**作业内容说明**

项目中模仿Unity实现了一个简单的游戏引擎，引擎包含渲染模块，脚本生命周期管理模块、场景管理模块、资源管理模块等。在引擎的基础上实现了作业中要求的场景。

完成了作业中包括进阶版在内的全部需求，即需求1-11。

1. 使用类库

使用Windows SDK生成窗口等，使用DirectXMath进行部分数学运算，使用DirectXTek进行贴图加载，使用tiny\_obj\_loader.h文件进行模型文件的加载。除此之外未使用额外的类库。

二．控制方法

控制方放和作业要求里基本相同。

1. 汽车操纵

w 前进 s后退 a左转 d右转

值得注意的是在骑车不前进不后退时按a或d骑车只有轮子会转动车身不会转动，这一点和现实中的汽车相同。

1. 相机操纵

相机分为第一人称相机和第三人称相机两种模式，二者按空格键切换。

第一人称相机放置在小车上方，跟随小车移动。在第一人称相机模式下鼠标长按屏幕左右两边即可转动相机朝向。

第三人称相机在较高处，始终保持看向小车。通过鼠标长按屏幕可以实现相机绕小车左右/上下旋转。滚动鼠标滚轮可以拉近/拉远相机。

三．项目结构介绍

项目代码主要分为两部分：引擎部分和应用部分。其中在工程文件中的RacingGame筛选器中的为应用部分，其他为引擎部分。

1. 引擎部分

引擎部分模仿Unity引擎，其中与Unity对应组件功能相近的有：

|  |  |  |
| --- | --- | --- |
| 编号 | 项目中文件 | Unity对应组件 |
| 1 | ArcBehaviour.h/.cpp | MonoBehaviour |
| 2 | ArcCamera.h/.cpp | Camera |
| 3 | ArcGameObject.h/.cpp | GameObject |
| 4 | ArcMaterial.h/.cpp | Material |
| 5 | ArcMesh.h/.cpp | Mesh |
| 6 | ArcScene.h/.cpp | Scene |
| 7 | ArcTexture.h/.cpp | Texture |
| 8 | ArcTime.h/.cpp | Time |
| 9 | ArcTransform.h/.cpp | Transform |

其他文件功能介绍如下：

|  |  |  |
| --- | --- | --- |
| 编号 | 项目中文件 | 功能 |
| 1 | ArcApplication.h/.cpp | 应用程序的抽象，一个app中包含多个Scene |
| 2 | ArcAsserLoader.h/.cpp | 用于各类资源加载 |
| 3 | ArcAsset.h/.cpp | 资源库文件的基类，用于创建、配置与保存资源 |
| 4 | ArcFramework.h/.cpp | 模仿GLFW，对窗口创建等功能进行封装 |
| 5 | ArcGeometry-  Generator.h/.cpp | 用于生成基本模型，如立方体，圆柱，球体等 |
| 6 | ArcGraphicSetting.h | 一些图形相关的配置 |
| 7 | ArcInput.h/.cpp | 对输入功能进行封装 |
| 8 | ArcMath.h | 对数学相关函数进行封装 |
| 9 | ArcRenerder.h/.cpp | 传入材质和网格，对场景中的组件进行渲染 |
| 10 | ArcRenderToTexture.h/.cpp | 用于做RTT相关工作（画阴影用） |
| 11 | ArcRHI.h/.cpp | 模仿UE4的RHI组件，（试图）对图形接口进行封装（还不太行） |
| 12 | ArcStructure.h | 一些数据结构，基本都是顶点结构啥的 |
| 13 | ArcTool.h | 实用小工具函数们 |
| 14 | ArcWindows.h/.cpp | 试图对各个平台的窗口这个概念进行封装 |
| 15 | CommonHeader,h | 常用头文件的集合 |
| 16 | DriverSetting.h | 用于配置用到的资源库和装载的应用程序 |
| 17 | Light.h/.cpp | 光照 |
| 18 | SimpleVertex.h/.cpp | 简单的顶点数据结构，用于Debug |
| 19 | Main.cpp | 程序入口，用于启动整个项目并开启主循环 |

1. 应用部分

应该部分分为资源文件、场景文件和脚本。

本项目中的资源文件为RacingGameAssets.h/.cpp。该脚本继承ArcAsset.h，用于加载项目中用到的网格、纹理、材质等所有资源。

项目中的应用文件有DebugApplication.h/.cpp和RacingGameApplication.h/.cpp两个，前者用于debug，后者为实际应用文件。应用文件通过资源名称获取资源并使用资源配置场景以及其中的GameObject。Main.cpp会轮询每个GameObject中的组件，执行渲染工作或运逻辑脚本。

剩下的几个文件为挂载在GameObject上的逻辑脚本。其中CameraController.h/cpp控制摄像机运动逻辑；CarMove.h/.cpp控制小车运动逻辑；SkyBoxFollowCamera.h/.cpp用于同步天空盒和相机的位置；WheelMove.h/.cpp用于控制车轮的运动逻辑；DebugCameraMove.h/.cpp用于控制用于调试的摄像机（实际项目未使用该逻辑脚本）。

三．闲扯几句

这个应用本来是想写成个适配多底层硬件接口的引擎的然而写到一半发现，一方面工作量实在太大，另一方面我都知识储备还不足以让我这么写，所以只能先以之前写的OpenGL引擎为原型搭一个简单的框架并先在框架上先实现功能，剩下的东西在继续学习后再继续添加或重构。

相比于之前用OpenGL写的引擎这个作业中大致有以下两点提升：

1. 将资源和场景分离，ArcAsset的派生类中加载资源后将资源存入库中，场景类再根据名称从库中获取资源。按理说用资源名hash出一个GUID来唯一标识资源可能会更好些，之后有空改改。
2. 把跨平台的一些位置留了出来，比如把平台独立层、数学库和窗口抽象出来等。但抽象的还有不少问题之后去学学虚幻RHI层的实现。

老实说写这东西花费时间不少，比起直接把网上大佬的框架搬过来用，自己搭建框架花费的精力实在是多太多。但这个过程也给我带来了一定的提升。例如熟悉了DX的封装和api、回顾了线性代数的知识、加深了对渲染管线的理解等。时间有限，接下来还是先写后面的作业，等到再有空了再去继续完善这玩意。

鄙人才疏学浅，写的东西难免有一些我无法发现的缺点和漏洞。如果大佬们在批改过程中发现了什么可以改善的地方欢迎您批评指正，在此先行谢过Orz。

——宋文博

2020年2月26日