计算机图形学 Project 3 文档

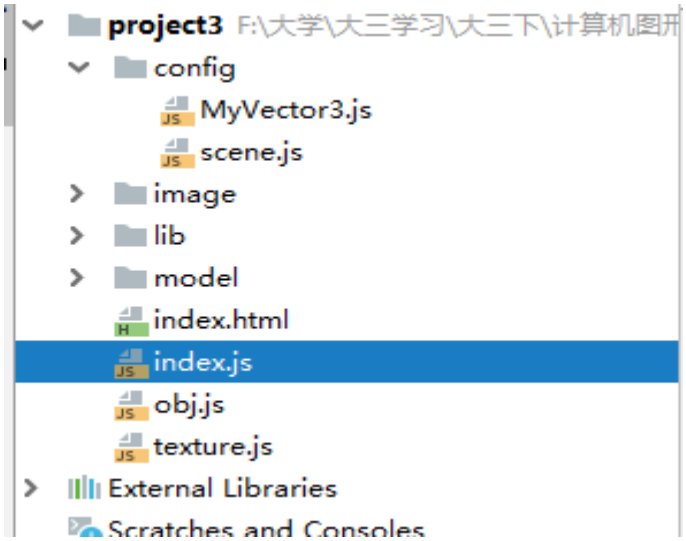
开发及运行环境

开发环境：PhpStorm 2019.1.2
运行环境：Chrom 浏览器 75+
　　FireFox 浏览器 67.0.2

运行方法

1：使用设置了允许跨域访问的 Chrom 浏览器或 Firefox 浏览器打开 index.html
2：使用 PhpStorm 打开文件目录，在其中运行 index.html

项目目录与文件说明



如上所示：
config 文件夹包含配置文件 scene.js 以及计算工具 MyVector3.js
image 文件夹包含纹理的图片
lib 文件夹包含一些 js 库
model 文件夹包含需要的 obj 文件
index.html：程序的 html 文件
index.js：主要的 js 文件，在其中实现了程序的逻辑
obj.js：实现一些对 obj 对象操作函数
texture.js：实现一些对纹理的操作函数

主要功能实现逻辑

加载纹理与 OBJ

1、 纹理和 OBJ 的加载采用了 switchShader 进行切换绘制
2、 创建纹理和 OBJ 的 program 时，会初始化其顶点并存储到 buffer 中，每次切换不需要再次初始化顶点
3、 加载 obj 时，使用同一个 program，遍历初始化好的 obj 数组，进行绘制 obj

键盘事件

1、 按下键盘时会将当前键存入一个自定义的键盘数组中
2、 每次判断事件时遍历数组，判断数组元素是哪个键，执行对应的处理方法
3、 每次松开键时，会键对应键从数组中删除

WASD 移动

1、 对于 WS 前后移动，首先获取 eye 到 at 的单位向量（视线向量），代表移动方向，然后计算上一帧到这一帧中应该移动的距离
2、 如果是 W，就将 eye 向量和 at 向量同时加上距离向量，如果是 S，就讲 eye 向量和 at 向量同时减去距离向量
3、 针对 AD 左右移动，首先需要获取视线向量与 up 向量的法向量，代表移动方向（需要根据），并计算上一帧到这一帧的移动距离
4、 根据 A 或者 D 选择加上或减去距离，更新 eye 和 at

IJKL 旋转

1、 针对 I K 旋转，首先需要更新 up 向量，比较便捷的方法是先找到 up 和视线向量所在平面的法向量 n，再根据 n 和视线向量获取其法向量，即为 up 向量
2、 根据 up 向量以及上一帧到这一帧应该旋转的角度，计算出移动向量，最后 eye 向量与移动向量相加，即可获取 at 所在位置
3、 针对 J L 旋转，首先根据 up 和视线向量获取他们的法向量，即旋转方向，不需要更新 up
4、 根据 up 向量以及上一帧到这一帧应该旋转的角度，计算出移动向量，最后 eye 向量与移动向量相加，即可获取 at 所在位置

光照与点光源

1、 对于光照，会在 obj 的 shader 进行绘制，根据其法向量等计算环境光、方向光
2、 在每一帧处理键盘事件时，如果是 F，就设置一个变量 isPointLight 为 true，代表需要点光源
3、 在每一帧设置光照时，根据变量设置，如果不需要点光源，则点光源设为 0，需要的话就设置点光源即可

动画

1、 在遍历绘制 obj 的时候，判断当前 obj 是否需要动画，（默认只有第二个 obj 需要），不需要的根据配置文件设置即可
2、 对于需要动画的 obj，即小鸟，需要获取计算小鸟当前帧旋转的角度
3、 设置小鸟的 modelMatrix，设置其缩放、根据角度旋转、平移，在平移时，y 向量是根据当前旋转角进行正弦计算获得

Bonus 完成

实现纹理与光照结合

1、 在 texture 的 shader 中，需要传入光源的信息
2、 计算方法与 obj 中的计算光源的方法类似，不过不同在于在 texture 的顶点着色器中没有 a_Color，所以计算主要放在片元着色器中完成
3、 在片元着色器中，通过 vec4 color=texture2D(u_Sampler, v_TexCoord)获取的变量 color 即与 a_Color 类似，代表纹理的颜色，所以通过使用 color 与光线计算，以此产生关线效果

雾化效果（同时实现了 obj 和纹理的雾化）

1、 根据书本教程，在 shader 中加入 fogColor、fogDist 等变量
2、 通过计算、混合颜色，即可实现雾化效果

phong shading

1、 将 obj 的 shader 的计算光线等过程移入片元着色器中
2、 在片元着色器中进行计算关线

项目亮点

1、 **实现了纹理雾化和 obj 雾化：**实现雾化的时候，不仅仅实现了 obj 雾化，也按照 obj 雾化的方法，实现了纹理雾化
2、 **性能较 sample 差不多或较好：**经过实验发现，移动旋转等操作与 sample 的速度没有什么差别，但是打开控制台查看内存的时候，发现我实现的 pj 需要的内存会比 sample 小一点点

遇到的问题

1、 在加载纹理的时候，已经按照教程实现正确，就是无法显示纹理
后面发现是由于仅仅绘制了一次纹理，而且图片采用异步加载方式，在绘制纹理时图片还没有加载完成，所以无法显示纹理
2、 进行移动旋转的时候，不知道如何更新向量
询问同学寻求解决方案，最后根据同学的说法，结合一些数学知识，每次会获取移动或者旋转方向上得单位向量，根据该向量进行更新 eye 和 at
3、 实现动画时，不知道怎么显示动画
通过学习与询问，最终绘制 obj 的时候，进行判断那个 obj 需要动画，并对该动画计算旋转角度等变量，进行设置 model 变化矩阵，实现动画
4、 实现纹理与光照结合时，不知道怎么获取 a_color
经过研究 shader 的代码与理解，发现在片元着色器中的纹理颜色即可代表平常的 a_color，所以在实现纹理与光线结合时，计算纹理光线颜色等操作放在了片元着色器中进行

存在缺陷

1、在第一次加载时，由于 obj 和图片存在，需要较长时间加载，打开刚开始运行时，会较慢显示画面