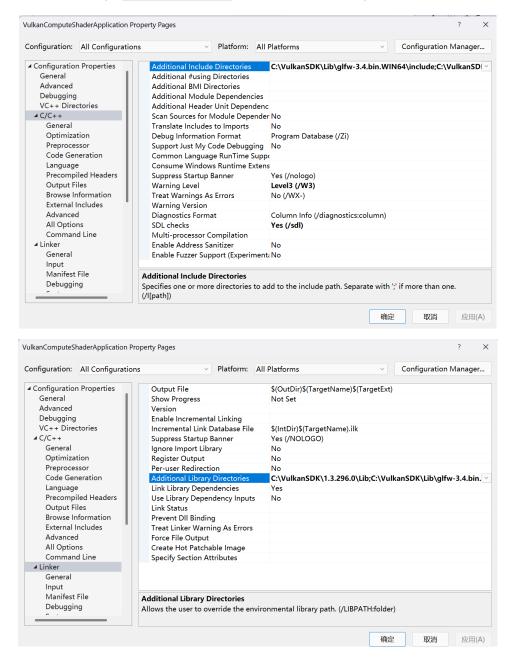
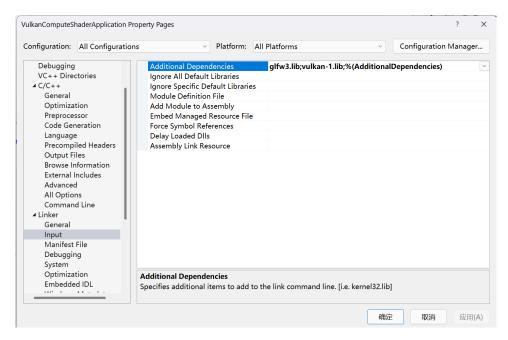
Final Project: minRTRender

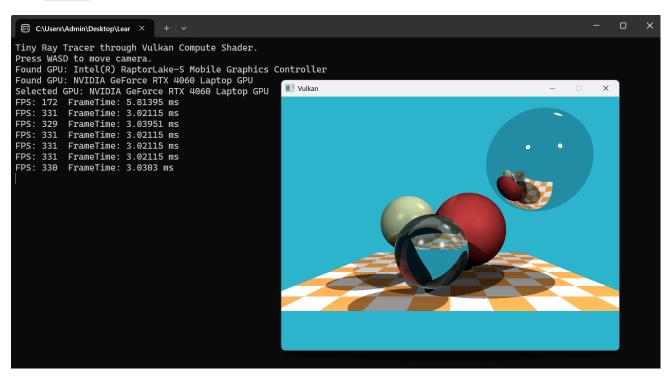
1. 编译运行基础代码(Vulkan Compute Shader)

- 使用 VS2022 打开 RTVulkan.sln 文件。
- 检查并更改以下项目设置(和 Vulkan Turorial 中的环境配置步骤相同)





• 在 VS2022 中运行项目。



2. 框架代码简介

- 框架代码基于 <u>Vulkan Tutorial Compute Shader</u> 中的粒子系统示例代码,可通过阅读该章了解Vulkan Compute Shader工作原理。
- 在提供的基础代码中,共有两个渲染管线,一个是graphics管线,另一个是compute shader管线,这两个管线相对独立,需要分别设置 descriptor set等属性。
 - o graphics管线用到了 shader.vert 和 shader.frag。通过阅读shader代码可以看到,graphics管线只渲染了一张带有纹理的矩形,纹理由compute shader管线计算得出。
 - o compute shader管线用到了 shader.comp 。通过阅读shader代码可以看到,shader.comp的输入为光线,场景,灯光,背景,三角形网格等数据,这些数据用于光线追踪的计算,输出为image2D数据,输出的数据即为graphics中使用的纹理。
- 在阅读框架代码的过程中, 你应该重点注意以下内容:

- o compute shader管线和graphics管线在设置上有什么不同,compute shader管线计算出的纹理是怎样从 传入graphics管线的?
- 。 数据是如何从CPU端传入compute shader管线的,又是怎样从compute shader管线中取出的?
- o 基础shader.comp代码在结构上与基于CPU的实现有何不同?glsl语法不允许使用函数递归,如何通过使用数据结构解决这一问题?

其他:

- 项目优先选择独立显卡,可在 main.cpp 中更改 preferHighPerformanceGPU 的值切换使用集成显卡
- 默认分辨率为 1024x768。可通过修改 main.cpp 和 shader.comp 中的 width 和 Height 的值改变分辨率。
- 。 使用包围盒对模型光追计算进行优化, 经测试可获得近一倍的性能提升。

3. 实现最终效果

- 最终效果参考 Final Project Ref 中的程序。
- 为方便同学们进行实现,main.cpp 和 shader.comp 中用 TODO 标记了需要实现或者修改的代码,同时给出了部分提示。
- 如果不想使用助教给的框架,也可以自行实现,最终评分以效果为准。