

Viewing

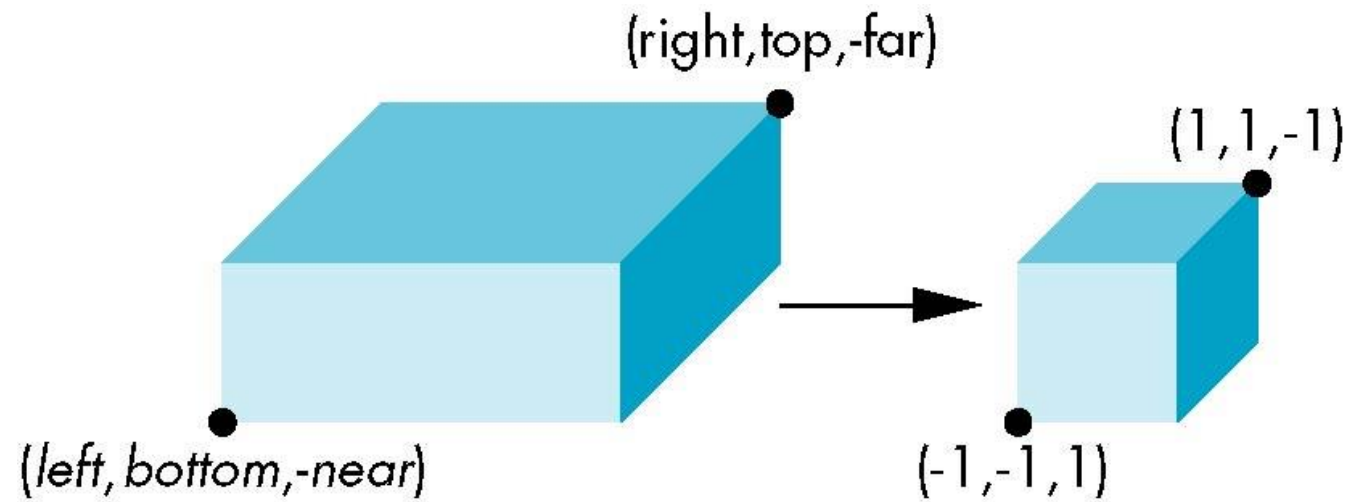
9TH WEEK, 2022



Orthogonal Normalization

`ortho(left, right, bottom, top, near, far)`

- Normalization \Rightarrow find transformation to convert specified clipping volume to default



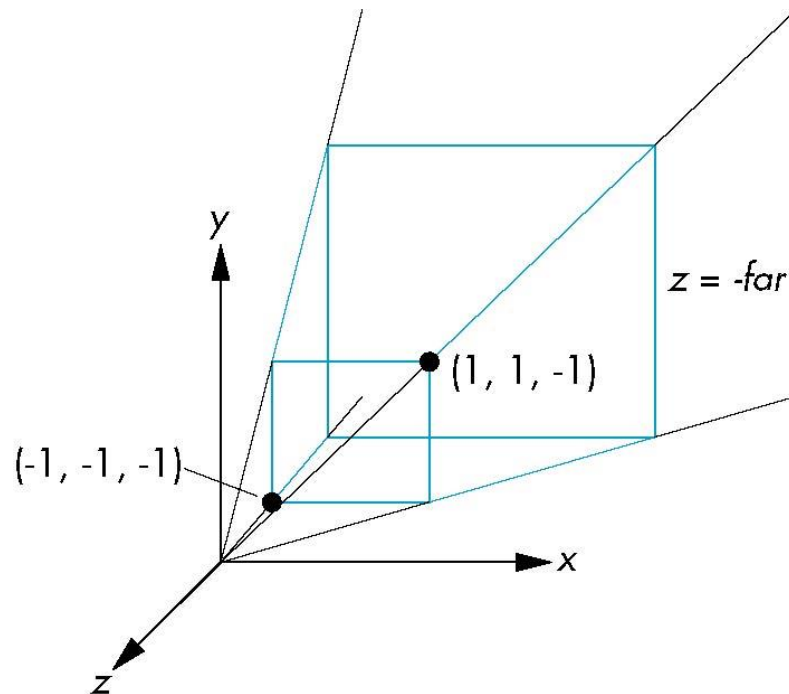
Orthogonal Matrix

- Two steps
 - Move center to origin: $T(-(left+right)/2, -(bottom+top)/2, (near+far)/2))$
 - Scale to have sides of length 2: $S(2/(left-right), 2/(top-bottom), 2/(near-far))$

$$\mathbf{P} = \mathbf{ST} = \begin{bmatrix} \frac{2}{right-left} & 0 & 0 & -\frac{right+left}{right-left} \\ 0 & \frac{2}{top-bottom} & 0 & -\frac{top+bottom}{top-bottom} \\ 0 & 0 & \frac{2}{near-far} & \frac{far+near}{far-near} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Simple Perspective

- Consider a simple perspective with the COP at the origin, the near clipping plane at $z = -1$, and a 90-degree field of view determined by the planes $x = \pm z$, $y = \pm z$



Perspective Matrices

- Simple projection matrix in homogeneous coordinates

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

- Note that this matrix is independent of the far clipping plane

Generalization

$$\mathbf{N} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \alpha & \beta \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

- after perspective division, the point $(x, y, z, 1)$ goes to

$$x'' = x/z$$

$$y'' = y/z$$

$$z'' = -(\alpha + \beta/z)$$

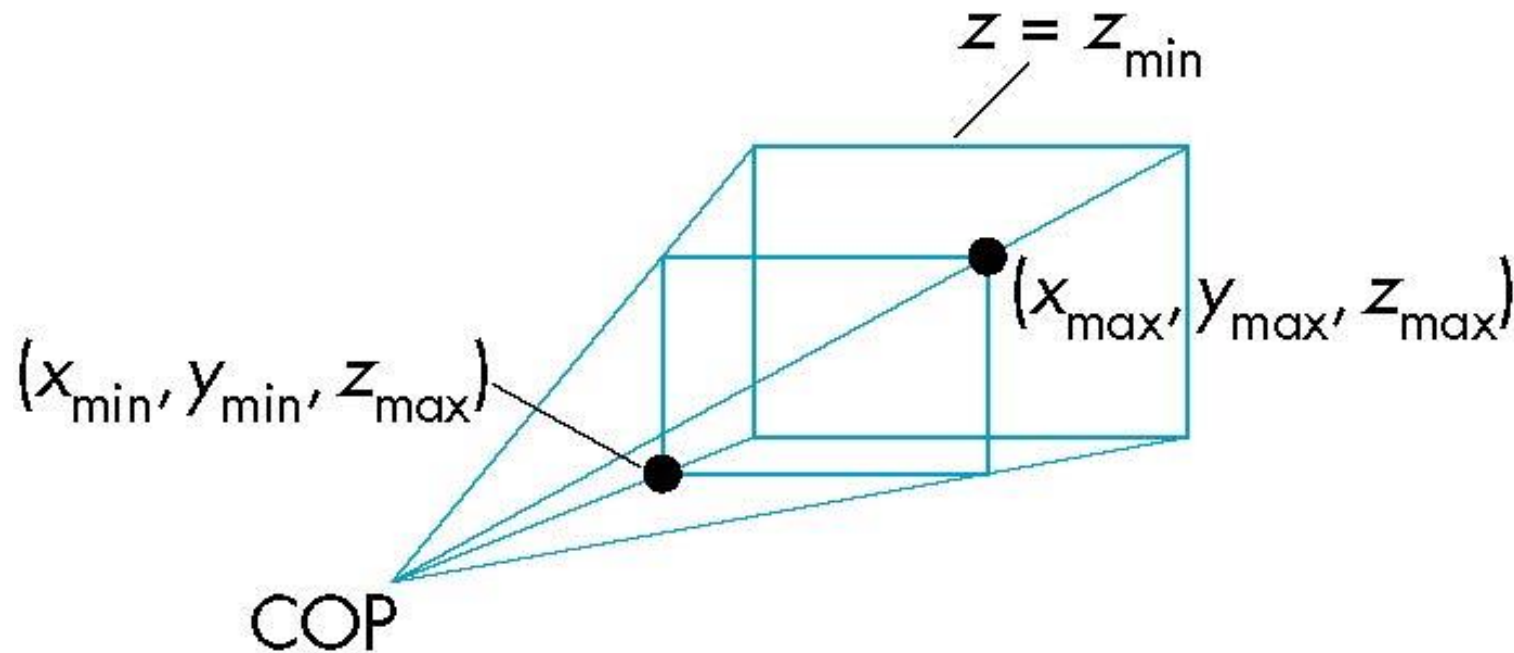
- which projects orthogonally to the desired point regardless of α and β

Picking α and β

- If we pick:
$$\alpha = \frac{\text{near} + \text{far}}{\text{far} - \text{near}}$$
$$\beta = \frac{2\text{near} * \text{far}}{\text{near} - \text{far}}$$
 - the near plane is mapped to $z = -1$
 - the far plane is mapped to $z = 1$
 - and the sides are mapped to $x = \pm 1, y = \pm 1$
- Hence the new clipping volume is the default clipping volume

WebGL Perspective

- **gl.frustum** allows for an unsymmetric viewing frustum (although **gl.perspective** does not)



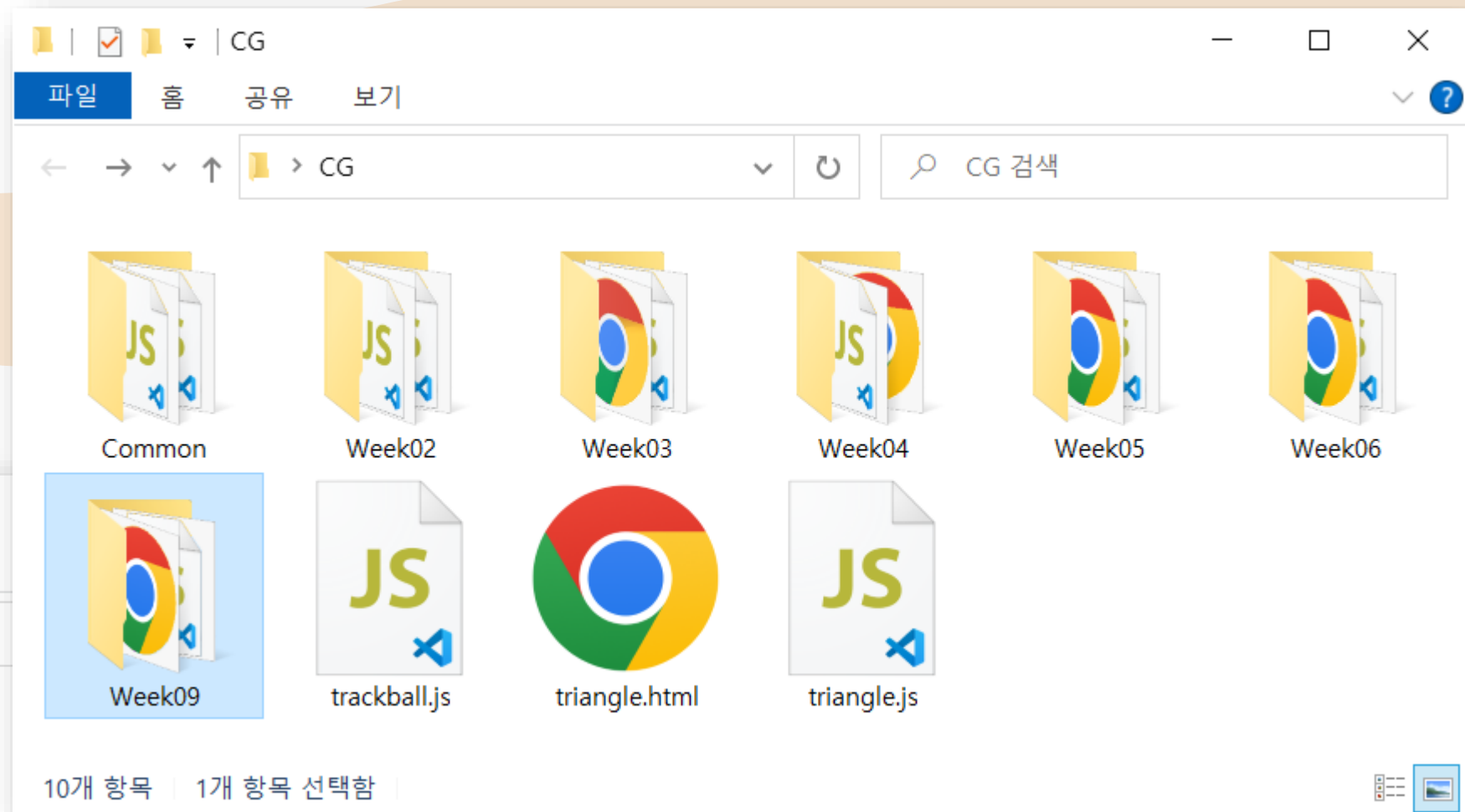
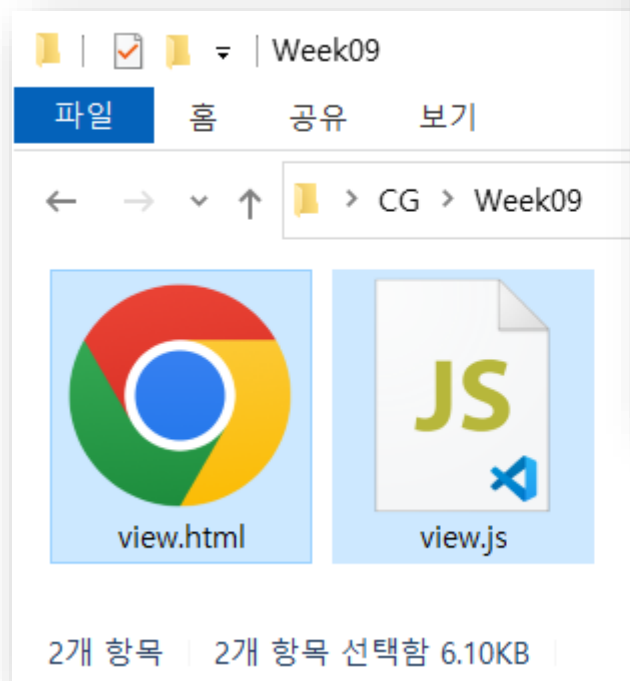
Perspective Matrices

- frustum

$$\mathbf{P} = \begin{bmatrix} \frac{2 * near}{right - left} & 0 & \frac{right - left}{right - left} & 0 \\ 0 & \frac{2 * near}{top - bottom} & \frac{top + bottom}{top - bottom} & 0 \\ 0 & 0 & -\frac{far + near}{far - near} & -\frac{2 * far * near}{far - near} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

- perspective

$$\mathbf{P} = \begin{bmatrix} \frac{near}{right} & 0 & 0 & 0 \\ 0 & \frac{near}{top} & 0 & 0 \\ 0 & 0 & -\frac{far + near}{far - near} & -\frac{2 * far * near}{far - near} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$



```

<> view.html x JS view.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > <> view.html > ...
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>학번 이름 - Viewing</title>
5      <script id="vertex-shader" type="x-shader/x-vertex">
6        attribute vec4 vPosition;
7        attribute vec4 vColor;
8        uniform mat4 modelViewMatrix;
9        uniform mat4 projectionMatrix;
10       varying vec4 fColor;
11
12       void main()
13       {
14         gl_Position = projectionMatrix * modelViewMatrix * vPosition;
15         fColor = vColor;
16       }
17     </script>
18
19     <script id="fragment-shader" type="x-shader/x-fragment">
20       precision mediump float;
21       varying vec4 fColor;
22
23       void main() {
24         gl_FragColor = fColor;
25       }
26     </script>
27
28     <script type="text/javascript" src="../Common/webgl-utils.js"></script>
29     <script type="text/javascript" src="../Common/initShaders.js"></script>
30     <script type="text/javascript" src="../Common/MV.js"></script>
31     <script type="text/javascript" src="../trackball.js"></script>
32     <script type="text/javascript" src="view.js"></script>
33   </head>
34   <body>
35     <div style="width:512px; text-align:center;">

```

```

12 void main()
13 {
14     gl_Position = projectionMatrix * modelViewMatrix * vPosition;
15     fColor = vColor;
16 }
17 </script>
18
19 <script id="fragment-shader" type="x-shader/x-fragment">
20 precision mediump float;
21 varying vec4 fColor;
22
23 void main() {
24     gl_FragColor = fColor;
25 }
26 </script>
27
28 <script type="text/javascript" src="../../Common/webgl-utils.js"></script>
29 <script type="text/javascript" src="../../Common/initShaders.js"></script>
30 <script type="text/javascript" src="../../Common/MV.js"></script>
31 <script type="text/javascript" src="../../trackball.js"></script>
32 <script type="text/javascript" src="view.js"></script>
33 </head>
34 <body>
35     <div style="width:512px; text-align:center;">
36         <button id="left">◀</button>
37         <button id="up">▲</button>
38         <button id="right">▶</button><br>
39         <button id="down">▼</button>
40     </div>
41     <canvas id="gl-canvas" width="512" height="512">
42         Oops... your browser doesn't support the HTML5 canvas element!
43     </canvas><br>
44 </body>
45 </html>

```

```

<> view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > ...
1  var gl;
2  var points = [];
3  var colors = [];
4
5  var modelViewMatrix, projectionMatrix;
6  var modelViewMatrixLoc, projectionMatrixLoc;
7  var eye = vec3(0.0, 0.0, 1.0);
8  var at = vec3(0.0, 0.0, 0.0);
9  var up = vec3(0.0, 1.0, 0.0);
10
11 var trballMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
12
13 window.onload = function init()
14 {
15     var canvas = document.getElementById("gl-canvas");
16
17     gl = WebGLUtils.setupWebGL(canvas);
18     if( !gl ) {
19         alert("WebGL isn't available!");
20     }
21
22     generateColorCube();
23
24     // virtual trackball
25     var trball = trackball(canvas.width, canvas.height);
26     var mouseDown = false;
27
28     canvas.addEventListener("mousedown", function(event) {
29         trball.start(event.clientX, event.clientY);
30
31         mouseDown = true;
32     });
33
34     canvas.addEventListener("mouseup", function(event) {
35         mouseDown = false;

```

```

view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > ...

33
34 canvas.addEventListener("mouseup", function(event) {
35     mouseDown = false;
36 });
37
38 canvas.addEventListener("mousemove", function(event) {
39     if (mouseDown) {
40         trball.end(event.clientX, event.clientY);
41
42         trballMatrix = mat4(trball.rotationMatrix);
43     }
44 });
45
46 // Configure WebGL
47 gl.viewport(0, 0, canvas.width, canvas.height);
48 gl.clearColor(0.9, 0.9, 0.9, 1.0);
49
50 // Enable hidden-surface removal
51 gl.enable(gl.DEPTH_TEST);
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

```



```
<> view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > ...

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 modelViewMatrix = lookAt(eye, at, up);
78 modelViewMatrixLoc = gl.getUniformLocation(program, "modelViewMatrix");
79 gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelViewMatrix));
80
81 projectionMatrix = ortho(-1, 1, -1, 1, -1, 1);
82 projectionMatrixLoc = gl.getUniformLocation(program, "projectionMatrix");
83 gl.uniformMatrix4fv(projectionMatrixLoc, false, flatten(projectionMatrix));
84
85 // Event listeners for buttons
86 document.getElementById("left").onclick = function () {
87
88 };
89 document.getElementById("right").onclick = function () {
90
91 };
92 document.getElementById("up").onclick = function () {
93
94 };
95 document.getElementById("down").onclick = function () {
96
97 };
98
99 render();
100 };
101
```

```
<> view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > ...
101
102 function render() {
103     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
104
105     //var modelView = mult(modelViewMatrix, trballMatrix);
106     //gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelView));
107
108     gl.drawArrays(gl.TRIANGLES, 0, points.length);
109
110     requestAnimationFrame(render);
111 }
112
113 function generateColorCube() {
114     quad(1, 0, 3, 2);
115     quad(2, 3, 7, 6);
116     quad(3, 0, 4, 7);
117     quad(4, 5, 6, 7);
118     quad(5, 4, 0, 1);
119     quad(6, 5, 1, 2);
120 }
121
122 const vertexPos = [
123     vec4(-0.5, -0.5, -0.5, 1.0),
124     vec4( 0.5, -0.5, -0.5, 1.0),
125     vec4( 0.5,  0.5, -0.5, 1.0),
126     vec4(-0.5,  0.5, -0.5, 1.0),
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 ];
132
133 const vertexColor = [
134     vec4(0.0, 0.0, 0.0, 1.0), // black
135     vec4(1.0, 0.0, 0.0, 1.0), // red
```


view.html

JS view.js

X

C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > ...

```

125     vec4( 0.5,  0.5, -0.5, 1.0),
126     vec4(-0.5,  0.5, -0.5, 1.0),
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 ];
132
133 const vertexColor = [
134     vec4(0.0, 0.0, 0.0, 1.0), // black
135     vec4(1.0, 0.0, 0.0, 1.0), // red
136     vec4(1.0, 1.0, 0.0, 1.0), // yellow
137     vec4(0.0, 1.0, 0.0, 1.0), // green
138     vec4(0.0, 0.0, 1.0, 1.0), // blue
139     vec4(1.0, 0.0, 1.0, 1.0), // magenta
140     vec4(1.0, 1.0, 1.0, 1.0), // white
141     vec4(0.0, 1.0, 1.0, 1.0)  // cyan
142 ];
143
144 function quad(a, b, c, d) {
145     points.push(vertexPos[a]);
146     colors.push(vertexColor[a]);
147     points.push(vertexPos[b]);
148     colors.push(vertexColor[b]);
149     points.push(vertexPos[c]);
150     colors.push(vertexColor[c]);
151     points.push(vertexPos[a]);
152     colors.push(vertexColor[a]);
153     points.push(vertexPos[c]);
154     colors.push(vertexColor[c]);
155     points.push(vertexPos[d]);
156     colors.push(vertexColor[d]);
157 }
158

```

Line 10

```

10  // ...
11  // ...
12  // ...
13  // ...
14  // ...
15  // ...
16  // ...
17  // ...
18  // ...
19  // ...
20  // ...
21  // ...
22  // ...
23  // ...
24  // ...
25  // ...
26  // ...
27  // ...
28  // ...
29  // ...
30  // ...
31  // ...
32  // ...
33  // ...
34  // ...
35  // ...
36  // ...
37  // ...
38  // ...
39  // ...
40  // ...
41  // ...
42  // ...
43  // ...
44  // ...
45  // ...
46  // ...
47  // ...
48  // ...
49  // ...
50  // ...
51  // ...
52  // ...
53  // ...
54  // ...
55  // ...
56  // ...
57  // ...
58  // ...
59  // ...
60  // ...
61  // ...
62  // ...
63  // ...
64  // ...
65  // ...
66  // ...
67  // ...
68  // ...
69  // ...
70  // ...
71  // ...
72  // ...
73  // ...
74  // ...
75  // ...
76  // ...
77  // ...
78  // ...
79  // ...
80  // ...
81  // ...
82  // ...
83  // ...
84  // ...
85  // ...
86  // ...
87  // ...
88  // ...
89  // ...
90  // ...
91  // ...
92  // ...
93  // ...
94  // ...
95  // ...
96  // ...
97  // ...
98  // ...
99  // ...
100 // ...

```

17

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

```

<> view.html JS MV.js X JS view.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Common > JS MV.js > ...
454 //-----
455 //
456 // Projection Matrix Generators
457 //
458
459 function ortho( left, right, bottom, top, near, far )
460 {
461     if ( left == right ) { throw "ortho(): left and right are equal"; }
462     if ( bottom == top ) { throw "ortho(): bottom and top are equal"; }
463     if ( near == far ) { throw "ortho(): near and far are equal"; }
464
465     var w = right - left;
466     var h = top - bottom;
467     var d = far - near;
468
469     var result = mat4();
470     result[0][0] = 2.0 / w;
471     result[1][1] = 2.0 / h;
472     result[2][2] = -2.0 / d;
473     result[0][3] = -(left + right) / w;
474     result[1][3] = -(top + bottom) / h;
475     result[2][3] = -(near + far) / d;
476
477     return result;
478 }
479
480 //-----
481
482 function perspective( fovy, aspect, near, far )
483 {
484     var f = 1.0 / Math.tan( radians(fovy) / 2 );
485     var d = far - near;
486
487     var result = mat4();
488     result[0][0] = f / aspect;

```

view.html

JS MV.js

JS view.js

C: > Users > Sun-Jeong Kim > Desktop > CG > Common > JS MV.js > ...

479

480 //-----

481

482 function perspective(fovy, aspect, near, far)

483 {

484 var f = 1.0 / Math.tan(radians(fovy) / 2);

485 var d = far - near;

486

487 var result = mat4();

488 result[0][0] = f / aspect;

489 result[1][1] = f;

490 result[2][2] = -(near + far) / d;

491 result[2][3] = -2 * near * far / d;

492 result[3][2] = -1;

493 result[3][3] = 0.0;

494

495 return result;

496 }

497

498 //-----

499 //

500 // Matrix Functions

501 //

502

503 function transpose(m)

504 {

505 if (!m.matrix) {

506 return "transpose(): trying to transpose a non-matrix";

507 }

508

509 var result = [];

510 for (var i = 0; i < m.length; ++i) {

511 result.push([]);

512 for (var j = 0; j < m[i].length; ++j) {

513 result[i].push(m[j][i]);

view.html

JS MV.js

JS view.js

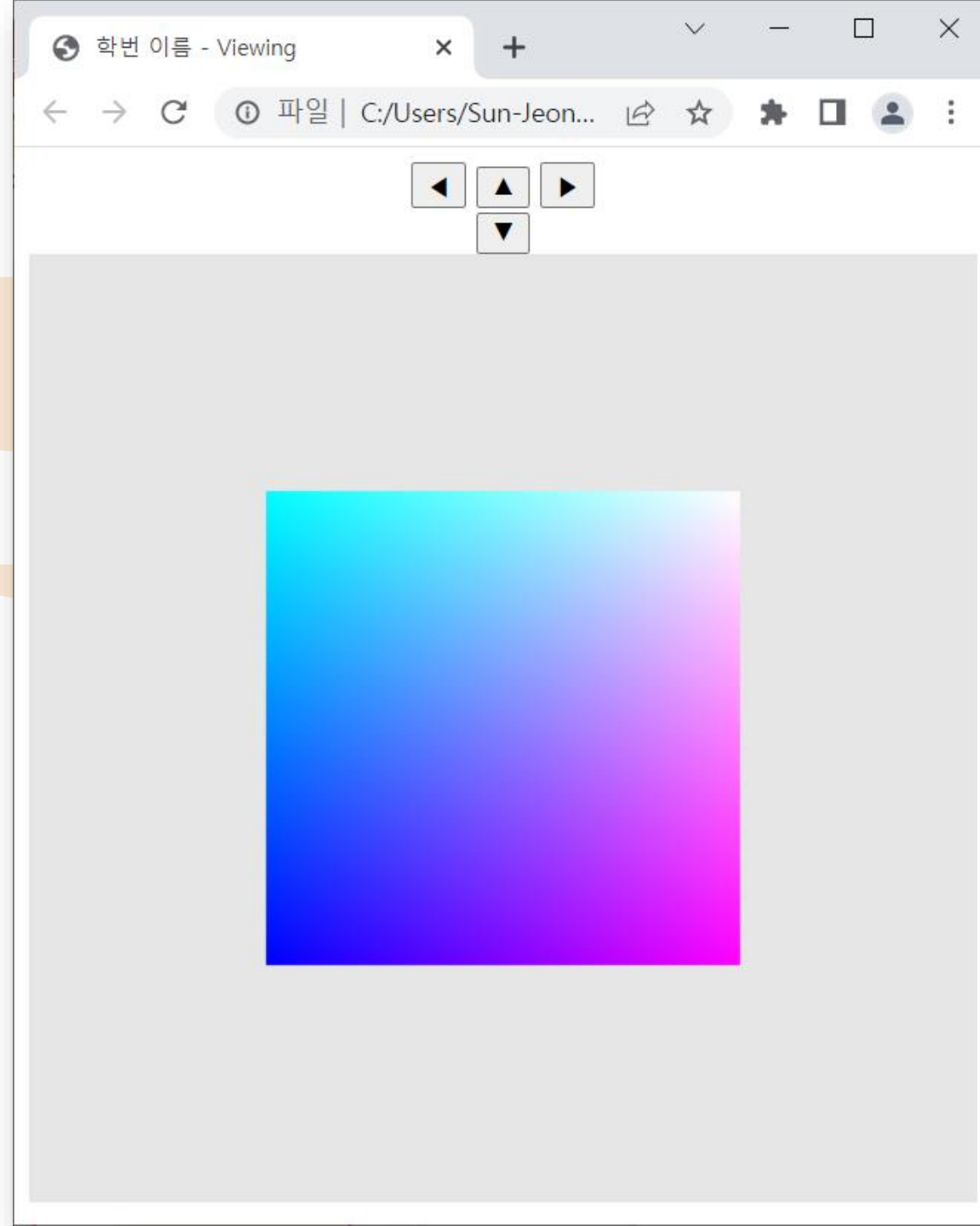
C: > Users > Sun-Jeong Kim > Desktop > CG > Common > JS MV.js > ...

```

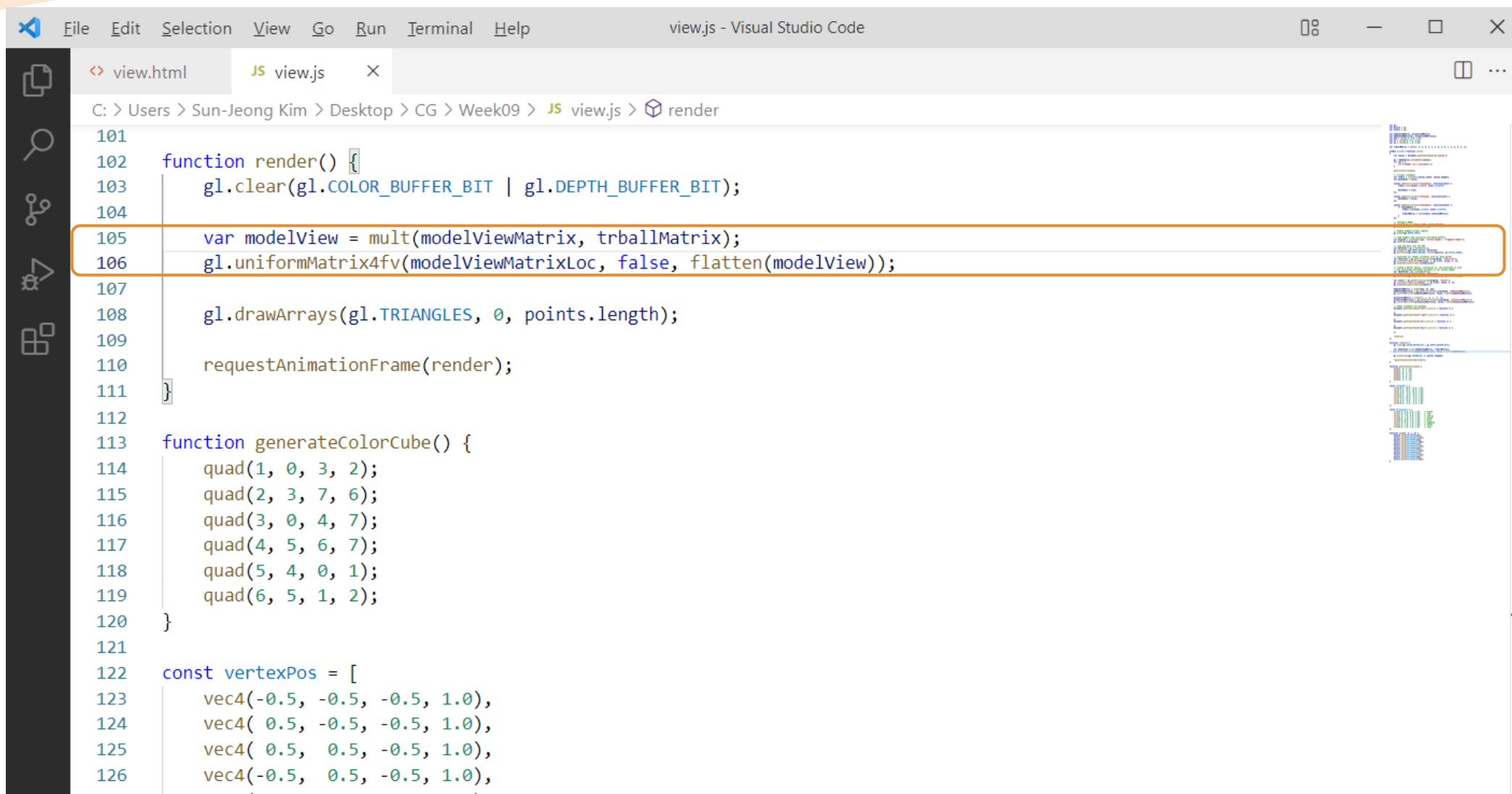
416 //
417 //  ModelView Matrix Generators
418 //
419
420 function lookAt( eye, at, up )
421 {
422     if ( !Array.isArray(eye) || eye.length !== 3 ) {
423         throw "lookAt(): first parameter [eye] must be an a vec3";
424     }
425
426     if ( !Array.isArray(at) || at.length !== 3 ) {
427         throw "lookAt(): first parameter [at] must be an a vec3";
428     }
429
430     if ( !Array.isArray(up) || up.length !== 3 ) {
431         throw "lookAt(): first parameter [up] must be an a vec3";
432     }
433
434     if ( equal(eye, at) ) {
435         return mat4();
436     }
437
438     var v = normalize( subtract(at, eye) ); // view direction vector
439     var n = normalize( cross(v, up) );     // perpendicular vector
440     var u = normalize( cross(n, v) );     // "new" up vector
441
442     v = negate( v );
443
444     var result = mat4(
445         vec4( n, -dot(n, eye) ),
446         vec4( u, -dot(u, eye) ),
447         vec4( v, -dot(v, eye) ),
448         vec4()
449     );
450

```

20

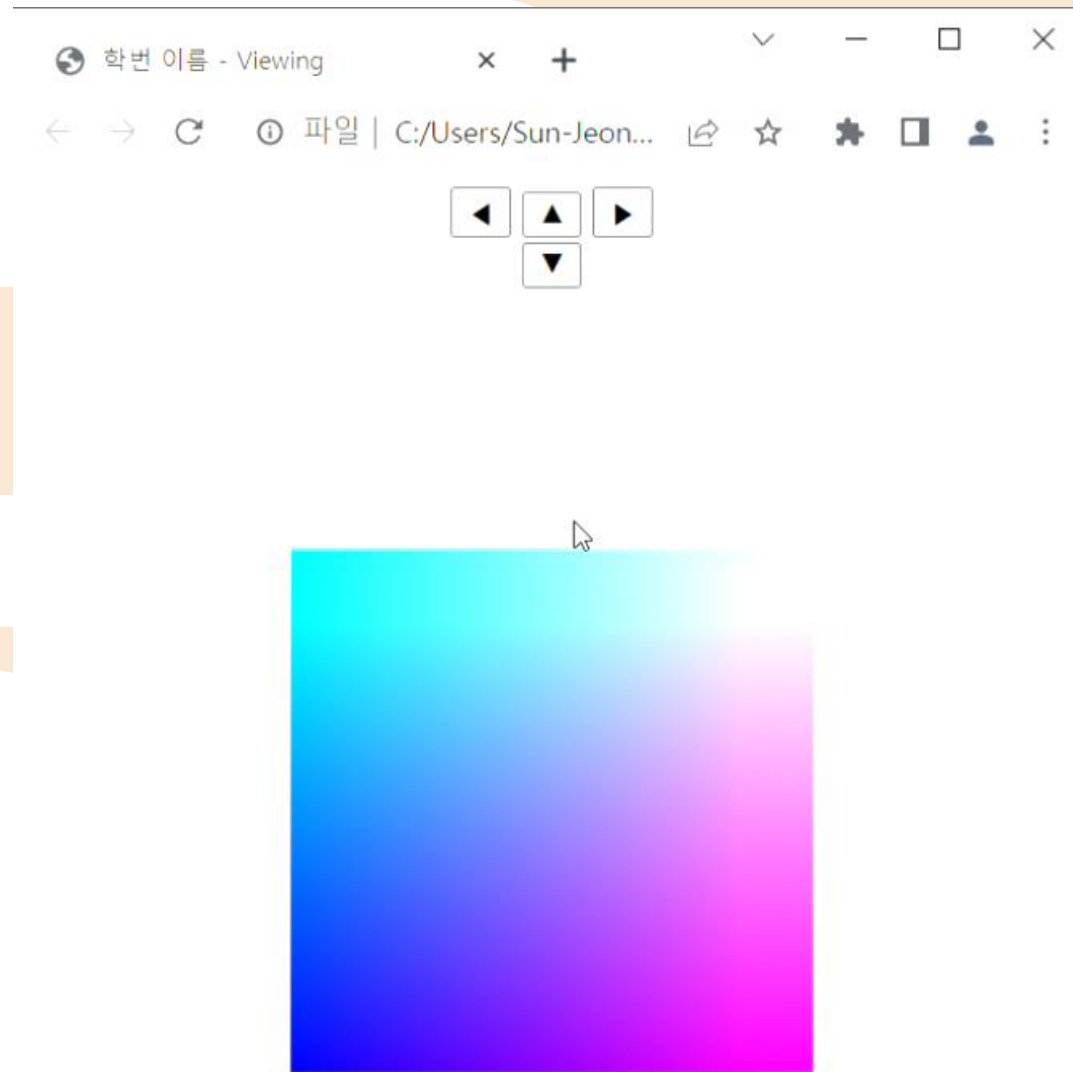


Rotation with a Quaternion



```
view.js - Visual Studio Code
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > render

101
102 function render() {
103     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
104
105     var modelView = mult(modelViewMatrix, trballMatrix);
106     gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelView));
107
108     gl.drawArrays(gl.TRIANGLES, 0, points.length);
109
110     requestAnimationFrame(render);
111 }
112
113 function generateColorCube() {
114     quad(1, 0, 3, 2);
115     quad(2, 3, 7, 6);
116     quad(3, 0, 4, 7);
117     quad(4, 5, 6, 7);
118     quad(5, 4, 0, 1);
119     quad(6, 5, 1, 2);
120 }
121
122 const vertexPos = [
123     vec4(-0.5, -0.5, -0.5, 1.0),
124     vec4( 0.5, -0.5, -0.5, 1.0),
125     vec4( 0.5,  0.5, -0.5, 1.0),
126     vec4(-0.5,  0.5, -0.5, 1.0),
```



```

view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > init
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 modelViewMatrix = lookAt(eye, at, up);
78 modelViewMatrixLoc = gl.getUniformLocation(program, "modelViewMatrix");
79 gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelViewMatrix));
80
81 // 3D orthographic viewing
82 //projectionMatrix = ortho(-1, 1, -1, 1, -1, 1);
83 var viewLength = 2.0;
84 if (canvas.width > canvas.height) { // landscape view
85     var aspectRatio = viewLength * canvas.width / canvas.height;
86     projectionMatrix = ortho(-aspectRatio, aspectRatio, -viewLength, viewLength, -viewLength, 1000);
87 }
88 else { // portrait view
89     var aspectRatio = viewLength * canvas.height / canvas.width;
90     projectionMatrix = ortho(-viewLength, viewLength, -aspectRatio, aspectRatio, -viewLength, 1000);
91 }
92 projectionMatrixLoc = gl.getUniformLocation(program, "projectionMatrix");
93 gl.uniformMatrix4fv(projectionMatrixLoc, false, flatten(projectionMatrix));
94
95 // Event listeners for buttons
96 document.getElementById("left").onclick = function () {
97
98 };
99 document.getElementById("right").onclick = function () {
100

```




연습 문제 (1)

- ortho() 함수의 left, right, bottom, top의 값을 변경해보고, 그 의미를 파악해 보시오.
 - 예) viewLength = 1.0; 또는 viewLength = 3.0;

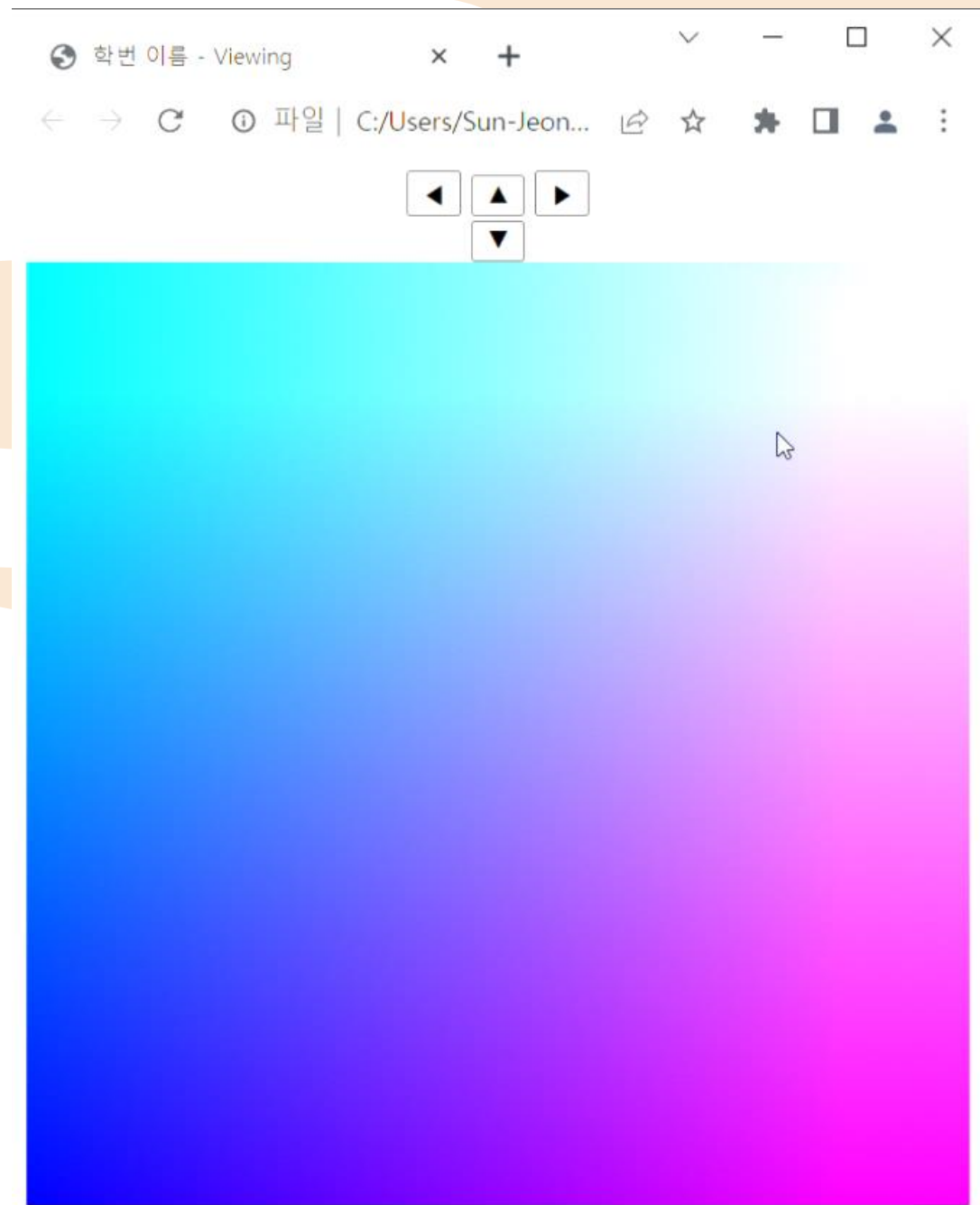
view.html

JS view.js

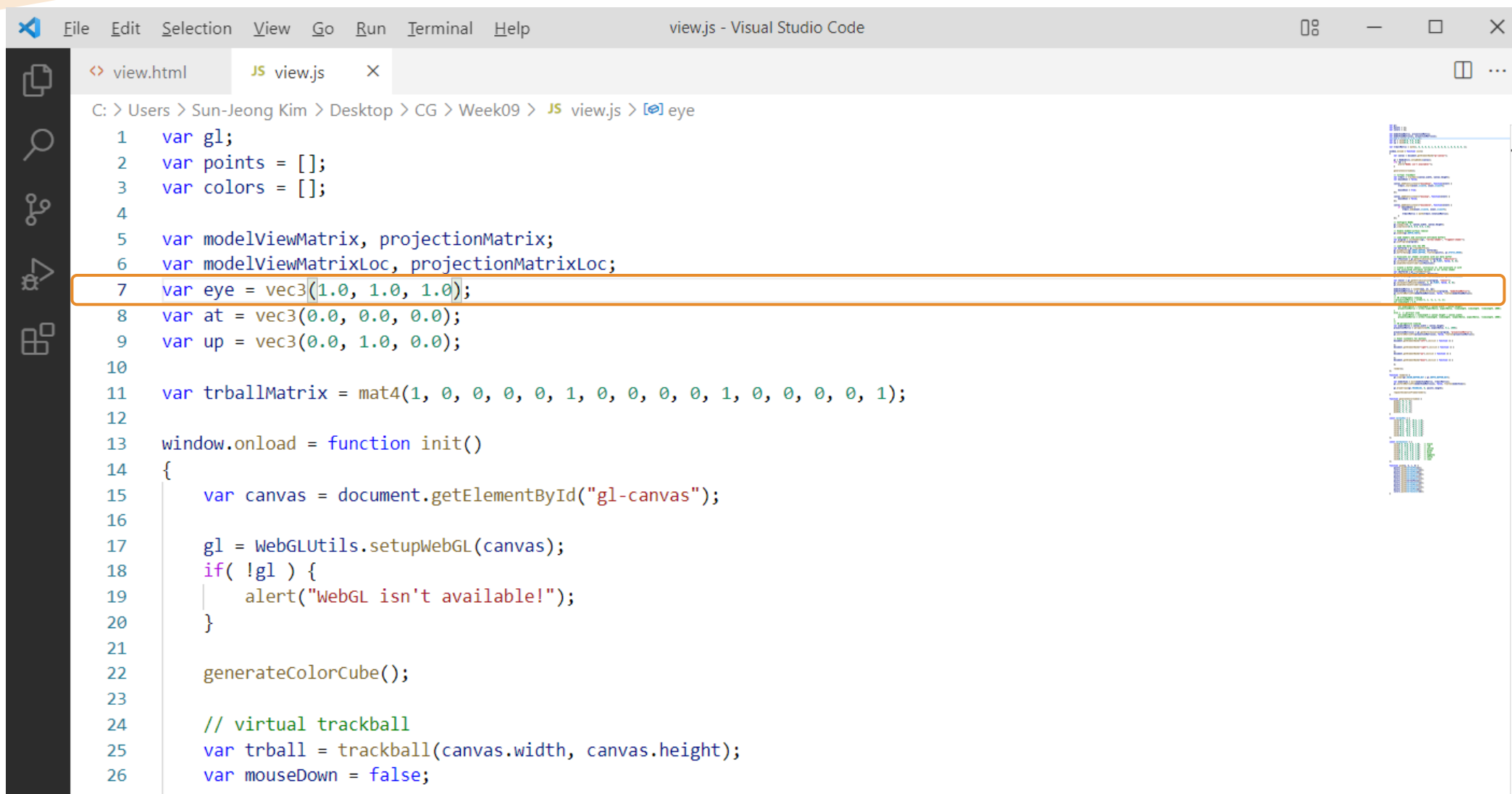
X

C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > init

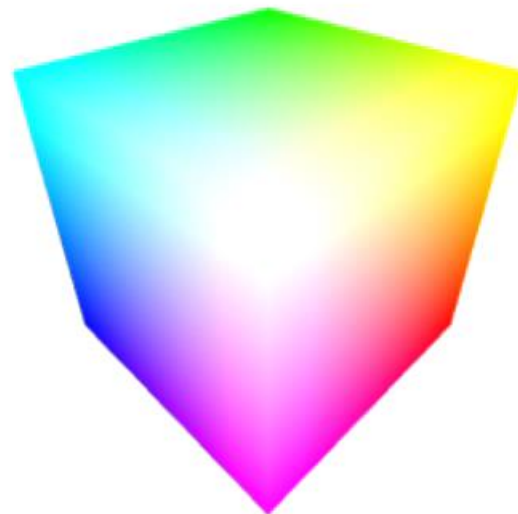
```
76
77     modelViewMatrix = lookAt(eye, at, up);
78     modelViewMatrixLoc = gl.getUniformLocation(program, "modelViewMatrix");
79     gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelViewMatrix));
80     /*
81     // 3D orthographic viewing
82     //projectionMatrix = ortho(-1, 1, -1, 1, -1, 1);
83     var viewLength = 2.0;
84     if (canvas.width > canvas.height) { // landscape view
85         var aspectRatio = viewLength * canvas.width / canvas.height;
86         projectionMatrix = ortho(-aspectRatio, aspectRatio, -viewLength, viewLength, -viewLength, 1000);
87     }
88     else { // portrait view
89         var aspectRatio = viewLength * canvas.height / canvas.width;
90         projectionMatrix = ortho(-viewLength, viewLength, -aspectRatio, aspectRatio, -viewLength, 1000);
91     }
92     */
93     // 3D perspective viewing
94     var aspectRatio = canvas.width / canvas.height;
95     projectionMatrix = perspective(90, aspectRatio, 0.1, 1000);
96
97     projectionMatrixLoc = gl.getUniformLocation(program, "projectionMatrix");
98     gl.uniformMatrix4fv(projectionMatrixLoc, false, flatten(projectionMatrix));
99
100    // Event listeners for buttons
101    document.getElementById("left").onclick = function () {
102
103    };
104    document.getElementById("right").onclick = function () {
105
106    };
107    document.getElementById("up").onclick = function () {
108
109    };
110    document.getElementById("down").onclick = function () {
```



Locating the Camera



```
view.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > [eye]
1  var gl;
2  var points = [];
3  var colors = [];
4
5  var modelViewMatrix, projectionMatrix;
6  var modelViewMatrixLoc, projectionMatrixLoc;
7  var eye = vec3(1.0, 1.0, 1.0);
8  var at = vec3(0.0, 0.0, 0.0);
9  var up = vec3(0.0, 1.0, 0.0);
10
11  var trballMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
12
13  window.onload = function init()
14  {
15      var canvas = document.getElementById("gl-canvas");
16
17      gl = WebGLUtils.setupWebGL(canvas);
18      if( !gl ) {
19          alert("WebGL isn't available!");
20      }
21
22      generateColorCube();
23
24      // virtual trackball
25      var trball = trackball(canvas.width, canvas.height);
26      var mouseDown = false;
```



연습 문제 (2)

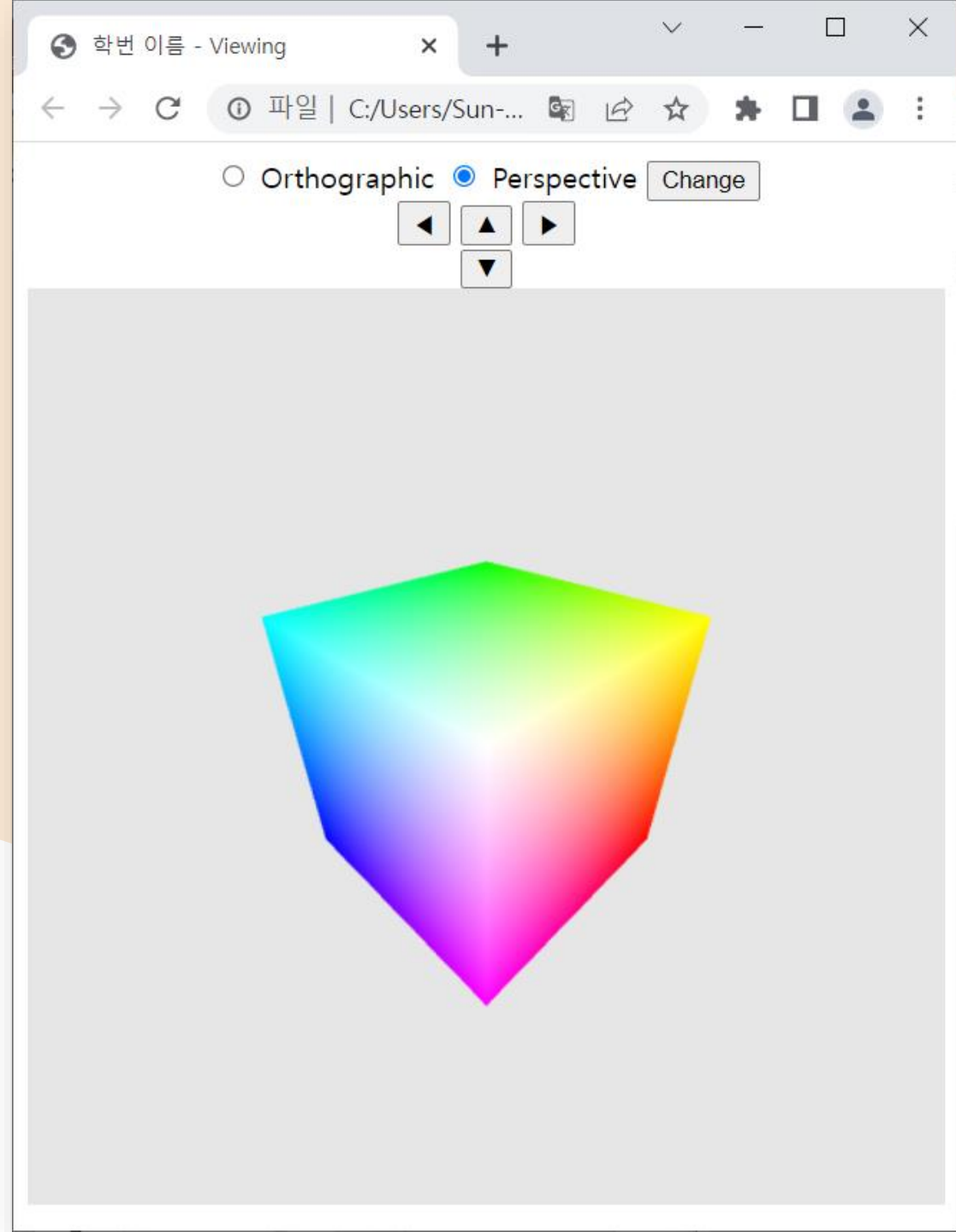
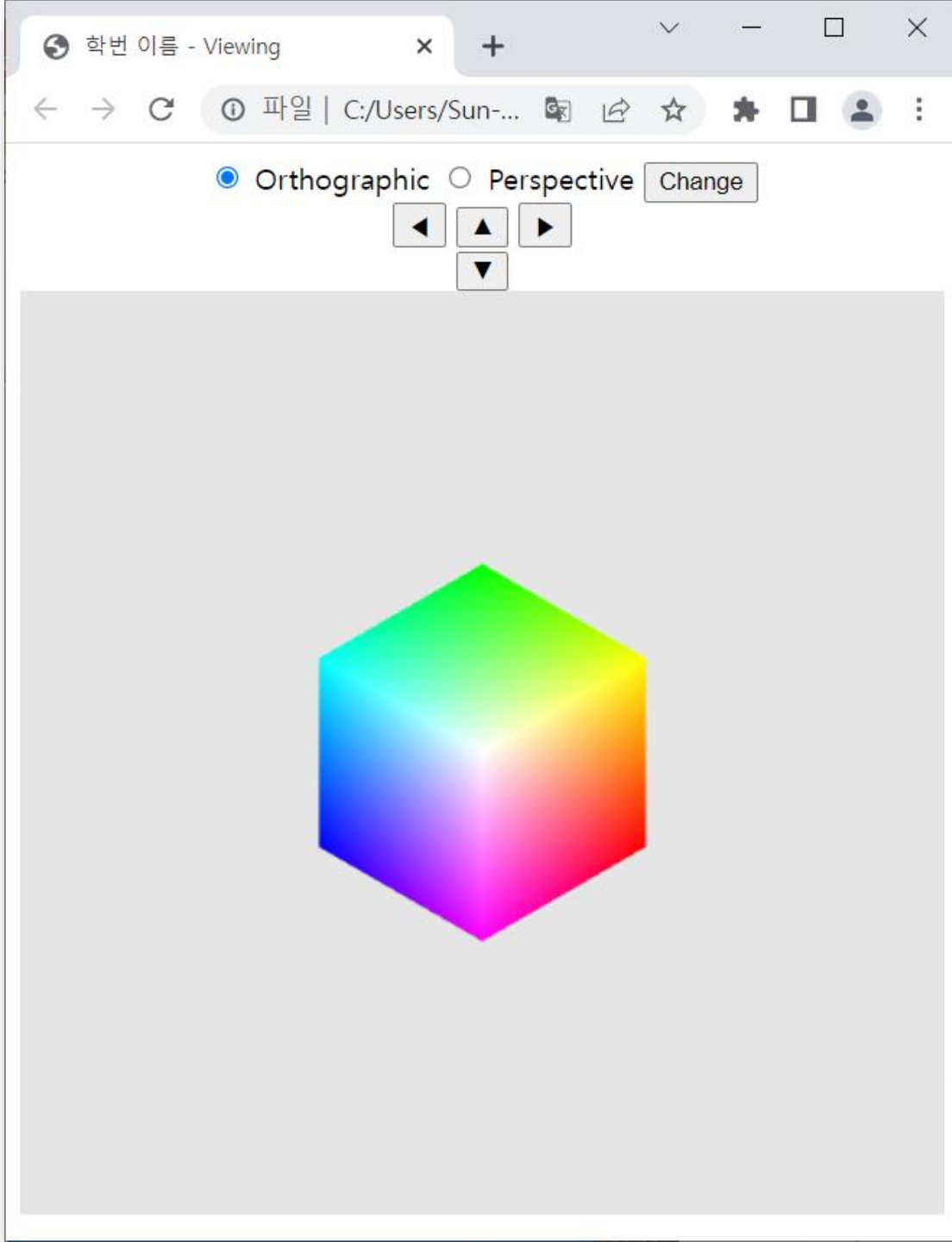
- perspective() 함수에서 fovy 파라미터 값을 변경해보고, 그 의미를 파악하시오.
 - 예) 90도 대신 60도 또는 120도
- perspective() 함수에서 near, far 파라미터 값을 변경해보고, 그 의미를 파악하시오.
 - 0.1 대신 0.01 또는 1.0
 - 1000 대신 100 또는 10000

```

<> view.html x JS view.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > <> view.html > html > body > div > button#change

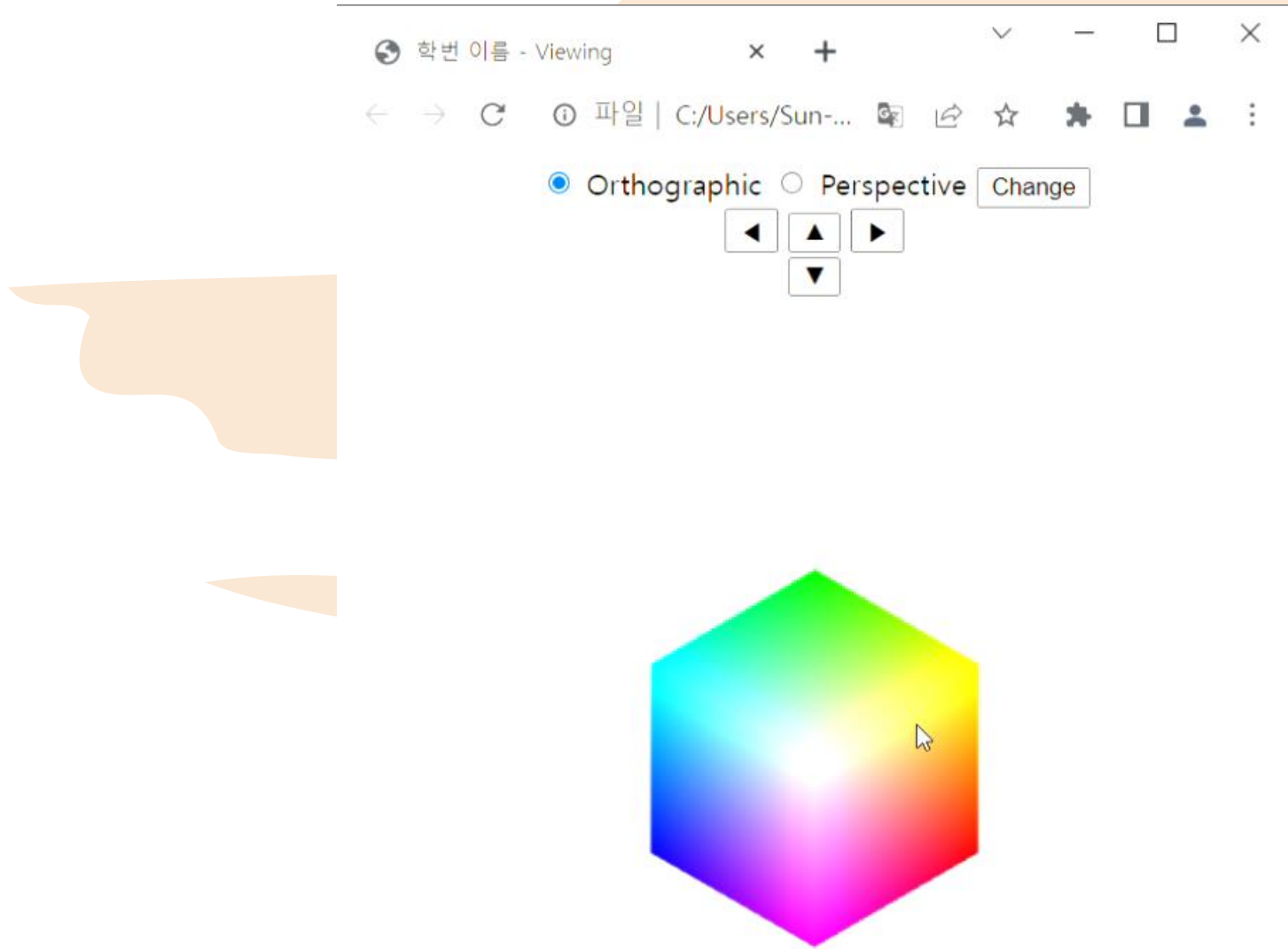
18
19     <script id="fragment-shader" type="x-shader/x-fragment">
20     precision mediump float;
21     varying vec4 fColor;
22
23     void main() {
24     |     gl_FragColor = fColor;
25     }
26     </script>
27
28     <script type="text/javascript" src="../Common/webgl-utils.js"></script>
29     <script type="text/javascript" src="../Common/initShaders.js"></script>
30     <script type="text/javascript" src="../Common/MV.js"></script>
31     <script type="text/javascript" src="../trackball.js"></script>
32     <script type="text/javascript" src="view.js"></script>
33 </head>
34 <body>
35     <div style="width:512px; text-align:center;">
36     |     <input type="radio" id="ortho" name="projection" checked> Orthographic
37     |     <input type="radio" id="persp" name="projection" > Perspective
38     |     <button id="change">Change</button>
39     </div>
40     <div style="width:512px; text-align:center;">
41     |     <button id="left"><img alt="left arrow icon" data-bbox="438 681 451 694"/></button>
42     |     <button id="up"><img alt="up arrow icon" data-bbox="438 706 451 719"/></button>
43     |     <button id="right"><img alt="right arrow icon" data-bbox="438 731 451 744"/></button><br>
44     |     <button id="down"><img alt="down arrow icon" data-bbox="438 756 451 769"/></button>
45     </div>
46     <canvas id="gl-canvas" width="512" height="512">
47     |     Oops... your browser doesn't support the HTML5 canvas element!
48     </canvas><br>
49 </body>
50 </html>

```

연습 문제 (3)

- 라디오 버튼을 만들어 Projection 방법을 선택할 수 있도록 구현하시오.
- Change 버튼을 누르면 선택된 Projection 방법으로 변경되도록 하시오.



Drawing the Ground



The image shows a Visual Studio Code window with a file named 'view.js'. The code is written in JavaScript and uses WebGL to draw a 3D scene. The scene includes a cube and a ground plane. The ground plane is drawn using the 'generateGround()' function, which is highlighted with an orange box. The code also includes a 'generateColorCube()' function and a 'trackball' for camera control. The browser's console is visible on the right, showing the output of the code.

```
view.js - Visual Studio Code

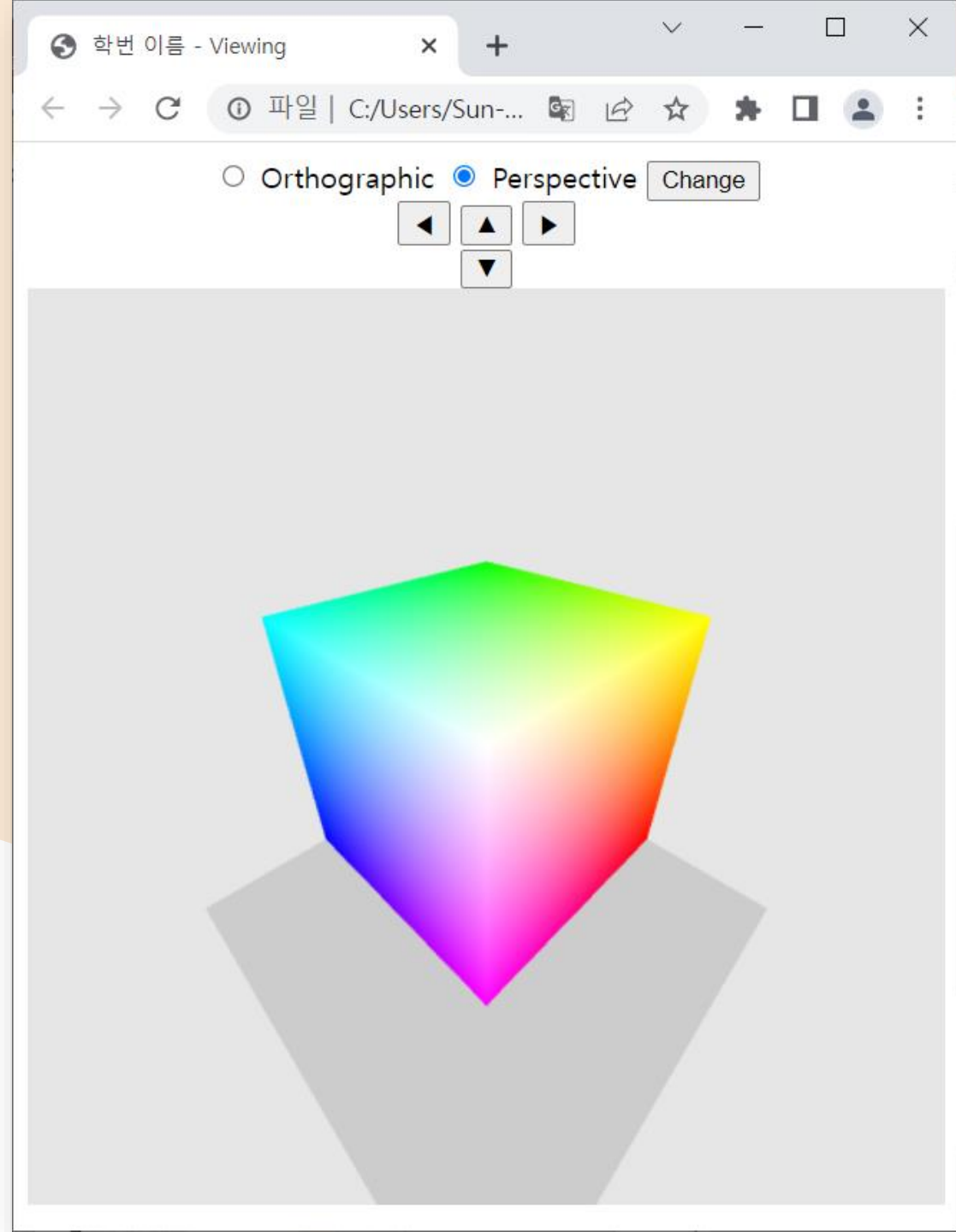
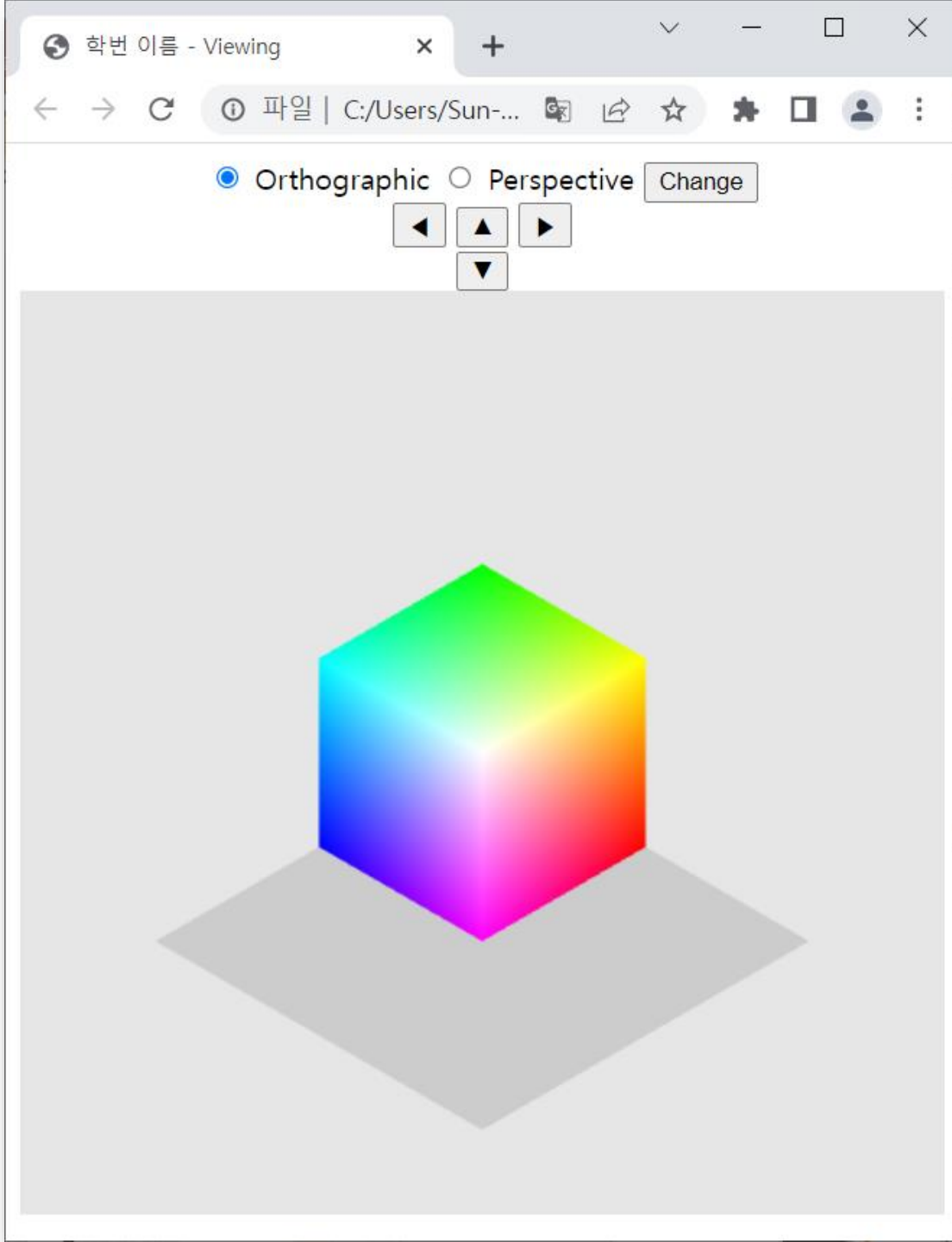
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > init

1  var gl;
2  var points = [];
3  var colors = [];
4
5  var modelViewMatrix, projectionMatrix;
6  var modelViewMatrixLoc, projectionMatrixLoc;
7  var eye = vec3(1.0, 1.0, 1.0);
8  var at = vec3(0.0, 0.0, 0.0);
9  var up = vec3(0.0, 1.0, 0.0);
10
11 var trballMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
12
13 window.onload = function init()
14 {
15     var canvas = document.getElementById("gl-canvas");
16
17     gl = WebGLUtils.setupWebGL(canvas);
18     if( !gl ) {
19         alert("WebGL isn't available!");
20     }
21
22     generateColorCube();
23     generateGround();
24
25     // virtual trackball
26     var trball = trackball(canvas.width, canvas.height);
```

```

<> view.html  JS view.js  X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > generateGround
181  };
182
183  function quad(a, b, c, d) {
184      points.push(vertexPos[a]);
185      colors.push(vertexColor[a]);
186      points.push(vertexPos[b]);
187      colors.push(vertexColor[b]);
188      points.push(vertexPos[c]);
189      colors.push(vertexColor[c]);
190      points.push(vertexPos[a]);
191      colors.push(vertexColor[a]);
192      points.push(vertexPos[c]);
193      colors.push(vertexColor[c]);
194      points.push(vertexPos[d]);
195      colors.push(vertexColor[d]);
196  }
197
198  function generateGround() {
199      // two triangles
200      points.push(vec4(1.0, -1.0, -1.0, 1.0));
201      colors.push(vec4(0.8, 0.8, 0.8, 1.0));
202      points.push(vec4(-1.0, -1.0, -1.0, 1.0));
203      colors.push(vec4(0.8, 0.8, 0.8, 1.0));
204      points.push(vec4(-1.0, -1.0, 1.0, 1.0));
205      colors.push(vec4(0.8, 0.8, 0.8, 1.0));
206
207      points.push(vec4(1.0, -1.0, -1.0, 1.0));
208      colors.push(vec4(0.8, 0.8, 0.8, 1.0));
209      points.push(vec4(-1.0, -1.0, 1.0, 1.0));
210      colors.push(vec4(0.8, 0.8, 0.8, 1.0));
211      points.push(vec4(1.0, -1.0, 1.0, 1.0));
212      colors.push(vec4(0.8, 0.8, 0.8, 1.0));
213  }
214

```



```

view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > generateGround

207 points.push(vec4(-1.0, -1.0, -1.0, 1.0));
208 colors.push(vec4(0.8, 0.8, 0.8, 1.0));
209 points.push(vec4(-1.0, -1.0, 1.0, 1.0));
210 colors.push(vec4(0.8, 0.8, 0.8, 1.0));
211
212 points.push(vec4(1.0, -1.0, -1.0, 1.0));
213 colors.push(vec4(0.8, 0.8, 0.8, 1.0));
214 points.push(vec4(-1.0, -1.0, 1.0, 1.0));
215 colors.push(vec4(0.8, 0.8, 0.8, 1.0));
216 points.push(vec4(1.0, -1.0, 1.0, 1.0));
217 colors.push(vec4(0.8, 0.8, 0.8, 1.0));
218
219 // boundary lines
220 points.push(vec4(1.0, -1.0, -1.0, 1.0));
221 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
222 points.push(vec4(-1.0, -1.0, -1.0, 1.0));
223 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
224
225 points.push(vec4(-1.0, -1.0, -1.0, 1.0));
226 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
227 points.push(vec4(-1.0, -1.0, 1.0, 1.0));
228 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
229
230 points.push(vec4(-1.0, -1.0, 1.0, 1.0));
231 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
232 points.push(vec4(1.0, -1.0, 1.0, 1.0));
233 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
234
235 points.push(vec4(1.0, -1.0, 1.0, 1.0));
236 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
237 points.push(vec4(1.0, -1.0, -1.0, 1.0));
238 colors.push(vec4(0.0, 0.0, 0.0, 1.0));
239
240

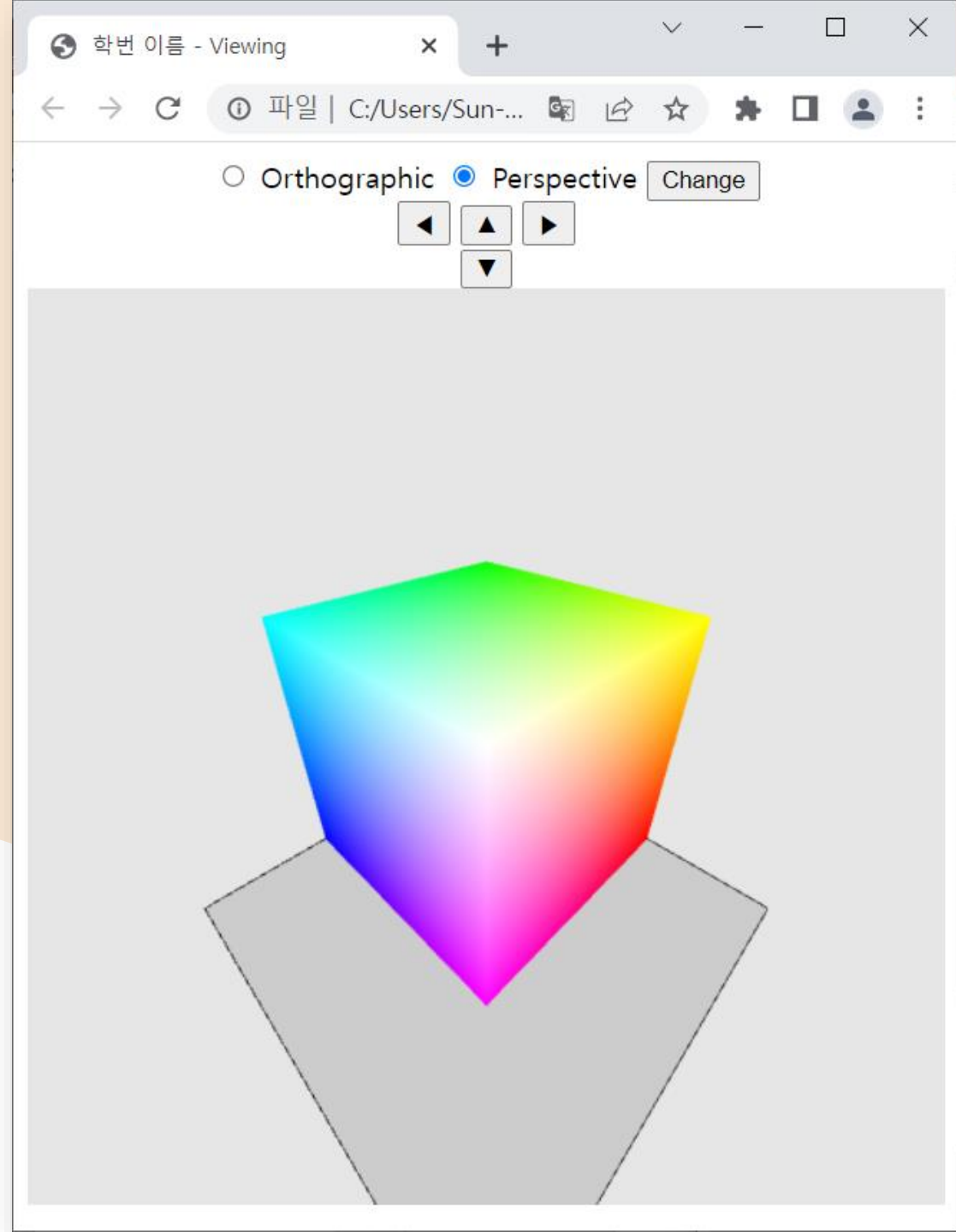
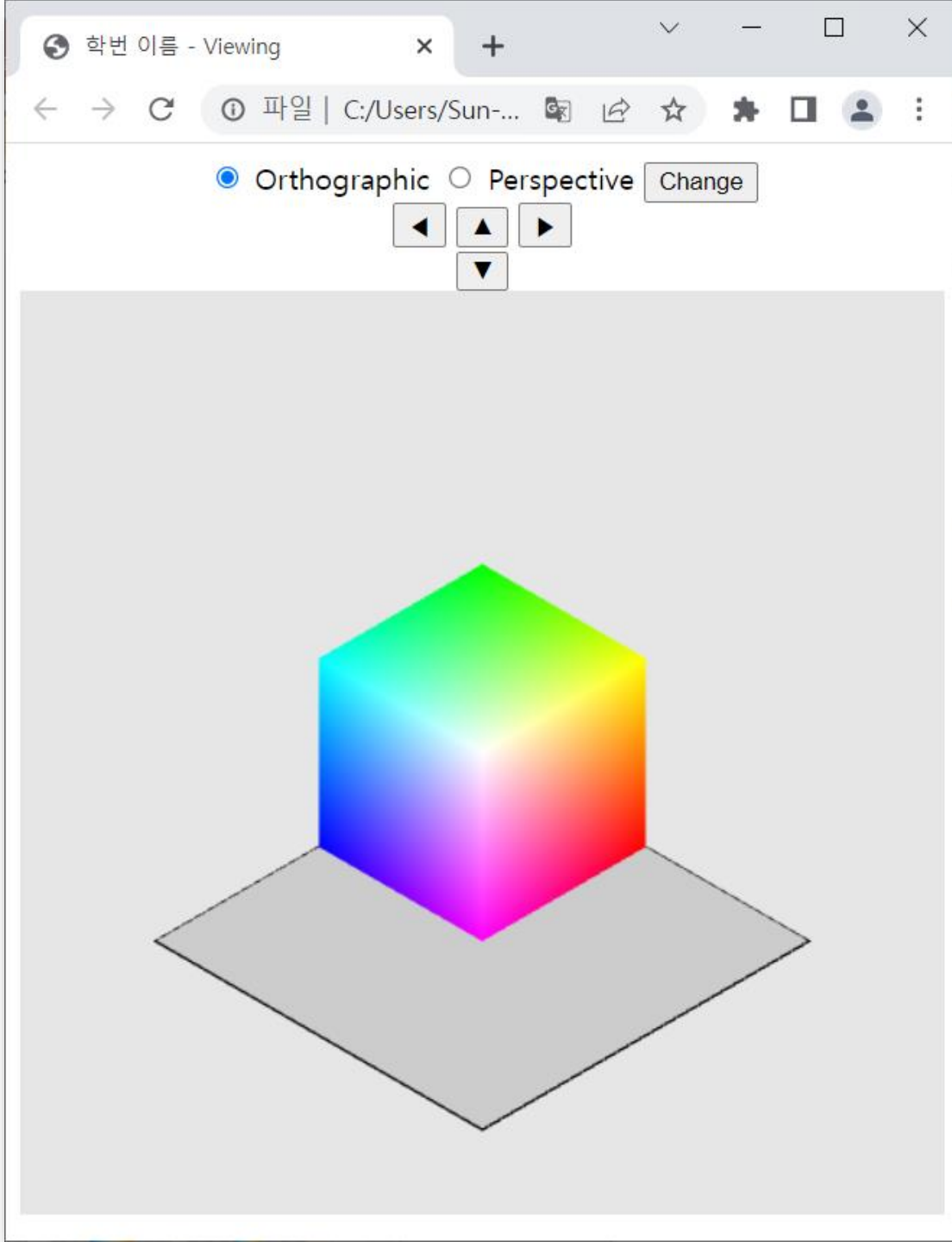
```

```

view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > render

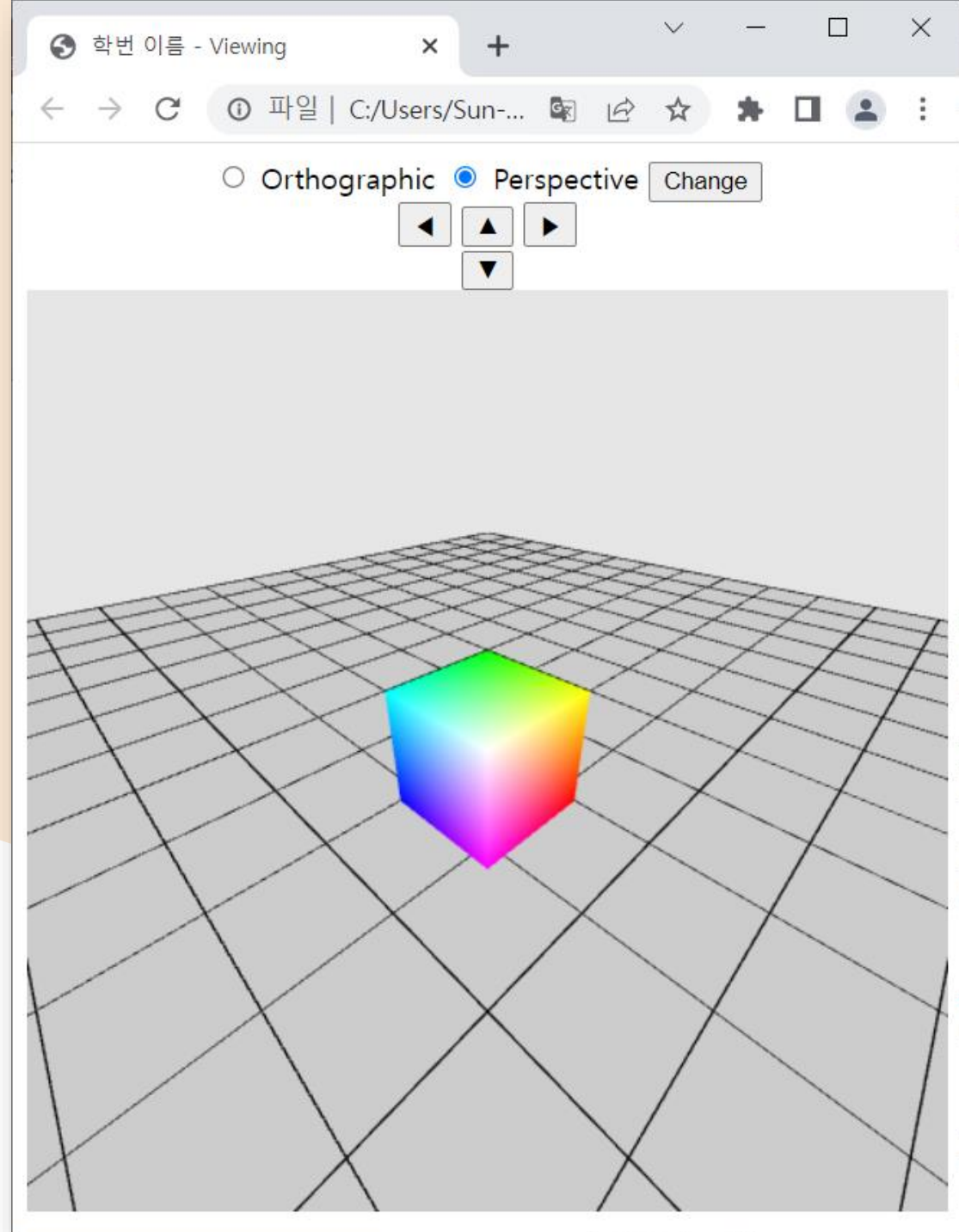
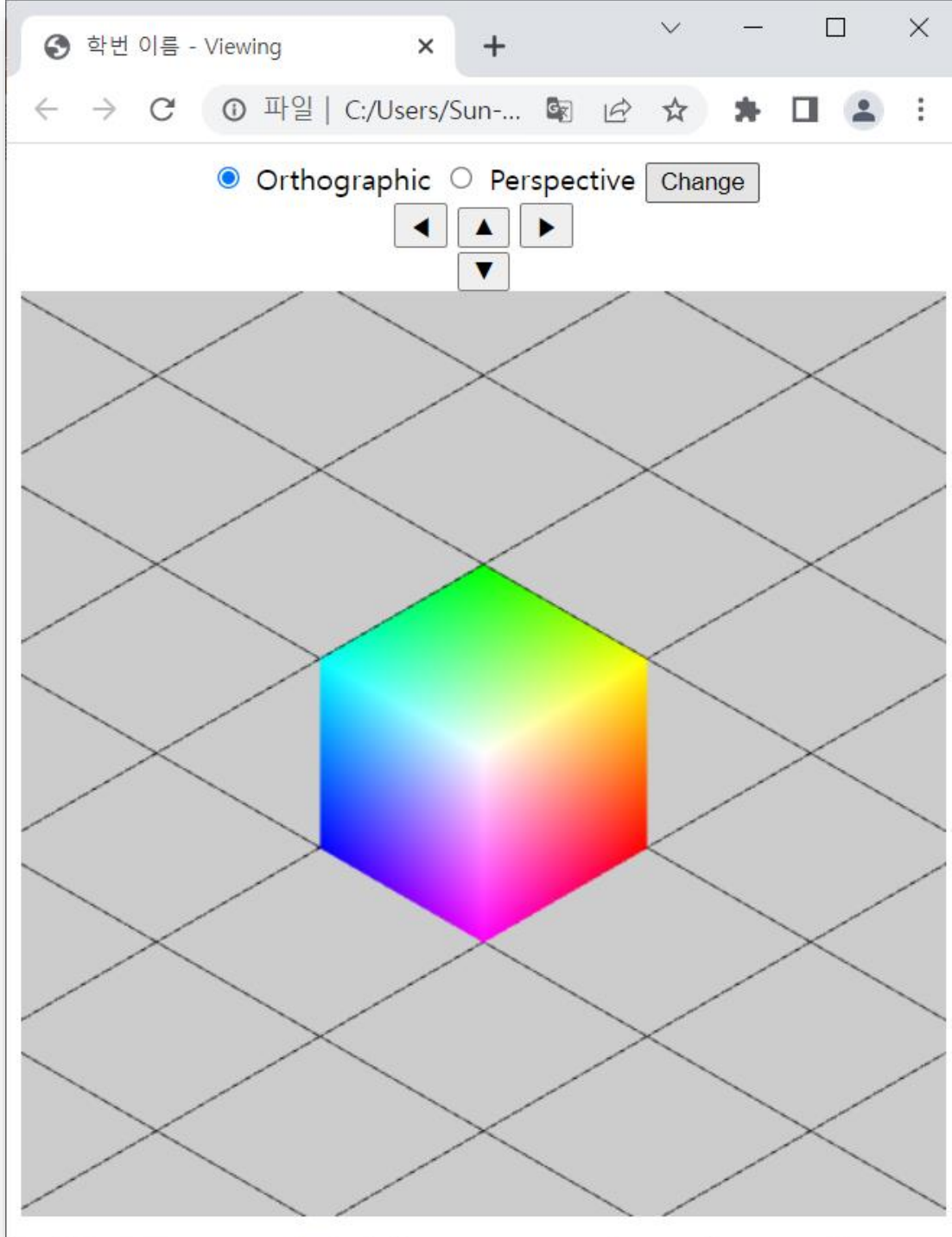
140
141 function render() {
142     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
143
144     var modelView = mult(modelViewMatrix, trballMatrix);
145     gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelView));
146
147     // draw a color cube
148     gl.drawArrays(gl.TRIANGLES, 0, 36);
149
150     // draw the ground
151     gl.drawArrays(gl.TRIANGLES, 36, 6);
152     gl.drawArrays(gl.LINES, 42, 8);
153
154     requestAnimationFrame(render);
155 }
156
157 function generateColorCube() {
158     quad(1, 0, 3, 2);
159     quad(2, 3, 7, 6);
160     quad(3, 0, 4, 7);
161     quad(4, 5, 6, 7);
162     quad(5, 4, 0, 1);
163     quad(6, 5, 1, 2);
164 }
165
166 const vertexPos = [
167     vec4(-0.5, -0.5, -0.5, 1.0),
168     vec4( 0.5, -0.5, -0.5, 1.0),
169     vec4( 0.5,  0.5, -0.5, 1.0),
170     vec4(-0.5,  0.5, -0.5, 1.0),
171     vec4(-0.5, -0.5,  0.5, 1.0),
172     vec4( 0.5, -0.5,  0.5, 1.0),
173     vec4( 0.5,  0.5,  0.5, 1.0),
174     vec4(-0.5,  0.5,  0.5, 1.0)

```

연습 문제 (4)

- Ground의 크기를 10배 늘리시오.
- Ground 간격 1마다 격자 선을 그리시오.
- 카메라의 위치를 (2, 2, 2)로 변경하시오.



view.js - Visual Studio Code

view.html JS view.js

C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > eye

```
1  var gl;
2  var points = [];
3  var colors = [];
4
5  var modelViewMatrix, projectionMatrix;
6  var modelViewMatrixLoc, projectionMatrixLoc;
7  var eye = vec3(2.0, 2.0, 2.0);
8  var at = vec3(0.0, 0.0, 0.0);
9  var up = vec3(0.0, 1.0, 0.0);
10
11 var trballMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
12
13 window.onload = function init()
14 {
15     var canvas = document.getElementById("gl-canvas");
16
17     gl = WebGLUtils.setupWebGL(canvas);
18     if( !gl ) {
19         alert("WebGL isn't available!");
20     }
21
22     generateColorCube();
23     generateGround();
24
25     // virtual trackball
26     var trball = trackball(canvas.width, canvas.height);
27     var mouseDown = false;
28
29     canvas.addEventListener("mousedown", function(event) {
30         trball.start(event.clientX, event.clientY);
31
32         mouseDown = true;
33     });
34
35     canvas.addEventListener("mouseup", function(event) {
```

14

Ln 7, Col 27 Spaces: 4 UTF-8 CRLF {} JavaScript

view.js - Visual Studio Code

view.html JS view.js

C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > init

```
46
47 // Configure WebGL
48 gl.viewport(0, 0, canvas.width, canvas.height);
49 gl.clearColor(0.9, 0.9, 0.9, 1.0);
50
51 // Enable hidden-surface removal
52 gl.enable(gl.DEPTH_TEST);
53
54 gl.enable(gl.POLYGON_OFFSET_FILL);
55 gl.polygonOffset(0.01, 1);
56
57 // Load shaders and initialize attribute buffers
58 var program = initShaders(gl, "vertex-shader", "fragment-shader");
59 gl.useProgram(program);
60
61 // Load the data into the GPU
62 var bufferId = gl.createBuffer();
63 gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
64 gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
65
66 // Associate our shader variables with our data buffer
67 var vPosition = gl.getAttribLocation(program, "vPosition");
68 gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
69 gl.enableVertexAttribArray(vPosition);
70
71 // Create a buffer object, initialize it, and associate it with
72 // the associated attribute variable in our vertex shader
73 var cBufferId = gl.createBuffer();
74 gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
75 gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
76
77 var vColor = gl.getAttribLocation(program, "vColor");
78 gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
79 gl.enableVertexAttribArray(vColor);
80
```

Ln 55, Col 31 Spaces: 4 UTF-8 CRLF {} JavaScript

```

<> view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > render

143
144 function render() {
145     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
146
147     var modelView = mult(modelViewMatrix, trballMatrix);
148     gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelView));
149
150     // draw a color cube
151     gl.drawArrays(gl.TRIANGLES, 0, 36);
152
153     // draw the ground
154     gl.drawArrays(gl.TRIANGLES, 36, 6);
155     gl.drawArrays(gl.LINES, 42, 84); // (21 + 21) * 2 = 84
156
157     requestAnimationFrame(render);
158 }
159
160 function generateColorCube() {
161     quad(1, 0, 3, 2);
162     quad(2, 3, 7, 6);
163     quad(3, 0, 4, 7);
164     quad(4, 5, 6, 7);
165     quad(5, 4, 0, 1);
166     quad(6, 5, 1, 2);
167 }
168
169 const vertexPos = [
170     vec4(-0.5, -0.5, -0.5, 1.0),
171     vec4( 0.5, -0.5, -0.5, 1.0),
172     vec4( 0.5,  0.5, -0.5, 1.0),
173     vec4(-0.5,  0.5, -0.5, 1.0),
174     vec4(-0.5, -0.5,  0.5, 1.0),
175     vec4( 0.5, -0.5,  0.5, 1.0),
176     vec4( 0.5,  0.5,  0.5, 1.0),
177     vec4(-0.5,  0.5,  0.5, 1.0)

```

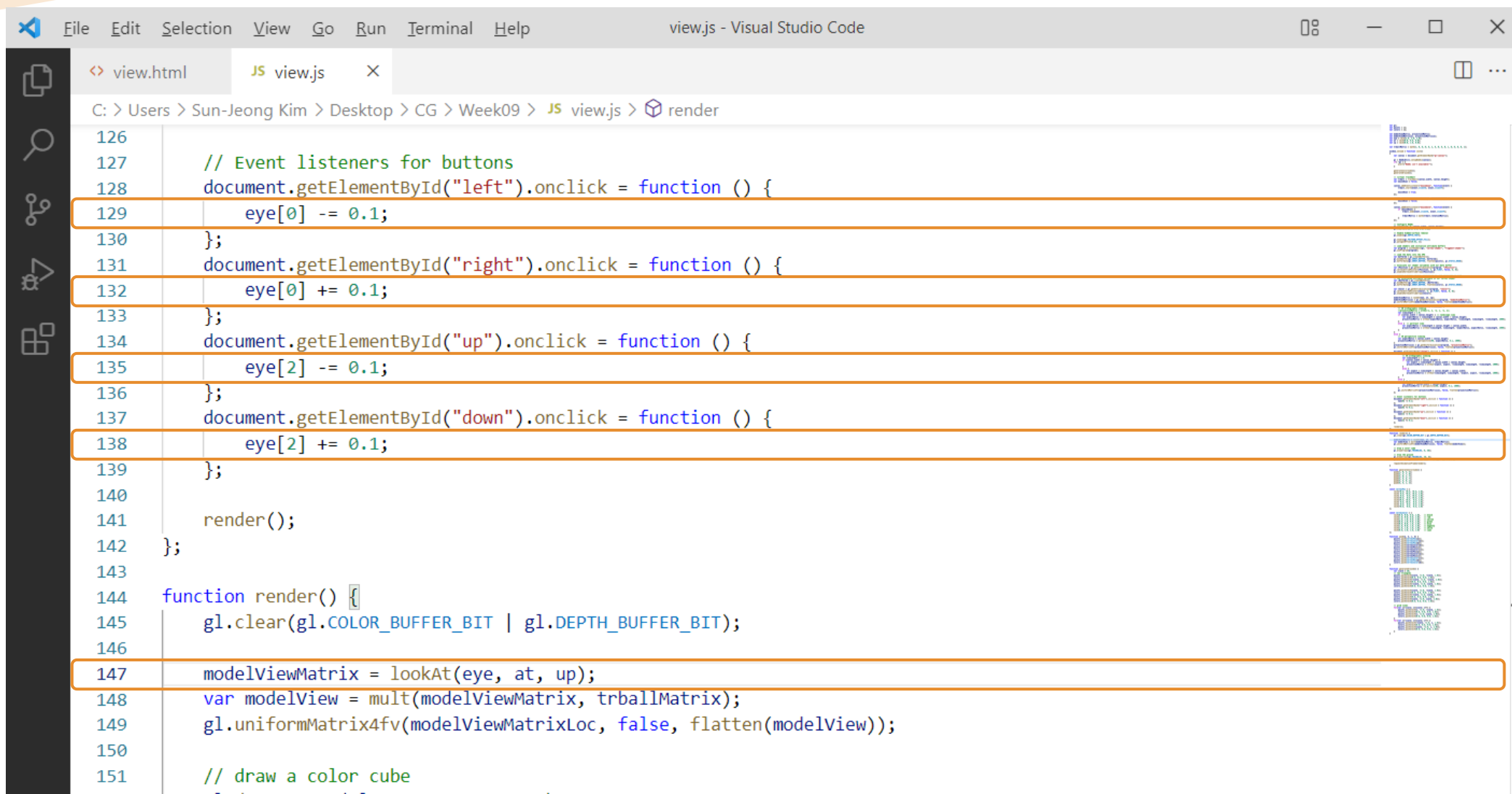
```

view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > generateGround

204   }
205
206   function generateGround() {
207       var scale = 10;
208       // two triangles
209       points.push(vec4(scale, -1.0, -scale, 1.0));
210       colors.push(vec4(0.8, 0.8, 0.8, 1.0));
211       points.push(vec4(-scale, -1.0, -scale, 1.0));
212       colors.push(vec4(0.8, 0.8, 0.8, 1.0));
213       points.push(vec4(-scale, -1.0, scale, 1.0));
214       colors.push(vec4(0.8, 0.8, 0.8, 1.0));
215
216       points.push(vec4(scale, -1.0, -scale, 1.0));
217       colors.push(vec4(0.8, 0.8, 0.8, 1.0));
218       points.push(vec4(-scale, -1.0, scale, 1.0));
219       colors.push(vec4(0.8, 0.8, 0.8, 1.0));
220       points.push(vec4(scale, -1.0, scale, 1.0));
221       colors.push(vec4(0.8, 0.8, 0.8, 1.0));
222
223       // grid lines
224       for(var x=-scale; x<=scale; x++) {
225           points.push(vec4(x, -1.0, -scale, 1.0));
226           colors.push(vec4(0.0, 0.0, 0.0, 1.0));
227           points.push(vec4(x, -1.0, scale, 1.0));
228           colors.push(vec4(0.0, 0.0, 0.0, 1.0));
229       }
230       for(var z=-scale; z<=scale; z++) {
231           points.push(vec4(-scale, -1.0, z, 1.0));
232           colors.push(vec4(0.0, 0.0, 0.0, 1.0));
233           points.push(vec4(scale, -1.0, z, 1.0));
234           colors.push(vec4(0.0, 0.0, 0.0, 1.0));
235       }
236   }
237

```

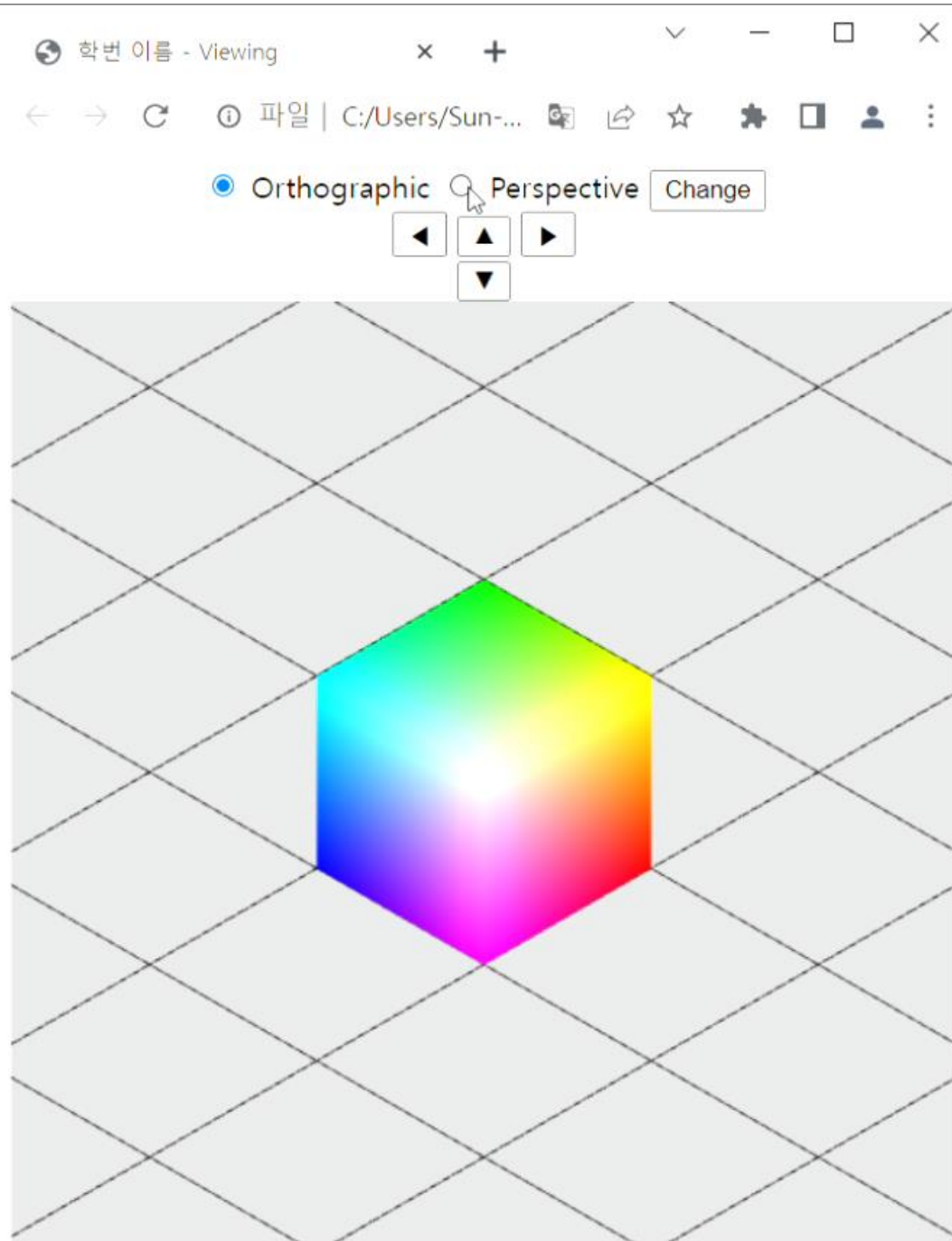

Walking Through



```
view.js - Visual Studio Code
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > render

126
127 // Event listeners for buttons
128 document.getElementById("left").onclick = function () {
129     eye[0] -= 0.1;
130 };
131 document.getElementById("right").onclick = function () {
132     eye[0] += 0.1;
133 };
134 document.getElementById("up").onclick = function () {
135     eye[2] -= 0.1;
136 };
137 document.getElementById("down").onclick = function () {
138     eye[2] += 0.1;
139 };
140
141 render();
142 };
143
144 function render() {
145     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
146
147     modelViewMatrix = lookAt(eye, at, up);
148     var modelView = mult(modelViewMatrix, trballMatrix);
149     gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelView));
150
151     // draw a color cube
```


What's Wrong?



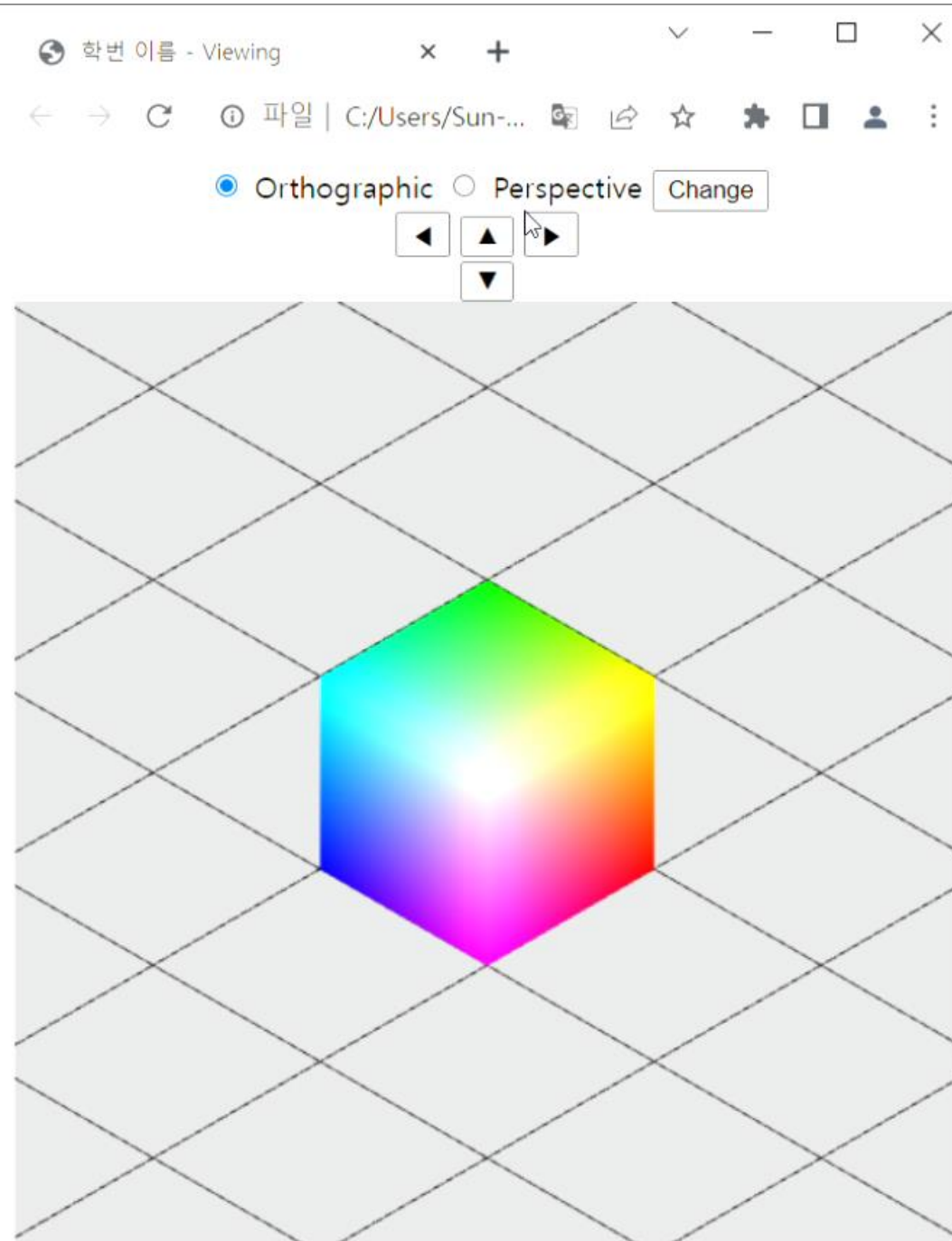
```

view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > init > onclick

126
127 // Event listeners for buttons
128 document.getElementById("left").onclick = function () {
129     eye[0] -= 0.1;
130     at[0] -= 0.1;
131 };
132 document.getElementById("right").onclick = function () {
133     eye[0] += 0.1;
134     at[0] += 0.1;
135 };
136 document.getElementById("up").onclick = function () {
137     eye[2] -= 0.1;
138     at[2] -= 0.1;
139 };
140 document.getElementById("down").onclick = function () {
141     eye[2] += 0.1;
142     at[2] += 0.1;
143 };
144
145 render();
146 };
147
148 function render() {
149     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
150
151     modelViewMatrix = lookAt(eye, at, up);
152     var modelView = mult(modelViewMatrix, trballMatrix);
153     gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelView));
154
155     // draw a color cube
156     gl.drawArrays(gl.TRIANGLES, 0, 36);
157
158     // draw the ground
159     gl.drawArrays(gl.TRIANGLES, 36, 6);
160     gl.drawArrays(gl.LINES, 42, 84); // (21 + 21) * 2 = 84

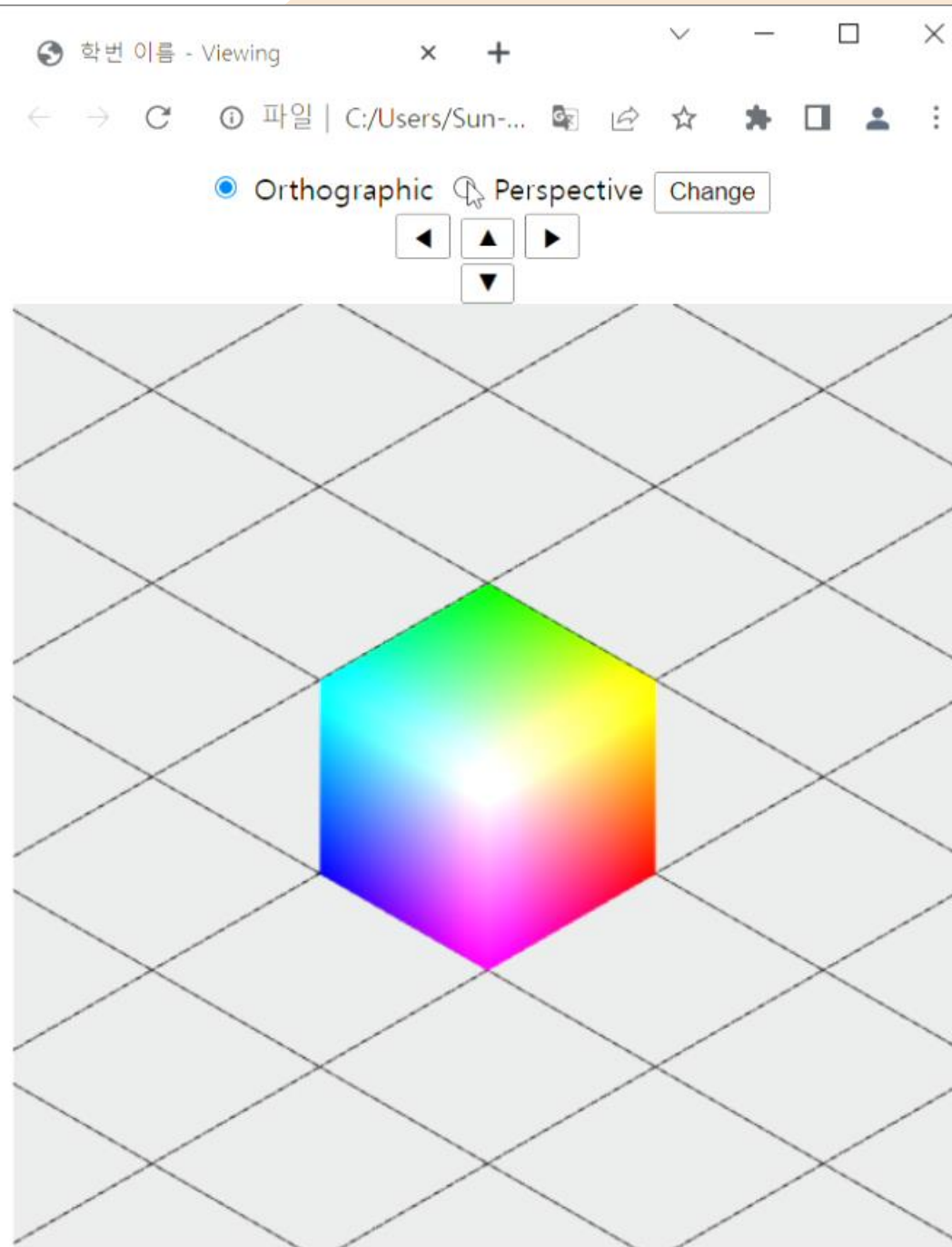
```

Is it Natural?



연습 문제 (5)

- ◀와 ▶ 버튼 입력에 대해,
 - 카메라의 로컬 y 축 회전하도록 구현하시오.
 - 즉, 제자리에서 바라보는 방향만 변경하시오.
- ▲와 ▼ 버튼 입력에 대해,
 - 카메라가 바라보는 방향으로 전진 또는 후진하도록 구현하시오.
- 카메라의 위치가 Ground 밖으로 나가지 못하도록 구현하시오.



```

<> view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > ...
1  var gl;
2  var points = [];
3  var colors = [];
4
5  var modelViewMatrix, projectionMatrix;
6  var modelViewMatrixLoc, projectionMatrixLoc;
7  var eye = vec3(2.0, 2.0, 2.0);
8  var at = vec3(0.0, 0.0, 0.0);
9  var up = vec3(0.0, 1.0, 0.0);
10 var cameraVec = vec3(-0.57735, -0.57735, -0.57735); // 0.57735 == 1.0/Math.sqrt(3.0)
11
12 var trballMatrix = mat4(1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1);
13
14 window.onload = function init()
15 {
16     var canvas = document.getElementById("gl-canvas");
17
18     gl = WebGLUtils.setupWebGL(canvas);
19     if( !gl ) {
20         alert("WebGL isn't available!");
21     }
22
23     generateColorCube();
24     generateGround();
25
26     // virtual trackball
27     var trball = trackball(canvas.width, canvas.height);
28     var mouseDown = false;
29
30     canvas.addEventListener("mousedown", function(event) {
31         trball.start(event.clientX, event.clientY);
32
33         mouseDown = true;
34     });
35

```

```

<> view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > init > onclick
127
128 // Event listeners for buttons
129 var sinTheta = Math.sin(0.1);
130 var cosTheta = Math.cos(0.1);
131 document.getElementById("left").onclick = function () {
132     var newVecX = cosTheta*cameraVec[0] + sinTheta*cameraVec[2];
133     var newVecZ = -sinTheta*cameraVec[0] + cosTheta*cameraVec[2];
134     cameraVec[0] = newVecX;
135     cameraVec[2] = newVecZ;
136 };
137 document.getElementById("right").onclick = function () {
138     var newVecX = cosTheta*cameraVec[0] - sinTheta*cameraVec[2];
139     var newVecZ = sinTheta*cameraVec[0] + cosTheta*cameraVec[2];
140     cameraVec[0] = newVecX;
141     cameraVec[2] = newVecZ;
142 };
143 document.getElementById("up").onclick = function () {
144     var newPosX = eye[0] + 0.5 * cameraVec[0];
145     var newPosZ = eye[2] + 0.5 * cameraVec[2];
146     if (newPosX > -10 && newPosX < 10 && newPosZ > -10 && newPosZ < 10 ) {
147         eye[0] = newPosX;
148         eye[2] = newPosZ;
149     }
150 };
151 document.getElementById("down").onclick = function () {
152     var newPosX = eye[0] - 0.5 * cameraVec[0];
153     var newPosZ = eye[2] - 0.5 * cameraVec[2];
154     if (newPosX > -10 && newPosX < 10 && newPosZ > -10 && newPosZ < 10 ) {
155         eye[0] = newPosX;
156         eye[2] = newPosZ;
157     }
158 };
159
160 render();
161 };

```

```

view.html JS view.js X
C: > Users > Sun-Jeong Kim > Desktop > CG > Week09 > JS view.js > render

162
163 function render() {
164     gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
165
166     at[0] = eye[0] + cameraVec[0];
167     at[1] = eye[1] + cameraVec[1];
168     at[2] = eye[2] + cameraVec[2];
169     modelViewMatrix = lookAt(eye, at, up);
170     var modelView = mult(modelViewMatrix, trballMatrix);
171     gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelView));
172
173     // draw a color cube
174     gl.drawArrays(gl.TRIANGLES, 0, 36);
175
176     // draw the ground
177     gl.drawArrays(gl.TRIANGLES, 36, 6);
178     gl.drawArrays(gl.LINES, 42, 84);    // (21 + 21) * 2 = 84
179
180     requestAnimationFrame(render);
181 }
182
183 function generateColorCube() {
184     quad(1, 0, 3, 2);
185     quad(2, 3, 7, 6);
186     quad(3, 0, 4, 7);
187     quad(4, 5, 6, 7);
188     quad(5, 4, 0, 1);
189     quad(6, 5, 1, 2);
190 }
191
192 const vertexPos = [
193     vec4(-0.5, -0.5, -0.5, 1.0),
194     vec4( 0.5, -0.5, -0.5, 1.0),
195     vec4( 0.5,  0.5, -0.5, 1.0),
196     vec4(-0.5,  0.5, -0.5, 1.0),

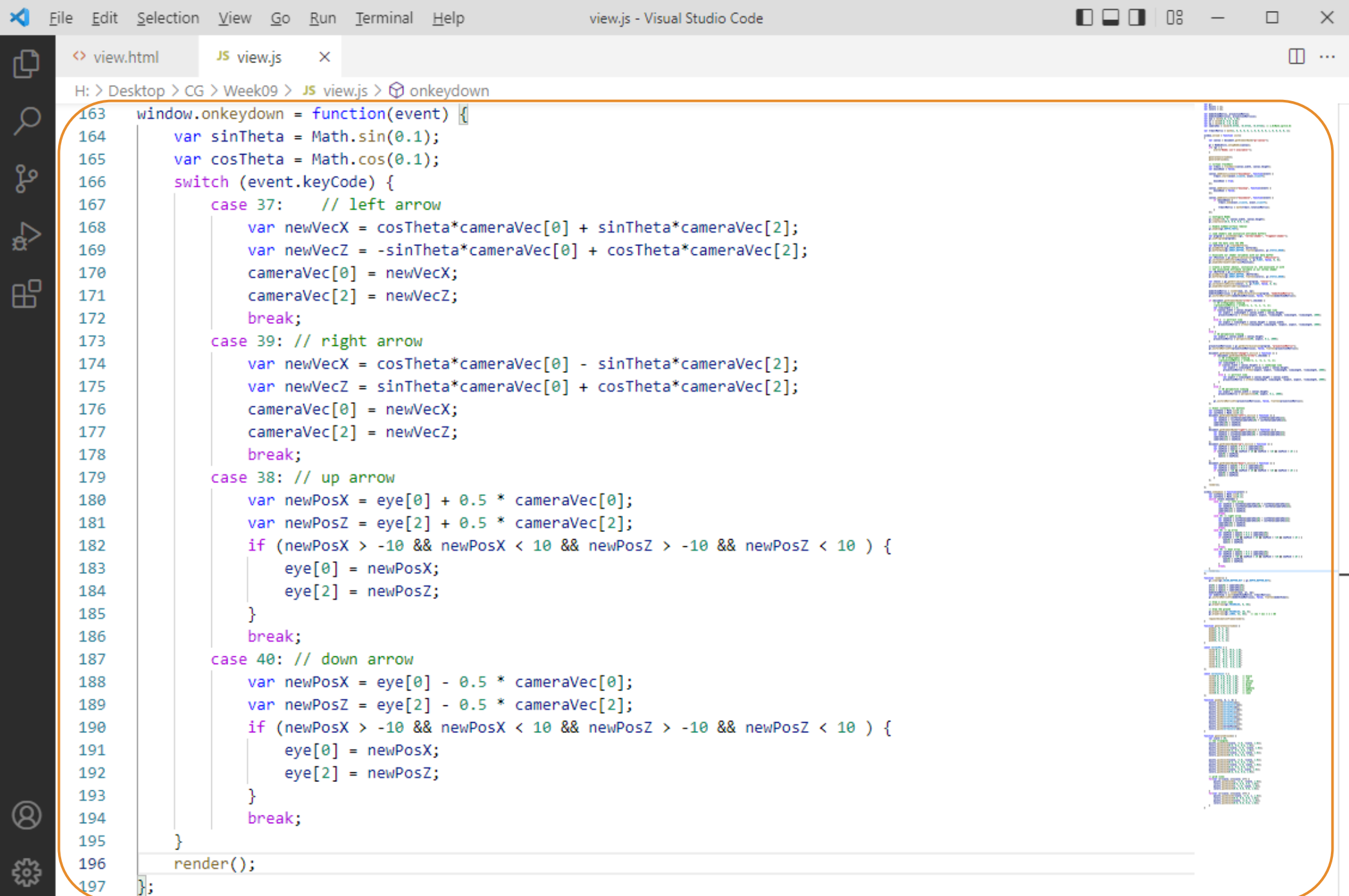
```


연습 문제 (6)

- 키보드 입력(화살표 키)을 받아 카메라를 이동시키시오.

```
window.addEventListener("keydown",  
function() {  
    switch (event.keyCode) {  
        case 49: // '1' key  
            direction = !direction;  
            break;  
        case 50: // '2' key  
            delay /= 2.0;  
            break;  
        case 51: // '3' key  
            delay *= 2.0;  
            break;  
    }  
});
```

```
window.onkeydown = function(event) {  
    var key = String.fromCharCode(event.keyCode);  
    switch (key) {  
        case '1':  
            direction = !direction;  
            break;  
        case '2':  
            delay /= 2.0;  
            break;  
        case '3':  
            delay *= 2.0;  
            break;  
    }  
};
```



```

<> view.html JS view.js x
H: > Desktop > CG > Week09 > JS view.js > onkeydown
163 window.onkeydown = function(event) {
164     var sinTheta = Math.sin(0.1);
165     var cosTheta = Math.cos(0.1);
166     var key = String.fromCharCode(event.keyCode);
167     switch (key) {
168         case 'A':
169         case 'a':
170             var newVecX = cosTheta*cameraVec[0] + sinTheta*cameraVec[2];
171             var newVecZ = -sinTheta*cameraVec[0] + cosTheta*cameraVec[2];
172             cameraVec[0] = newVecX;
173             cameraVec[2] = newVecZ;
174             break;
175         case 'D':
176         case 'd':
177             var newVecX = cosTheta*cameraVec[0] - sinTheta*cameraVec[2];
178             var newVecZ = sinTheta*cameraVec[0] + cosTheta*cameraVec[2];
179             cameraVec[0] = newVecX;
180             cameraVec[2] = newVecZ;
181             break;
182         case 'W':
183         case 'w':
184             var newPosX = eye[0] + 0.5 * cameraVec[0];
185             var newPosZ = eye[2] + 0.5 * cameraVec[2];
186             if (newPosX > -10 && newPosX < 10 && newPosZ > -10 && newPosZ < 10 ) {
187                 eye[0] = newPosX;
188                 eye[2] = newPosZ;
189             }
190             break;
191         case 'S':
192         case 's':
193             var newPosX = eye[0] - 0.5 * cameraVec[0];
194             var newPosZ = eye[2] - 0.5 * cameraVec[2];
195             if (newPosX > -10 && newPosX < 10 && newPosZ > -10 && newPosZ < 10 ) {
196                 eye[0] = newPosX;
197                 eye[2] = newPosZ;

```