Project 2 编程画一个真实感静态景物

使用 OpenGL 库[1]，python 语言编程，绘制了一个真实感木板图。

1. 文件目录

```
.
├── main.py           // 源代码
├── readme.md         // markdown报告
├── report_PJ2.pdf    // pdf报告
├── result.png        // 渲染结果
└── texture.jpg       // 纹理贴图
```

2. 算法说明

2.1 绘制立方体

在`draw_board()`函数中定义了木板的点、面、坐标，设置角度，并绘制3个显示的面。

```
glMatrixMode(GL_MODELVIEW)
# 旋转角度
glRotatef(20., 0., 1., 0.)
glRotatef(20., 0., 0., 0.)
# 绘制3个显示的面
glBegin(GL_QUADS)
# 前
glNormal3f(0.0, 0.0, 1.0)
for f in faces[0]:
    glTexCoord2fv(coords[f])
    glVertex3fv(vertices[f])
# 上
glNormal3f(0.0, 1.0, 0.0)
for f in faces[1]:
    glTexCoord2fv(coords[f])
    glVertex3fv(vertices[f])
# 左
glNormal3f(-1.0, 0.0, 0.0)
for f in faces[2]:
    glTexCoord2fv(coords[f])
    glVertex3fv(vertices[f])
glEnd()
```

2.2 设置纹理

为了使物体具有真实感，准备了 .jpg 格式的木质纹理贴图，这一步进行纹理设置。首先调用`glGenTextures()`函数生成一个纹理对象`texture`。调用`glBindTexture()`函数将`texture`纹理绑定到`GL_TEXTURE_2D`纹理目标上。调用`glTexParameterf()`函数对纹理进行设置：当纹理坐标超出边界时，采用`GL_REPEAT`方式重复边界的纹理；当绘制的图像与贴图尺寸不一致时，采用`GL_NEAREST`方式使用纹理中坐标最接近的一个像素的颜色作为需要绘制的像素颜色。调用`glTexImage2D()`函数定义材质图片。

```
# 读取材质贴图
img = Image.open(filename)
img = np.asarray(img, dtype=np.uint8)
# 生成纹理
texture = glGenTextures(1)
# # 将texture纹理绑定到GL_TEXTURE_2D纹理目标上
glBindTexture(GL_TEXTURE_2D, texture)
# 纹理坐标超出边界时，采用GL_REPEAT
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT)
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT)
# 绘制图像大于或小于贴图尺寸时，采用GL_NEAREST
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST)
glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST)
# 定义材质图片
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, img.shape[0], img.shape[1], 0, GL_RGB, GL_UNSIGNED_BYTE, img)
```

2.3 设置光源

使用 `glLightfv()` 函数创建光源，设置光源位置。使用`glMaterialfv()`函数设置材质， `GL_FRONT` 和 `GL_DIFFUSE` 表示设置多边形正面的材质散射颜色。

```
# 设置光源位置
glLightfv(GL_LIGHT0, GL_POSITION, [-.5, .5, 1.5, 1.])
# 设置材质对各种光的反光率
glMaterialfv(GL_FRONT, GL_DIFFUSE, [1., 1., 1., 1.])
```

3. 运行方式

在IDE中运行 `main.py` 文件或者在 Teminal 中输入命令 `python main.py`，查看渲染结果。

4. GitHub 链接

https://github.com/jiayunz/Computer_Graphics/tree/master/PJ2

---

1. https://www.opengl.org/ ↩