

## Assignment #2 – 3D Models and Viewing

**DUE DATE: Thursday March 13<sup>th</sup> (4 weeks)**

In this assignment you will build your own 3D scene. You will: (1) use VBOs and vertex attributes for vertices and texture coordinates, (2) build matrix transforms sent in uniforms, (3) use texture images, texture units, and tiling, (4) build 3D models, and (5) build a simple camera controller for viewing a 3D world from different vantage points.

### World Objects

Your scene must contain at least four (4) different objects, and also satisfy the following:

- at least one object that is fixed at or near the origin
- at least one moving object
- at least one rotating object (that doesn't count as the moving object)
- at least one model designed by you, by hand, including the vertices and texture coordinates
- at least one model must be read in from an .OBJ file
- a single set of lines colored red/green/blue showing the world XYZ axes (these don't count towards the 4 required objects)

Each of the objects should be initially defined in its own local coordinate system, and placed in the world with a *Model* matrix.

The moving object(s) should have a reasonably interesting trajectory, and should be viewable (i.e., not fly out of range of view). Movement should be based on elapsed time.

The rotating object(s) should have rotation that is clearly visible, either due to the shape of the object or its texture.

The model designed by you can be simple, but not copied from the book. It may be procedural, or made of explicit vertices (not exported from a tool like Blender or Maya).

It is **not** a requirement to deal with collisions between objects; you may allow moving objects to “pass through” each other, although your scene will look better if they don't.

### Textures

Your scene must utilize at least four (4) texture images, and also satisfy the following:

- at least one texture must be created by you (such as with a paint tool)
- at least one must be from the book's provided textures
- at least one must come from another source (such as the web)
- at least two must clearly reveal and align with the object geometry (not random patterns)

At least one of your models must also utilize texture tiling, where the texture coordinates in some areas of the model fall outside the range (0...1), with tiling making use of that.

You must clearly attribute the source(s) of every texture and .OBJ model used in your project, and provide evidence that you have permission to use it. Do this as follows:

- If you created the texture or model yourself, or if you got it from the textbook, just state that in your readme document.
- If the model or texture is from the web, provide evidence that you have permission to use it from the copyright holder (e.g., an email). Or, provide a link that leads **directly** to a page showing that the asset is public domain. Or, provide a **direct** link to the terms of use, and **briefly describe how you complied with that.**

## Camera Control

See the provided supplemental notes for implementing this. Your program must define a *camera* object that has variables for location and UVN vectors corresponding to its orientation axes. These values are manipulated as follows:

w / s – move the camera forward / backward a small amount (i.e. in the positive / negative N direction).  
a / d – move the camera left / right a small amount (i.e., in the negative / positive U direction)  
q / e – move the camera up / down a small amount (i.e., in the positive / negative V direction).  
← and → (left and right arrow) – rotate the camera by a small amount left/right around its V axis (“pan”).  
↑ and ↓ (up and down arrow) – rotate the camera by a small amount up/down around its U axis (“pitch”).  
<space bar> - toggle the visibility of the world axes.

## Additional Notes

- The objects should all be visible on screen at the same time. They shouldn’t be so small, or so distant, or rotating so fast, that they or their textures are hard to see.
- Use `KeyListener` methods to handle the keyboard commands. The listener methods should update the camera’s location or orientation as described above. Camera movement does not need to be based on elapsed time (but object movement does).
- Your `display()` method will need to build the appropriate *perspective* and *model* transforms, and also get the camera state to build the *view* matrix.
- The lines showing the positive X, Y, and Z world axes should (when enabled) be colored RGB corresponding in order to XYZ. See the provided tech tip for more info.
- Use only *relative paths* to access the texture image, model, and shader files.
- Your code must be in a Java package named “a2” (lower case). The “main” class be named exactly “code”. Compiling and running are otherwise the same as for A1.
- You should use the “Imported Model” and “Utils” classes from the textbook, and you may also use any other code provided with the textbook.
- Your scene must be coherent and logical, and shouldn’t just duplicate a scene from the textbook. You are expected to *apply* the methods, not just copy them.

## Deliverables

- This is an INDIVIDUAL assignment. You may use models / textures from the web as described above. You may use code from the textbook, but you may not use code obtained from the web or elsewhere.
- Submit to Canvas ONE item – a ZIP folder with your A2 submission, that contains the following:
  - a .PDF report as described below
  - `compile.bat` and `run.bat` files that work with your program.
  - Texture image files and `.OBJ` model file(s) that work with your program
  - A subfolder “a2” that contains:
    - Java source (.java) and compiled (.class) files
    - GLSL vertex and fragment shader files.
- Your .PDF report file must consist of the following seven numbered items (number them!!):
  1. your name, “assignment 2”, CSc-155, section number, and “Spring 2025”.
  2. a screenshot of your running program
  3. a list of which requirements you were NOT able to get fully working
  4. a description of the object that you created yourself by hand
  5. a description of which object(s) is moving, which object(s) is rotating, and which use tiling
  6. source information for each texture and model that you used, as described above on page 1
  7. indicate on which RVR-5029 machine you tested your program, and if it works on it