Assignment 3 – CS360

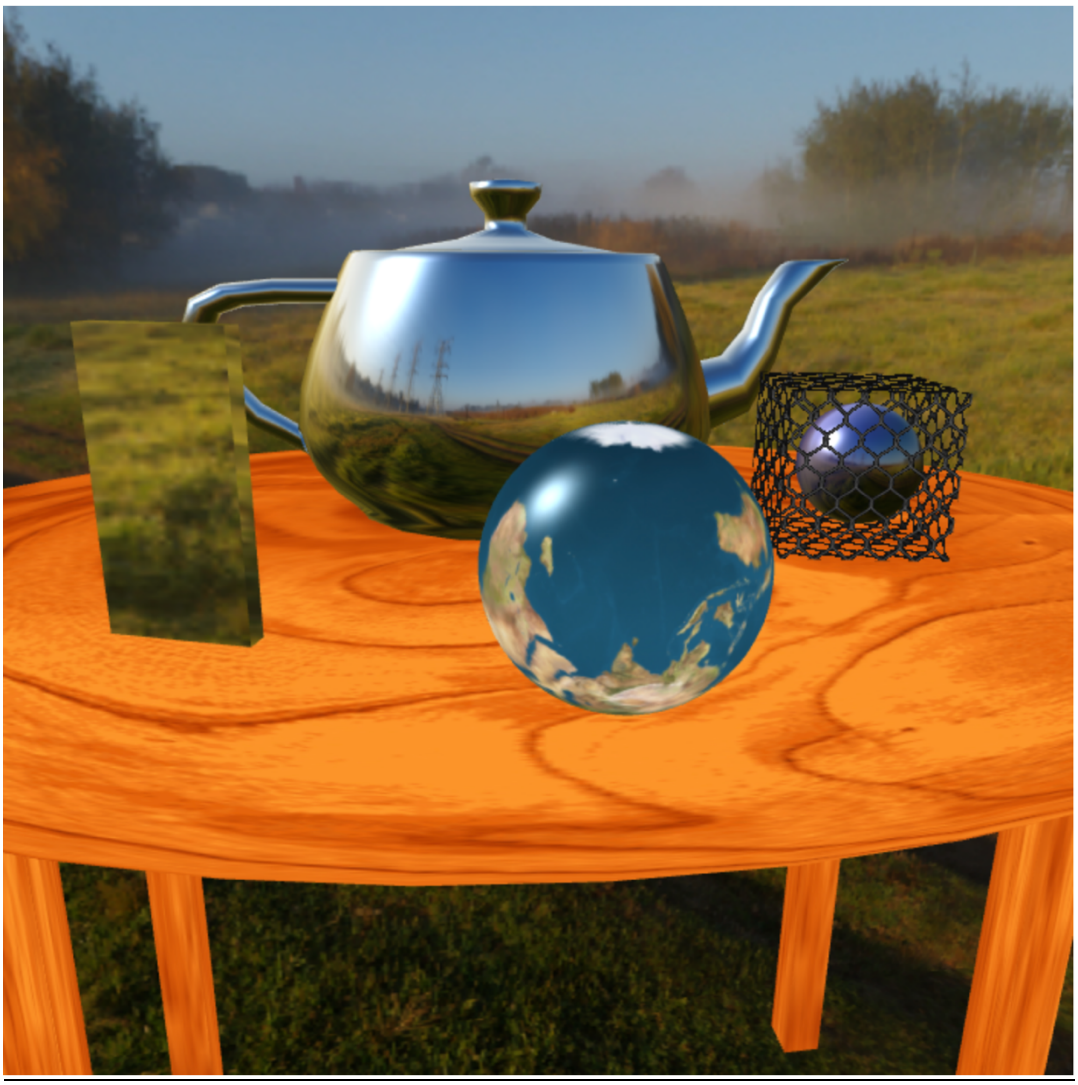
Texture Mapping and Cubemap Environment Mapping

Due date: October 13, 2024, 11:59pm

Grade: 100 points (10% of the course grade)

In this assignment, you will create a 3D animated scene that will showcase texture mapping on objects and cubemap environment reflection mapping on a teapot. You will also create a skybox to give context to the objects in the scene, as shown in the image below. You will use the glMatrix-0.9.5.min.js JavaScript matrix-vector manipulation library to perform transformations using the APIs provided by it. Example usage of this library is discussed in class, and sample codes are provided as references. Other than the glMatrix-0.9.5.min.js library, no other additional library is allowed to be used to complete your assignment.

Here is the scene that you will create (See attached video for):



Pointers for your assignment:

1. All the necessary texture files and teapot.json file is provided.
2. You should first understand the Loading_Meshes and Texture_mapping examples that I uploaded sometime back in HelloIITK. Those two example codes already provide you with some of the code segments you will need for this assignment.
3. Skybox: The skybox is constructed by putting together six square plane objects and then adding each of the six cubemap texture images to each face of the cube. This has been discussed in the class in detail. Refer to those slides to understand how to build a simple skybox.
4. In the scene, the teapot, one of the spheres have a cubemap environment reflection on them.
    - The teapot is modeled as a perfectly reflecting shiny material and hence has the final color = reflection texture color of the cube-mapped environment.
    - The color of the bluish sphere (inside the cube) has two components. One component is the environment reflection color (same as the teapot), and the other color component is the sphere's own color, which can be computed using per-fragment Phong shading that you have implemented in Assignment 2. Then, the reflection color and Phong color are mixed to get the final color. That is why the sphere has reflections on them, but they also show their own blue color.
    - The other larger sphere has a texture mapping on it showing a texture of a globe.
    - The smaller cube on the right is texture mapped using an image with an alpha channel It uses the images alpha channel information to selectively discard fragments on cube so that we can see what is inside the cube. We have discussed in class about how to discard fragments using textures. The texture file is provided.
    - The mirror-like scaled cube on the left is modeled as a transparent block of glass. Note that this does not reflect the environment color. Instead, this block of glass object is refracting lights, i.e., some part of the background is seen through this glass block. To get this refraction effect, we can use cubemap reflection texture. We have seen in the class how we can get the refraction vector using the refract () function in the fragment shader. You can take the refractive index, the third parameter of the refract () function = 0.99.
    - Finally, the tabletop is modeled as a scaled sphere, and the four legs of the table are modeled as scaled cubes. The table has simple wood texture color.

I suggest that you follow the divide-and-conquer strategy for this assignment. Several pieces are put together to form the scene. Build them separately by writing separate functions for them. For example, write a function for building the skybox. You already know how to render spheres and cubes. So, extend those functions and add texture mapping ability. Then finally, implement the cubemap environment reflection mapping in the shader. Finally, put together all the objects with desired lighting effects to construct the final scene. Then, add the camera rotation animation. Check the accompanying video carefully for all the visual effects on the objects.

How to submit?

The **HelloIITK** portal will be set up for submission. There will be a time limit set. Please start early and finish it by the deadline. Your submission should ideally contain three files, one main JavaScript, one HTML file, and the glMatrix.js file. Zip everything into a single compressed file and upload it. Your code should run out-of-the-box on TA's computer without needing to do any modification. You can test it in both Chrome and Firefox before submitting it. Name your compressed submission file as "**Last-name_roll-number_Assignment3.zip**", and replace last-name, roll-number with your last name and roll number.

Grading:

We will grade your submitted version only. So DO NOT MISS THE DEADLINE, else you may get 0.

Start early and have fun!