CSE 333/533: Computer Graphics

Lab 8: Environment Mapping

Instructor: Ojaswa Sharma , TAs: Vishwesh Vhavle , Aadit Kant Jha

Due date: 23:59, 24 November 2023

Introduction

In this lab, we will look at how to use cube-map and use it to map the environment. A cube-map is a texture that contains 6 individual 2D textures that each form one side of a cube: a textured cube. A cube map essentially maps a hemisphere above the surface, but in many cases, maps the entire 360 degree scene around a point. Two popular techniques that make use of cube maps are **sky boxes**, which provide the illusion of a 3d background behind the scenery, and **environment mapping**, which can make a very shiny surface (think lakes, chrome, or car paint) appear to reflect the environment scenery.

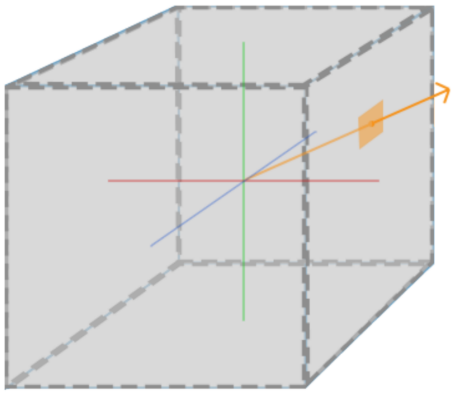

Fig. 1: *Cube Maps*

- A cube-map is a texture like any other texture, so to create one, we generate a texture and bind it to the proper texture target before we do any further texture operations. This time binding it to `GL_TEXTURE_CUBE_MAP`. *(For reference, check line 142 in skybox.cpp)*

- Because a cube-map contains 6 textures, one for each face, we have to either call `glTexImage2D` six times with their parameters set *(For reference, check line 145)*. This time, however, we have to set the texture target parameter to match a specific face of the cube-map, telling OpenGL which side of the cube-map we're creating a texture for. This means we have to call `glTexImage2D` once for each face of the cube-map.

- Then you have to set the filers for the texture format.

Environment Mapping

An environment map behaves like a mirror to translucent surfaces to either reflect or refract the skybox. For this, we have to work out a direction vector from the viewpoint (camera position) to the surface. We then reflect the direction of the surface based on the surface normal. We use the reflected direction vector to sample the cube map. GLSL has a built-in `reflect()` function which takes an input vector and a normal and produces the output vector.

Similarly, translucent objects should refract light. This means that it doesn't go straight through; the ray bends when it hits the surface and changes material. Looking into a swimming pool, for example. This works much the same way as reflection, except that we perturb the eye-to-surface vector about the surface normal, instead of reflecting it. The actual change in direction is governed by Snell's Law. GLSL also has a built-in `refract()` function with a vector-based version of Snell's Law.

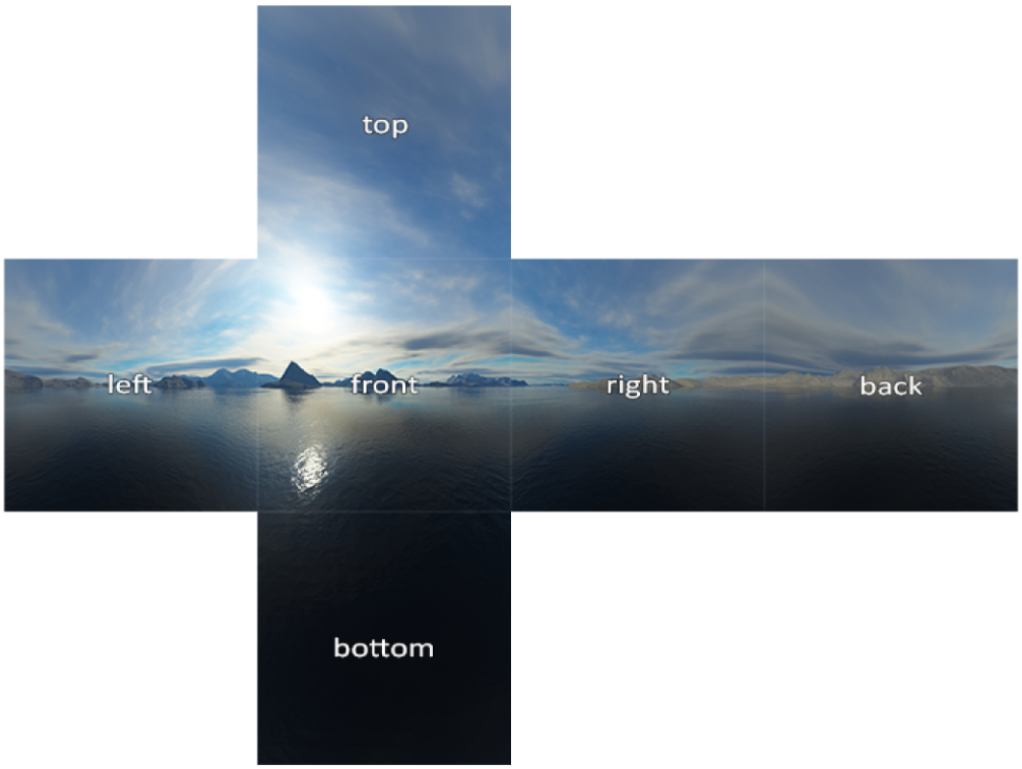

Fig. 2: *Skybox Texture*

Deliverables

Fig. 1: *Reflective Sphere*

Fig. 2: *Refractive Sphere*

The current code just renders the skybox. If you set `ENABLE_REFLECTIVE_SPHERE` to 1, then you will see the reflective sphere. The task for the lab is to convert the reflective sphere to a refractive sphere. For this you just have to add the refraction part in the fragment shader. To have a realistic glass sphere that both reflects and refracts, you have to use Schlick's approximation to get the final color. Submit the screenshot of outputs (Only refraction and reflection refraction together) along with the code for evaluation in a zip file.

Name the zip file as lab08_<name>_<roll number>.zip
Example: lab08_vishwesh_2020156.zip

References

https://www.opengl.org/documentation/
LearnOpenGL - Cubemaps. site: https://learnopengl.com/Advanced-OpenGL/Cubemaps
Cube Maps: Sky Boxes and Environment Mapping - https://antongerdelan.net/opengl/cubemaps.html

---

Note: Your code should be written by you and be easy to read. You are NOT permitted to use any code that is not written by you. (Any code provided by the instructor/TA can be used with proper credits within your program). Theory questions need to be answered by you and not copied from other sources. Please refer to IIIT-Delhi's Policy on Academic Integrity here.