CS112 Programming Assignment 1

1. Introduction

In this assignment, we will get familiar with the transformation which are fundamental to creating any scene, using WebGL without the use of plug-ins. In this assignment, you would need to write some code (maybe dozens of lines) and it will take some time to understand every part, so **please start early**.

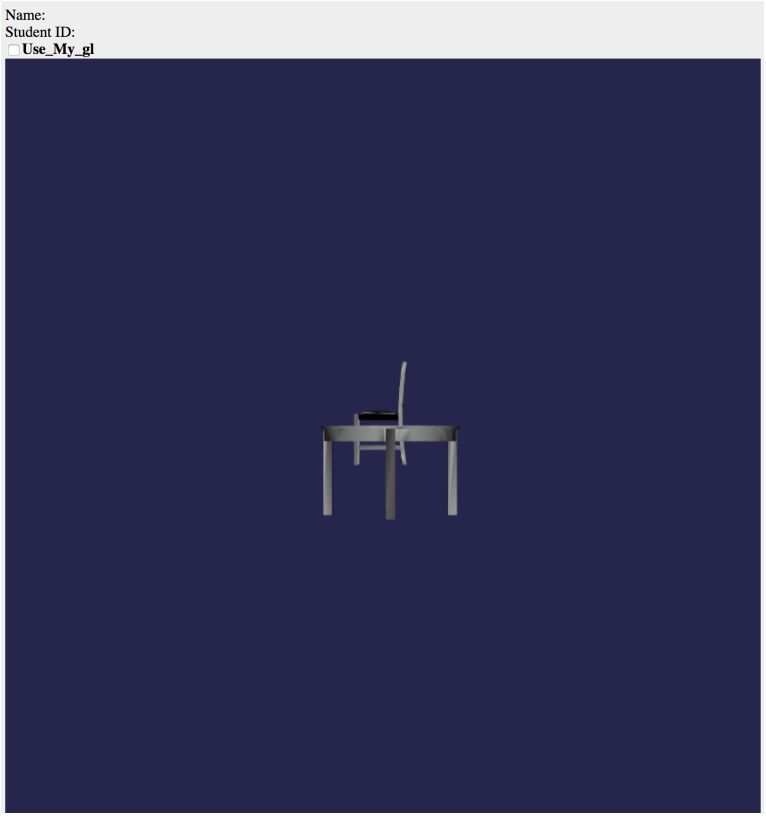
Software and hardware requirement: WebGL runs within the browser, so is independent of the operating and window systems. You may finish the assignment using any operating system you like, e.g. Windows, OSX or Linux. **Programming language**: The assignment will be implemented in JavaScript. As we will minimize the use of obscure Javascript language features, it should not be too difficult for anyone with experience in a dynamic language like Python or familiar with the principles of Object Oriented Programming (like C++ or Java) to get a handle on Javascript syntax by reading through some of the code in code skeleton. For a more formal introduction to Javascript, checkout the nice tutorial from https://javascript.info/.

Cooperation and third-party code: This is an individual programming assignment, and you should implement the code **by your own**. You may not share final solutions, and you must write up your own work, expressing it in your own words/notation. Third party codes are not allowed unless with professor's permission.

2. Getting started with the code skeleton.

1) Download pa1.zip and extract it any folder you like. You should have six files in the folder:
    a. pa1.html : html file which shows the WebGL canvas, you don't need to change this file.
    b. pa1.js : the actual functions for this assignment.
    c. gl-matrix-min.js : utility functions for operating matrices
    d. webgl-utils.js : utility functions for WebGL animation
    e. models.js : file describing the models in IFS format.
    f. trackball-rotator.js : utility functions for rotating the scene using the cursor.

2) Open pa1.html in the extracted folder with Chrome. In the given code skeleton, a scene is drawn with 4 chairs (which are overlapping with each other as of now) and a table as shown below.



3) Note, **for this project, you will be modifying only two files – models.js and pa1.js**. But try to read and understand what other functions are doing as well as it would help you better implement things. There are lots of comments throughout the codebase and the functions have similar structure as programming assignment 0 for better/easy understanding.

   There are two tasks which need to be completed in this programming assignment. They are as follows.

3. Model a scene of a dining room

TASK 1: Using the **inbuilt matrix transformations** (such as mat4.scale(), mat4.translate() and mat4.rotate()) provided by gl-matrix-min.js, model a scene of a typical dining room. Check the references provided below on their implementation.

1) First build the center-piece (i.e. a cube) by determining the coordinates, normals for each of the 8 vertices and also its faces. Use a scratch paper if you need to, similar to what you did in the PA 0. Note, you would need to compute the normals for the vertices of the cube which are different than normals of the faces. Once you have computed all values, complete the function *cube()* in model.js. (Refer to models.js to see how other models are represented)

2) Complete the function *draw()* in pa1.js by computing the transformation for each of the six objects (table,4 chairs, box) where you have a table in the center surrounded by four chairs and a center piece on the table. Complete your functions i.e.

   i. *perspective(input parameters)*
   ii. *translation(input parameters)*
   iii. *scaling(input parameters)*
   iv. *rotation(input parameters)*

   and use them to complete the function *draw()*.Think what input and output parameters you would need for each of these function. The description of these have been provided in the file. Note, the *rotate()* function rotates the model about an arbitrary axis, hence one of the input would be the axis vector. Note you will still be calling all the 4 transformation functions in *draw()*).

   Things to keep in mind:

   - Appropriately scale all the objects such that the scene looks more realistic.
   - All the four chairs should be properly oriented. i.e. facing the table.
   - The box should be much smaller and placed approximately on the center of the table.
   - Try to avoid any intersections between the models.

TASK 2: Using the task 1 as reference, now model the **same** scene which you built above but this time you **CANNOT** use any of the inbuilt matrix transformations.

1) Complete the functions *perspective(), translate(), rotate() and scale()* in pa1.js such that when the user clicks "Use_My_gl", the scene should be using your hand coded transformations and when the checkbox is unchecked, it should be using the inbuilt transformation. See below image for example. (Feel free to use your own variable names)

```
function translate(inputmodel,trans_vector){

    if (document.getElementById("my_gl").checked) {
        /*
        TODO: Your code goes here.
        Write the code to perform translation transformation.
        Think about what would be the input and output to the function would be
        */
    }
    else {

        mat4.translate(inputmodel,inputmodel,trans_vector);
        return inputmodel;
    }
}
```

2) Feel free to write helper functions to perform matrix-matrix multiplication, matrix-vector multiplication etc. if you have to.

4. Submission

1) Make sure your name and ID is filled above the canvas, in pa1.html.
2) You will need to submit the following files on EEE in **a zip archive**, please DO NOT submit individual files.
    a. pa1.html
    b. p1.js
    c. models.js
    d. gl-matrix-min.js
    e. trackball-rotator.js
    f. webgl-utils.js

5. Grading

1) Your program should be able to place and render all the six objects (1 table, 4 chairs and 1 box) in their correct positions as described above.
2) When the user checks the checkbox "Use_My_gl", the scene should be rendered using your own transformations and when its unchecked it should be using the inbuilt transformations. Note, the scene should remain the same whether the checkbox is checked or unchecked.

6. Useful references

1) WebGL tutorial:
   http://learningwebgl.com/blog/?page_id=1217
2) JavaScript
   https://developer.mozilla.org/en-US/docs/Web/JavaScript
3) JS style guide:
   https://google.github.io/styleguide/javascriptguide.xml?showone=Comments#Comments
4) glMatrix Documentation:
   http://glmatrix.net/