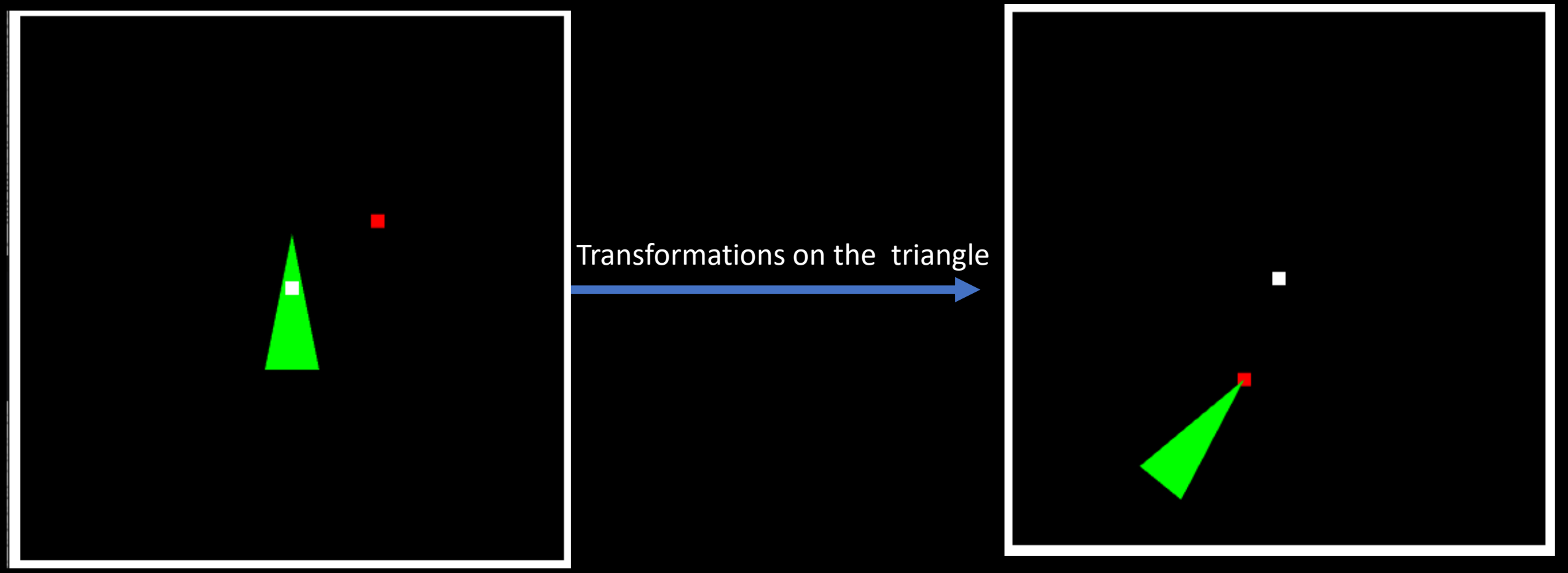
Lab 3

- Download this lab and run it, you will see the left figure.
- By adding some transformations on the triangle, make the triangle move with the red dot and also fix the tip point (triangle) to rotate around the red dot.
  - Check the video here: https://www.youtube.com/watch?v=SSud8JLqvU8&list=PLsId7efYPyAah0Z64j9DpedSVAcvzOSKb&index=2&t=0s&ab_channel=Ko-ChihWangKo-ChihWang



- In WebGL.js, I set vertices information here (check the comment), do NOT change any information here

```
var transformMat = new Matrix4(); //cuon 4x4 matrix

//NOTE: You are NOT allowed to change the vertex information here
var centerPointLoc = [0.0, 0.0, 0.0]; //center white point location
var centerPointColor = [1.0, 1.0, 1.0 ]; //center white point color

//NOTE: You are NOT allowed to change the vertex information here
var rotatingPointLoc = [0.0, 0.0, 0.0]; //rotating red point location
var rotatingPointColor = [1.0, 0.0, 0.0 ]; //rotating red point color

//NOTE: You are NOT allowed to change the vertex information here
var triangleVertices = [0.0, 0.2, 0.0, -0.1, -0.3, 0.0, 0.1, -0.3, 0.0]; //green rotating triangle vertices
var triangleColor = [0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0 ]; //green trotating riangle color

var pointAngle = 0;  //angle for rotatin red point
var triangleAngle = 0; //angle for ratating green triangle
```

- Also, do not change any code here

- The only thing you can do is to add some transformations here

- **Firstly, try to understand how I make the red point rotate.**

```
function draw(gl)
{
    ////clear background color by black
    gl.clearColor(0.0, 0.0, 0.0, 1.0);
    gl.clear(gl.COLOR_BUFFER_BIT);

    ////////////// Begin: draw the rotating green triangle
    transformMat.setIdentity(); //set identity matrix to transformMat
    ///******TODO: you can multiple transformMat.translate() and transformMat.rotate() to make the rotating grenn triangle


    //Note: You are NOT Allowed to change the following code
    initAttributeVariable(gl, program.a_Position, triangle.vertexBuffer);//set triangle  vertex to shader varibale
    initAttributeVariable(gl, program.a_Color, triangle.colorBuffer); //set triangle  color to shader varibale
    gl.uniformMatrix4fv(program.u_modelMatrix, false, transformMat.elements);//pass current transformMat to shader
    gl.drawArrays(gl.TRIANGLES, 0, triangle.numVertices);//draw the triangle
    ////////////// END: draw the rotating green triangle

    //// Begin: draw the center white point
    transformMat.setIdentity(); //set identity matrix
    initAttributeVariable(gl, program.a_Position, centerPoint.vertexBuffer); //set center point vertex to shader varibale
    initAttributeVariable(gl, program.a_Color, centerPoint.colorBuffer); //set center point color into shader varibale
    gl.uniformMatrix4fv(program.u_modelMatrix, false, transformMat.elements); //pass current transformMat to shader
    gl.drawArrays(gl.POINTS, 0, centerPoint.numVertices); //draw the center white point
    //// END: draw the center white point

    ////////*******Suggestion: read the following code and understand what's going on here (about the transofmation)
    //// Begin: draw the rotating red point
    pointAngle++; //rotating angle of the red point
    transformMat.setIdentity(); //set identity matrix to transformMat
    transformMat.rotate(pointAngle, 0, 0, 1);
    transformMat.translate(0, 0.4, 0);
    initAttributeVariable(gl, program.a_Position, rotatingPoint.vertexBuffer); //set rotating point vertex to shader varibale
    initAttributeVariable(gl, program.a_Color, rotatingPoint.colorBuffer); //set rotating point color to shader varibale
    gl.uniformMatrix4fv(program.u_modelMatrix, false, transformMat.elements); //pass current transformMat to shader
    gl.drawArrays(gl.POINTS, 0, rotatingPoint.numVertices); //draw the rotating red point
    //// END: draw the rotating red point

}
```

```
function main(){
    //////Get the canvas context
    var canvas = document.getElementById('webgl');
    var gl = canvas.getContext('webgl2');
    if(!gl){
        console.log('Failed to get the rendering context for WebGL');
        return ;
    }

    /////compile shader and use it
    program = compileShader(gl, VSHADER_SOURCE, FSHADER_SOURCE);
    gl.useProgram(program);

    /////prepare attribute reference of the shader
    program.a_Position = gl.getAttribLocation(program, 'a_Position');
    program.a_Color = gl.getAttribLocation(program, 'a_Color');
    program.u_modelMatrix = gl.getUniformLocation(program, 'u_modelMatrix');
    if(program.a_Position<0 || program.a_Color<0 || program.u_modelMatrix < 0)console.log('Error: f(program.a_Position<0 || program.a_Color<0 || .....');

    /////create vertex buffer of rotating point, center points, rotating triangle for later use
    centerPoint = initVertexBufferForLaterUse(gl, centerPointLoc, centerPointColor);
    rotatingPoint = initVertexBufferForLaterUse(gl, rotatingPointLoc, rotatingPointColor);
    triangle = initVertexBufferForLaterUse(gl, triangleVertices, triangleColor);

    ////For creating animation, in short this code segment will keep calling "draw(gl)"
    ////btw, this needs "webgl-util.js" in the folder (we include it in index.html)
    var tick = function() {
        draw(gl);
        requestAnimationFrame(tick);
    }
    tick();

}
```

- You may be confused about how I create and use the vertex buffer object. I will explain that next week. Just assume everything is alright. Just take care of the transformations in draw().

- This is how we make the animation. In short, the function, draw(), is called repeatedly

What You Should Do for “Submission”

Submission Instruction

- Create a folder
  - Put the html and js files in the folder
  - Zip the folder
  - Rename the zip file to your student ID
    - For example, if your student ID is “40312345s”, rename the zip file to “40312345s.zip”
  - Submit the renamed zip file to Moodle

- Make sure
  - you put all files in the folder to zip
  - You submit the zip file with correct name
- You won’t get any point if
  - the submitted file does not follow the naming rule,
  - TA cannot run your code,
  - or cannot unzip your zip file.