17341112 刘异橦

# CG 水族馆Project作业报告

## 选题

本小组选题为制作一段动画，选择了水族馆Project。水族馆由海底地板，蟹，章鱼，珊瑚，鱼类和海星生物构成，可以模块化的设计和分工。

## 分工

水族馆的构成主要分为：观察、视角（相机）类用于调整观测方向，物体类用于生成渲染的物体，光照（场景）类用于调节局部和全局光照等。我负责海底地面的生成和基本光照类的代码。

## 设计思路

### 1. 光线

水族馆有两种光，一个是聚光灯，一个是顶部平行光照射，需要有两个光源，第一个角度为90度垂直照射，第二个与视角相关。

### 2. 地板

地板用四边形表示，有颜色属性，实现时要尽可能无缝连接。

综上，需要一个场景类负责光线参数设定和一个四边形类负责生成地板

## 主要代码

Scene.h

场景类和光照参数：

```
class Scene
{
private:
```

```
......
  // light0的光照参数
  static GLfloat ambient0[4];
  static GLfloat diffuse0[4];
  static GLfloat specular0[4];
  static GLfloat position0[4];

  // light1的光照参数
......

  static GLfloat spotAngle; // 聚光灯最大扩展角度

public:
  Camera camera;
  int objects[5]; // 五种物体的计数器
......
  bool light0On;  // 开启/关闭光照0
  bool light1On;
  bool fogMode; // 开启关闭雾气效果
  bool lightMode; // 开关光照功能
......

public:
......
  bool render(void); // 渲染一帧
  void add(Renderable *object); // 添加一个物体

private:
......
};
```

Scene.cpp

```
/**
* 场景类的定义
**/

#include "Scene.h"

using namespace std;

int Scene::width;
int Scene::height;
```

```
// 光照0的参数，环境光，漫反射和反射光和光源位置
GLfloat Scene::ambient0[4]  = {0.1f, 0.1f, 0.1f, 1.0f};
GLfloat Scene::diffuse0[4]  = {0.4f, 0.4f, 0.4f, 1.01f};
GLfloat Scene::specular0[4] = {0.2f, 0.2f, 0.2f, 1.0f};
GLfloat Scene::position0[4] = {0.0f, -1.0f, 0.0f, 0.0f};

// 光照1参数，有光照方向
GLfloat Scene::ambient1[4]  = {0.1f, 0.1f, 0.1f, 1.0f};
GLfloat Scene::diffuse1[4]  = {0.45f, 0.45f, 0.45f,
1.0f};
GLfloat Scene::specular1[4] = {0.5f, 0.5f, 0.5f, 1.0f};
GLfloat Scene::position1[4] = {0.0f, 0.0f, 1.0f, 1.0f};
GLfloat Scene::direction1[4]  = {0.0f, 0.0f, -1.0f};

// 光照1是聚光灯，设置光源最大扩展角度，与法线夹角超过的部分被遮蔽
GLfloat Scene::spotAngle  = 15.f;


Scene::Scene()
{
......
  // 设置光源0
  glLightfv(GL_LIGHT0, GL_AMBIENT, ambient0);
  glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse0);
  glLightfv(GL_LIGHT0, GL_SPECULAR, specular0);
  // 光源1
  glLightfv(GL_LIGHT1, GL_AMBIENT, ambient1);
  glLightfv(GL_LIGHT1, GL_DIFFUSE, diffuse1);
  glLightfv(GL_LIGHT1, GL_SPECULAR, specular1);
}
......
/**
* 仅渲染一帧
**/
bool Scene::render(void)
{
  GLenum error;
  // 有错误则输出
  while ((error = glGetError()) != GL_NO_ERROR)
    cout << ">> Error: " << gluErrorString(error) <<
endl;

  // 清除屏幕，准备渲染和输出
```

```
  clear();

  // 设置聚光灯光照
  glLightfv(GL_LIGHT1, GL_POSITION, position1);
  glLightf(GL_LIGHT1, GL_SPOT_CUTOFF, spotAngle);
  glLightfv(GL_LIGHT1, GL_SPOT_DIRECTION, direction1);

  // 摆放摄像机（调整观察角度）
  camera.position();

  // 顶部光照设定
  glLightfv(GL_LIGHT0, GL_POSITION, position0);

  // 画图
  if (elements->size() > 0)
  {
    iter = elements->begin();
    while (iter != elements->end())
      (*iter++)->draw();
  }

  drawHUD();

  glutSwapBuffers();
  return true;
}

/**
*  颜色和深度缓冲区都被清空，防止影响下一次画图
**/
void Scene::clear()
{
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
}
```

Quad.cpp 地板

```
/**
* Renderable中四边形基类的定义，用于生成水族馆的地板
```

```
**/
#include "Renderable.h"
using namespace std;
// 地板材质颜色和粗糙程度的静态变量
GLfloat Quad::material[4] = {1.f, 1.f, 1.f, 1.f};
GLfloat Quad::shininess = 120.f;
......
// 四边形描绘，先设定材质属性和纹理，然后画四边形
void Quad::_draw(void)
{
  // 设置材质属性（只需要前面，背面在底部看不到，参数material和shininess为static，已初始化
  glMaterialfv(GL_FRONT, GL_AMBIENT, material); // 环境颜色
  glMaterialfv(GL_FRONT, GL_DIFFUSE, material); // 散射颜色
  glMaterialfv(GL_FRONT, GL_SPECULAR, material); // 环境和散射颜色
  glMaterialf(GL_FRONT, GL_SHININESS, shininess); // 镜面反射颜色

  // 开启纹理
  glEnable(GL_TEXTURE_2D);
  // 绑定自定义纹理
  glBindTexture(GL_TEXTURE_2D, FLOOR_TEXTURE);

  // 纹理参数，REPEAT对越界的纹理采用重复填充，消除板块之间的缝隙
  glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
  glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
  // 加载不同大小纹理，自动选择放大缩小最优的纹理
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR_MIPMAP_LINEAR);
  glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
  // glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);

  glColor3f(0.9f, 0.6f, 0.0f);
  // 设定顶点顺序逆时针为正面
  glFrontFace(GL_CCW);
  glBegin(GL_QUADS);
    glNormal3f(0.f, 0.f, 1.f);
```

```
    glTexCoord2f(0.0f, 0.0f); glVertex2f(-0.5f, -0.5f);
    glTexCoord2f(1.0f, 0.0f); glVertex2f(0.5f,  -0.5f);
    glTexCoord2f(1.0f, 1.0f); glVertex2f(0.5f, 0.5f);
    glTexCoord2f(0.0f, 1.0f); glVertex2f(-0.5f, 0.5f);
  glEnd();

  glDisable(GL_TEXTURE_2D);
}
```

我负责部分的代码没有视角和角度控制，只有预定义和初始化的一些光照角度参数，具体相机观察的改变由Camera文件定义。

光照部分需要设置两种光的参数，第一个光源是平行光，第二个是聚光灯（点光源设定一个最大扩展角度形成锥体），两者分别设置环境光、漫反射和全反射光的大小和方向。

地板的四边形部分需要完成的是材质的参数设定，包括颜色和漫反射强度（粗糙度），以及纹理贴图的设定。