深 圳 大 学 实 验 报 告

课程名称：计算机图形学

实验项目名称：实验四　带纹理的 OBJ 文件读取和显示

学院：计算机与软件学院

专业：软件工程（腾班）

指导教师：熊卫丹

报告人：洪子敬　学号：2022155033　班级：腾班

实验时间：2024 年 12 月 4 日 至 2024 年 12 月 11 日

实验报告提交时间：2024 年 12 月 5 日

教务部制

实验目的与要求：

1. 了解三维曲面和纹理映基本知识
2. 了解从图片文件载入纹理数据基本步骤
3. 掌握三维曲面绘制过程中纹理坐标和几何坐标的使用
4. 在程序中读取带纹理的 obj 文件，载入相应的纹理图片文件，将带纹理的模型显示在程序窗口中。

实验过程及内容：

1. 读取带纹理的 obj 文件

给定的 TriMesh 类的 readObj 函数，补充代码使得其满足：

vertex_positions 存储顶点坐标

vertex_textures 存储 UV 坐标数据

vertex_normals 存储顶点法向量数据

faces 存储三角面片的顶点索引数据

texture_index 存储面片顶点对应的纹理坐标的索引下标

normal_index 存储面片顶点法向量的索引下标

其中：由于 obj 文件无法存储颜色数据，可以采用法向量作为颜色数值。

解答：根据实验指导我们知道 obj 文件读入的格式，对于不同的数据，每行给的关键词都不同，因此根据关键词可以对每行数据进行读入：“v ”代表顶点（由于有几种类型都是 v 开头，这里加上一个空格作为区分）、“vn”代表顶点法向量、“vt”代表纹理坐标（注意纹理坐标空间是二维的）和“f”代表面片（空格可加可不加）；对于前面三种我们可以很简单的按格式读入得到 vertex_positions、vertex_textures 和 vertex_normals，而对顶点颜色 vertex_colors，我们简单通过顶点法向量坐标赋值得到即可；详细代码如下：

```
// @TODO: Task2 读取obj文件，记录里面的这些数据，可以参考readOff的写法
// vertex_positions
if (line.substr(0, 2) == "v ") {
    sin >> type >> _x >> _y >> _z;
    vertex_positions.push_back(glm::vec3(_x, _y, _z));
}
// vertex_normals
else if (line.substr(0, 2) == "vn") {
    sin >> type >> _x >> _y >> _z;
    vertex_normals.push_back(glm::vec3(_x, _y, _z));
    //用法向量的数值作为颜色
    vertex_colors.push_back(glm::vec3(_x, _y, _z));
}
// vertex_textures
else if (line.substr(0, 2) == "vt") {
    sin >> type >> _x >> _y;
    vertex_textures.push_back(glm::vec2(_x, _y));
}
```

HZJ

最后面片的读入，需要明确面片的输入格式是:f 顶点索引/uv 点索引/法线索引（共三组），因此对于每行读入，需要进行三次读取，对每次各类结果进行存储，最后打包成一个 vec3i 类型数据存储在各个类型的索引数组中即可；详细代码如下：

```
// index
else if (line.substr(0, 2) == "f ") {                                    HZJ
    sin >> type;
    int vertexIndex[3];
    int textureIndex[3];
    int normalIndex[3];
    for (int i = 0; i < 3; i++) {…
    }
    //faces
    faces.push_back(vec3i(vertexIndex[0] - 1, vertexIndex[1] - 1, vertexIndex[2] - 1));
    // normal_index
    normal_index.push_back(vec3i(normalIndex[0] - 1, normalIndex[1] - 1, normalIndex[2] - 1));
    color_index = normal_index;
    // texture_index
    texture_index.push_back(vec3i(textureIndex[0] - 1, textureIndex[1] - 1, textureIndex[2] - 1));

}
```

2. 完善数据的读取

给定的 TriMesh 类的 storeFacesPoints 函数，结合前面的 obj 文件的读取方式，补充代码完成顶点坐标 points、顶点颜色 colors、顶点法线 normals 和纹理坐标 textures 的转换，并将数据传入 GPU。

解答：根据完成的 obj 文件，我们知道了读取的数组是按照三角形面片存储的，所以我们每次取出一个面片，根据该面片的三个索引往对应的数组里面加入对应的坐标即可；例如每次从 faces 数组取出一个面片 face，根据它的三个索引 x、y 和 z，对应到顶点数组 vertex_positions 中的三个顶点坐标，将三个坐标依此加入 points 中即可，循环往复，直至所有面片加入完毕；其他数组也通过类似方法得到，详细代码如下：

```
// @TODO Task1 根据每个三角面片的顶点下标存储要传入GPU的数据
for (int i = 0; i < faces.size(); i++)
{
    vec3i face = faces[i];
    vec3i normal = normal_index[i];
    vec3i color = color_index[i];
    vec3i texture = texture_index[i];
    // 坐标
    points.push_back(vertex_positions[face.x]);
    points.push_back(vertex_positions[face.y]);
    points.push_back(vertex_positions[face.z]);
    // 颜色
    colors.push_back(vertex_colors[color.x]);
    colors.push_back(vertex_colors[color.y]);
    colors.push_back(vertex_colors[color.z]);
    // 法向量
    normals.push_back(vertex_normals[normal.x]);
    normals.push_back(vertex_normals[normal.y]);
    normals.push_back(vertex_normals[normal.z]);
    // 纹理
    textures.push_back(vertex_textures[texture.x]);
    textures.push_back(vertex_textures[texture.y]);
    textures.push_back(vertex_textures[texture.z]);
}                                                                        HZJ
```

3. 模型和纹理显示

给定 main.cpp 中的 init 函数，完成玩偶模型和桌子模型的显示，并实现贴图。

解答：仿照实验 4.1 模型的显示以及模型旋转位移的设置，这里实现基本一致，唯一要注意的是如果进行旋转的话，模型的旋转是朝外的，所以需要沿 x 轴，逆时针旋转 90

度达到摆正的效果，即设置旋转三维坐标的 x 坐标为-90；此外，为了使得显示结果明显，这里对每维度放大 1.5 倍；详细代码如下：

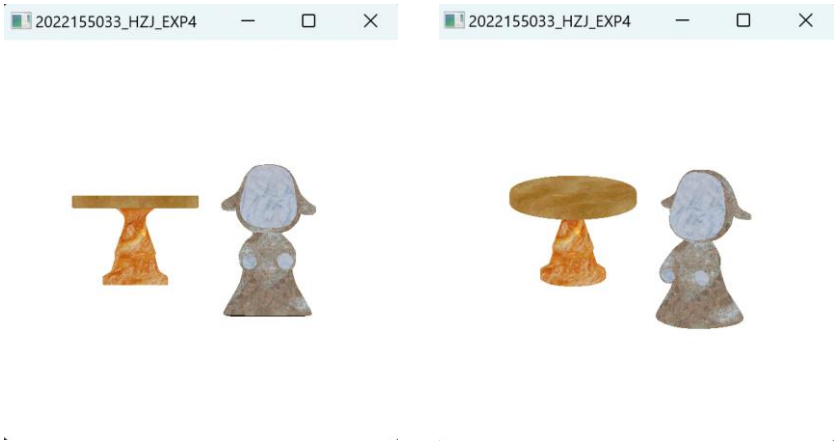
```
TriMesh* table = new TriMesh();
// @TODO: Task2 读取桌子模型
table->setNormalize(true);
table->readObj("./assets/table.obj");
// 设置物体的旋转位移
table->setTranslation(glm::vec3(-0.5, 0.0, 0.0));
table->setRotation(glm::vec3(-90.0, 0.0, 0.0));
table->setScale(glm::vec3(1.5, 1.5, 1.5));
// 加到painter中
painter->addMesh(table, "table", "./assets/table.png", vshader, fshader);

TriMesh* wawa = new TriMesh();
// @TODO: Task2 读取娃娃模型
wawa->setNormalize(true);
wawa->readObj("./assets/wawa.obj");
// 设置物体的旋转位移
wawa->setTranslation(glm::vec3(0.5, 0.0, 0.0));
wawa->setRotation(glm::vec3(-90.0, 0.0, 0.0));
wawa->setScale(glm::vec3(1.5, 1.5, 1.5));
// 加到painter中
painter->addMesh(wawa, "wawa", "./assets/wawa.png", vshader, fshader);
```

HZJ

4. 结果展示

运行补充好代码的程序，可以得到部分实验结果如下：



可以看到效果与实验要求和预期结果一致，实验成功完成。

实验结论：

本实验回顾了三维曲面和纹理映基本知识，并初步了解了从图片文件载入纹理数据基本步骤，掌握了三维曲面绘制过程中纹理坐标和几何坐标的使用，学会了在程序中读取带纹理的 obj 文件、载入相应的纹理图片文件和将带纹理的模型显示在程序窗口中，最后成功展示了带纹理的娃娃模型和桌子模型。本次实验圆满结束。

指导教师批阅意见：

成绩评定：

指导教师签字：

年　　月　　日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。