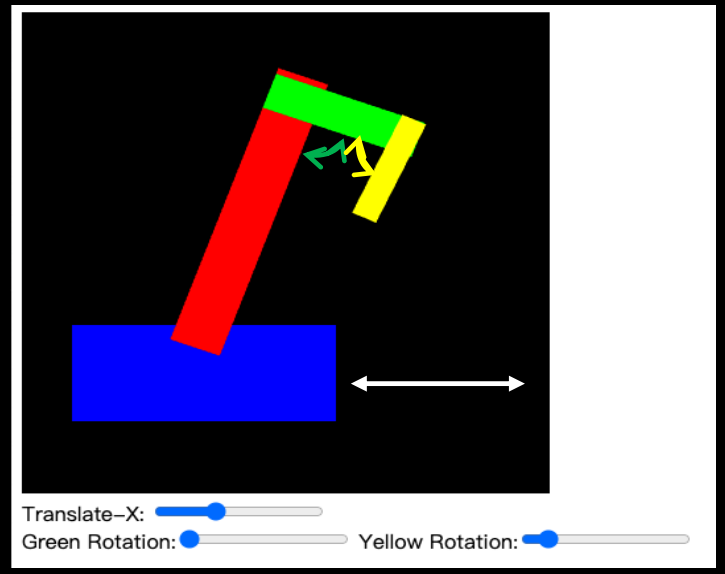
Lab 4

- Download the lab4 template, you can see this figure
- Goal
  - If you move the ”translate-x” slider, the robot will move left or right
  - If you move the ”green rotation” slider, the green arm will rotate
  - If you move the “yellow rotation” slider, the yellow arm will rotate
- Or you can check this video
  - https://www.youtube.com/watch?v=mJn6BklPGmM&feature=youtu.be&ab_channel=Ko-ChihWang



Index.html

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>WebPage Title </title>
    </head>

    <body onload="main()">
        <canvas id="webgl" width = "400" height = "400">
            Please use a browser that support "canvas"
        </canvas>
        <script src="cuon-matrix.js"></script>
        <script src="WebGL.js"></script>
        <div class="slidecontainer0">
            Translate-X: <input type="range" min="-100" max="100" value="0" class="slider" id="Translate-X">
        </div>
        <div class="slidecontainer1">
            Red Rotation:<input type="range" min="0" max="45" value="0" class="slider" id="jointForRed">
            Green Rotation:<input type="range" min="0" max="45" value="0" class="slider" id="jointForGreen">
            Yellow Rotation:<input type="range" min="0" max="45" value="0" class="slider" id="jointForYellow">
        </div>
    </body>
</html>
```

Min, max and initial value

sliders

WebGL.js – main()

```
//variables for tx, red,green and yellow arms angle
var tx = 0;
var redAngle = 0;
var greenAngle = 0;
var yellowAngle = 0;

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 ;
    }

    program = compileShader(gl, VSHADER_SOURCE, FSHADER_SOURCE);
    redraw(gl); //call redarw here to show the initial image

    //setup the call back function of tx Sliders
    var txSlider = document.getElementById("Translate-X");
    txSlider.oninput = function() {
        tx = this.value / 100.0; //convert sliders value to -1 to +1
        redraw(gl);
    }

    //setup the call back function of red arm rotation Sliders
    var jointRedSlider = document.getElementById("jointForRed");
    jointRedSlider.oninput = function() {
        redAngle = this.value;
        redraw(gl);
    }

    //setup the call back function of green arm rotation Sliders
    var jointGreenSlider = document.getElementById("jointForGreen");
    jointGreenSlider.oninput = function() {
        greenAngle = this.value; //convert sliders value to 0 to 45 degrees
        redraw(gl);
    }

    //setup the call back function of yellow arm rotation Sliders
    var jointYellowSlider = document.getElementById("jointForYellow");
    jointYellowSlider.oninput = function() {
        yellowAngle = this.value *  -1; //convert sliders value to 0 to -45 degrees
        redraw(gl);
    }
}
```

Store the input from sliders

Function to draw a frame

Register the call back function of sliders

WebGL.js – redraw()

```
function redraw(gl)
{
  gl.clearColor(0.0, 0.0, 0.0, 1.0);
  gl.clear(gl.COLOR_BUFFER_BIT);

  gl.useProgram(program);
  u_modelMatrix = gl.getUniformLocation(gl.getParameter(gl.CURRENT_PROGRAM), 'u_modelMatrix');

  rectVertices = [-0.5, 0.5, 0.5, 0.5, -0.5, -0.5, 0.5, -0.5];
  var redColor = [1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0 ];
  var greenColor = [0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0 ];
  var blueColor = [0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0 ];
  var yellowColor = [1.0, 1.0, 0.0, 1.0, 1.0, 0.0, 1.0, 1.0, 0.0, 1.0, 1.0, 0.0 ];
  buffer0 = initArrayBuffer(gl, new Float32Array(rectVertices), 2, gl.FLOAT, 'a_Position');
  buffer1 = initArrayBuffer(gl, new Float32Array(blueColor), 3, gl.FLOAT, 'a_Color');

  transformMat.setIdentity();
  //TODO-1: translate whole robot here
  transformMat.translate(0.0, -0.5, 0.0);
  pushMatrix();
  transformMat.scale(1.0, 0.4, 0.0);
  gl.uniformMatrix4fv(u_modelMatrix, false, transformMat.elements);
  gl.drawArrays(gl.TRIANGLE_STRIP, 0, rectVertices.length/2);//draw the blue one

  popMatrix();
  buffer1 = initArrayBuffer(gl, new Float32Array(redColor), 3, gl.FLOAT, 'a_Color');
  //TODO-2: make the red arm rotate
  transformMat.translate(0.0, 0.2, 0.0);
  transformMat.rotate(-20, 0.0, 0.0, 1.0);
  transformMat.translate(0.0, 0.5, 0.0);
  pushMatrix();
  transformMat.scale(0.2, 1.2, 0.0);
  gl.uniformMatrix4fv(u_modelMatrix, false, transformMat.elements);
  gl.drawArrays(gl.TRIANGLE_STRIP, 0, rectVertices.length/2);//draw the red one

  popMatrix();
  buffer1 = initArrayBuffer(gl, new Float32Array(greenColor), 3, gl.FLOAT, 'a_Color');
  //TODO-3: you may add some functions here
  //        and modify translate() in next line to rotate the green bar
  transformMat.translate(0.2, 0.5, 0.0);
  pushMatrix(); //for one more yellow
  transformMat.scale(0.6, 0.15, 0.0);
  gl.uniformMatrix4fv(u_modelMatrix, false, transformMat.elements);
  gl.drawArrays(gl.TRIANGLE_STRIP, 0, rectVertices.length/2);//draw the green one

  //TODO-4: add code here to draw and rotate the yelloe block
}
```

Check these TODO

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.