COMP 370 assignment #5: Projection, Lighting, and Shading in a WebGL Application

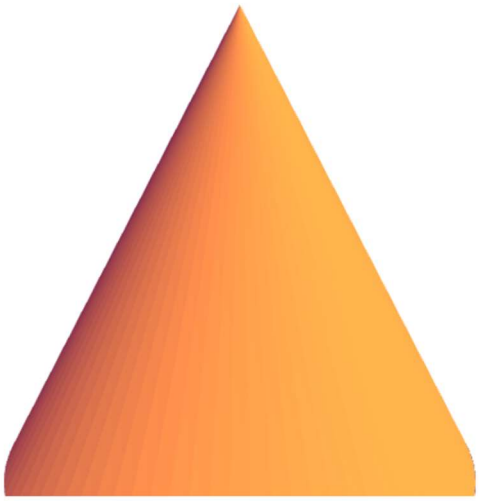
Thomas Williamson

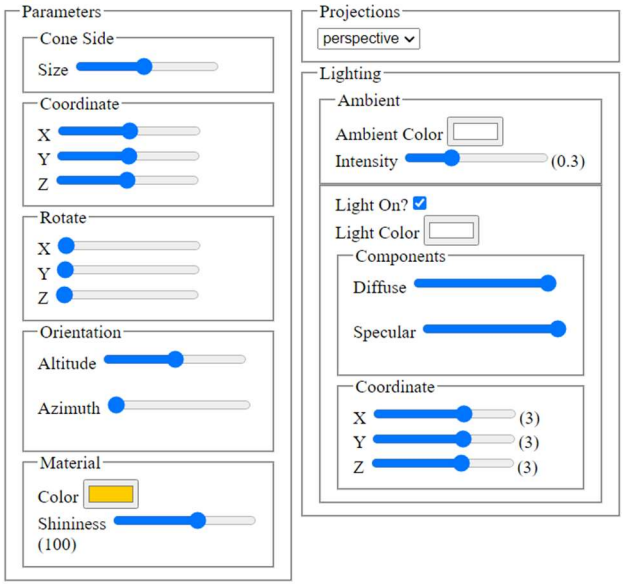
id: 588206

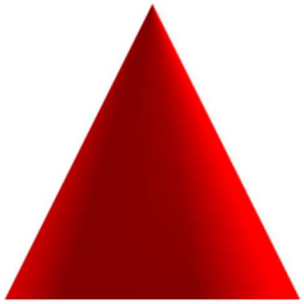
2021/11/29

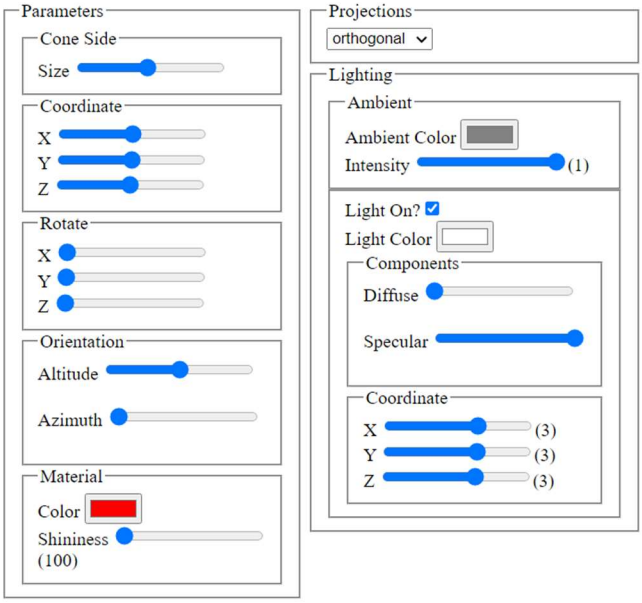
The goal of this project is to render a shape with lighting and shading and controllable sliders to change variables of the location and size of the object, perspective vs orthogonal, location of the light, color of the material, shininess of the material, and colour and intensity of the ambient light, specular light, and diffuse light.

Startup

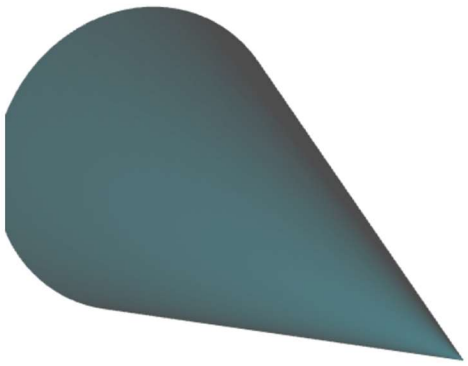


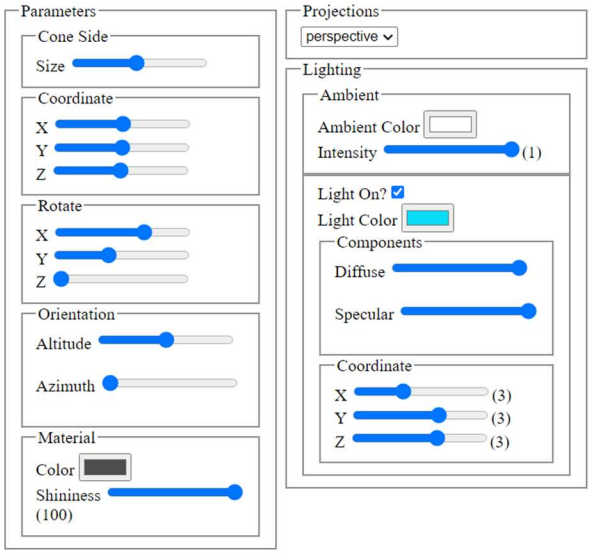







Parameters
Cone Side
Size
Coordinate
X
Y
Z
Rotate
X
Y
Z
Orientation
Altitude
Azimuth
Material
Color
Shininess
(100)
Projections
perspective
Lighting
Ambient
Ambient Color
Intensity
(1)
Light On?
Light Color
Components
Diffuse
Specular
Coordinate
X
(3)
Y
(3)
Z
(3)




Parameters
Cone Side
Size
Coordinate
X
Y
Z
Rotate
X
Y
Z
Orientation
Altitude
Azimuth
Material
Color
Shininess
(100)
Projections
perspective
Lighting
Ambient
Ambient Color
Intensity
(1)
Light On?
Light Color
Components
Diffuse
Specular
Coordinate
X
(3)
Y
(3)
Z
(3)

Code

```
<!--
COMP 370 assignment #5: Projection, Lighting, and Shading in a WebGL Application
Thomas Williamson
id: 588206
2021/11/29
-->
<html>
<body>
    <canvas id="gl-canvas" width="512" height="512" style="float: left"></canvas>


    <div style="display:inline-block; vertical-align:top">
        <form id="parameters">
            <fieldset>
                <legend>Parameters</legend>
                <fieldset>
                    <legend>Cone Side</legend>
                    <label>Size </label><input id="cube_side_size" type="range" min="0.1" max="2" step="0.1" value="1.0"><label id="cube_side_value"></label><br>
                </fieldset>
                <fieldset>
                    <legend>Coordinate</legend>
                    <label>X </label><input id="coor_x" type="range" min="-2" max="2" step="0.1" value="0.0"><label id="x_value"></label><br>
                    <label>Y </label><input id="coor_y" type="range" min="-2" max="2" step="0.1" value="0.0"><label id="y_value"></label><br>
                    <label>Z </label><input id="coor_z" type="range" min="-10" max="10" step="0.5" value="0.0"><label id="z_value"></label><br>
                </fieldset>
                 <fieldset>
                    <legend>Rotate</legend>
```

```
32                      <label>X </label><input id="rotate_x" type="
                        range" min="0" max="360" step="5"
33                      value="0.0"><label id="rotate_x"></label><br>
34                      <label>Y </label><input id="rotate_y" type="
                        range" min="0" max="360" step="5"
35                      value="0.0"><label id="rotate_y"></label><br>
36                      <label>Z </label><input id="rotate_z" type="
                        range" min="0" max="360" step="5"
37                      value="0.0"><label id="rotate_z"></label><br>
38                  </fieldset>
39                  <fieldset>
40                      <legend>Orientation</legend>
41                      <label>Altitude </label><input id="orien_al"
                        type="range" min="-90" max="90"
42                      step="1" value="0"><br><label id="al_value"></
                        label><br>
43                      <label>Azimuth </label><input id="orien_az"
                        type="range" min="0" max="359"
44                      step="1" value="0"><br><label id="az_value"></
                        label><br>
45                  </fieldset>
46                  <fieldset>
47                      <legend>Material</legend>
48                      <label for="material_color">Color </label><
                        input id="material_color" type="color"
49                      value="#ffcc00"><br>
50                      <label>Shininess </label><input id="
                        material_shn" type="range" min="1" max="16"
51                      step="1" value=10><br><label id="shn_value">(
                        100)</label><br>
52                  </fieldset>
53              </fieldset>
54          </form>
55      </div>
56
57      <div style="display:inline-block; vertical-align:top">
58          <fieldset>
59              <legend>Projections</legend>
```

```
60              <select id="select_projs">
61                   <option value="1">perspective</option>
62                  <option value="2">orthogonal</option>
63              </select>
64          </fieldset>
65          <fieldset>
66              <legend>Lighting</legend>
67              <fieldset>
68                  <legend>Ambient</legend>
69                  <label for="amb_color">Ambient Color </label><
                    input id="amb_color" type="color"
70                  value="#ffffff"><br>
71                  <label>Intensity </label><input id="amb_inten"
                    type="range" min="0" max="1" step="0.1"
72                  value="0.3"><label id="amb_inten_value">(0.3)</
                    label><br>
73              </fieldset>
74              <fieldset>
75                  <label for="light_on">Light On?</label><input id="
                    light_on" type="checkbox"
76                  checked="true"><br>
77                  <label for="light_color">Light Color </label><
                    input id="light_color" type="color"
78                  value="#ffffff"><br>
79                  <fieldset>
80                      <legend>Components</legend>
81                      <label>Diffuse </label><input id="light_dif"
                        type="range" min="0" max="1"
82                      step="0.1" value="1"> <br> <label id="
                        light_dif_value"></label> <br>
83                      <label>Specular </label><input id="light_spe"
                        type="range" min="0" max="1"
84                      step="0.1" value="1"> <br> <label id="
                        light_spe_value"></label> <br>
85                  </fieldset>
86                  <fieldset>
87                      <legend>Coordinate</legend>
88                      <label>X </label><input id="light_x" type="
```

```
                    value="3"><label id="light_x_value">(3)</label
                    ><br>
                    <label>Y </label><input id="light_y" type="
                    range" min="-10" max="10" step="1"
                    value="3"><label id="light_y_value">(3)</label
                    ><br>
                    <label>Z </label><input id="light_z" type="
                    range" min="-10" max="10" step="1"
                    value="3"><label id="light_z_value">(3)</label
                    ><br>
                </fieldset>
            </fieldset>
        </fieldset>
    </div>
    <script id="vertex-shader" type="x-shader/x-vertex">
    #version 300 es

    in  vec4 aPosition;
    in  vec3 aNormal;
    out vec4 vColor;

    uniform vec4 uAmbientProduct, uDiffuseProduct,
    uSpecularProduct;
    uniform mat4 uModelViewMatrix;
    uniform mat4 uProjectionMatrix;
    uniform vec4 uLightPosition;
    uniform float uShininess;

    void main()
    {


        vec3 pos = -(uModelViewMatrix * aPosition).xyz;

        //fixed light postion

        vec3 light = uLightPosition.xyz;
        vec3 L = normalize(light - pos);
```

```
121
122
123         vec3 E = normalize(-pos);
124         vec3 H = normalize(L + E);
125
126         vec4 NN = vec4(aNormal,0);
127
128         // Transform vertex normal into eye coordinates
129
130         vec3 N = normalize((uModelViewMatrix*NN).xyz);
131
132         // Compute terms in the illumination equation
133         vec4 ambient = uAmbientProduct;
134
135         float Kd = max(dot(L, N), 0.0);
136         vec4  diffuse = Kd*uDiffuseProduct;
137
138         float Ks = pow( max(dot(N, H), 0.0), uShininess );// (n
            dot h)^b
139         vec4  specular = Ks * uSpecularProduct; // Ks Is(n dot
            h)^b
140
141         if( dot(L, N) < 0.0 ) {
142           specular = vec4(0.0, 0.0, 0.0, 1.0);
143         }
144
145         gl_Position = uProjectionMatrix * uModelViewMatrix
            *aPosition;
146         vColor = ambient + diffuse +specular;
147
148         vColor.a = 1.0;
149     }
150     </script>
151
152     <script id="fragment-shader" type="x-shader/x-fragment">
153     #version 300 es
154
155     precision mediump float;
```

```
156
157
158         in vec4 vColor;
159         out vec4 fColor;
160
161         void
162         main()
163         {
164             fColor = vColor;
165         }
166         </script>
167
168         <script src="../Common/initShaders.js"></script>
169         <script src="../Common/MVnew.js"></script>
170         <script src="Assignment5.js"></script>
171     </body>
172     </html>
```

```
/*
COMP 370 assignment #4: Affine Transformations in a WebGL Applica
Thomas Williamson
id: 588206
2021/11/16
*/
"use strict";

var canvas;
var gl;

var axis = 0;
var xAxis = 0;
var yAxis =1;
var zAxis = 2;
var theta = [0, 0, 0];
var thetaLoc;
var transf = [0, 0, 0];
var translationLoc;
var scaile = [1, 1, 1];
var scailLoc;
var numElements = 15;


    var vertices = [
        vec3(-0.5, -0.5,  0.5),
        vec3(-0.5,  0.5,  -0.5),
        vec3(0.5,  0.5,  0.5),
        vec3(0.5, -0.5,  -0.5),
        // vec3(-0.5, -0.5, -0.5),
        // vec3(-0.5,  0.5, -0.5),
        // vec3(0.5,  0.5, -0.5),
        // vec3(0.5, -0.5, -0.5)
    ];

    var vertexColors = [
        vec4(1.0, 1.0, 1.0, 1.0),  // white
```

```
38          vec4(1.0, 0.0, 0.0, 1.0),  // red
39          vec4(0.0, 1.0, 0.0, 1.0),  // green
40          vec4(0.0, 0.0, 1.0, 1.0),  // blue
41          vec4(1.0, 1.0, 0.0, 1.0),  // yellow
42          vec4(1.0, 0.0, 1.0, 1.0),  // magenta
43          vec4(0.0, 0.0, 0.0, 1.0),  // black
44          vec4(0.0, 1.0, 1.0, 1.0)   // cyan
45      ];
46
47  // indices of the 12 triangles that compise the cube
48
49  var indices = [
50      0, 1, 2, 255,
51      0, 1, 3, 255,
52      0, 2, 3, 255,
53      3, 2, 1,
54  ];
55
56  window.onload = function init()
57  {
58      canvas = document.getElementById("gl-canvas");
59
60      gl = canvas.getContext('webgl2');
61      if (!gl) alert("WebGL 2.0 isn't available");
62
63
64      gl.viewport(0, 0, canvas.width, canvas.height);
65      gl.clearColor(210/255, 210/255, 210/255, 1.0);
66
67      gl.enable(gl.DEPTH_TEST);
68      //gl.enable(gl.PRIMITIVE_RESTART_FIXED_INDEX);
69
70      //
71      //  Load shaders and initialize attribute buffers
72      //
73      var program = initShaders(gl, "vertex-shader", "fragment-shade
74      gl.useProgram(program);
```

```
75
76      // array element buffer
77
78      var iBuffer = gl.createBuffer();
79      gl.bindBuffer(gl.ELEMENT_ARRAY_BUFFER, iBuffer);
80      gl.bufferData(gl.ELEMENT_ARRAY_BUFFER, new Uint8Array(indices
81
82      // color array atrribute buffer
83
84      var cBuffer = gl.createBuffer();
85      gl.bindBuffer(gl.ARRAY_BUFFER, cBuffer);
86      gl.bufferData(gl.ARRAY_BUFFER, flatten(vertexColors), gl.STAT
87
88      var colorLoc = gl.getAttribLocation(program, "aColor");
89      gl.vertexAttribPointer(colorLoc, 4, gl.FLOAT, false, 0, 0);
90      gl.enableVertexAttribArray(colorLoc);
91
92      // vertex array attribute buffer
93
94      var vBuffer = gl.createBuffer();
95      gl.bindBuffer(gl.ARRAY_BUFFER, vBuffer);
96      gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_D
97
98      var positionLoc = gl.getAttribLocation( program, "aPosition")
99      gl.vertexAttribPointer(positionLoc, 3, gl.FLOAT, false, 0, 0)
100     gl.enableVertexAttribArray(positionLoc );
101
102     thetaLoc = gl.getUniformLocation(program, "uTheta");
103     translationLoc = gl.getUniformLocation(program, "transform");
104     scailLoc = gl.getUniformLocation(program, "scaling");
105
106    // console.log(gl.getUniformLocation(program, ""));
107
108     //event listeners for buttons
109     document.getElementById("Test1Button").onclick = function(){
110
111         document.getElementById("RotateY").value = parseFloat(doc
```

```
112         document.getElementById("ScaleX").value = parseFloat(docu
113     };
114
115     document.getElementById("Test2Button").onclick = function(){
116         document.getElementById("ScaleX").value = parseFloat(docu
117         document.getElementById("RotateY").value = parseFloat(doc
118     };
119
120     document.getElementById("Reset").onclick =  function(){
121         document.getElementById("ScaleY").value = 1;
122         document.getElementById("ScaleX").value = 1;
123         document.getElementById("ScaleZ").value = 1;
124         document.getElementById("TranslateX").value = 0;
125         document.getElementById("TranslateY").value = 0;
126         document.getElementById("TranslateZ").value = 0;
127         document.getElementById("RotateX").value = 180;
128         document.getElementById("RotateY").value = 180;
129         document.getElementById("RotateZ").value = 180;
130     };
131
132     render();
133 }
134
135 function render()
136 {
137     gl.clear( gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
138  //   console.log(document.getElementById("RotateX").value);
139     theta[xAxis] = document.getElementById("RotateX").value;
140     theta[yAxis] = document.getElementById("RotateY").value;
141     theta[zAxis] = document.getElementById("RotateZ").value;
142     transf[0] = document.getElementById("TranslateX").value;
143     transf[1] = document.getElementById("TranslateY").value;
144     transf[2] = document.getElementById("TranslateZ").value;
145     scaile[0] = document.getElementById("ScaleX").value;
146     scaile[1] = document.getElementById("ScaleY").value;
147     scaile[2] = document.getElementById("ScaleZ").value;
148
```

```
149
150       gl.uniform3fv(thetaLoc, theta);
151       gl.uniform3fv(translationLoc, transf);
152       gl.uniform3fv(scailLoc, scaile);
153
154
155       gl.drawElements(gl.TRIANGLE_FAN, numElements, gl.UNSIGNED_BYTE
156       requestAnimationFrame(render);
157   }
158
```