# *计算机图形学入门概论*

## 作业二：纹理和照明

到期时间：2025年4月25日（星期五）晚上11:59

## I. 介绍

在本任务中，您需要使用OpenGL构建一个更加逼真和复杂的场景。为了完成这项任务，您将在OpenGL中体验更多功能，包括照明、复杂的模型构建和加载、纹理映射和交互式事件。您将使用基本体图形或直接从.obj文件加载三维模型，然后查看/建模变换以创建此三维场景。将使用纹理贴图和照明使场景和对象更加逼真。鼠标/键盘输入和窗口事件处理将有助于实现交互式动画。

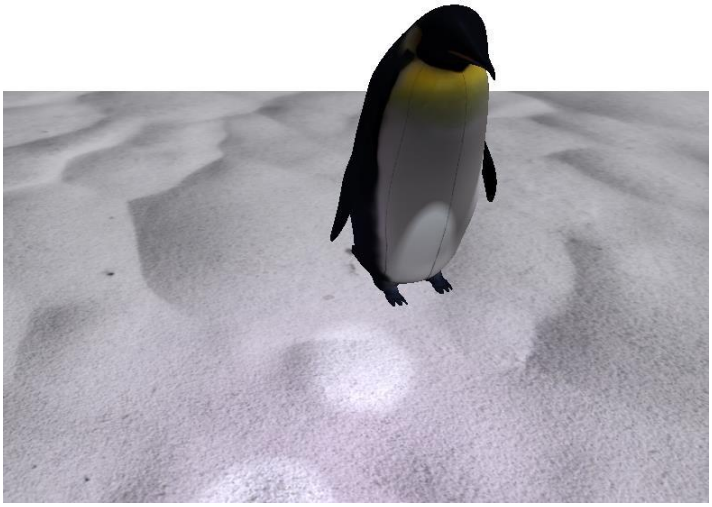

**图1：演示中捕捉到的场景**

在该指定中，场景中有两个模型。其中一个（背景雪原）比较简单，另一个（企鹅）比较复杂。我们可以自己设计背景雪原的顶点属性。然而，对于企鹅来说，它是如此复杂，以至于我们需要通过.obj文件加载模型。此外，雪原和企鹅被渲染成不同的纹理和灯光效果。所显示的场景可以通过用户的交互式输入来控制。您还可以丰富在指定1中创建的场景。

## II. 实施详细信息

### 任务1：加载复杂对象

使用Open Asset Import Library，或我们拥有的函数“Model loadOBJ（const char*objPath）”

给定加载至少一个复杂模型，即演示程序中的企鹅。在本部分中，您可以通过修改“void sendDataToOpenGL（）”子例程来使用“Model loadOBJ（const char*objPath）”函数。

我们在演示程序中提供了模型，即snowfield.obj和penguin.obj。我们鼓励您从Internet下载其他.obj文件或使用Blender来设计您的对象。

（你需要检查penguin.obj，因为如果你直接画企鹅，它会非常巨大。具体来说，你需要做一些转换。）

**任务2：纹理映射和照明**

您需要将不同的纹理映射到两个模型，即演示程序中的雪地和企鹅。我们将使用（请参阅“依赖项/stb_image”）来加载纹理图像。您需要使用键盘交互来更改企鹅的纹理。您首先需要生成一个OpenGL纹理，并通过修改“void texture：：setupTexture（const char*texturePath）”子例程来设置纹理参数。

然后，分别在“void sendDataToOpenGL（）”

和“void paintGL（void）”子例程中加载纹理并将其绑定到不同的模型。**stb imagelibrary**

在这里，我们还在演示程序中提供了两个模型的纹理，也鼓励您从互联网上下载其他纹理或自己绘制/过滤纹理。

此外，3D场景应使用至少两个光源进行照明。一个应该是环境（定向）灯。对于其他光源，您可以自己决定位置和颜色。添加此类光源的主要目的是在模型上产生漫射光和镜面光效果。您可以通过修改“void paintGL（void）”子例程来完成此操作。

**任务3：互动活动和动画**

在此任务中，您需要实现以下交互式事件和动画：

(a) 照明控制

按“w”键和“s”键可分别增加和减少定向光的亮度。

(b) 纹理控制

按“1”和“2”键为企鹅切换两种不同的纹理，我们还提供了两种可以应用于企鹅的纹理。（即penguin/dolpihin_01.jpg，penguin/penguin_02.jpg）

按“3”和“4”键可以切换雪原的两种不同纹理，我们还提供了两种可以应用于企鹅的纹理。（即雪地/雪地_01.jpg，雪地/雪地_02.jpg）

(c) 对象控件

按下箭头键“↑↓←→” 以控制企鹅的运动。具体而言，“↑↓” 分别指示向前和向后移动。“←→” 分别指示向左和向右旋转。（请参阅演示程序中企鹅的动画）

(d) 视图控件

通过鼠标控制摄像机的视图，这意味着：

当单击左键，鼠标上下移动时，您看到的整个场景会相应地上下移动。

（请参阅演示程序。不需要右键单击功能。）

在本任务中，您可以修改以下子程序来实现上述要求：

**void mouse_button_callback（GLFWwindow*窗口，int按钮，int操作，int mods）**

**{**

**// Sets the mouse-button callback for the current window.**

**}**

**void cursor_position_recallback（GLFWwindow*窗口，双x，双y）**

**{**

**// Sets the cursor position callback for the current window**

**}**

**void scroll_callback（GLFWwindow*窗口，双xoffset，双yoffset）**

**{**

**// Sets the scoll callback for the current window.**

**}**

**void key_callback（GLFWwindow*窗口，int键，int扫描代码，int操作，int mods）**

**{**

**// Sets the Keyboard callback for the current window.**

**}**

## 额外任务：增强场景的视觉效果（最高20%）

OpenGL为您的程序提供了许多功能来创建各种视觉效果。你可以自己研究它们，并将它们引入作业中。以下是一些建议的改进：

- 加载更复杂的模型，并将其他纹理映射到它们上，以形成一个有意义的场景。(10%)
- 使用不同类型的光源来制作有意义的场景，例如Pointlight、Spotlight等的组合（10%）
- 复杂模型上的阴影映射。(10%)
- 绘制点或线来跟踪其中一个复杂模型的运动。(10%)
- 任何其他有趣的效果。

## III. 分级方案

您的作业将按照以下评分方案进行评分：

| **基本（80%）** | |
|---|---|
| 加载复杂模型 | 10% |
| 纹理贴图 | 10% |
| 环境（定向）灯照明 | 10% |
| 您设计的光源照明 | 10% |
| 照明控制 | 10% |
| 纹理控制 | 10% |
| 对象控件 | 10% |
| 视图控件 | 10% |
| Bonus | 20% |
| **总计：** | **100%** |

**注意：如果程序不完整或编译失败，则不予评分。**

## IV. 提交编程作业的指南

1) 官方的评分平台应该是带有Visual Studio的Windows。如果我们在执行/编译程序时遇到问题，助教会与你联系，重新提交。

2) 修改提供的main.cpp和VertexShaderCode.glsl和FragmentShaderCode.glsl，并在此文件中提供所有代码。如果您想实现阴影映射技术，可以创建额外的.glsl文件。此外，我们鼓励您为相机类添加额外的.h和.cpp文件。在main.cpp中键入您的全名和学生证。缺少这些基本信息将导致扣分（最高10分）。

3) 我们只接受在可编程管道中编写的OpenGL代码。如果您的解决方案是在固定管道中编写的，则不会得分。

4) 我们只接受使用GLFW和GLEW实现的OpenGL代码。如果您使用其他窗口和OpenGL扩展库（除非您有足够的理由），则不会得分。

5) 将源代码文件（即main.cpp&VertexShaderCode.glsl&FragmentShaderCode.glsl）、可执行文件（例如，如果使用windows，则为openGL.exe）、自述文件（即，readme.txt）、您创建或下载的.obj文件以及您创建或下载的图像文件压缩到.Zip中。用你自己的学生id命名（例如，20210810XXX+张三.zip）。

**6) 请在截止日期的晚上11:59之前提交作业。通过电子邮件提交作业给助教.**

7) 如果有多份提交文件，则只考虑最新的一份。

8) 如果你抄了，课程不及格。