开发日志.md

文档跳转

技术文档

测试文档

演示视频

2024年

10.23

完成选题

10.24

1.组员开始进行对QT技术的学习与实践。

2.安装QT Creator等相关软件。

3.使用Qmake搭建第一个QT项目，熟悉QT中提供的各种组件与模块。

10.25

1.组员开始对计算机图形学相关知识进行学习。

2.在QT中引入OpenGL模块，并使用该模块自定义一个图形窗口。

10.26

1.学习着色器相关知识。

2.编写顶点着色器与片段着色器，实现了一个简单的2D渲染。

10.27

1.翻阅QT RHI官方文档，尝试使用RHI的接口实现经典OpenGL任务

2.尝试失败，使用RHI的过程中频繁报错，查询原因，缺乏社区支持，无法debug。

10.28

1.查阅官方文档，进行多番尝试，没有找到报错解决方法。

2.联系官方，无应答。

10.29

1.了解到QT RHI接口是OpenGL，Vulkan等的抽象与封装，现阶段缺乏社区支持与学习案例。

2.经过全体组员商讨，决定使用QT中内置的OpenGL模块进行项目功能实现。

10.30

1.决定使用加权平均计算解决图片主色调计算。

2.在使用GPU计算主色调前，使用CPU进行计算验证方法可行性。

CPU方式:设置步长，遍历整张图片所有像素点的方式统计RGB直方图信息。

10.31

调试并完成了使用CPU进行主色调计算的demo项目

11.13

1.着手进行使用GPU对主色调进行计算的开发

2.OpenGL4.3版本开始支持计算着色器(compute shader),使用OpenGLFunctions4.5 Core。

3.项目构建从Qmake更换为CMake

11.14

1.学习GLSL，编写compute shader

2.进行简单地像素点统计。

11.15

1.尝试将图像传入GPU进行处理，失败。

2.将图像转换为OpenGL纹理后，传入GPU进行读取尝试，成功。

3.各个工作区访问缓冲区同一位置产生竞争，利用原子计算的原子性，解决竞态问题。

4.成功实现了对整张图片的像素点个数统计，统计出的像素点个数随图片尺寸增大而增加。

11.16

1.尝试用各工作区采集图片像素点的RGBA信息，并传回CPU，失败。

2.检查到是图片与纹理转换出错，查阅文档后进行了更改，成功实现在GPU的局部工作区中采集像素点RGBA信息。

3.CPU能正常从缓冲区中取出RGBA直方图，并成功计算出图片的主色调颜色信息。

11.17

1.调试发现bug：统计出的RGBA直方图中发生溢出。

解决方案：采用能承载更大正数范围的类型uint。

2.调试发现bug：在单次程序启动后，连续进行多张图片的主色调计算时，计算结果与预期不符，且同一张图片的计算结果与之前不同。

原因:计算结束前没有处理缓冲区中直方图的数据，对之后的计算产生影响。

解决方案：每次计算前，将缓冲区中直方图的区域置0，使上一次统计结果不会影响到下一次计算。

11.18

新增用户交互界面，提供图片上传功能和主色调信息显示。