山东大学 计算机科学与技术 学院

计算机图形学 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202200130119 | 姓名：于斐 | | 班级：学堂计机22 |
| 实验题目：模型绘制与交互 | | | |
| 实验学时： 4 | | 实验日期： 2024年11月23日 | |
| 实验目的：  通过使用OpenGL编程实现一个真实感3D模型的绘制与交互，培养学生对计算机图形学的基本概念、技术和开发能力的掌握。通过从OBJ等格式文件中加载复杂模型并进行真实感显示，理解3D模型的渲染过程及其与纹理、光照的交互原理。通过实现平移、旋转、缩放等基本操作，熟悉3D模型的变换矩阵及其在OpenGL中的应用。最终，通过优化交互方式，提升用户对3D模型操作的自然性和直观性，为后续复杂的图形学应用开发奠定基础。 | | | |
| 实验步骤与内容：  实验环境：OpenGL 4.6 及 GLFW, GLM 等附属库。  实验步骤：   1. 项目使用 CMake 管理，原则上任何支持CMake的编辑器均可使用。所有未包含在项目文件中的库均使用 CMake 的FetchContent导入，不需额外手动安装任何库。 程序自动生成到 dist 目录下，但运行时pwd需包含assets。因此需要在根目录运行 dist/Renderer 或将 assets 拷贝到 dist 下。 2. 建立OpenGL窗体及 argument parser, config parser。前者代码位于 Renderer::init()中，在 src/renderer/renderer.h, renderer.cpp 下可以找到。后者代码位于 src/utils 中。       部分实验代码  使用 ImGUI 构建交互选单，用于控制程序运行时的行为。相关代码位于 src/gui/ 下。     1. 构建 Model 类用于从 .obj 中导入模型（包含vertices position, normal（如果没有则插值获得）, texcoord）。此处使用 tiny obj loader 库辅助实现。支持导入同一 obj 文件中的不同 object，并导入 diffuse, specular, normal texture。 导入的数据首先在 CPU 侧储存，后建立OpenGL Vertex Arrays, Buffer Data, Texture 送入 GPU 侧。     程序支持运行时替换模型。读入的模型会被归一化为统一大小（默认为 XYZ -1 ~ 1）。     1. 构建Shader类用于加载 Shader Program。加载方式与大多 OpenGL 加载器相同，在此不再赘述。     本实验用到了 5 个 shader（位于 assets/shaders/ 下）：   * 1. rasterization/object.frag/vert：渲染物体的 shader，其中 vertex shader 做变换（model \* view \* projection），fragment shader 做 phong 模型的绘制，其中 ambient, diffuse 使用 diffuse 贴图 + 光照，specular 使用 specular 贴图。变换矩阵、光照通过 uniform 送入。   2. rasterization/lighting.frag/vert：在屏幕中绘制一个虚拟的光球，代表此时的光源位置。通过 opengl 的点绘制（glDrawArrays GL\_POINTS）实现。Uniform 变量同上。   3. rasterization/axis.frag/vert：坐标轴绘制，绘制出一个 x, y, z 坐标轴，类似一般的 object viewer。   4. normal/object.frag/vert：绘制物体的法线。将 fragment shader 中的输出颜色改为了normalized normal。   5. depth/object.frag/vert：绘制物体的深度图。通过 gl\_FragCoord.z 得到数据。     程序支持运行时更换渲染模式。支持运行时光照调整。     1. 构建 Camera 类，用于变换相机。可供变换的参数有控制姿态的position, pitch, yaw、控制成像的fov, near, far和控制操作的speed, sensitivity。 程序支持运行时相机调整，包含两种调整模式：FPS鼠标调整、选单调整。使用 ` 键切换 FPS 模式（称 focus mode）。   在 focus mode 下，窗口将锁定鼠标，可使用类似FPS游戏的方式调整相机。  在选单中，可拖拽变换相机的 speed，sensitivity，fov。  相关代码位于 src/renderer/camera.cpp 下。     1. 构建物体变换方式。项目中构建了Motion类，支持vec3形式的 translation, scale 和 quaternion 形式的 rotation。在 gui 交互界面中，rotation 被表示为 euler angle，但是是由 quaternion 转译来的。     程序支持两种交互方式：通过gui设置数值，鼠标拖拽交互。  前者不再赘述，对于后者，提供了切换交互方式的选单，在选单中可以选择不同的交互方式。    在选择了某种交互方式后，使用鼠标拖拽模型，即可以符合人类直觉的形式自然交互。  **此处“拖拽交互”形式是实验的创新点**。对三种交互方式，实现方式如下：   * 1. Translation：鼠标左键按下时，cast 一条 ray 与物体做 intersection test，得到交点。在鼠标左键按住的过程中，实时 cast 新的 ray，尝试将之前的交点以最短距离的方式移动到新的 ray 上。即，找到原交点在新的 ray 上的投影点，二者之差即可作为 translation。   2. Rotation：鼠标左键按下时记录指针的屏幕空间坐标，在鼠标左键按住的过程中，实时追踪指针的屏幕空间坐标。二者在 x, y 轴上的差距，映射到相机的 right 和up轴上。对物体的四元数做反向变换。   3. Scale：记录物体交点之差，令 scale 增量为差值即可。 | | | |
| 实验总结：  通过本次实验，实现了一个 OpenGL 的渲染器，支持 3D Object 的读取和查看，并支持多种变形和多种视角。对 OpenGL 的理解加深，为后续复杂的图形学应用开发奠定了基础。 | | | |