CS 543 Project 3 Manual

Zhixin Yan

# Introduction

There are four different types of shaders implemented in this project.

The shader for the chessboard like ground is a simple shader rendering texture with phong lighting.

The box on the left side is rendered by bump map shader. The texture and bump map are two pictures given in the project 3 web page: Bump-Picture.jpg and Bump-Map.jpg

The sphere in the middle looks like a metal one has environment shader bounded. The environment maps are six pictures from <http://www.humus.name/index.php?page=Cubemap&item=Lycksele3>

The cylinder on the right side is bounded to parallax map shader. The parallax map implemented here is not steep parallax map or relief parallax map, but the naïve parallax map with fixed scale and bias similar to the example showed in the class.

The only light source in the environment is a directional light from (0.0, 1.0, -1.0). You could press “t” to render without texture but only the normals. If the bump map or parallax map effect are not very noticeable, please rotate the view by press control left or control right.

The web page might crush in one or two minutes sometimes, due to memory issue.

|  |  |
| --- | --- |
|  |  |
| a) | b) |

Figure 1: Screenshots. a) is rendered with texture; b) only shows the normals in bump map and parallax map

# Keyboard Controls

|  |  |
| --- | --- |
| **Keystroke** | **Action** |
| Right Arrow | Slide camera 1 unit in the positive X direction |
| Left Arrow | Slide camera 1 unit in the negative X direction |
| Up Arrow | Slide camera 1 unit in the positive Y direction |
| Down Arrow | Slide camera 1 unit in the negative Y direction |
| Shift Up Arrow | Slide camera 1 unit in the positive Z ("in") direction |
| Shift Down Arrow | Slide camera 1 unit in the negative Z ("out") direction |
| Control Down Arrow | Change camera pitch by 2 degrees |
| Control Up Arrow | Change camera pitch by -2 degrees |
| Control Right Arrow | Change camera yaw by 2 degrees |
| Control Left Arrow | Change camera yaw by -2 degrees |
| < | Change camera roll by 2 degrees |
| > | Change camera roll by -2 degrees |
| r | Reset to the default position and orientation |
| t | Only show normal map in bump map and parallax map shaders |

# File Organization

## Shaders

All the shaders are in project\_3\_2.html

The shader of the ground is vertex-shader-ground and fragment-shader-ground, from line 7.

The shader of the sphere is vertex-shader-envmap and fragment-shader- envmap, from line 95.

The shader of the box is vertex-shader-bumpmap and fragment-shader- bumpmap, from line 137.

The shader of the cylinder is vertex-shader-parallaxmap and fragment-shader- parallaxmap, from line 247 to line 361.

## Highlighted Functions and Classes

### InitializeGLShader in project3\_2.js

Even though the program started from function init();

InitializeShader is the real main function to initialize shader. Inside this function, I created a quad, a cube, a sphere and a cylinder and bind with different textures. To make the shader programming more flexible, I pass a function to the Spirit3D class as a parameter to execute shader related scripts at each frame. Those functions are GroundShaderScript (line 346), EnvMapShaderScript (line 335), ParallaxMapShaderScript (line 313), and BumpMapShaderScript (line 290). Textures and uniform variables are passed to shader in those functions.

The function initLights (line 279) is also called by those shader functions to pass light and material information to shader.

### checkKey & cancelKey in project3\_2.js

All the keyboard events are collected by the two functions from line 477. Most of the keys are used to change three vec3 type variable, eye, at and up to change the camera view.

### Class Diagram

Spirit3D is the class representing 3d objects in the scene. It has mesh, shader and transform information inside. A Spirit3D object could also have children. While rendering, it will call the render function of its children after render itself.

Mesh is an abstract class which is derived by all kinds of geometries such as cube, sphere, and cylinder. After a mesh is created, call DumpToVertexArray function of mesh, which is a general function for converting mesh representation to points, normal, texture coordinates, and color arrays webgl vertex buffer.

Figure 2: Class Diagram.