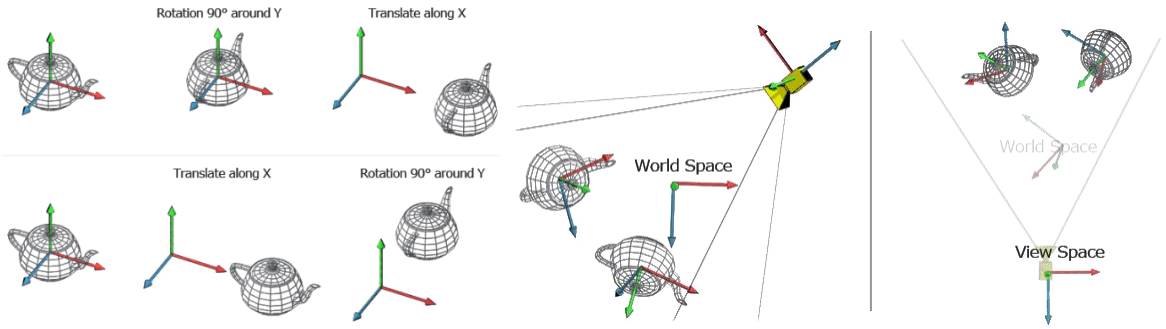
Exercise #4

The purpose of this lab is to introduce you to working with transformations in OpenGL, and to get you familiar with the model/view, and projection, stages of the vertex pipeline.



1. This assignment is strictly **individual** (no groupwork).

2. You are required to create a transformation matrix to move an object around the scene. Your code must use matrices to calculate the transformations and perspective projection. You are then required to create a 3D scene and a camera to allow you to navigate the scene. Details provided below.
3. Do not wait until the last minute to start this lab. This assignment should be straightforward, but it may not go as smoothly as you would like. Assume something unexpected will happen and give yourself time to deal with it. Be sure to attend online labs and ask the demonstrators for help.

4. You can download the sample scene on Blackboard from Lab #3 which should provide you with a good starting point. Note: the shaders have been externalised so you need to put them (simpleVertexShader.txt and simpleFragmentShader.txt) in a "Shader" folder in your project.

5. This week, you will be required to submit via Blackboard on 22nd October:
    a. Evidence that you completed/attempted the lab (screenshot of result, video, screenshot of error message you got with short description of what went wrong, etc.). This will **not be graded**, but it will give us an indication of where everybody is at currently. If you do not submit anything, we will be contacting you to know why. Please do not request an extension, as this is not a graded assignment.

6. Note: The next lab, Exercise #5 will be a mid-term report and video of your progress to date and will be graded, so please make sure that you are up to date on your exercises.

- Your program should have the following features:
    - Keyboard control
        a. Keypress to show: Rotation around the x- y- and z-axis
        b. Keypress to show: Translation in the x- y- and z- direction

    c. Keypress to show: Uniform and non-uniform Scaling
    d. Keypress to show: Combined Transformations
    e. Keypress to show: switching from orthographic to perspective projection
- Virtual Camera: Create an interactive walkthrough that allows the user to move around the scene using the keyboard and/or the mouse. At a minimum, implement moving forwards and backwards, turning left and turning right. Extra marks will be given for extra functionality (i.e., additional degrees of freedom of rotation of the camera).

Common mistakes:

- ***My shape disappeared!*** - check how far you are moving it; maximum -1 to 1 on x and y. Also check your matrix is being sent to the shader - all uniform variables are zero by default.
- ***Shader compilation error!*** - Trying to multiply a `vec3` by a `mat4`. Recast like this for a point: `vec4 result = matrix * vec4 (input_vector, 1.0);`
- ***My transformation is inside-out!*** - Row and column-order multiplication are mixed up. Check the order of multiplication, and compare to the layout of your matrix.
- ***My translation isn't doing anything!*** - `vec4` point has a 0 for the last component, when it should have a 1.
- ***My translation still isn't doing anything!*** - check that your matrix uniform location is correct, and is being sent with `glUniform`.

Notes

- You will need to refer to the documentation of your supporting library e.g. GLFW or FreeGLUT to find the functions and call-backs to use for keyboard input.
- Make sure that you have implemented error checks. Are you checking for shader compilation and linking errors? Do you print the logs in this case? Are you checking that the result of glGetUniformLocation() is not negative?
- Remember to read the OpenGL 4 man-pages for every function that you use from OpenGL - know what the parameters do, and what other functions they depend on. The http://docs.gl/project is a very much improved presentation of the man-pages, and has examples.
- This assignment is where you may first encounter 3d geometry chaos and lose track of what you are computing. Use your pencil and paper to draw the scene. Know what the numbers should be. Use your calculator to hand calculate the transformations for the first vertex. Compare this to the computer's result to help diagnose any problems. You should be able to hand-calculate a matrix multiplication - use a cheat sheet to help you if you are unsure. I added print() functions to my maths code to help debug calculation problems. Think about the efficiency of your code - what calculations should be done in C, and what should be computed in shaders?