**컴퓨터 그래픽스(SW) 과제 2**

**:** **Pyramid Control**



**학번: 32193430**

**이름: 이재원**

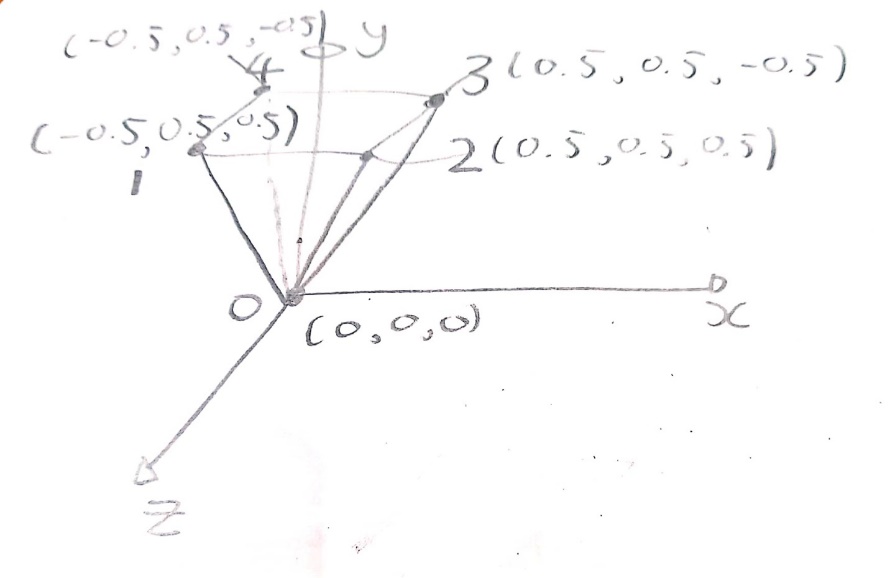
**담당교수: 송 인 식 교수님**

**분반: 2분반**

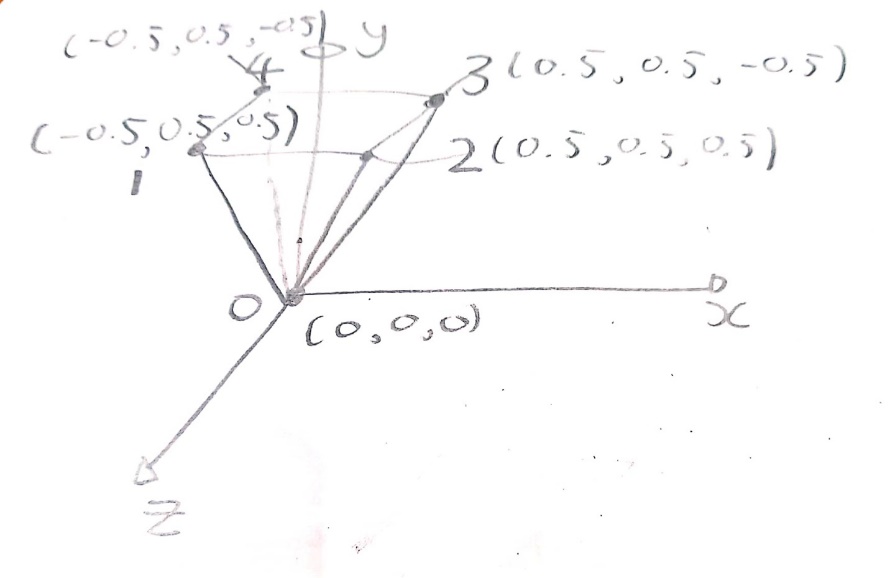
**제출일: 2021. 10. 24**

코드 편집은 Visual Studio Code 소프트웨어를 이용하였다.

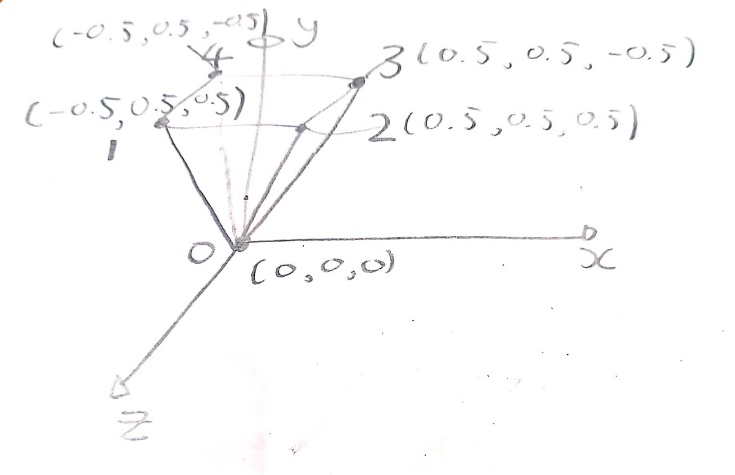
1. **기본 아이디어(피라미드 구현)**
2. **피라미드의 각 정점을 다음 그림과 같이 정한다.**

****

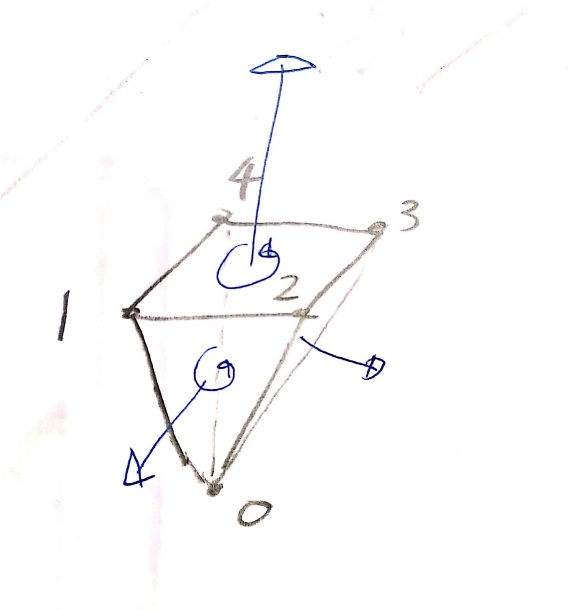
1. **정점(1,2,3,4)를 연결하여 피라미드의 밑면(사각형)을 구현한다.**

****

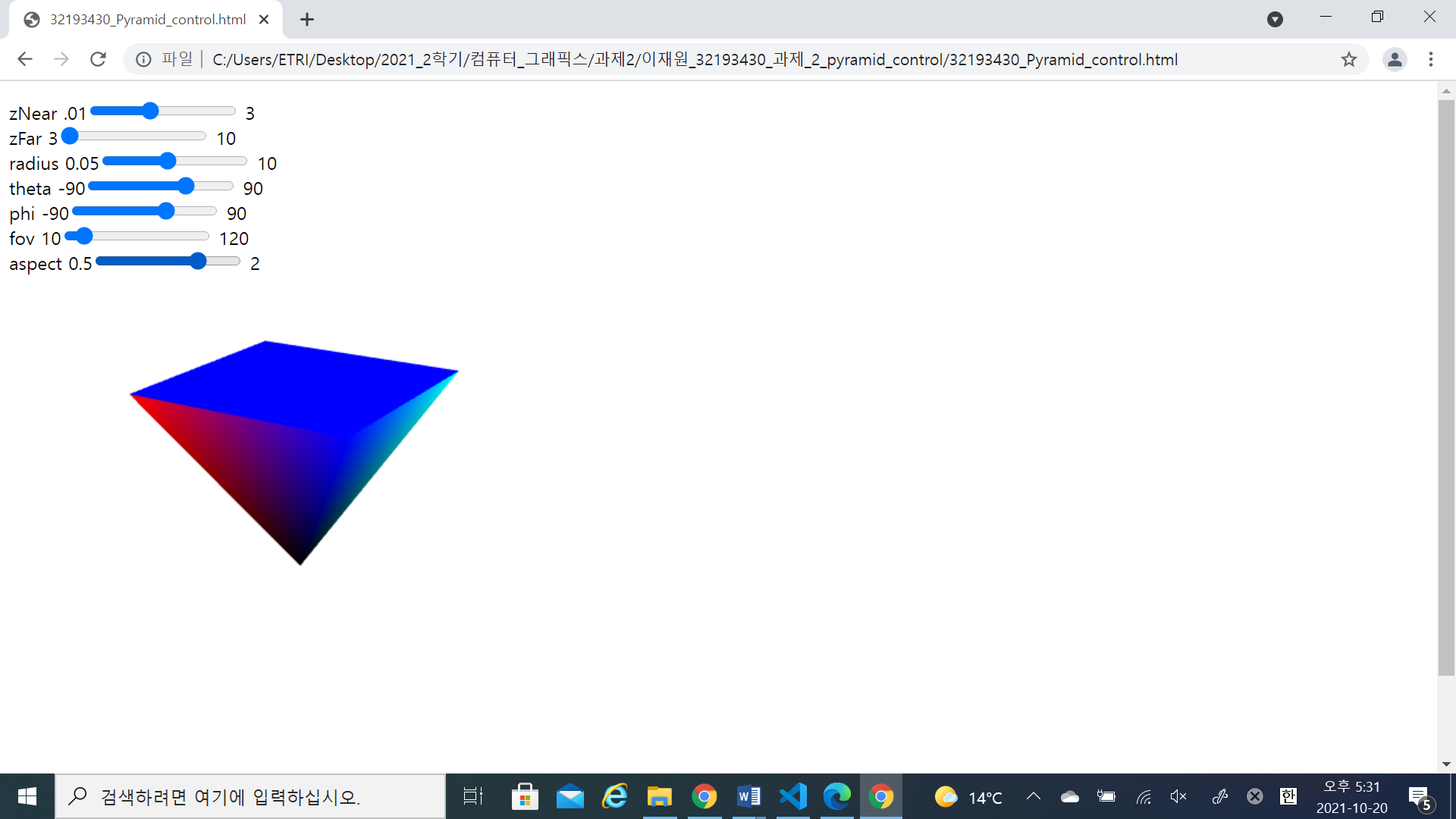
1. **정점(0,2,1), (0, 3, 2) … (0,1,4)를 연결하여 피라미드의 옆면(삼각형)을 구현한다.**

****

1. **오른손 법칙에 유의하여 따라 각 면의 바깥쪽에 색을 칠한다.**

****

1. **피라미드를 완성한다.**



1. **주요 코드 설명**

* Javascript

1. **변수선언: 필요한 정점의 수: 18개**

// pyramid는 4개의 면, 4개의 삼각형으로 사각형을 그린다

삼각형은 3개의 꼭짓점 -> 2\*3 + 3\*4 = 18

var NumVertices  = 18;

1. **변수선언: 회전, 이동, 축소/확대**

var near = 0.3;

var far = 3.0;

var radius = 4.0;

var theta  = 0.0;

var phi    = 0.0;

var dr = 5.0 \* Math.PI/180.0;

var  fovy = 45.0;  // Field-of-view in Y direction angle (in degrees)

var  aspect = 1.0;       // Viewport aspect ratio

1. **변수 선언: 카메라 위치 지정(lookAt 함수)**

var eye;

const at = vec3(0.0, 0.0, 0.0);

const up = vec3(0.0, 1.0, 0.0);

1. **Parameter를 보기 위한 slider 구현**
2. // sliders for viewing parameters
3. document.getElementById("zFarSlider").onchange = function(event) {
4. far = event.target.value;
5. };
6. document.getElementById("zNearSlider").onchange = function(event) {
7. near = event.target.value;
8. };
9. document.getElementById("radiusSlider").onchange = function(event) {
10. radius = event.target.value;
11. };
12. document.getElementById("thetaSlider").onchange = function(event) {
13. theta = event.target.value\* Math.PI/180.0;
14. };
15. document.getElementById("phiSlider").onchange = function(event) {
16. phi = event.target.value\* Math.PI/180.0;
17. };
18. document.getElementById("aspectSlider").onchange = function(event) {
19. aspect = event.target.value;
20. };
21. document.getElementById("fovSlider").onchange = function(event) {
22. fovy = event.target.value;
23. };
24. render();
25. **사각형 면 만들기(두 개의 삼각형으로 구현)**
26. // 사각형 면 만들기
27. function quad(a, b, c, d)
28. {
29. var vertices = [
30. vec4(  0.0,  0.0,  0.0, 1.0 ),
31. vec4( -0.5,  0.5,  0.5, 1.0 ),
32. vec4(  0.5,  0.5,  0.5, 1.0 ),
33. vec4(  0.5,  0.5, -0.5, 1.0 ),
34. vec4( -0.5,  0.5, -0.5, 1.0 ),


38. ];
39. // 각 면에 색칠하기
40. var vertexColors = [
41. [ 1.0, 0.0, 0.0, 1.0 ],  // red
42. [ 0.0, 0.0, 1.0, 1.0 ],  // blue
43. [ 0.0, 1.0, 1.0, 1.0 ],  // cyan
44. [ 1.0, 1.0, 1.0, 1.0 ]   // white
45. ];
46. // We need to parition(분할하다) the quad into two triangles in order for
47. // WebGL to be able to render it.  In this case, we create two
48. // triangles from the quad indices
50. //vertex color assigned by the index of the vertex
51. // 인덱스 리스트로부터 두 개의 삼각형의 위치와 데이터를 vertices 배열에 저장
52. var indices = [ a, b, c, a, c, d ];
53. for ( var i = 0; i < indices.length; ++i ) {
54. pointsArray.push( vertices[indices[i]] );
55. //colors.push( vertexColors[indices[i]] );
57. // for solid colored faces use
58. colorsArray.push(vertexColors[a]);
60. }
61. }

**6. 피라미드 밑면 그리기**

// pyramid 밑면 그리기

function colorside()

{

    quad( 1, 2, 3, 4 );

}

**7. 삼각형 만들기**

// 삼각형 만들기

function tri(a, b, c)

{

    var vertices = [

        vec4(  0.0,  0.0,  0.0, 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 ),

    ];

    var vertexColors = [

        [ 0.0, 0.0, 0.0, 1.0 ],  // black

        [ 1.0, 0.0, 0.0, 1.0 ],  // red

        [ 0.0, 0.0, 1.0, 1.0 ],  // blue

        [ 0.0, 1.0, 1.0, 1.0 ],  // cyan

        [ 1.0, 1.0, 1.0, 1.0 ]   // white

    ];

    var indices = [ a, b, c ]; //0,2,1

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

        pointsArray.push( vertices[indices[i]] ); // points = [a,b,c]

        colorsArray.push( vertexColors[indices[i]] );

        // for solid colored faces use

        // colors.push(vertexColors[a]);

    }

}

**8. 피라미드 옆면 구현하기**

//피라미드 옆면 구현하기

function colorPyramid()

{

    tri(0, 2, 1);

    tri(0, 3, 2);

    tri(0, 4, 3);

    tri(0, 1, 4);

}

**9. 피라미드 그리기**

colorPyramid();

colorside();

**10. 회전, 이동, 축소/확대를 위한 계산 및 Rendering**

// rendering

var render = function(){

    gl.clear( gl.COLOR\_BUFFER\_BIT | gl.DEPTH\_BUFFER\_BIT);

    eye = vec3(radius\*Math.sin(theta)\*Math.cos(phi),

        radius\*Math.sin(theta)\*Math.sin(phi), radius\*Math.cos(theta));

    modelViewMatrix = lookAt(eye, at , up);

    projectionMatrix = perspective(fovy, aspect, near, far);

    gl.uniformMatrix4fv( modelViewMatrixLoc, false, flatten(modelViewMatrix) );

    gl.uniformMatrix4fv( projectionMatrixLoc, false, flatten(projectionMatrix) );

    gl.drawArrays( gl.TRIANGLES, 0, NumVertices );

    requestAnimFrame(render);

}

* HTML

**사용자 인터페이스 구현**

<div>

zNear .01<input id="zNearSlider" type="range"

 min=".01" max="3" step="0.1" value="0.3" />

 3

</div>

<div>

zFar 3<input id="zFarSlider" type="range"

 min="3" max="10" step="3.0" value="3" />

 10

</div>

<div>

radius 0.05<input id="radiusSlider" type="range"

 min="0.05" max="10" step="0.1" value="4" />

 10

</div>

<div>

theta -90<input id="thetaSlider" type="range"

 min="-90" max="90" step="5" value="0" />

 90

</div>

<div>

phi -90<input id="phiSlider" type="range"

 min="-90" max="90" step="5" value="0" />

 90

</div>

<div>

fov 10<input id="fovSlider" type="range"

 min="10" max="120" step="5" value="45" />

 120

</div>

<div>

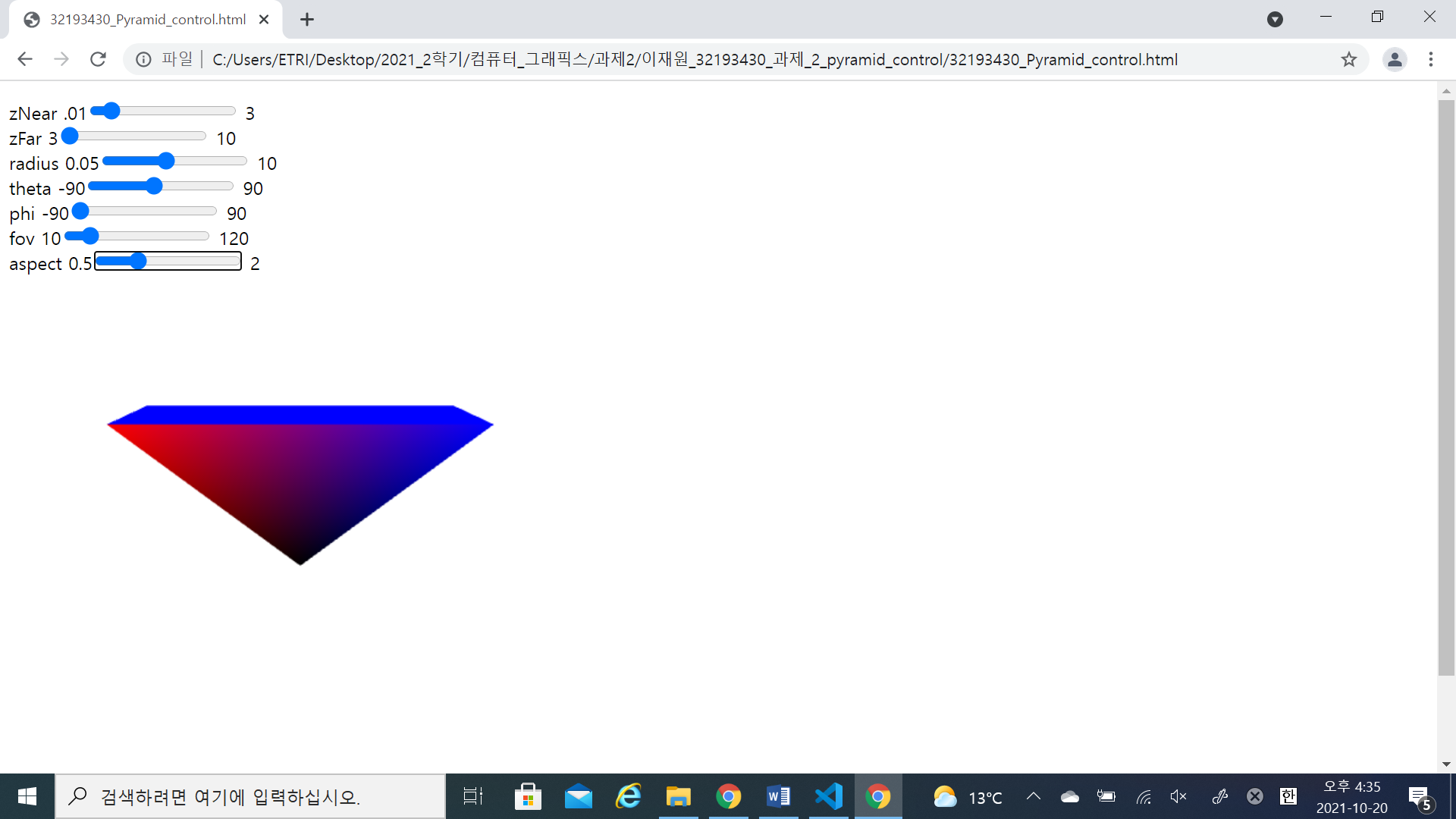
aspect 0.5<input id="aspectSlider" type="range"

 min="0.5" max="2" step="0.1" value="1" />

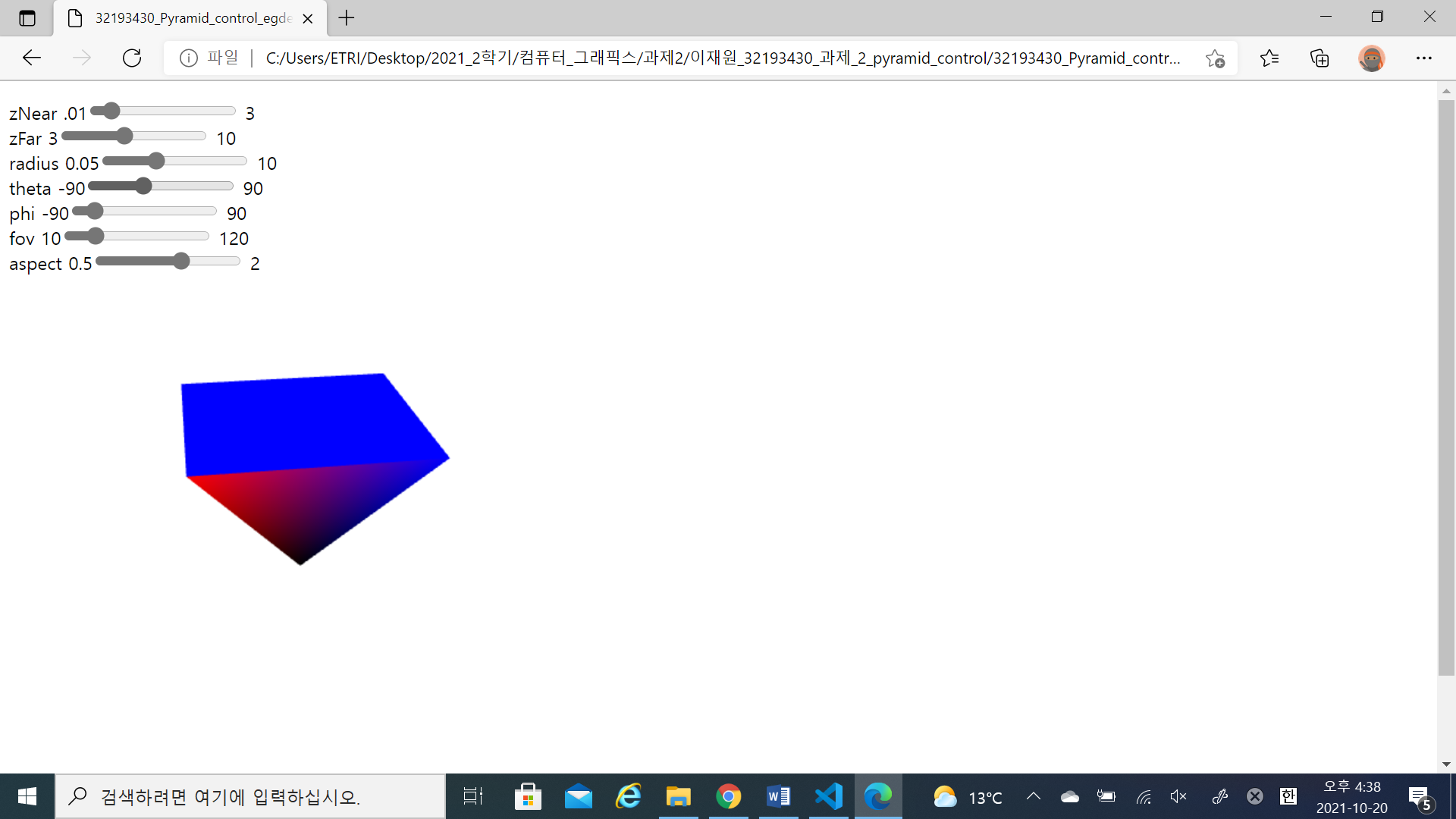
 2

</div>

1. **브라우저 테스트 결과(Chrome, Edge)**
2. **Chrome**



1. **Edge**



**참고문헌**

Edward Angel, <Interactive Computer Graphics: A Top-down Approach with WebGL 7th edition>, Pearson Education, 2015

<https://www.cs.unm.edu/~angel/BOOK/INTERACTIVE_COMPUTER_GRAPHICS/SEVENTH_EDITION/CODE/05/perspective2.html>