计算机图形学 2017 项目 1 文档

刘洋　515030910486

0. 操作指南

- 打开项目后，会出现命令提示行窗口与显示窗口两个窗口。
- 首先在命令提示行窗口中输入要加载的.obj 文件名。（项目目录下自带有 `sphere.obj`，`mario.obj` 与 `monkey.obj`）如果加载失败或直接回车跳过，项目将会自动加载“`sphere.obj`”（球面模型）。
- 此时，显示窗口中将会开始渲染过程。鼠标左键增加小球向上的速度，鼠标右键增加小球向下的速度，键盘“WASD”控制相机环绕，鼠标滚轮控制视野范围。按“ESC”键便可退出项目程序。

1. 项目完成情况

功能点	完成情况
绘制球面	完成，球面为外部.obj 文件
小球弹跳物理仿真	完成
Loader 载入三维模型	完成，但 Loader 有局限性
球面塑性变形	有，触地时会发生变形
相机移动	有，相机沿球面移动

2. 项目架构概述

由于 glut 为闭源软件，且不再更新维护，本项目使用了 glfw + glew 的开源外部库处理 Windows 窗口化、IO 输入、OpenGL 支持函数查询等操作。数学相关的操作使用了 glm 库。项目高度模块化，封装了 `Application` 类，而 `Loader`、`Camera` 等功能都在不同的文件中。项目实现了简单的 Vertex Shader 与 Fragment Shader，为提高兼容性，GLSL 版本选择了 3.3。物体的下落根据牛顿第二定律，塑性变形遵从胡克定律。

3. 项目各组件详述

3.1 application 类 (sb7.hpp, sb7.cpp)

主要参考了 *OpenGL SuperBible, 6th Edition* 中的 `application` 类，对初始化窗口、渲染等操作用类进行了封装。主要的方法有：

- `init()`负责初始化窗口，设置窗口属性
- `startup()`负责绘制前的初始化，包括加载 shader 等
- `render()`为最重要的绘制单帧函数，application 类在调用时会传入当前时间
- `shutdown()`在应用程序关闭前调用，可以在这里执行释放内存等操作

3.2 `shader.cpp, shader.hpp`

包含了 `LoadShaders()`方法，可以通过此方法编译 Fragment Shader 与 Vertex Shader。

3.3 objloader.cpp, objloader.hpp

包含 `loadOBJ()`方法，实现了非常简单的.obj 文件 vertex，normal，uv 以及 face 的加载。加载的.obj 文件必须要包含法线信息。

3.4 camera.cpp, camera.hpp

包含关于球面相机环绕的代码。将会根据键盘的输入计算输出 `Model`，`View`，`Perspective` 中的 `View` 与 `Perspective` 矩阵。

3.5 Bounds.h

参考 Unity3D，包含了 `Bounds` 数据结构，负责储存导入模型的包围盒。

3.5 SimpleTransform.vert, SimpleFragment.frag

分别为项目的 Vertex Shader 与 Fragment Shader， Vertex Shader 负责根据输入的线性变换矩阵计算物体顶点的新位置，Fragment Shader 中则定义了输出的颜色。

4. 项目参考文献

1. OpenGL SuperBible: Comprehensive Tutorial and Reference, 7th edition
2. GLFW project website: http://www.glfw.org/
3. opengl-tutorial: http://www.opengl-tutorial.org

5. Github 链接

https://github.com/GhostatSpirit/OpenGL-projects