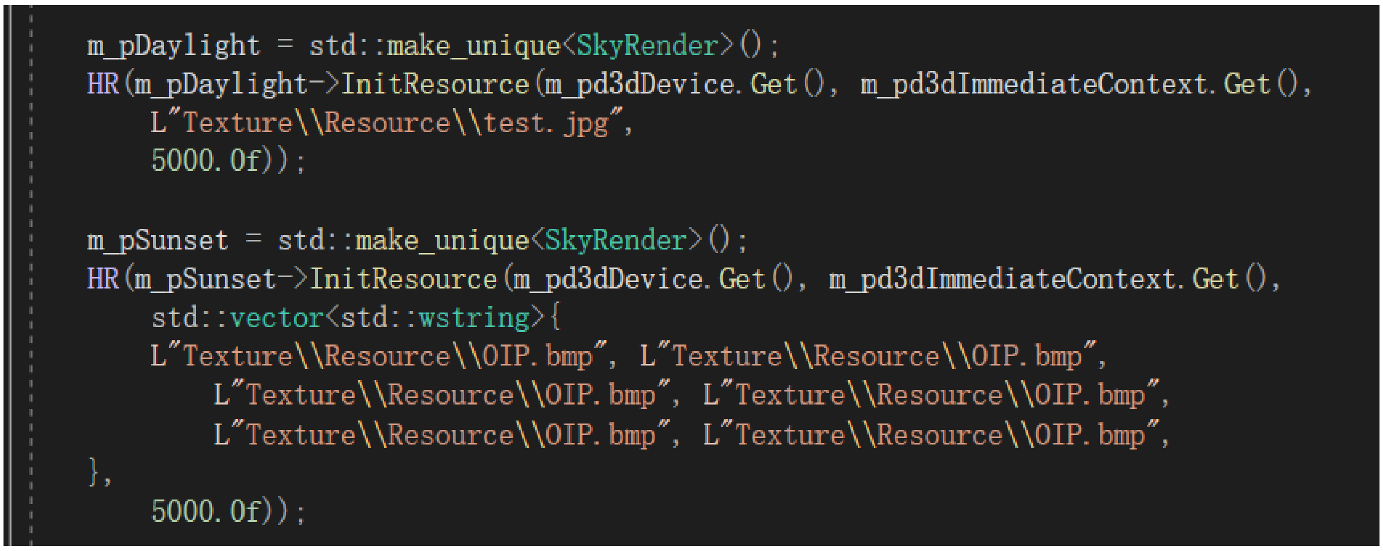
准备工作

新建项目,将DX11对应教程代码的项目22中的HLSL文件夹,Texture文件夹和C++头文件,源文件复制到新建的项目中,并修改相应的配置

天空盒的实现

在网络上寻找贴图,放置在Texture文件夹的Resource文件夹内

替换了原先项目的daylight天空盒和sunset天空盒

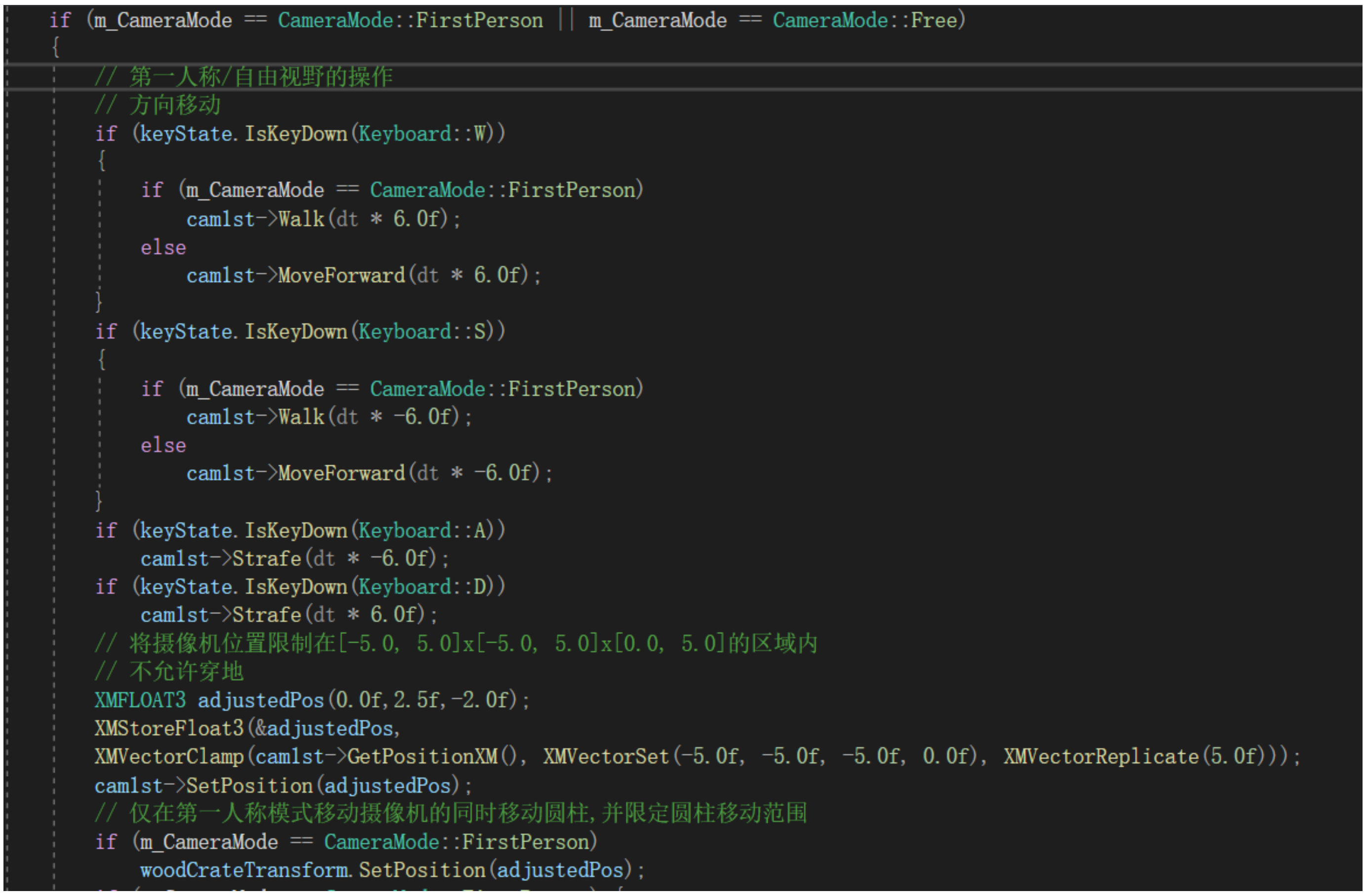


```
    m_pDaylight = std::make_unique<SkyRender>();
    HR(m_pDaylight->InitResource(m_pd3dDevice.Get(), m_pd3dImmediateContext.Get(),
        L"Texture\\Resource\\test.jpg",
        5000.0f));

    m_pSunset = std::make_unique<SkyRender>();
    HR(m_pSunset->InitResource(m_pd3dDevice.Get(), m_pd3dImmediateContext.Get(),
        std::vector<std::wstring>{
        L"Texture\\Resource\\OIP.bmp", L"Texture\\Resource\\OIP.bmp",
            L"Texture\\Resource\\OIP.bmp", L"Texture\\Resource\\OIP.bmp",
            L"Texture\\Resource\\OIP.bmp", L"Texture\\Resource\\OIP.bmp",
    },
        5000.0f));
```

第一人称和自由视野的实现

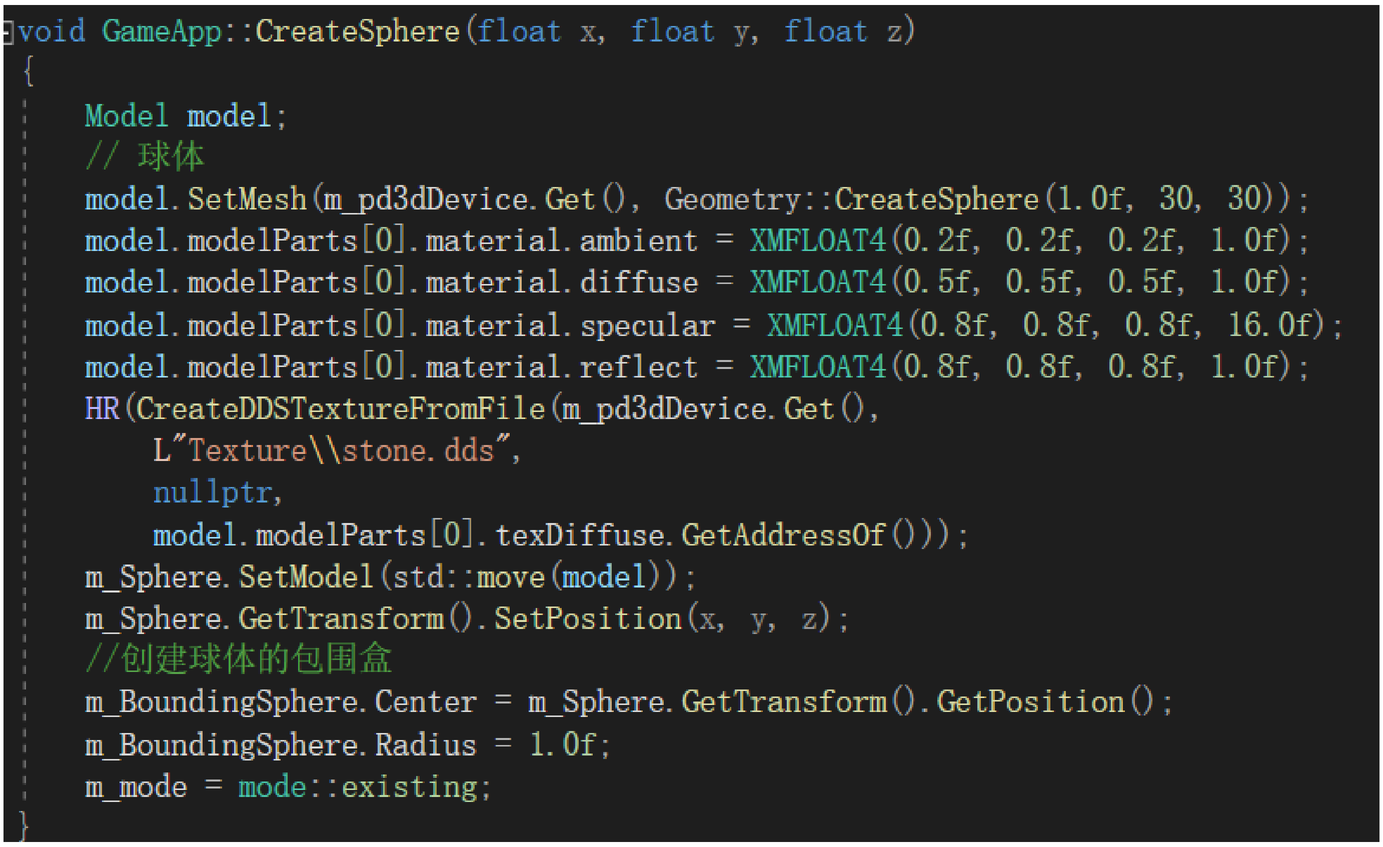
以圆柱作为第一人称,来进行对球体的拾取,放置和摧毁



```
    if (m_CameraMode == CameraMode::FirstPerson || m_CameraMode == CameraMode::Free)
    {
        // 第一人称/自由视野的操作
        // 方向移动
        if (keyState.IsKeyDown(Keyboard::W))
        {
            if (m_CameraMode == CameraMode::FirstPerson)
                cam1st->Walk(dt * 6.0f);
            else
                cam1st->MoveForward(dt * 6.0f);
        }
        if (keyState.IsKeyDown(Keyboard::S))
        {
            if (m_CameraMode == CameraMode::FirstPerson)
                cam1st->Walk(dt * -6.0f);
            else
                cam1st->MoveForward(dt * -6.0f);
        }
        if (keyState.IsKeyDown(Keyboard::A))
            cam1st->Strafe(dt * -6.0f);
        if (keyState.IsKeyDown(Keyboard::D))
            cam1st->Strafe(dt * 6.0f);
        // 将摄像机位置限制在[-5.0, 5.0]x[-5.0, 5.0]x[0.0, 5.0]的区域内
        // 不允许穿地
        XMFLOAT3 adjustedPos(0.0f,2.5f,-2.0f);
        XMStoreFloat3(&adjustedPos,
        XMVectorClamp(cam1st->GetPositionXM(), XMVectorSet(-5.0f, -5.0f, -5.0f, 0.0f), XMVectorReplicate(5.0f)));
        cam1st->SetPosition(adjustedPos);
        // 仅在第一人称模式移动摄像机的同时移动圆柱,并限定圆柱移动范围
        if (m_CameraMode == CameraMode::FirstPerson)
            woodCrateTransform.SetPosition(adjustedPos);
```

鼠标拾取和对球体的破坏,放置,拾取的实现

类似硬件实例化教程中,创建一堆树的方式,自己构造一个创造球体的函数

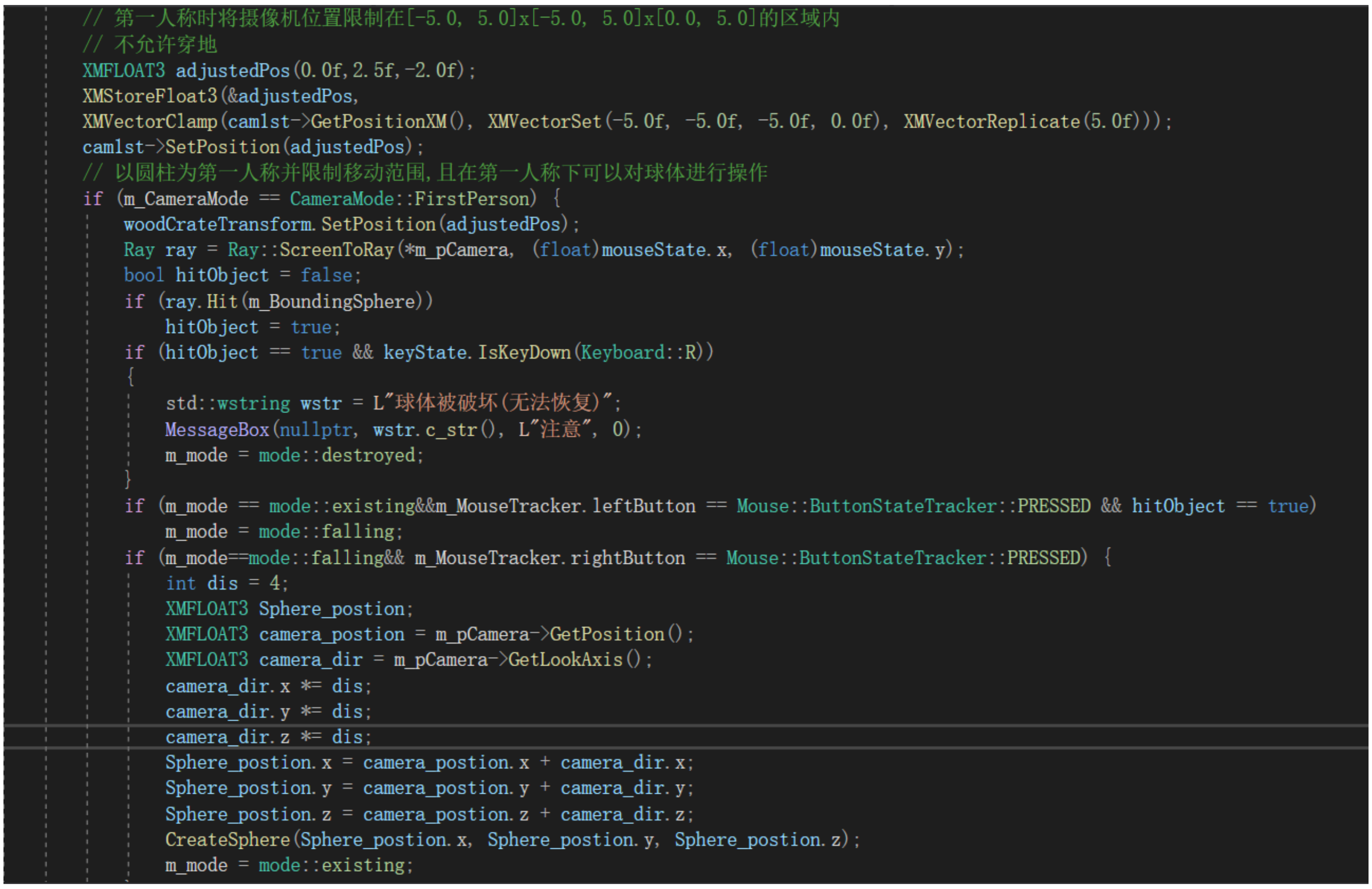


```
void GameApp::CreateSphere(float x, float y, float z)
{
    Model model;
    // 球体
    model.SetMesh(m_pd3dDevice.Get(), Geometry::CreateSphere(1.0f, 30, 30));
    model.modelParts[0].material.ambient = XMFLOAT4(0.2f, 0.2f, 0.2f, 1.0f);
    model.modelParts[0].material.diffuse = XMFLOAT4(0.5f, 0.5f, 0.5f, 1.0f);
    model.modelParts[0].material.specular = XMFLOAT4(0.8f, 0.8f, 0.8f, 16.0f);
    model.modelParts[0].material.reflect = XMFLOAT4(0.8f, 0.8f, 0.8f, 1.0f);
    HR(CreateDDSTextureFromFile(m_pd3dDevice.Get(),
        L"Texture\\stone.dds",
        nullptr,
        model.modelParts[0].texDiffuse.GetAddressOf()));
    m_Sphere.SetModel(std::move(model));
    m_Sphere.GetTransform().SetPosition(x, y, z);
    //创建球体的包围盒
    m_BoundingSphere.Center = m_Sphere.GetTransform().GetPosition();
    m_BoundingSphere.Radius = 1.0f;
    m_mode = mode::existing;
}
```

根据m_mode变量和ray.hit函数的返回值来判断该球体是否可以进入对应的状态

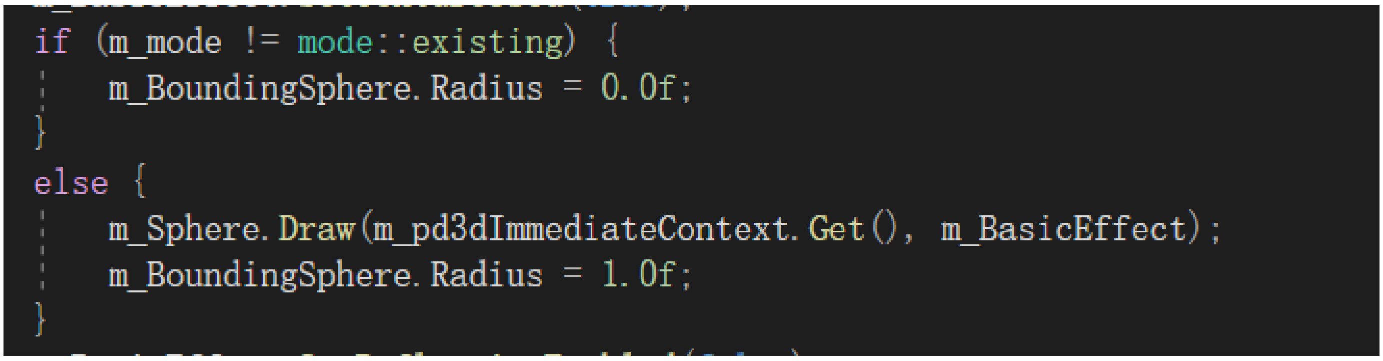
同时在放置时,通过摄像机的位置和摄像机的朝向来得到球体的绝对位置,

从而将球体绘制在摄像机前



```
// 第一人称时将摄像机位置限制在[-5.0, 5.0]x[-5.0, 5.0]x[0.0, 5.0]的区域内
// 不允许穿地
XMFLOAT3 adjustedPos(0.0f,2.5f,-2.0f);
XMStoreFloat3(&adjustedPos,
XMVectorClamp(cam1st->GetPositionXM(), XMVectorSet(-5.0f, -5.0f, -5.0f, 0.0f), XMVectorReplicate(5.0f)));
cam1st->SetPosition(adjustedPos);
// 以圆柱为第一人称并限制移动范围,且在第一人称下可以对球体进行操作
if (m_CameraMode == CameraMode::FirstPerson) {
    woodCrateTransform.SetPosition(adjustedPos);
    Ray ray = Ray::ScreenToRay(*m_pCamera, (float)mouseState.x, (float)mouseState.y);
    bool hitObject = false;
    if (ray.Hit(m_BoundingSphere))
        hitObject = true;
    if (hitObject == true && keyState.IsKeyDown(Keyboard::R))
    {
        std::wstring wstr = L"球体被破坏(无法恢复)";
        MessageBox(nullptr, wstr.c_str(), L"注意", 0);
        m_mode = mode::destroyed;
    }
    if (m_mode == mode::existing&&m_MouseTracker.leftButton == Mouse::ButtonStateTracker::PRESSED && hitObject == true)
        m_mode = mode::falling;
    if (m_mode==mode::falling&& m_MouseTracker.rightButton == Mouse::ButtonStateTracker::PRESSED) {
        int dis = 4;
        XMFLOAT3 Sphere_postion;
        XMFLOAT3 camera_postion = m_pCamera->GetPosition();
        XMFLOAT3 camera_dir = m_pCamera->GetLookAxis();
        camera_dir.x *= dis;
        camera_dir.y *= dis;
        camera_dir.z *= dis;
        Sphere_postion.x = camera_postion.x + camera_dir.x;
        Sphere_postion.y = camera_postion.y + camera_dir.y;
        Sphere_postion.z = camera_postion.z + camera_dir.z;
        CreateSphere(Sphere_postion.x, Sphere_postion.y, Sphere_postion.z);
        m_mode = mode::existing;
```

摧毁和拾取后为确保球体不可见,在drawscene函数中加入判断,判断是否绘制球体,并修改包围球的大小



```
if (m_mode != mode::existing) {
    m_BoundingSphere.Radius = 0.0f;
}
else {
    m_Sphere.Draw(m_pd3dImmediateContext.Get(), m_BasicEffect);
    m_BoundingSphere.Radius = 1.0f;
}
```

已知问题

1,目前的知识储备不足,只能通过定义一个m_mode变量作为标志该球体的三种状态(拾取,放置,摧毁)

2,目前只能对一个球体进行操作,无法创建多个球体,否则无法区分不同球体的不同状态.

(对于问题1,2初步想法是在game object类内定义一个新变量作为标志)

3,ray类似乎是与鼠标的绝对位置作为方向向量,因此在相对模式下只有将球体移到屏幕左上角才能进行操作,

(怀疑原因是鼠标进入相对模式时,绝对位置在屏幕左上角)