**Computer Graphics Final - Documentation**

todo: 改语句，加截图，加代码

一、基本介绍

Nova是一个基于OpenGL的太阳系模拟系统。

二、基本操作

W键向前移动，S键向移动，A键向左移动，D键向右移动，Q键向上移动，E键向下移动。

1键可以切换显示网格模型或面模型，2键可以切换地球的着色器（普通的着色器和优化后的着色器），3键可以切换粒子模型。

三、设计思路

四、技术实现

五、难点问题

1. 整体

软件整体使用OpenGL的核心管线模式，使用VBO, EBO, shader, 实例化等技术以获得更高效的性能。使用Assimp库进行模型的读取，stb\_???进行纹理图片的读取，???进行文字的渲染显示。

1. 粒子系统

粒子系统用于控制宇宙中飞行的陨石。陨石分为两类，一类是由大量陨石绕着太阳进行公转，构成太阳系中的小行星带。另一类则在宇宙中任意飞行，从指定边界上随机生成，并朝太阳系中心附近进行匀速直线运动，直到离开指定边界，进行销毁。

粒子的显示使用实例化技术，一次性将所有粒子的空间信息发送到GPU，极大的提高了整体性能。

刚开始使用了多个指针分别存储粒子的各项物理信息，后添加了一个Class将所有信息放在其中，使得访问数据时更好的满足Locality，提升了不少帧率。

之后经调试发现，此时性能的瓶颈主要在于陨石旋转运动时调用glm::rotate()带来的过高性能损耗，不使用glm::rotate()时的帧率在100左右，而使用了glm::rotate()后帧率降到了40。但是陨石全都不旋转的话，场景整体感觉不够真实，顾折衷使25%左右的陨石不旋转，帧率提高到了60左右，但场景的变化不仔细看的话却难以察觉。

最后还优化了一点，并不删除死亡的粒子，而是将其速度反转重新摄入宇宙中，获得了少量的帧率提升。

【随机生成算法】

3. 星球显示

地球使用了4个贴图，一个普通的颜色贴图，一个法线贴图以使陆地有微微的突起感，一个高光贴图以使水面的高光亮度较高而陆地几乎没有高光，另外有一个夜晚贴图，使得地球在背对太阳的半球上，陆地会亮起星星点点的灯。

看各种各样电影里的宇宙场景可以发现，星球想要有更加真实惊艳的外观，离不开边缘上渐变的亮光。原本看网上的各类攻略，似乎需要使用HDR保存高亮度区域，并使用帧缓冲提取此区域进行高斯模糊以进行实现，实现过程总体十分繁琐。几经研究后改用了一个较为简单但是效果还不错的方法，在星球的着色器中，通过对视线和法线角度的判断以判断物体边界，进而对边界进行额外的亮度提升，以实现星球的边缘照明，获得较为生动的场景同时，对性能几乎没有影响。此时各个星球都有微微发亮的感觉，而且太阳也让人觉得更亮了。

4. 模型

鉴于大多数模型都是相同的网格结构但不同的贴图，对模型读取部分进行了改良，使得模型读取时可以使用自己指定的纹理而不是mtl中设定的纹理，减少了模型的存储空间及读取时间。

粒子的模型可以切换为一个Low-poly风格的宇宙机器人，这个模型是我自己用Cinema 4D进行建模的，虽然较为简陋，但是实现过程还是耗了不少时间。Cinema 4D导出的不少格式在我使用的Assimp模型读取库中会产生贴错纹理位置的情况，研究一番并不知道怎么解决。而Cinema 4D生成Obj格式文件只有少部分可以正常读取显示，复杂的模型会出现读取后卡死的情况。经调试发现，问题在于部分生成的obj文件中会没有法线输入，故需要自己在读取的时候根据indices进行生成，这才终于将自己的模型放入了场景中。

2. 天空盒

天空盒用于显示宇宙中的浩瀚的星辰背景，使用立方体贴图实现。

6. 文字显示

使用什么什么库，

5.未解决的难点问题

Cinema 4D自制的模型远看会有裂缝。

OpenGL的抗锯齿不生效，调用设置查看语句发现显卡并未分配抗锯齿缓冲。

好不容易找到了切换到独显的宏定义，但是奇怪的是在粒子静止的时候独显的性能高于集显，而粒子移动的时候独显的性能低于集显。