fragment-shader

```
<!DOCTYPE html>
<html>
    <head>
        <title>Texture Mapping</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform mat4 modelViewMatrix;
        uniform mat4 projectionMatrix;

        varying vec4 fColor;

        void main() {
            gl_Position = projectionMatrix * modelViewMatrix * vPosition;

            fColor = vColor;
        }
        </script>

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

        varying vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </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>
```

⊗ 0 ⚠ 0 Ln 26, Col 35 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 > sunje > Desktop > 2021cg > alphaCube.html > html > head > script#fragment-shader
<script type="text/javascript" src="alphaCube.js"></script>
</head>
<body>
<canvas id="gl-canvas" width="512" height="512">
Oops... your browser doesn't support the HTML5 canvas element!
</canvas>
</body>
</html>
Ln 26, Col 35
Spaces: 4
UTF-8
CRLF
HTML

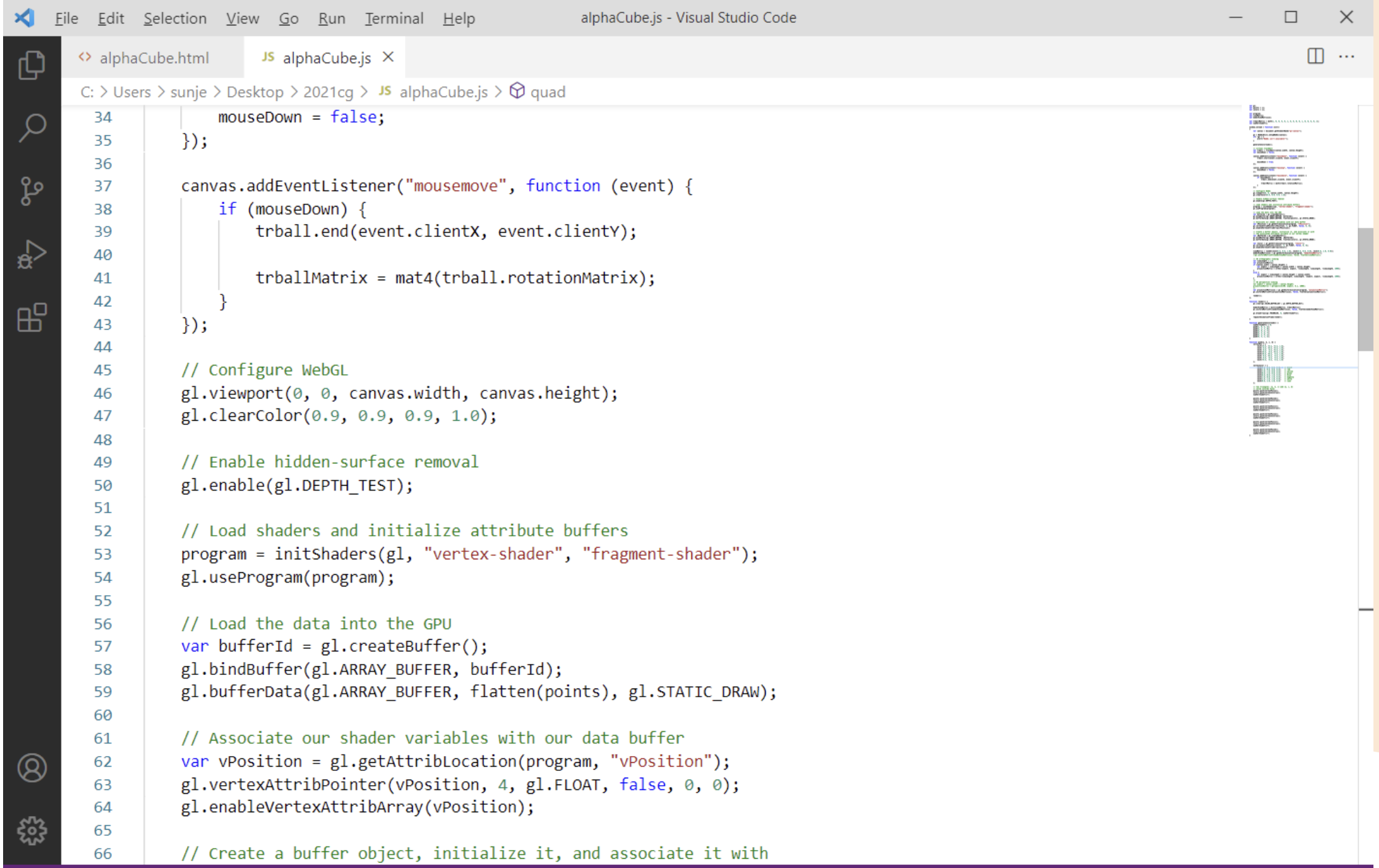
File Edit Selection View Go Run Terminal Help alphaCube.js - Visual Studio Code

alphaCube.html | alphaCube.js

C: > Users > sunje > Desktop > 2021cg > alphaCube.js > quad

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

var program;
var viewMatrix;
var modelViewMatrixLoc;

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) {
```

0 0 Ln 136, Col 34 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 > sunje > Desktop > 2021cg > alphaCube.js > quad
mouseDown = false;
});
canvas.addEventListener("mousemove", function (event) {
if (mouseDown) {
trball.end(event.clientX, event.clientY);
trballMatrix = mat4(trball.rotationMatrix);
}
});
// Configure WebGL
gl.viewport(0, 0, canvas.width, canvas.height);
gl.clearColor(0.9, 0.9, 0.9, 1.0);
// Enable hidden-surface removal
gl.enable(gl.DEPTH_TEST);
// Load shaders and initialize attribute buffers
program = initShaders(gl, "vertex-shader", "fragment-shader");
gl.useProgram(program);
// Load the data into the GPU
var bufferId = gl.createBuffer();
gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
// Associate our shader variables with our data buffer
var vPosition = gl.getAttribLocation(program, "vPosition");
gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
gl.enableVertexAttribArray(vPosition);
// Create a buffer object, initialize it, and associate it with
0 0
Ln 136, Col 34 Spaces: 4 UTF-8 CRLF JavaScript

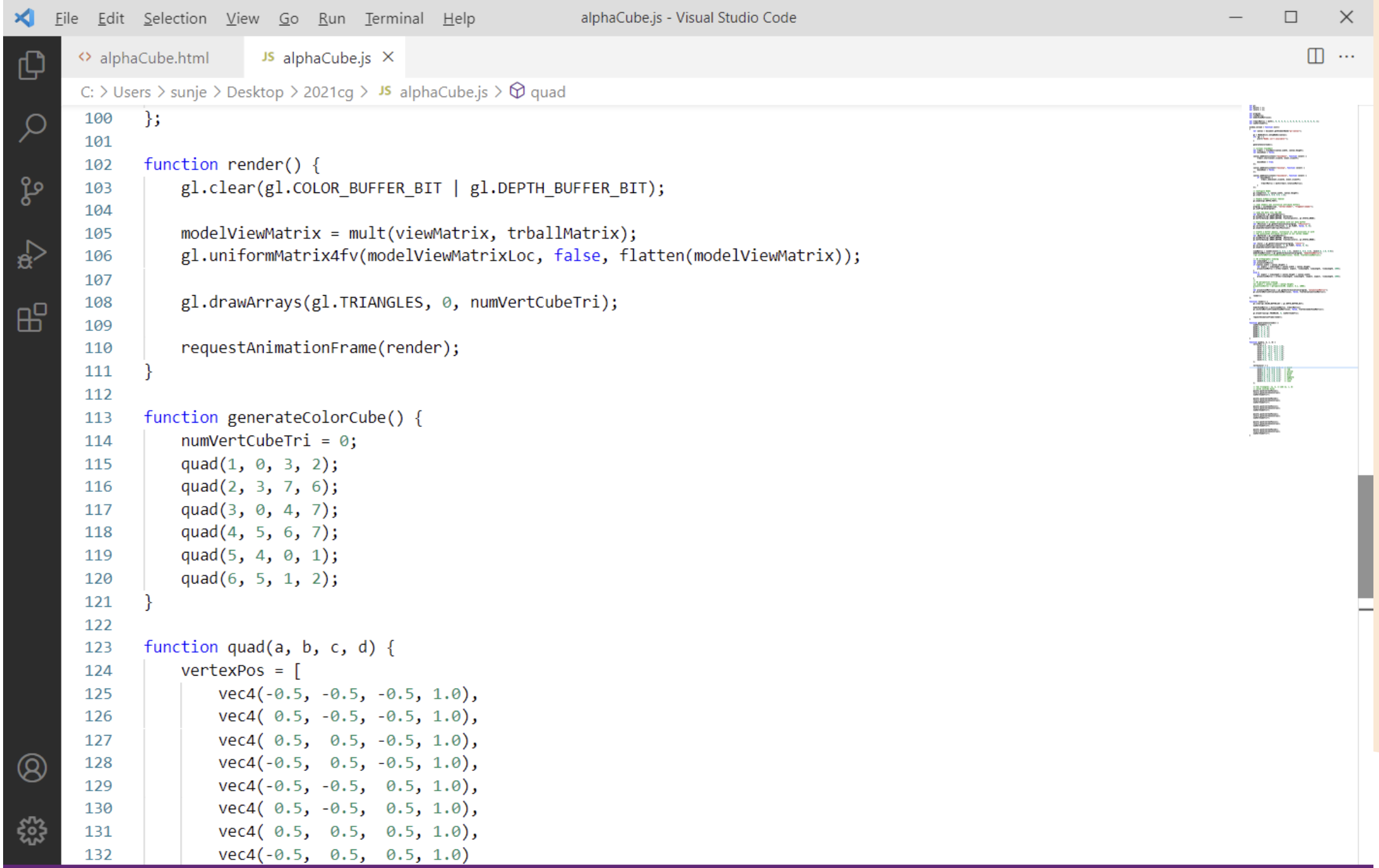
File Edit Selection View Go Run Terminal Help alphaCube.js - Visual Studio Code

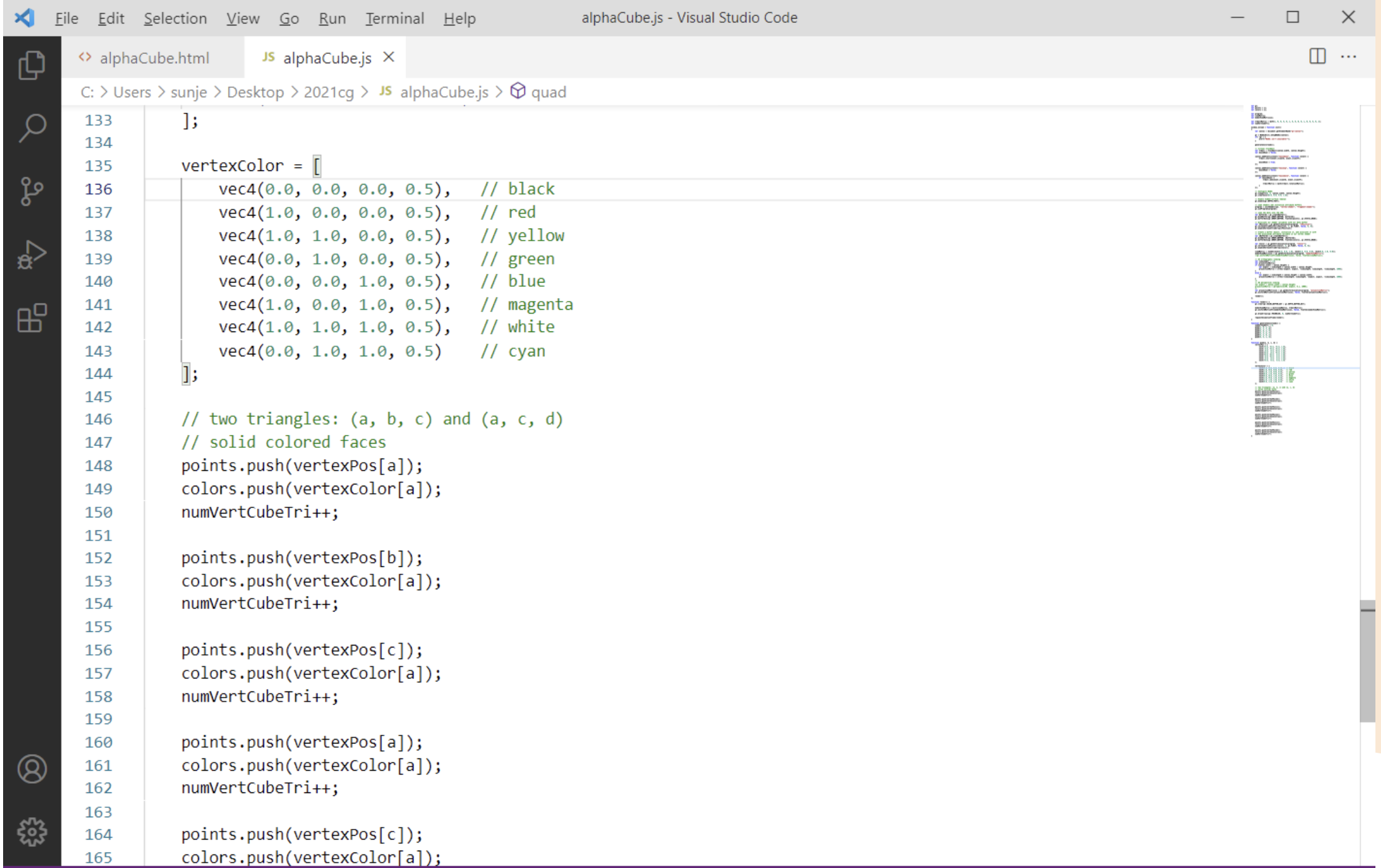
alphaCube.html | alphaCube.js

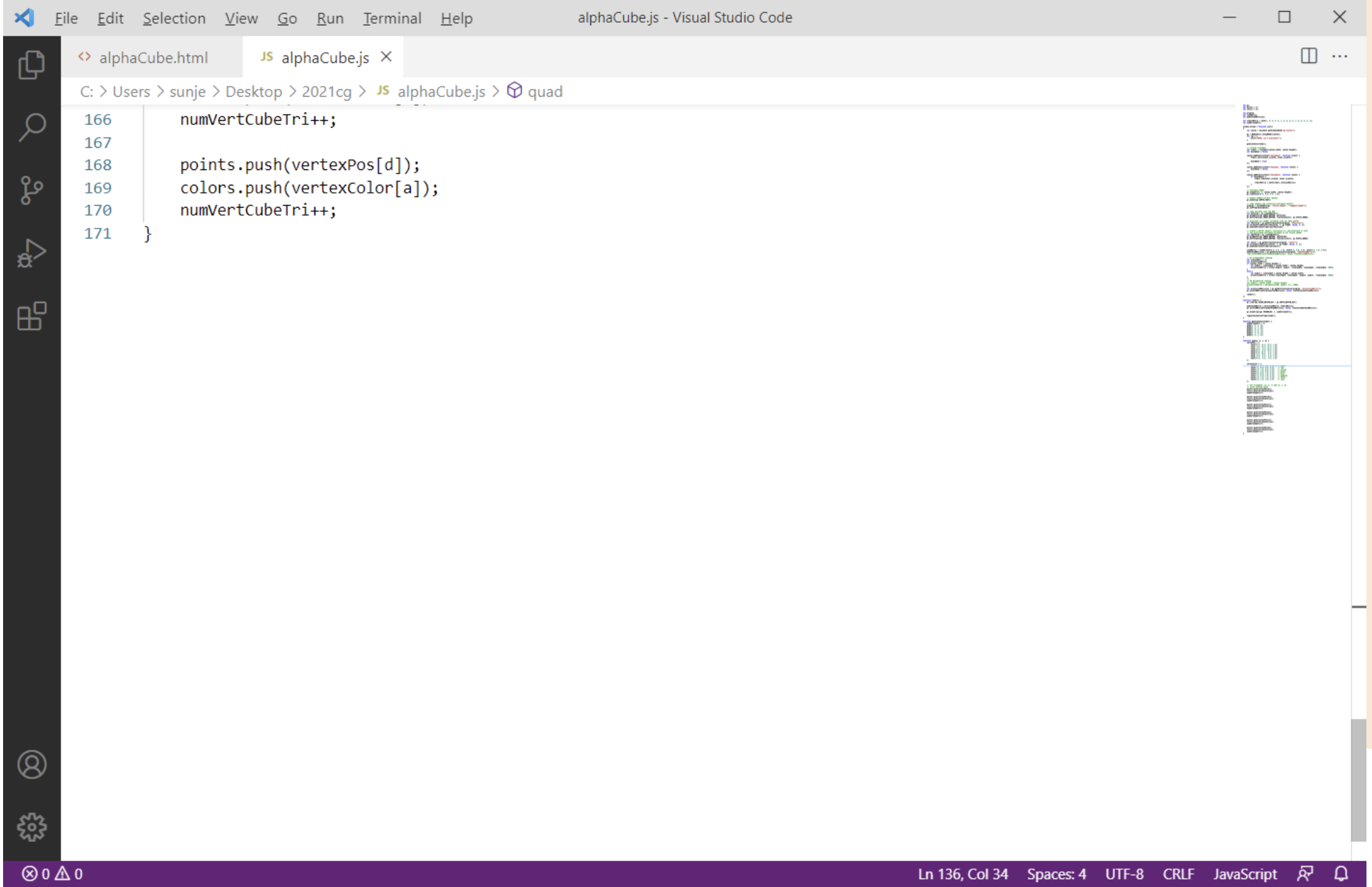
C: > Users > sunje > Desktop > 2021cg > alphaCube.js > quad

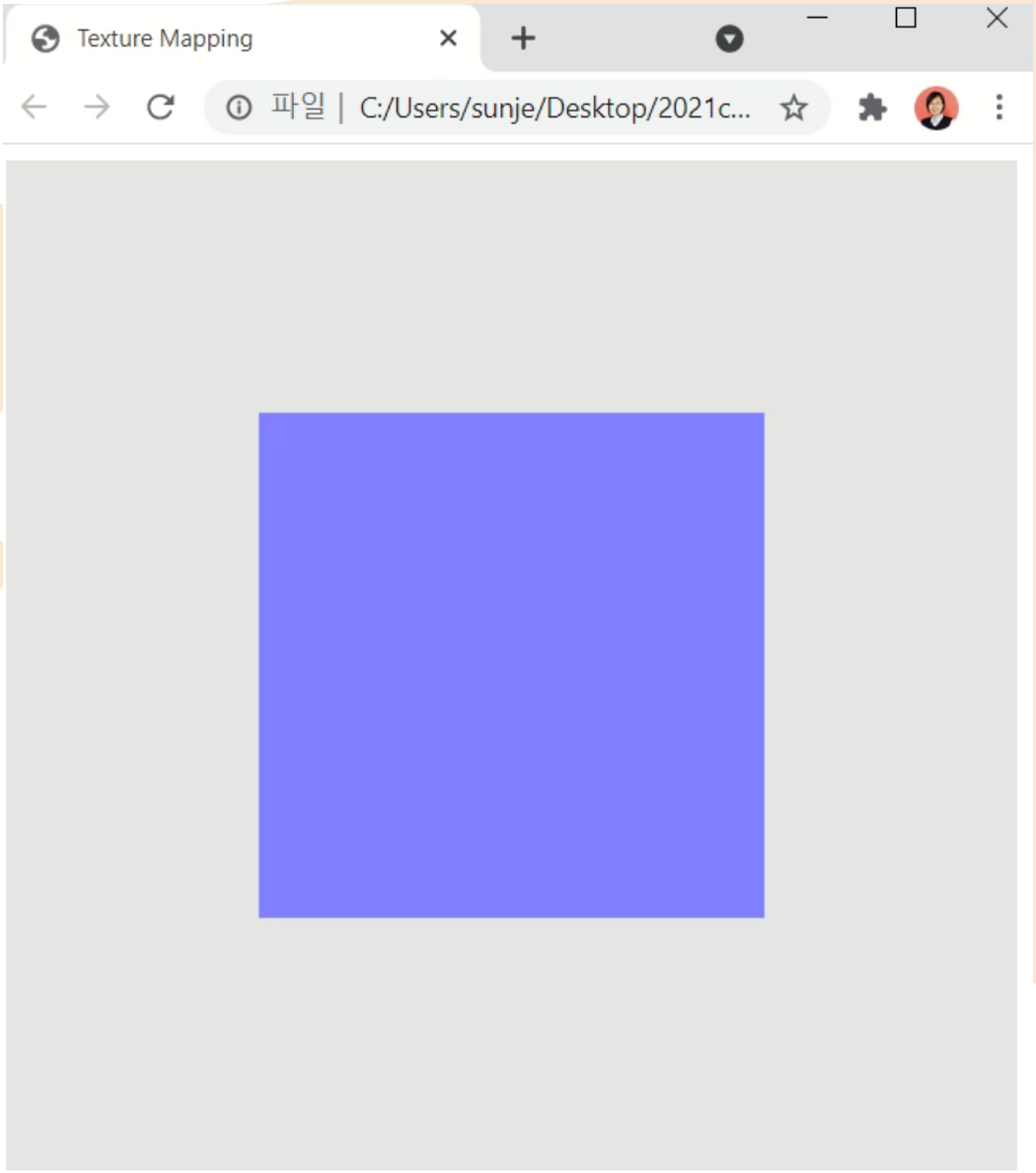
```
67    // the associated attribute variable in our vertex shader
68    var cBufferId = gl.createBuffer();
69    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
70    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
71
72    var vColor = gl.getAttribLocation(program, "vColor");
73    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
74    gl.enableVertexAttribArray(vColor);
75
76    viewMatrix = lookAt(vec3(0.0, 0.0, 1.0), vec3(0.0, 0.0, 0.0), vec3(0.0, 1.0, 0.0));
77    modelViewMatrixLoc = gl.getUniformLocation(program, "modelViewMatrix");
78    //gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(viewMatrix));
79
80    // 3D orthographic viewing
81    var viewLength = 1.0;
82    var projectionMatrix;
83    if (canvas.width > canvas.height) {
84        var aspect = viewLength * canvas.width / canvas.height;
85        projectionMatrix = ortho(-aspect, aspect, -viewLength, viewLength, -viewLength, 1000);
86    }
87    else {
88        var aspect = viewLength * canvas.height / canvas.width;
89        projectionMatrix = ortho(-viewLength, viewLength, -aspect, aspect, -viewLength, 1000);
90    }
91    /*
92    // 3D perspective viewing
93    var aspect = canvas.width / canvas.height;
94    projectionMatrix = perspective(90, aspect, 0.1, 1000);
95    */
96    var projectionMatrixLoc = gl.getUniformLocation(program, "projectionMatrix");
97    gl.uniformMatrix4fv(projectionMatrixLoc, false, flatten(projectionMatrix));
98
99    render();
```

⊗ 0 ⚠ 0 Ln 136, Col 34 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 > sunje > Desktop > 2021cg > alphaCube.js > quad
};
function render() {
gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
modelViewMatrix = mult(viewMatrix, trballMatrix);
gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelViewMatrix));
gl.drawArrays(gl.TRIANGLES, 0, numVertCubeTri);
requestAnimationFrame(render);
}
function generateColorCube() {
numVertCubeTri = 0;
quad(1, 0, 3, 2);
quad(2, 3, 7, 6);
quad(3, 0, 4, 7);
quad(4, 5, 6, 7);
quad(5, 4, 0, 1);
quad(6, 5, 1, 2);
}
function quad(a, b, c, d) {
vertexPos = [
vec4(-0.5, -0.5, -0.5, 1.0),
vec4(0.5, -0.5, -0.5, 1.0),
vec4(0.5, 0.5, -0.5, 1.0),
vec4(-0.5, 0.5, -0.5, 1.0),
vec4(-0.5, -0.5, 0.5, 1.0),
vec4(0.5, -0.5, 0.5, 1.0),
vec4(0.5, 0.5, 0.5, 1.0),
vec4(-0.5, 0.5, 0.5, 1.0)
0 0
Ln 136, Col 34 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 > sunje > Desktop > 2021cg > alphaCube.js > quad
];
vertexColor = [
vec4(0.0, 0.0, 0.0, 0.5), // black
vec4(1.0, 0.0, 0.0, 0.5), // red
vec4(1.0, 1.0, 0.0, 0.5), // yellow
vec4(0.0, 1.0, 0.0, 0.5), // green
vec4(0.0, 0.0, 1.0, 0.5), // blue
vec4(1.0, 0.0, 1.0, 0.5), // magenta
vec4(1.0, 1.0, 1.0, 0.5), // white
vec4(0.0, 1.0, 1.0, 0.5) // cyan
];
// two triangles: (a, b, c) and (a, c, d)
// solid colored faces
points.push(vertexPos[a]);
colors.push(vertexColor[a]);
numVertCubeTri++;
points.push(vertexPos[b]);
colors.push(vertexColor[a]);
numVertCubeTri++;
points.push(vertexPos[c]);
colors.push(vertexColor[a]);
numVertCubeTri++;
points.push(vertexPos[a]);
colors.push(vertexColor[a]);
numVertCubeTri++;
points.push(vertexPos[c]);
colors.push(vertexColor[a]);
0 0
Ln 136, Col 34 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 > sunje > Desktop > 2021cg > alphaCube.js > quad
numVertCubeTri++;
points.push(vertexPos[d]);
colors.push(vertexColor[a]);
numVertCubeTri++;
}
Ln 136, Col 34
Spaces: 4
UTF-8
CRLF
JavaScript


Texture Mapping
파일 | C:/Users/sunje/Desktop/2021c...

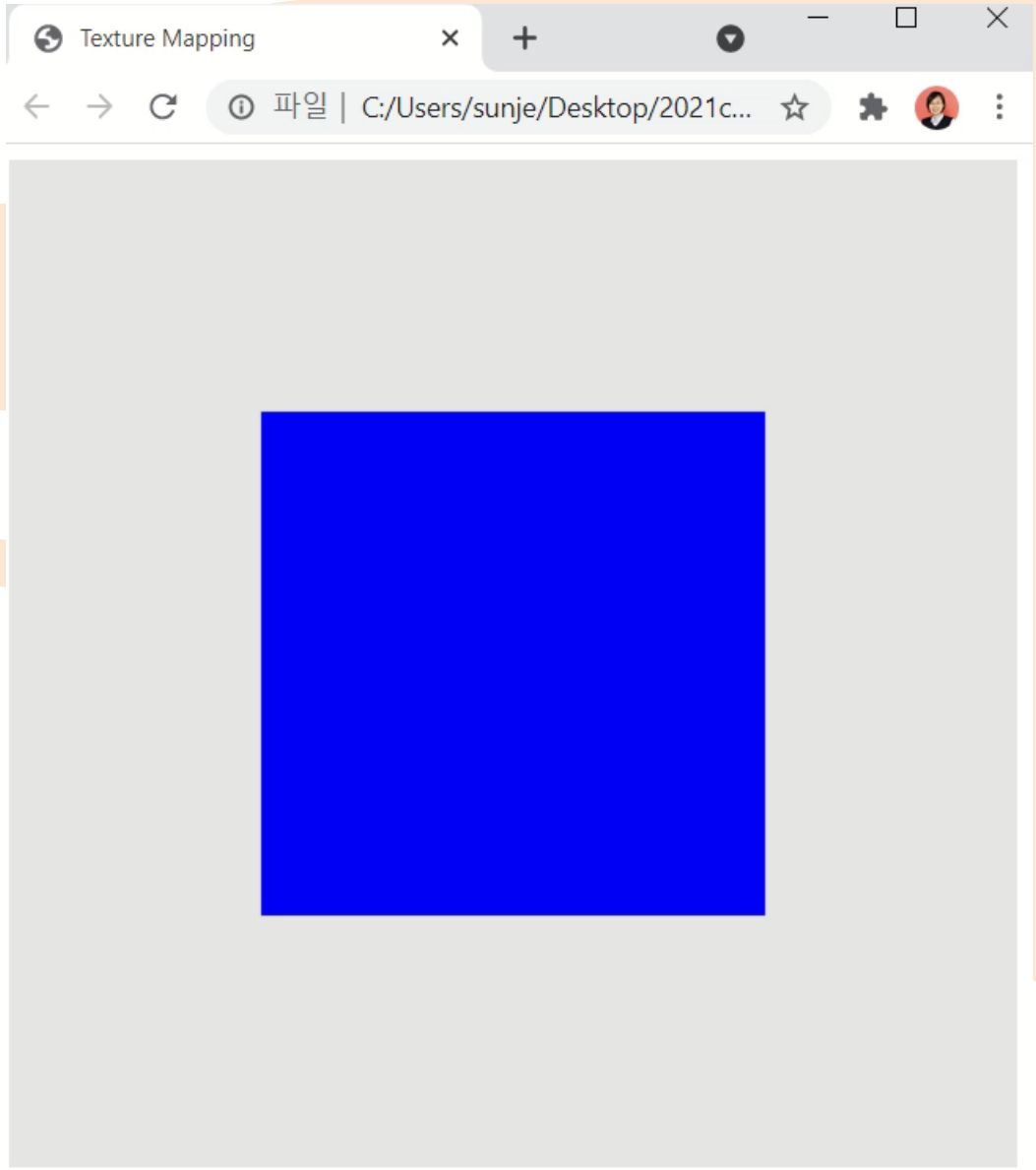
File Edit Selection View Go Run Terminal Help alphaCube.html - Visual Studio Code

alphaCube.html alphaCube.js

C: > Users > sunje > Desktop > 2021cg > alphaCube.html > html > head > script#fragment-shader

```
<!DOCTYPE html>
<html>
    <head>
        <title>Texture Mapping</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform mat4 modelViewMatrix;
        uniform mat4 projectionMatrix;

        varying vec4 fColor;

        void main() {
            gl_Position = projectionMatrix * modelViewMatrix * 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>
```

⊗ 0 ⚠ 0 Ln 27, Col 34 Spaces: 4 UTF-8 CRLF HTML


Texture Mapping
파일 | C:/Users/sunje/Desktop/2021c...

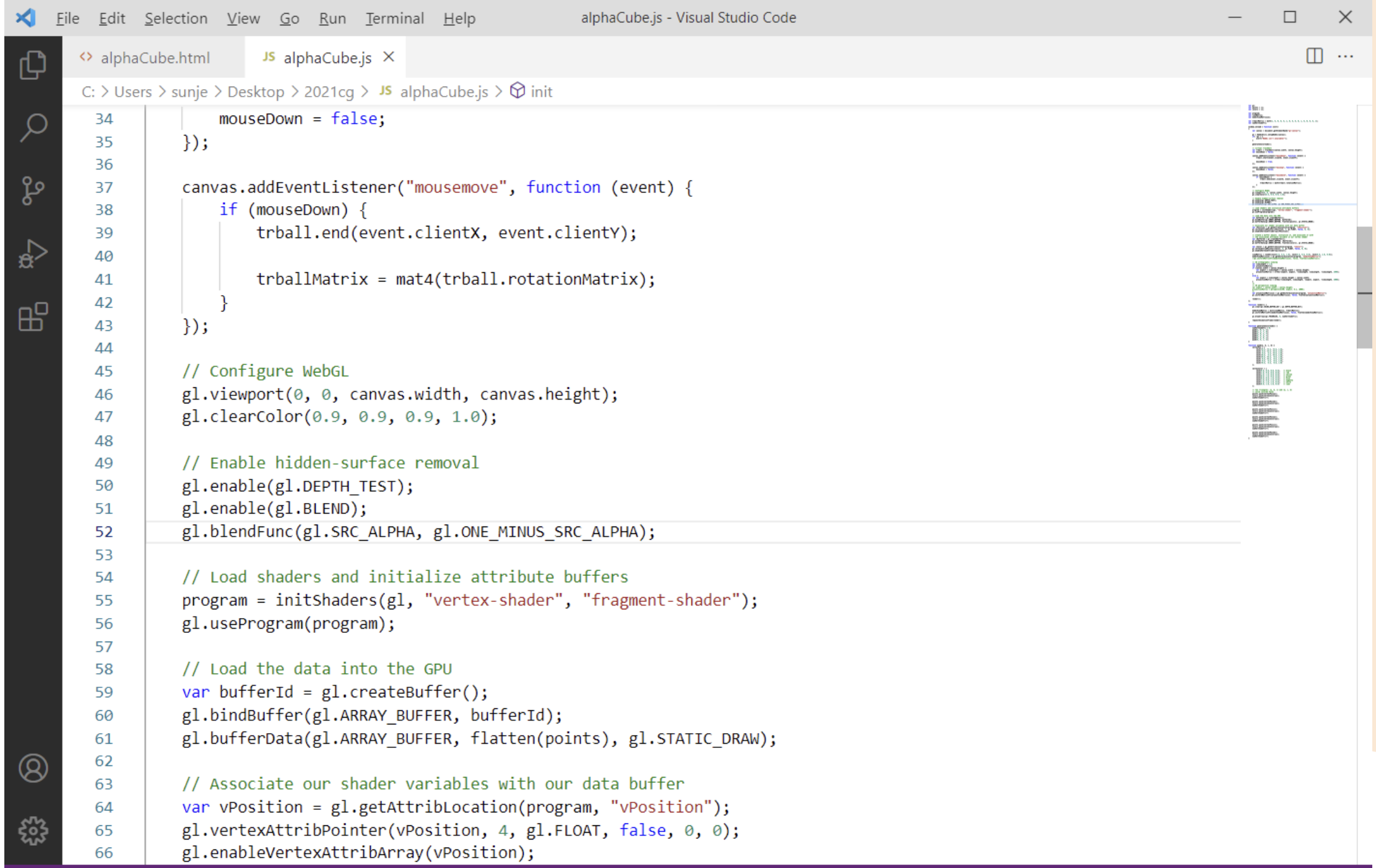
File Edit Selection View Go Run Terminal Help alphaCube.html - Visual Studio Code

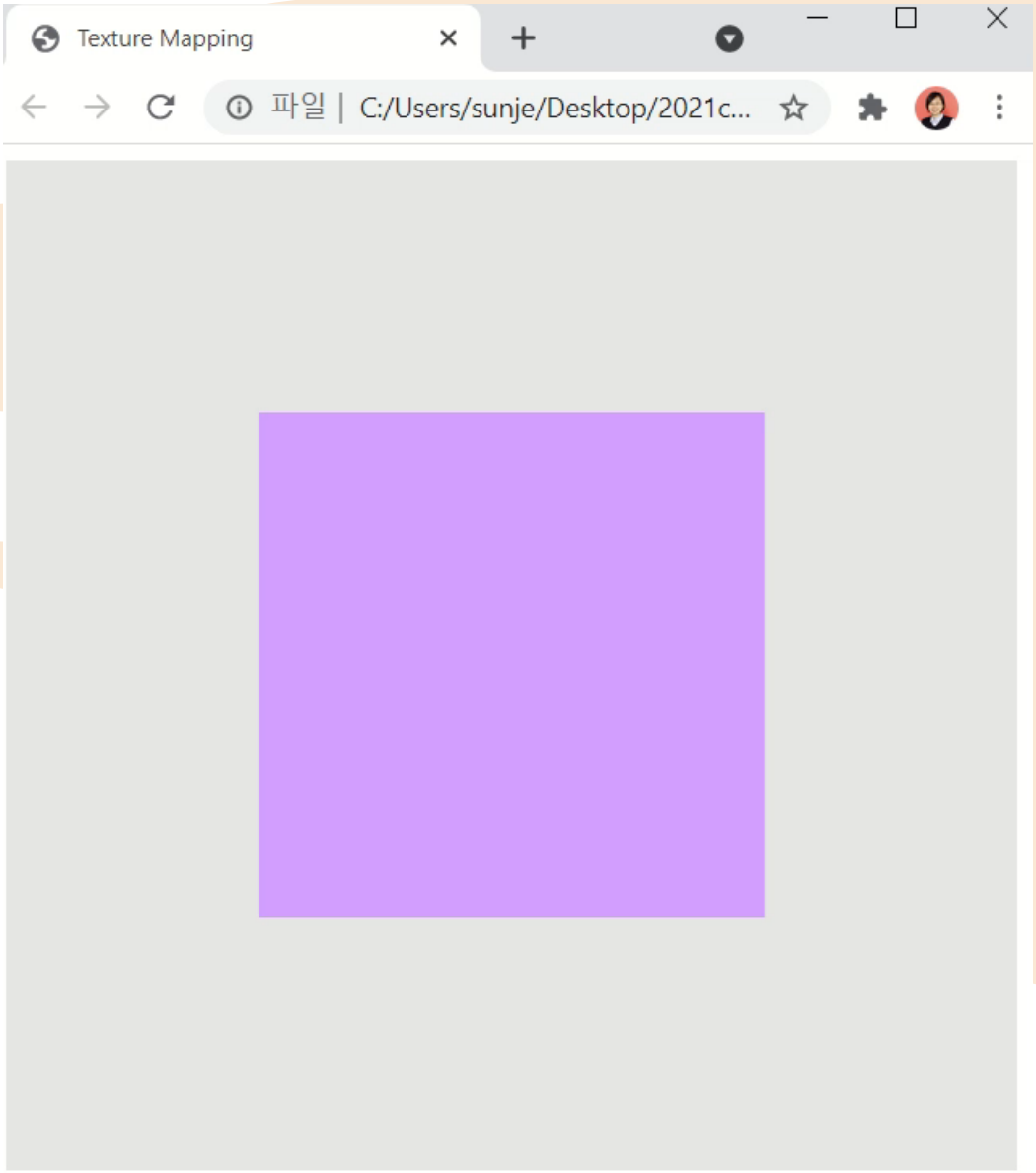
alphaCube.html alphaCube.js

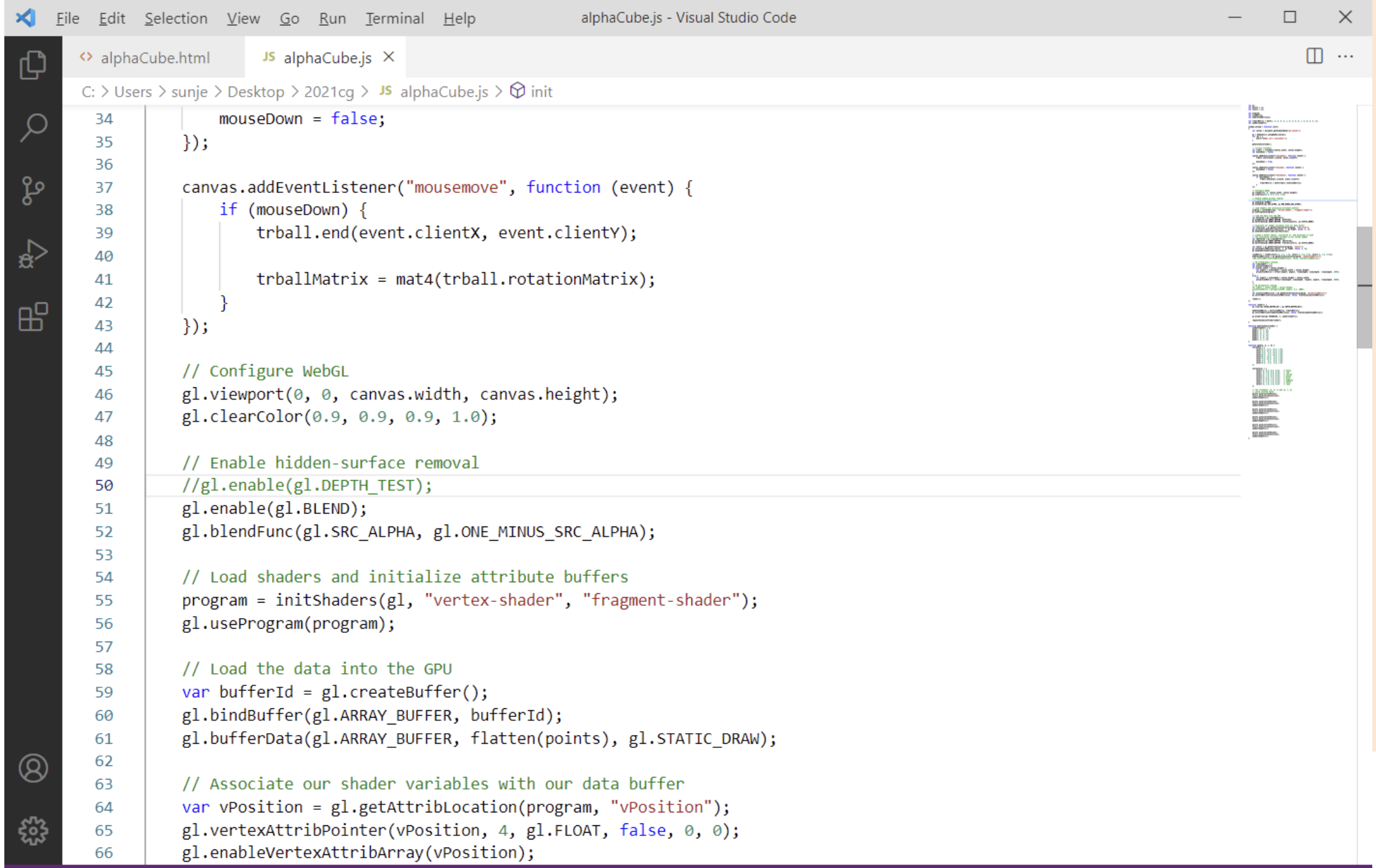
C: > Users > sunje > Desktop > 2021cg > alphaCube.html > html > head > script#fragment-shader

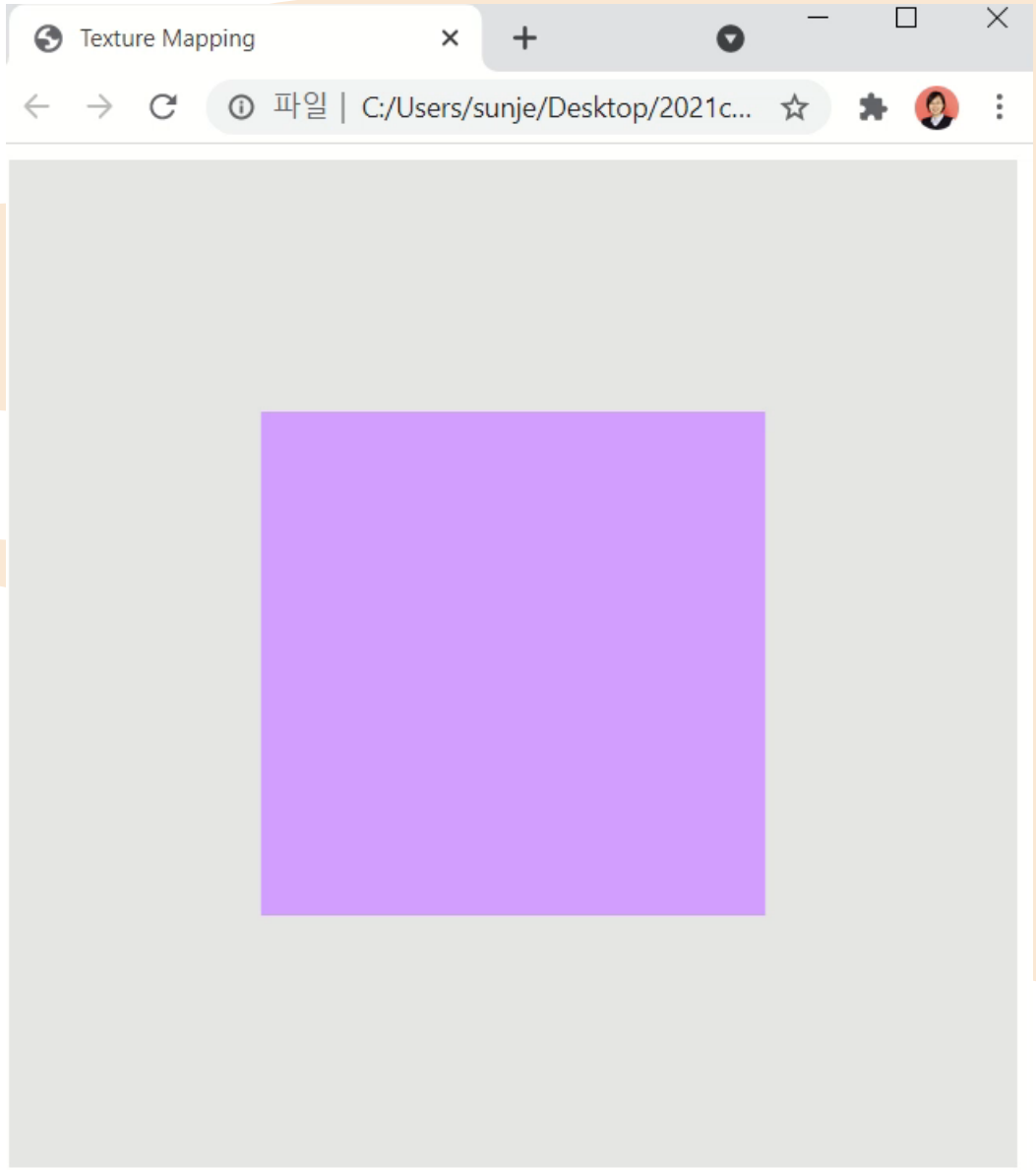
```
<!DOCTYPE html>
<html>
    <head>
        <title>Texture Mapping</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform mat4 modelViewMatrix;
        uniform mat4 projectionMatrix;

        varying vec4 fColor;

        void main() {
            gl_Position = projectionMatrix * modelViewMatrix * 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>
```

⊗ 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 > sunje > Desktop > 2021cg > alphaCube.js > init
mouseDown = false;
});
canvas.addEventListener("mousemove", function (event) {
if (mouseDown) {
trball.end(event.clientX, event.clientY);
trballMatrix = mat4(trball.rotationMatrix);
}
});
// Configure WebGL
gl.viewport(0, 0, canvas.width, canvas.height);
gl.clearColor(0.9, 0.9, 0.9, 1.0);
// Enable hidden-surface removal
gl.enable(gl.DEPTH_TEST);
gl.enable(gl.BLEND);
gl.blendFunc(gl.SRC_ALPHA, gl.ONE_MINUS_SRC_ALPHA);
// Load shaders and initialize attribute buffers
program = initShaders(gl, "vertex-shader", "fragment-shader");
gl.useProgram(program);
// Load the data into the GPU
var bufferId = gl.createBuffer();
gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
// Associate our shader variables with our data buffer
var vPosition = gl.getAttribLocation(program, "vPosition");
gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
gl.enableVertexAttribArray(vPosition);
0 0
Ln 52, Col 56 Spaces: 4 UTF-8 CRLF JavaScript


Texture Mapping
파일 | C:/Users/sunje/Desktop/2021c...


File Edit Selection View Go Run Terminal Help
alphaCube.js - Visual Studio Code
alphaCube.html
alphaCube.js
C: > Users > sunje > Desktop > 2021cg > alphaCube.js > init
34 mouseDown = false;
35 });
36
37 canvas.addEventListener("mousemove", function (event) {
38 if (mouseDown) {
39 trball.end(event.clientX, event.clientY);
40
41 trballMatrix = mat4(trball.rotationMatrix);
42 }
43 });
44
45 // Configure WebGL
46 gl.viewport(0, 0, canvas.width, canvas.height);
47 gl.clearColor(0.9, 0.9, 0.9, 1.0);
48
49 // Enable hidden-surface removal
50 //gl.enable(gl.DEPTH_TEST);
51 gl.enable(gl.BLEND);
52 gl.blendFunc(gl.SRC_ALPHA, gl.ONE_MINUS_SRC_ALPHA);
53
54 // Load shaders and initialize attribute buffers
55 program = initShaders(gl, "vertex-shader", "fragment-shader");
56 gl.useProgram(program);
57
58 // Load the data into the GPU
59 var bufferId = gl.createBuffer();
60 gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
61 gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
62
63 // Associate our shader variables with our data buffer
64 var vPosition = gl.getAttribLocation(program, "vPosition");
65 gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
66 gl.enableVertexAttribArray(vPosition);
⊗ 0 ⚠ 0
Ln 50, Col 7 Spaces: 4 UTF-8 CRLF JavaScript


Texture Mapping
파일 | C:/Users/sunje/Desktop/2021c...

수고하셨습니다