Report

16340221 王睿泽 16340203 谭江华 16340257 熊思佳

项目介绍以及实现结果

1. 项目介绍

   本次我们小组完成的项目内容主要是对“吃鸡”游戏的场景模拟，场景中包括树木、草地、房屋等，同时也需要考虑光源对环境光影变化的影响。

2. 实现结果

   效果展示

开发环境以及使用到的第三方库

```
win10+vs2017
assimp3.3.1
```

实现功能列表及简单介绍

- camera roaming

```
// Processes input received from any keyboard-like input system. Accepts input parameter in the form of camera defined
void ProcessKeyboard(Camera_Movement direction, float deltaTime)
{
  float velocity = MovementSpeed * deltaTime;
  if (direction == FORWARD)
    Position += Front * velocity;
  if (direction == BACKWARD)
    Position -= Front * velocity;
  if (direction == LEFT)
    Position -= Right * velocity;
  if (direction == RIGHT)
    Position += Right * velocity;
  if (direction == UP)
    Position += Up * velocity;
  if (direction == DOWN)
    Position -= Up * velocity;
}

// Processes input received from a mouse input system. Expects the offset value in both the x and y direction.
void ProcessMouseMovement(float xoffset, float yoffset, GLboolean constrainPitch = true)
{
  xoffset *= MouseSensitivity;
  yoffset *= MouseSensitivity;

  Yaw += xoffset;
  Pitch += yoffset;

  // Make sure that when pitch is out of bounds, screen doesn't get flipped
  if (constrainPitch)
  {
    if (Pitch > 89.0f)
      Pitch = 89.0f;
    if (Pitch < -89.0f)
      Pitch = -89.0f;
  }
```

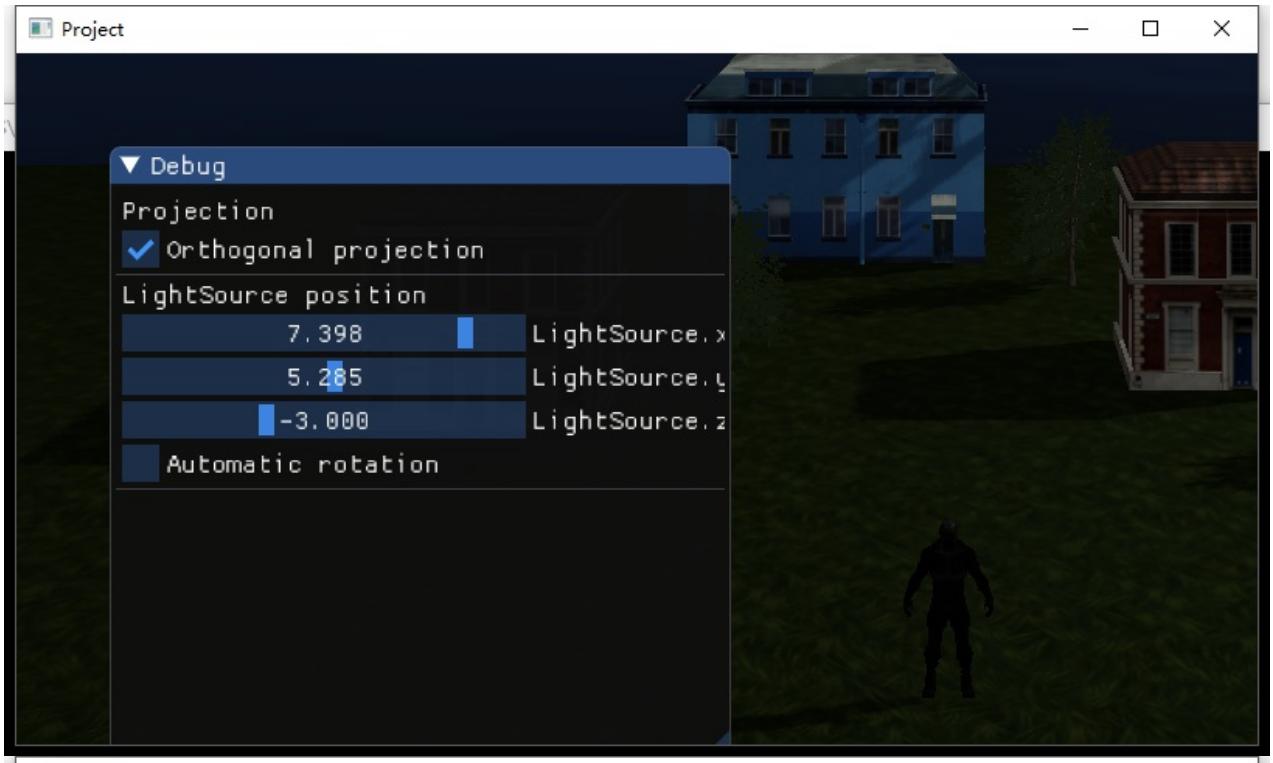
  相机的视角会跟随键盘的上下左右键或鼠标的移动进行变换

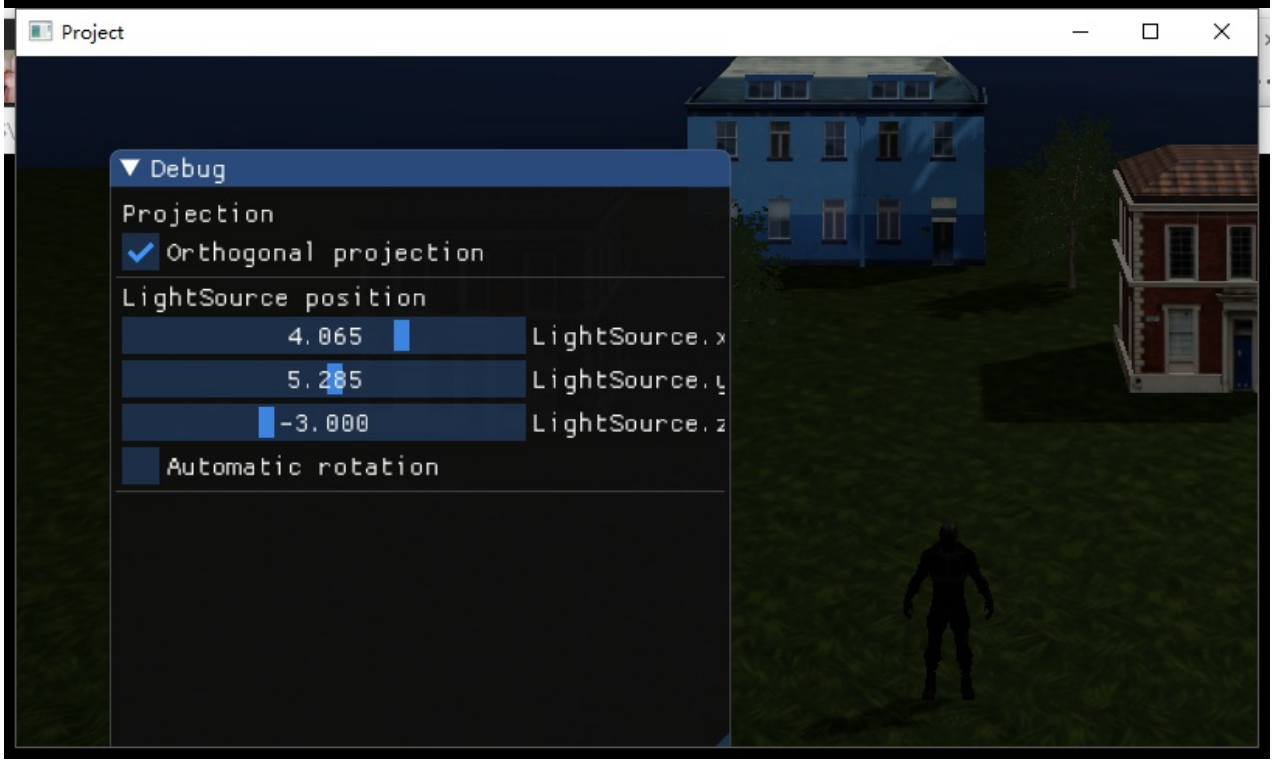
  实际移动效果见效果展示

- simple lighting and shading

```
lightView = glm::lookAt(lightPos, glm::vec3(0.0f), glm::vec3(0.0, 1.0, 0.0));
lightSpaceMatrix = lightProjection * lightView;
// render scene from light's point of view
simpleDepthShader.use();
simpleDepthShader.setMat4("lightSpaceMatrix", lightSpaceMatrix);
glViewport(0, 0, SHADOW_WIDTH, SHADOW_HEIGHT);
glBindFramebuffer(GL_FRAMEBUFFER, depthMapFBO);
glClear(GL_DEPTH_BUFFER_BIT);
glActiveTexture(GL_TEXTURE0);
glBindTexture(GL_TEXTURE_2D, woodTexture);
```

场景中模型的影子变化会随着光源位置的改变而改变





- texture mapping

生成纹理

```
const unsigned int SHADOW_WIDTH = 1024, SHADOW_HEIGHT = 1024;
unsigned int depthMapFBO;
glGenFramebuffers(1, &depthMapFBO);
unsigned int depthMap;
glGenTextures(1, &depthMap);
glBindTexture(GL_TEXTURE_2D, depthMap);
glTexImage2D(GL_TEXTURE_2D, 0, GL_DEPTH_COMPONENT, SHADOW_WIDTH, SHADOW_HEIGHT, 0, GL_DEPTH_COMPONENT, GL_FLOAT, NULL);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_BORDER);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_BORDER);
float borderColor[] = { 1.0, 1.0, 1.0, 1.0 };
glTexParameterfv(GL_TEXTURE_2D, GL_TEXTURE_BORDER_COLOR, borderColor);
glBindFramebuffer(GL_FRAMEBUFFER, depthMapFBO);
glFramebufferTexture2D(GL_FRAMEBUFFER, GL_DEPTH_ATTACHMENT, GL_TEXTURE_2D, depthMap, 0);
glDrawBuffer(GL_NONE);
glReadBuffer(GL_NONE);
glBindFramebuffer(GL_FRAMEBUFFER, 0);
```

例如下面是应用纹理到地面，使用纹理坐标更新顶点数据

```
void renderFloor()
{
  float planeVertices[48] = {
    // positions            // normals         // texcoords
    25.0f, -0.5f,  25.0f,  0.0f, 1.0f, 0.0f,  25.0f,  0.0f,
    -25.0f, -0.5f,  25.0f,  0.0f, 1.0f, 0.0f,   0.0f,  0.0f,
    -25.0f, -0.5f, -25.0f,  0.0f, 1.0f, 0.0f,   0.0f, 25.0f,

    25.0f, -0.5f,  25.0f,  0.0f, 1.0f, 0.0f,  25.0f,  0.0f,
    -25.0f, -0.5f, -25.0f,  0.0f, 1.0f, 0.0f,   0.0f, 25.0f,
    25.0f, -0.5f, -25.0f,  0.0f, 1.0f, 0.0f,  25.0f, 25.0f
  };
```

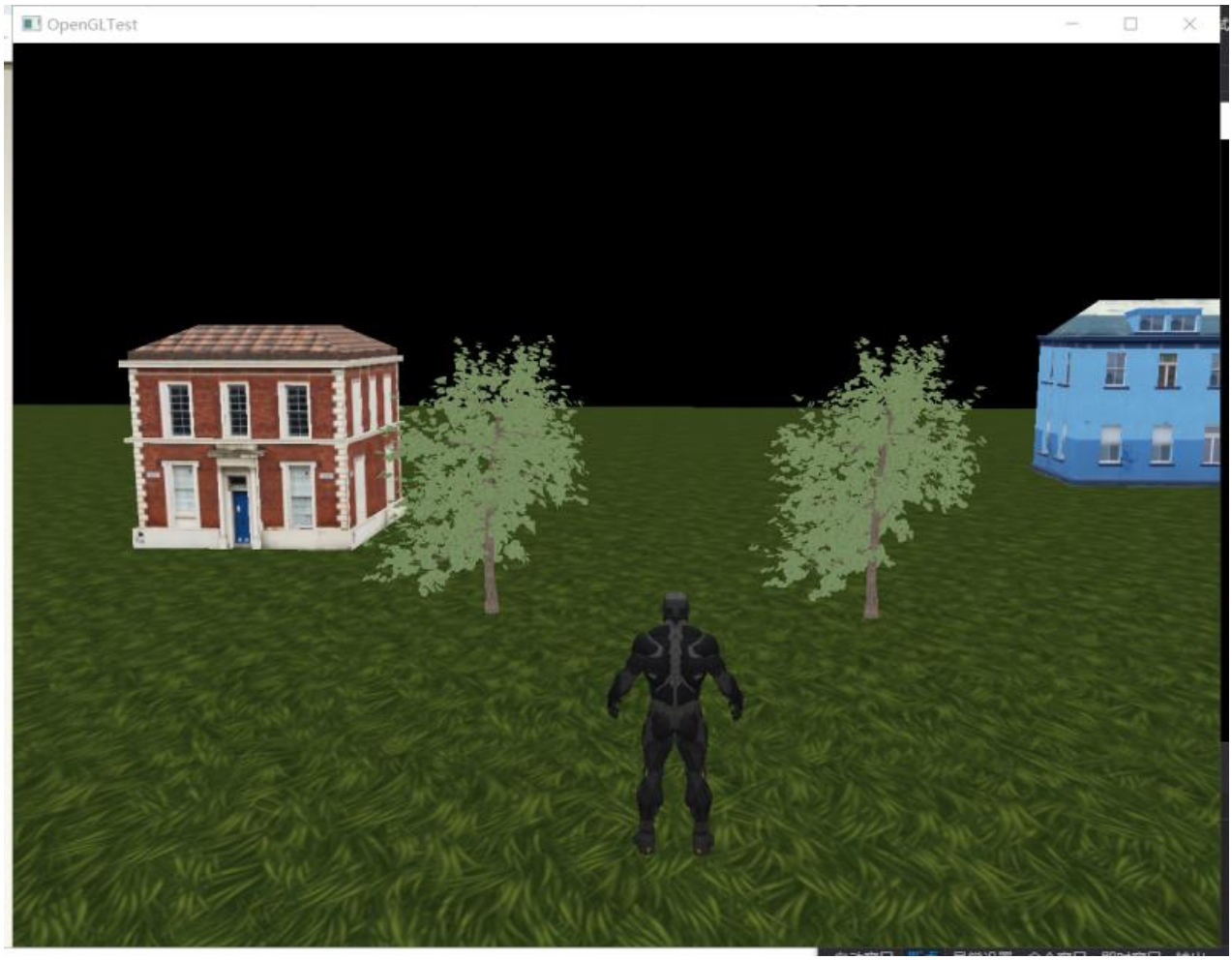
- model import & mesh viewing

```
Model tree("resource/tree/tree.obj");
Model player("resource/nanosuit/nanosuit.obj");
Model house1("resource/Building Apartment/Mesh/Building_Apartment_10.fbx");
Model house2("resource/Building Apartment/Mesh/Building_Apartment_13.fbx");


vector<Model> allModels;
allModels.push_back(player);
allModels.push_back(tree);
allModels.push_back(house1);
allModels.push_back(house2);
```

```
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(6.0f, -1.5f, -4.0f));
model = glm::scale(model, glm::vec3(0.05f, 0.05f, 0.05f));
shader.setMat4("model", model);
allModels[1].Draw(shader);
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(1.0f, -1.5f, -8.0f));
model = glm::scale(model, glm::vec3(0.05f, 0.05f, 0.05f));
shader.setMat4("model", model);
allModels[1].Draw(shader);
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(-1.5f, -1.5f, -3.5f));
model = glm::scale(model, glm::vec3(0.05f, 0.05f, 0.05f));
shader.setMat4("model", model);
allModels[1].Draw(shader);
model = glm::mat4(1.0f);
model = glm::translate(model, glm::vec3(3.5f, -1.5f, -3.5f));
model = glm::scale(model, glm::vec3(0.05f, 0.05f, 0.05f));
shader.setMat4("model", model);
allModels[1].Draw(shader);
```

将模型导入场景后

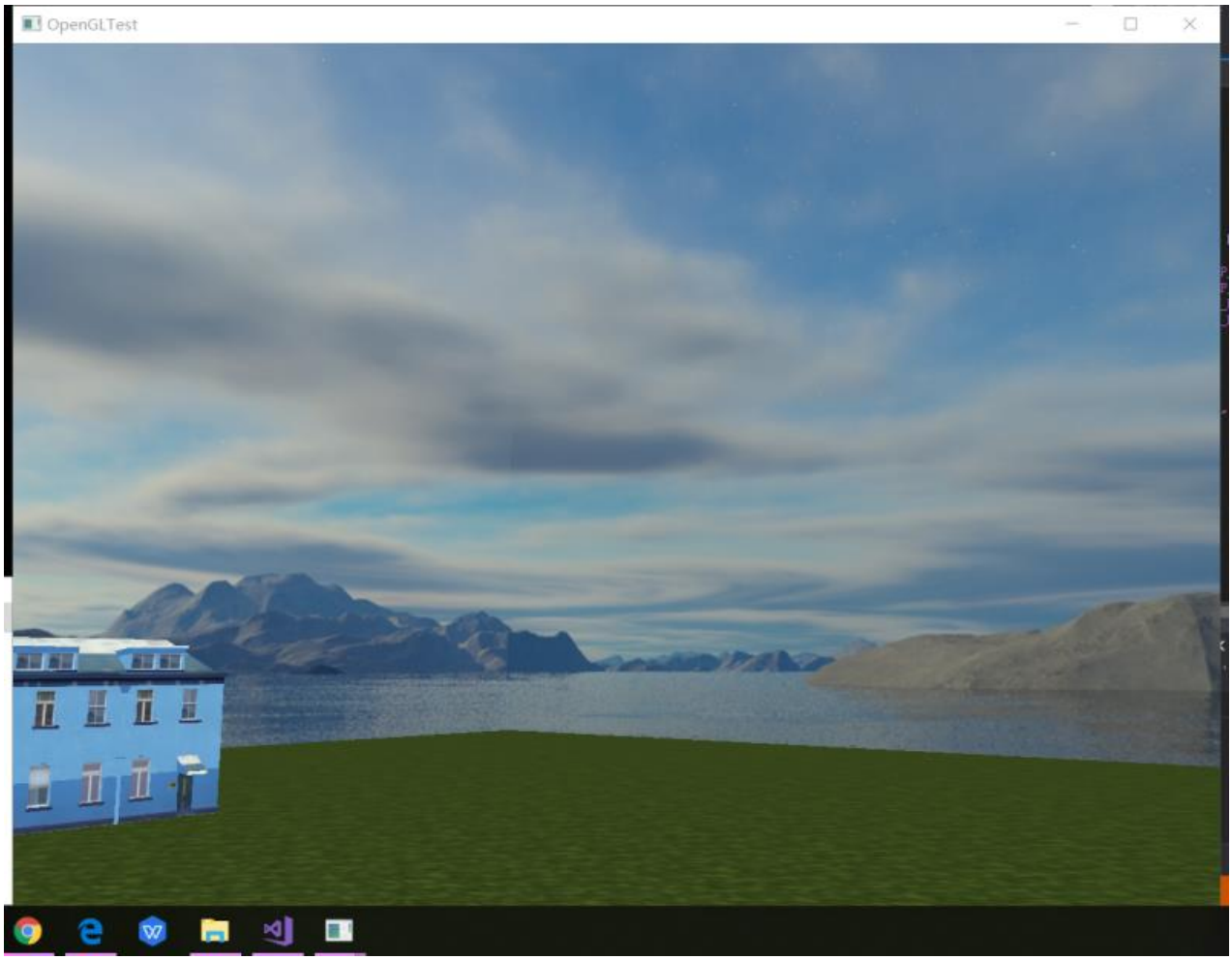


- sky box

```
vector<std::string> faces
{
  "resource/skybox/left.tga",
  "resource/skybox/right.tga",
  "resource/skybox/top.tga",
  "resource/skybox/bottom.tga",
  "resource/skybox/back.tga",
  "resource/skybox/front.tga"
};
unsigned int cubemapTexture = loadCubemap(faces);

shader.use();
shader.setInt("diffuseTexture", 0);
shader.setInt("shadowMap", 1);
```

创建天空盒后



遇得到的问题以及解决方案

1. 在将小组成员各自完成的内容结合起来的时候，出现了多个函数已有主体和重定义的报错。这是由于cpp中重复引用头文件生成了重复的obj，所以在将引用的头文件都只放在hpp里之后问题得到了解决。

小组成员分工

成员	分工	贡献度
谭江华	找素材、建立模型贴图与天空盒	45%
王睿泽	完成简单光照与纹理	45%
熊思佳	找素材、写报告	10%