Assignment #2: Simple and Interactive Animation
Shane Steiner
T00622768
2/3/2021

Problem descriptions:
In this problem we need to create an animation with colors and some effects

```
<html>

<script id="vertex-shader" type="x-shader/x-vertex">
#version 300 es

in vec4 aPosition;
in vec4 aColor;//attribute

uniform float uLRTranslation;
uniform mat3 u_matrix;

out vec4 vColor;//vertex

void main()
{
  //gl_Position.x = (aPosition.x  + uLRTranslation); //* u_matrix;
  //gl_Position.y = 0.5 +aPosition.y;
  //gl_Position.z =0.0;
  //gl_Position.w =1.0;

  vec2 xypos = vec2((aPosition.x  + uLRTranslation),0.5 +aPosition.y);
  xypos = (vec3(xypos,1) * u_matrix).xy; //mat3(-1,0,0,0,1,0,0,0,1)).xy;

  gl_Position =vec4(xypos,0,1);
  vColor = aColor;
}
</script>

<script id="fragment-shader" type="x-shader/x-fragment">
#version 300 es

precision mediump float;

in vec4 vColor;
```

```
out vec4  fColor;

void main()
{
    //fColor = vec4( 1.0, 0.0, 0.0, 1.0 );
    fColor = vColor;
}
</script>

<script type="text/javascript" src="../Common/initShaders.js"></script>
<script type="text/javascript" src="assignment2.js"></script>
<script type="text/javascript" src="../Common/MV.js"></script>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></s
cript>

<body>
    <canvas id="gl-canvas" width="512" height="512" style=" margin: 0
25%;"> </canvas>
    <div></div>
    <button id="startButton">Start</button>
    <button id="stopButton">Stop</button>
    <button id="directionButton">Change Direction</button>
    <select id="colorControls" size="5">
      <option value="0">Red</option>
      <option value="1">Green</option>
      <option value="2">Blue</option>
      <option value="3">Yellow</option>
      <option value="4">Reset</option>
  </select>
  <p id="debugP">forDebug</p>

  Speed:  Slow <input id="speedSlider" type="range"
 min="0.02" max="0.14" step="0.02" value="0.05" />
Fast
</body>

</html>
```

```
"use strict";

var gl;
var points ,points2;
var LRTranslation, matrixLocation ;
var backAndForth = 0.0;
var matrix =[-1,0,0,0,1,0,0,0,1];
var angle =0,directionMutiplyer = 1,angleIncroment=0.05,
savedAngleIncroment;
var program;
var FishPoints;
var colorArray = [];


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

    gl = canvas.getContext('webgl2');
    if (!gl) { alert( "WebGL 2.0 isn't available" ); }

    //
    //  Initialize our data for a single triangle
    //

    // First, initialize the  three points.

     points = new Float32Array([
       -0.3, -0.3 ,
          0,  0.3 ,
          0.3, -0.3,
          0.5,0.1
        ]);

        const numVerts = 100;
    var radius = 0.8
    FishPoints = [];
    FishPoints.push(vec2(-.2,0));
    for (var i = 0; i < numVerts; i++) {
        var u = i / numVerts;
```

```
        var angle = u * 3.14159 * 2.0;
        var a=0.28;//for fish curve
      //  var pos = vec2(Math.cos(angle) * radius, Math.sin(angle) *
radius);
        var pos =
vec2(a*Math.cos(angle)-((a*Math.pow(Math.sin(angle),2))/Math.sqrt(2)) ,
(a*Math.cos(angle)*Math.sin(angle)) );
        FishPoints.push(pos);
    }
    FishPoints.push(FishPoints[1]);
    //
    //  Configure WebGL
    //
    gl.viewport( 0, 0, canvas.width, canvas.height );
    gl.clearColor( 1.0, 1.0, 1.0, 1.0 );

    //  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(FishPoints), gl.STATIC_DRAW );

    // Associate out shader variables with our data buffer

    var aPosition = gl.getAttribLocation( program, "aPosition" );
    gl.vertexAttribPointer( aPosition, 2, gl.FLOAT, false, 0, 0 );
    gl.enableVertexAttribArray( aPosition );

    //for color
    //
    var colors = [];
    var numPoints = 102;
    for (var i = 0; i < numPoints; i++)colors.push(0.3, 0.01, 0.7);

    var cBuffer = gl.createBuffer();
```

```
    gl.bindBuffer(gl.ARRAY_BUFFER, cBuffer);
    gl.bufferData(gl.ARRAY_BUFFER, new Float32Array(colors),
gl.STATIC_DRAW); //new Float32Array(colors)

    var aColor = gl.getAttribLocation(program, "aColor");
    gl.vertexAttribPointer(aColor, 3, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(aColor);
    //

    //lookups
    LRTranslation = gl.getUniformLocation(program, "uLRTranslation");
    matrixLocation = gl.getUniformLocation(program, "u_matrix");



    $("#startButton").click(start);
    $("#stopButton").click(stop);
    $("#directionButton").click(chnageDirection);
    $("#speedSlider").on('input',sliderChange);
    $("#colorControls").on('input',setColor)

    render();
};

function sliderChange()
{
    //alert( typeof $("#speedSlider").val());
    angleIncroment = parseFloat($("#speedSlider").val())
*directionMutiplyer;
    savedAngleIncroment = angleIncroment;
}

//start the animation
function start()
{
    angleIncroment =savedAngleIncroment;
}
//stop the animation
function stop()
{
    if(angleIncroment != 0)
```

```
    savedAngleIncroment = angleIncroment;
    angleIncroment = 0;
}
//change the animations direction
function chnageDirection()
{
    directionMutiplyer=directionMutiplyer*-1;
    angleIncroment = angleIncroment * -1;
    savedAngleIncroment =savedAngleIncroment *-1;
}
//render function
function render() {
    gl.clear( gl.COLOR_BUFFER_BIT );

    angle = angle >=2*3.14159 ?0:angle+=angleIncroment;
    if (angle < 0) {
        angle = 0;
        chnageDirection();
    }
    else{
        matrix = angle >=1*3.14159
?[1*directionMutiplyer,0,(Math.cos(angle)/1.43),0,1,Math.sin(2*angle)/10,0
,0,1]:[-1*directionMutiplyer,0,(Math.cos(angle)/1.43),0,1,Math.sin(2*angle
)/10,0,0,1]
    }

    //update parameters
    gl.uniformMatrix3fv(matrixLocation, false, matrix);
    $("#debugP").text(angle.toFixed(2));

    //colors
    var numPoints = 102;
    var colors = [];
    for (var i = 0; i < numPoints; i++)colors.push(...colorArray);

    gl.bufferSubData(gl.ARRAY_BUFFER, FishPoints, new
Float32Array(colors));
    gl.clear(gl.COLOR_BUFFER_BIT);
    gl.drawArrays( gl.TRIANGLE_FAN, 0, 102 );
```

```
    setTimeout(
        function () {requestAnimationFrame(render);},
        5.0// speed
    );
}
function setColor()
{
    var color =  $("#colorControls option:selected").text();
    if (color == "Red") colorArray = [1, 0, 0];
    else if (color == "Green") colorArray = [0, 1, 0];
    else if (color == "Blue") colorArray = [0, 0, 1];
    else if (color == "Yellow") colorArray = [1, 1, 0];
    else if (color == "Reset") colorArray = [0.3, 0.01, 0.7];
}
```

Red
Green
Blue
Yellow
Reset

Start | Stop | Change Direction

3.35

Speed: Slow Fast

Red
Green
Blue
Yellow
Reset

Start Stop Change Direction

2.48

Speed: Slow Fast