**深 圳 大 学 实 验 报 告**

**课程名称：­ 计算机图形学**

**实验项目名称： 实验二 三维模型读取与控制**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 胡瑞珍**

**报告人： 学号： 班级：**

**实验时间：2024年 10月15日 -- 2024年 10月29日**

**实验报告提交时间： 2024 年 10 月 22 日**

**教务部制**

|  |
| --- |
| 实验目的与要求：   1. 熟悉OpenGL 三维模型的读取与处理；理解三维模型的基本变换操作；掌握鼠标键盘交互控制逻辑；掌握着色器中uniform关键字的使用以及数据传输方法。 2. OFF格式三维模型文件的读取：完成对OFF格式三维模型文件的读取与显示，可改变物体的显示颜色。 3. 三维模型的旋转动画：结合模型进行旋转变换的过程，为模型添加自动的旋转动画。 4. 键盘鼠标的交互：通过键盘设定选择绕x、y、z轴进行旋转，鼠标左右键控制动画的开始与暂停。 |
| 实验过程及内容：   1. OFF格式三维模型文件的读取 2. 读取OFF格式模型的函数     该函数用于读取OFF格式的三维模型文件。首先，函数检查输入的文件名是否有效并尝试打开文件。如果文件成功打开，读取文件头信息，获取顶点和面片的数量。接着，通过循环读取每个顶点的坐标并将其存入顶点位置和颜色的容器中。随后，读取每个面片的顶点索引并将其存储到面片索引的容器中。最后，调用 storeFacesPoints 函数以整理面片和点的信息，准备后续的图形渲染。   1. 存储三角面片点和渐变颜色的函数     该函数主要负责将三角面片的顶点坐标和对应的渐变颜色存储到 GPU 可以使用的数据结构中。首先，它遍历所有的面片，每个面片由三个顶点组成，分别获取它们的位置。接着，使用顶点的坐标值计算每个顶点的颜色，采用将坐标值映射到0到1范围内的方式，实现颜色的渐变效果。具体来说，顶点的 x、y 和 z 坐标分别映射到 RGB 颜色通道。最后，将顶点的位置和对应的颜色分别存储到 points 和 colors 容器中，以供后续渲染使用。这样，渲染时每个三角面片的颜色将基于其顶点的坐标，实现了视觉上的渐变效果。   1. 读取OFF模型       给出OFF模型地址并传入到实例cow的方法readOff，即可成功读取到OFF模型。   1. 三维模型的旋转动画   为了实现三维模型的旋转动画，需要每次调用 display() 函数时，都会根据当前的角度和旋转速度进行重新计算和渲染，使得模型在屏幕上看起来是连续且流畅的旋转效果。   1. 基础变量      * X\_AXIS, Y\_AXIS, Z\_AXIS：用于表示旋转的轴，分别对应X轴、Y轴和Z轴（0, 1, 2）。这些常量用于指定当前模型绕哪个轴进行旋转。 * DELTA\_DELTA：控制 rotateDelta 的变化速率，表示每次调整时旋转速度的增量为0.3。用于控制旋转速度的变化步长。 * DEFAULT\_DELTA：默认的旋转速度值，初始为0.4。程序开始时旋转速度会被设为这个值，后续可以根据需要动态调整。 * rotateDelta：当前的旋转速度，初始为 DEFAULT\_DELTA。通过该变量控制模型的旋转速度，我们可以通过按键或其他方式动态调整此值。 * rotateTheta：一个 glm::vec3 类型的向量，表示绕X、Y、Z三个轴的当前旋转角度。每个轴的旋转角度可以分别控制，且随时间或输入更新。 * currentRotateAxis：当前正在旋转的轴，初始为 X\_AXIS。该变量用于控制模型绕哪个轴旋转，可以在运行时动态切换旋转轴。 * stopFlag：控制旋转的暂停标志，值为0时模型持续旋转，值为1时暂停旋转。用于实现模型旋转的开始与暂停。 * currentRotateDirection：当前的旋转方向，1表示顺时针旋转，-1表示逆时针旋转。用于控制旋转的方向，允许我们在运行时切换方向。  1. 渲染与旋转变换     以上代码的作用是清空OpenGL窗口，并根据当前的旋转角度绘制模型。在代码中，首先通过 glClear 函数清除颜色和深度缓存，以准备新一帧的绘制。接着使用绑定的着色器程序和顶点数组对象（VAO），这使得后续绘制操作能够正确使用模型的顶点数据。  代码段中对变换矩阵的初始化和旋转处理是关键部分。首先，初始化一个4×4的单位变换矩阵 m，这个矩阵用于描述物体在三维空间中的位置和方向。单位矩阵表示没有任何变换。接下来，通过 glm::rotate 函数依次对矩阵 m 执行绕X轴、Y轴和Z轴的旋转，旋转的角度由 rotateTheta 向量中的值指定，并通过 glm::radians 将角度转换为弧度。这些旋转操作会累积在变换矩阵中，从而实现对模型的组合旋转。  最后，使用 glUniformMatrix4fv 将更新后的变换矩阵 m 传入着色器中的对应 uniform 变量 matrixLocation，这使得在绘制过程中，着色器能够根据该矩阵正确地变换模型的位置和方向。通过这个过程，模型能够根据当前的旋转角度在屏幕上实时显示旋转效果。   1. 旋转控制与参数更新     以上代码实现了对模型旋转参数的管理与更新。updateTheta 函数通过接收一个轴和一个符号，利用当前的 rotateDelta 值来调整 rotateTheta 向量中的旋转角度，允许动态改变模型的旋转。resetTheta 函数则用于重置所有旋转相关参数，将 rotateTheta 清零，并将 rotateDelta 设置为默认值，从而方便地恢复初始状态。最后，updateDelta 函数允许根据输入的符号调整 rotateDelta，实现对旋转速率的动态调节。   1. 旋转动画实现     当 stopFlag 为 0 时，表示旋转未被暂停，程序会通过 updateTheta() 函数不断更新模型在指定旋转轴 currentRotateAxis 上的旋转角度，旋转方向由 currentRotateDirection 决定。updateTheta 函数根据当前 rotateTheta 值来调整模型的旋转角度。这允许模型在每帧都根据当前的状态进行旋转。  随后，display() 函数被调用，用于进行当前帧的绘制，将更新后的模型渲染到窗口中。整个逻辑通过一个无限循环不断执行，直到窗口关闭，从而实现实时的旋转显示效果。   1. 键盘鼠标的交互 2. 设置鼠标按钮回调函数     以上代码实现了一个鼠标按钮的回调函数，用于处理左键和右键的点击事件。函数首先检查按下的按钮是左键还是右键，然后根据鼠标的动作（按下）来更新 stopFlag 的值。如果按下左键，stopFlag 被设置为 0，表示模型继续旋转，并打印 "旋转继续" 的消息。如果按下右键，stopFlag 被设置为 1，表示暂停旋转，并打印 "旋转停止" 的消息。通过这种方式，我们可以通过鼠标操作动态控制模型的旋转状态。   1. 设置键盘输入回调函数     以上代码实现了键盘输入回调函数，用于处理我们的按键输入。按下不同的键，会触发对应的操作，例如关闭窗口、改变旋转轴、加快或减慢旋转速度、调整旋转方向等。核心逻辑是当按键被按下时，根据按键对应的功能修改全局变量（如 currentRotateAxis 和 rotateDelta），以控制模型的旋转行为。每个按键功能的结果会通过控制台输出消息，提供即时反馈给我们。   1. 输出帮助信息     以上代码实现了一个帮助信息的输出函数 printHelp，用于在程序启动时向我们提供控制3D模型旋转动画的键盘和鼠标操作说明。每个选项对应不同的功能操作，例如：按 "X" 键控制模型绕 X 轴旋转，按 "U" 键加快旋转速度，按 "Esc" 键退出程序等。通过这些提示信息，我们可以快速了解如何与程序进行交互，控制3D模型的旋转动画。   1. 设置回调函数     这两行代码通过 GLFW 库为窗口 window 设置回调函数：   * glfwSetKeyCallback：用于设置键盘输入的回调函数 key\_callback，每当我们按下键盘时，程序会调用该函数来处理按键事件（例如旋转方向、调整速度等）。 * glfwSetMouseButtonCallback：用于设置鼠标按钮输入的回调函数 mouseButtonCallback，当我们点击鼠标左键或右键时，程序会调用该函数来控制动画的开始或停止。   这些回调函数能够帮助程序实时响应我们输入，从而控制3D模型的旋转行为。 |

深圳大学学生实验报告用纸

|  |
| --- |
| 实验结论：    成功导入OFF模型和实现了旋转动画，并且可以用鼠标和键盘进行相应的交互，即可以通过键盘设定选择绕x、y、z轴进行旋转，通过鼠标左右键控制动画的开始与暂停等等。 |
| 实验心得：  通过这次实验，我对OpenGL处理三维模型有了更深的认识。读取并解析OFF格式文件，让我了解到如何提取并组织几何数据以供渲染。使用矩阵进行变换操作，尤其是实现模型的旋转动画，增强了我对三维空间变换的理解。同时，通过键盘和鼠标的交互控制，使我能更直观地操作场景中的对象，这让我体会到良好用户界面设计的重要性。此外，掌握uniform变量的应用，使得在着色器中动态调整参数成为可能，进一步提高了渲染效果的灵活性。 |
| 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。