

计算机图形学 – Final Report

1 项目介绍以及实现结果

Over the sea, under the water 的灵感来自于台湾乐团 Cicada 的同名歌曲。原歌曲是为了纪念莫拉克水灾逝去的人而作，意在表达水面上下生死相隔的主题。我们取水上水下的意象，创作了一组对照鲜明的场景。不同的人欣赏我们的作品时会有不同的感受——哪些东西是你留给自己的秘密？哪些光鲜是你挑选出来给别人看的？飞翔的鱼宛如诡谲的梦呓，海底的幽暗场景好比心底的一片秘密。我们想展现的就是这样的一组抽象，但是每个人都很熟悉的，外在世界和里世界的对比。

实现的过程中，我们希望将这个对比清晰的展现出来。于是我们挑选了很多夸张的表现形式：飞翔的鱼，汪洋中的孤岛，旋转的红色云彩。我们将这些具有冲击力的，有些特别的意象放置在海面上，尝试表达这个追求独特、追求个性的外在世界。而对于海底，我们挑选了很多常见的阴暗形式，像是海草、泡泡等。我们将这些常见的形象放置在海底，用以表现“太阳底下无新事，大家本质都是类似的孤独”的里世界。

经过大家的努力，我们的成果截图如下：

- 海上世界：



- 海底世界：



2 开发环境以及使用到的第三方库

- WebGL
- Three.js

3 实现功能列表(Basic与Bonus)

3.1 Basic

- Camera Roaming
- Simple lighting and phong shading
- Texture mapping
- Shadow mapping
- Cloth Simulation
- Model import & Mesh viewing

3.2 Bonus

- SkyBox
- Display Text
- Stencil Test
- Complex Lighting
- Gravity System and Collision Detection
- Skeletal Animation
- Particle System
- Explosion Effect

4 实现的功能点的简单介绍(Bonus主要介绍实现原理)

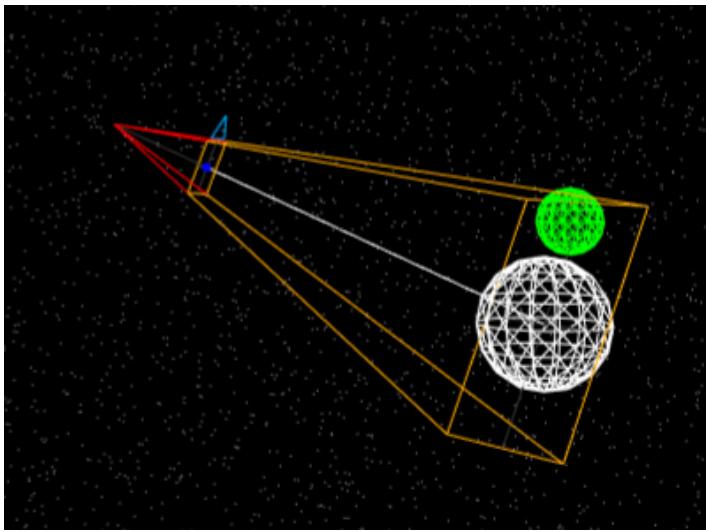
4.1 Basic

4.1.1 Camera Roaming

实验原理:

使用 THREE 的透视摄像机 PerspectiveCamera, 其属于 THREE 中的 Object3D 类, THREE 对 Object3D 封装了很多方法, 要改变其 position, rotation 等属性直接赋值即可, 除此之外还有 lookAt() 等方法

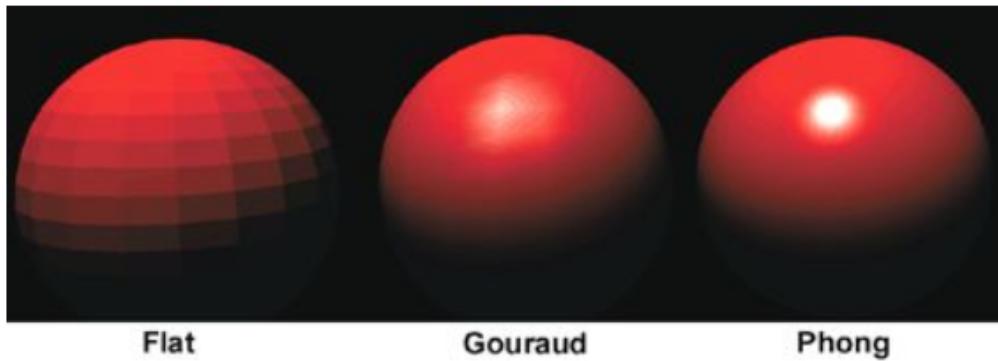
实验截图:



4.1.2 Simple lighting and phong shading

实验原理:

THREE 中封装了对应光源对象，本场景使用了一个环境光 THREE.AmbientLight(0x808080, 0.5) 和若干平行光，shadings 的方式由每个物体的 material 决定，本场景绝大多数物体使用了 MeshStandardMaterial，THREE 的标准材质，其使用的即为冯氏光照模型，但是由于本场景为低多边形风格，所以设置了平面着色（flat shading），并且提高了材质的粗糙度，以减少镜面反射。



实验截图:



4.1.3 Texture mapping

气泡，太阳光晕，海底发光的浮游生物

实验原理:

在THREE中，创建一个物体时，需要几何体geometry和材质material两个要素，在创建材质时，初始化map属性为载入的贴图，即可在物体的表面贴上纹理。

实验截图:



4.1.4 Shadow mapping

实验原理:

使用一个正交摄像机生成阴影，在 THREE 中，该摄像机挂载在 DirectionalLights 方向光下，设置其对应的参数即可实现（贴图大小，边界等，可以用 GUI 打开阴影摄像机的 helper）。同时对于需要投射和接受阴影的物体，需要设置其对应的 castShadow 和 receiveShadow 属性。

实验截图:



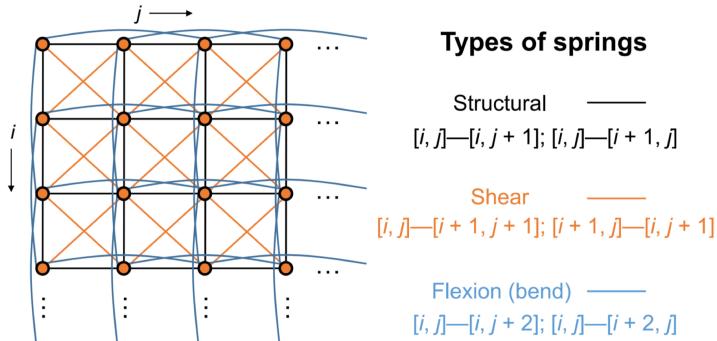
4.1.5 Cloth Simulation

实验原理:

旗帜受到两个外力，恒定的重力与实时更新风向与风强的风力，旗帜内部还存在三角形的各顶点之间的作用力，structure springs、shear springs和bend springs：

Mass-Spring System for Cloth Simulation

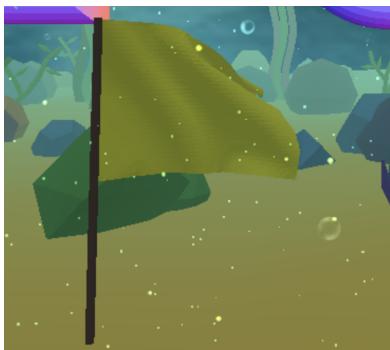
- Consider a rectangular cloth with $m \times n$ particles



具体实现：

- 通过几千个三角面片绘制一面旗帜，在初始化三角面片时，同时也将每个点受到的结构力、剪切力和弯曲力进行记录；
- 每帧随机生成风的方向和强度，当受到风作用时，计算风向和每个三角面片法线的夹角求得风导致的位移，结合重力产生的位移，可得出理论上在外力作用下的位移
- 外力作用的位移会受到内部各点的牵制与拖曳的影响，得到每一帧每个三角形实际的位移。
- 在无穷问题离散化上采用欧拉方法。

实验截图：



4.1.6 Model import & Mesh viewing

实验原理：

使用 THREE 的官方 Loader 导入各种格式的模型，如 THREE.OBJLoader() 等，若存在关联的材质和贴图，则会自动导入，也可以通过 THREE.MTLLoader() 单独导入材质。导入异步进行，导入成功后添加到场景并进行相应更改。

本实验中岛屿上的树皆为模型导入，模型来源包括自己建模和在已有的开源模型基础上做出更改。

实验截图：

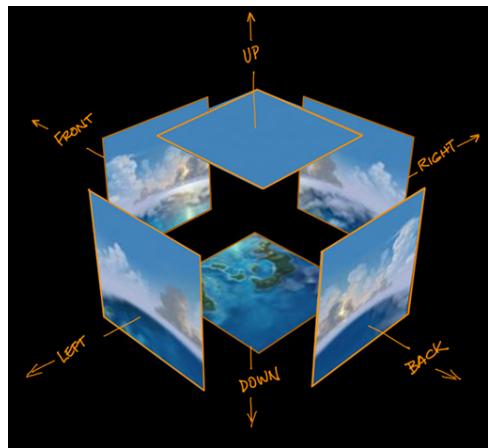


4.2 Bonus

4.2.1 SkyBox

实验原理:

天空盒实际是立方体贴图，通过设置一个很大的立方体的六个面的纹理贴图，将摄像机放置在立方体内，达到增强场景的表现力，减少场景搭建花费的效果。



具体实现:

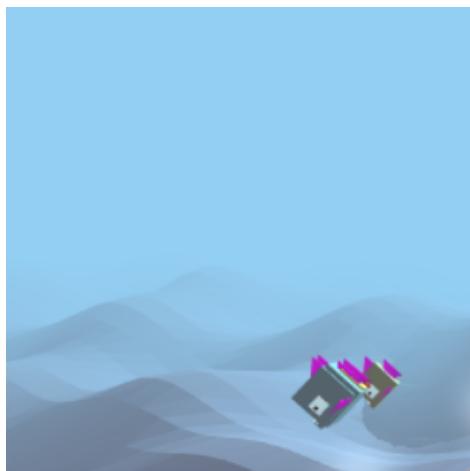
THREE.js可以直接设置场景的background为一个立方体贴图，因此只需按right, left, up, down, back, front的顺序导入立方体各面的贴图纹理即可。

由于找不到海底场景的天空盒资源，因此找了一些图片稍加处理做成了海底场景的天空盒。

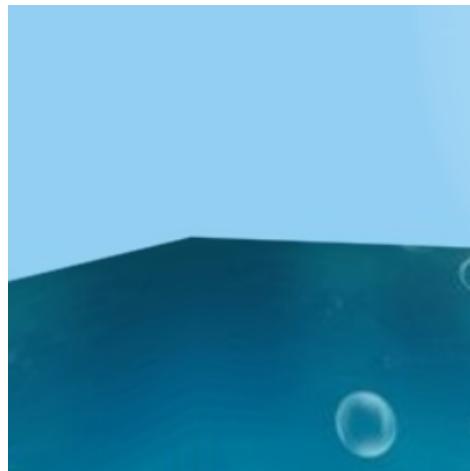
海上的天空盒由天空和海底场景组成，简单的将天空的颜色与海底贴图进行拼图后得到。

实验截图:

海上场景：



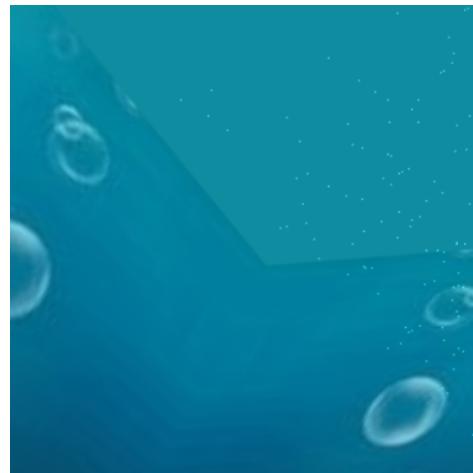
海上天空盒交界处：



海底场景：



海底天空盒的三个面交界处：



4.2.2 Display Text

实验原理:

通过THREE.js的Text Geometry文字几何体来生成立体文字，其实际是将每个字看成一个平面图形，将这个图形扩充至三维图形，从而生成立体文字。需要提供文字的字体，文字内容以及一些扩充至三维图形时的参数。

实验截图:



4.2.3 Stencil Test

实验原理:

模板测试利用了Stencil Buffer，其发生在深度测试前，alpha测试之后。模板缓冲一般为8位，开启模板测试后，描绘一个物体实际就是写入模板缓冲的过程。我们可以通过设置模板缓冲的值来保留或丢弃一些片元信息。利用模板测试进行一些遮蔽操作时，需要正确的关闭和开启深度测试，否则可能因为深度测试而覆盖了我们想要的效果。

具体实现:

- 1、首先设置全局的深度和模板测试相关设置，打开深度和模板测试，设置模板缓冲区可写，否则画面将无法渲染，然后设置深度和模板测试都通过时缓存值为1；
- 2、清除深度和模板缓冲区，然后设置模板缓冲区可写，设置通过模板测试的条件为ALWAYS，即总是通过模板测试；
- 3、利用正交摄影机和可视区域所在场景渲染可视区域，可视区域设为一个圆形，渲染过程即为写入模板缓冲区过程。用正交摄影机是为了保持可视区域形状的不变形，完成此步后圆形区域的目标缓冲区值为1，其他为0；

4、设置通过模板测试的条件为缓冲区值为1才能通过，即可视区域（圆形区域）内才能通过模板测试，禁止写入模板缓冲（防止修改缓冲区值），关闭深度测试；

5、渲染海底场景，海底场景使用透视投影摄影机，使渲染效果更真实，此时只有通过模板测试的区域才能被渲染出来。

实验截图：



4.2.4 Complex Lighting -- Normal Mapping

实验原理：

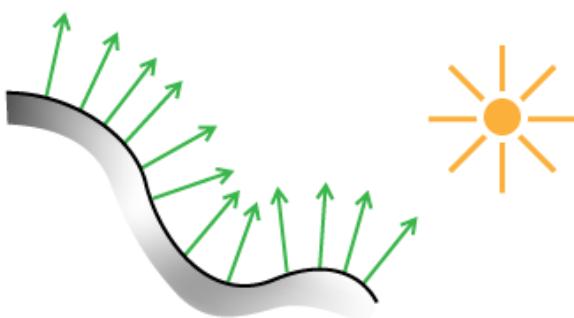
法线贴图可以为已简化的网格重新添加细节。在三维计算机图形学中，法线贴图 (Normal mapping) 是一种模拟凹凸处光照效果的技术，是凸凹贴图的一种实现。法线贴图可以在不添加多边形的前提下，为模型添加细节。例如，皮肤的皱纹，砖墙表面的凹凸，衣服的褶皱，树皮的纹路...

简单地说，法线贴图的出现，是为了低面数的模型模拟出高面数的模型的”光照信息”，其中光照信息最重要的部分是光入射方向与入射点的法线夹角。法线贴图本质上就是记录了这个夹角的相关信息，光照的计算与某个面上的法线方向息息相关。

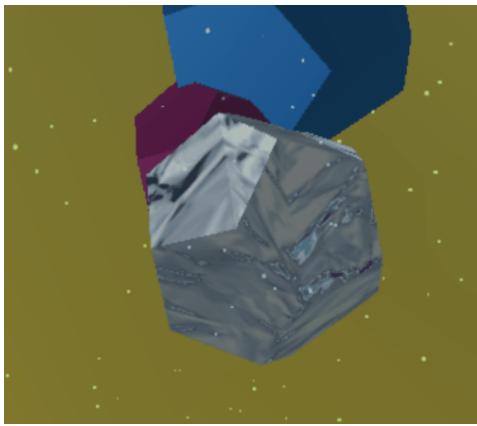
在实际计算计算表面光照的漫反射值时，只需计算以下两个单位向量的点积：

- 从浓淡点到光源的单位向量
- 该表面的法线单位向量

得到的结果就是光线在表面上的强度，这使得模型表面更加细致，尤其是与先进的光照技术一起使用的时候更是如此。



实验截图：



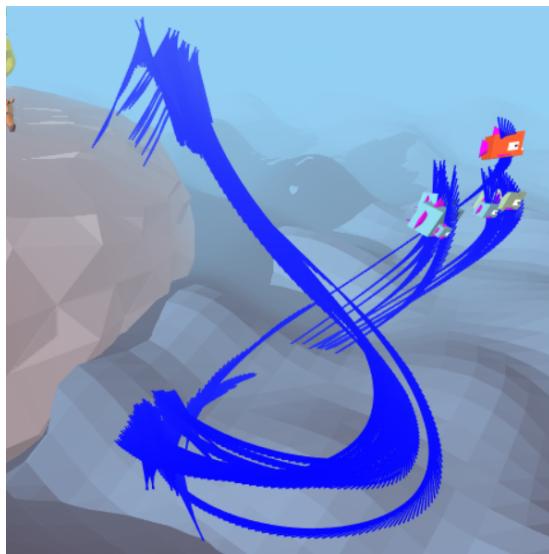
4.2.5 Gravity System and Collision Detection

实验原理:

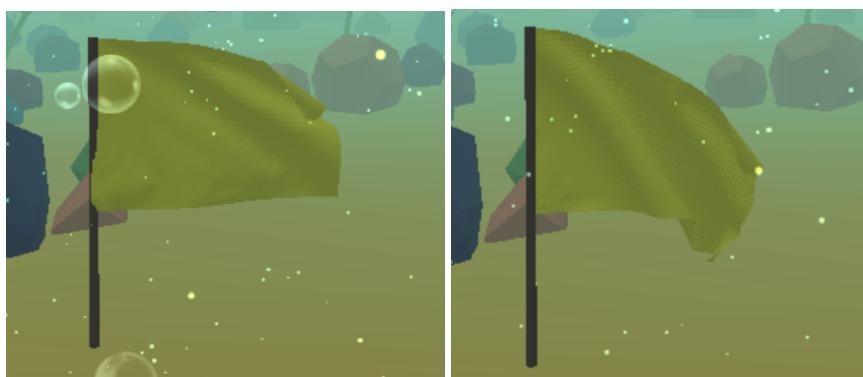
- Collision Detection。每个时刻，对物体移动的下一个位置进行判断，通过判断下一个位置的点和岛屿所有节点的距离来判断是否发生碰撞
- Gravity System。在织物模拟中，旗帜受风与重力两个外力，由于重力的影响，每个时刻旗子会有垂直方向的位移，但由于将旗子左边两个顶点固定在了旗杆上所以旗子不会持续下落

实验截图:

- Collision Detection: 为了方便观察，使用蓝线记录鱼的运动轨迹，这里实现的是鱼和岛的碰撞检测，使得鱼与岛将要碰撞时改变方向。



- Gravity System



4.2.6 Skeletal Animation

实验原理:

Skeletal animation is a technique in computer animation in which a character (or other articulated object) is represented in two parts: a surface representation used to draw the character (called skin or mesh) and a hierarchical set of interconnected bones (called the skeleton or rig) used to animate (pose and keyframe) the mesh.

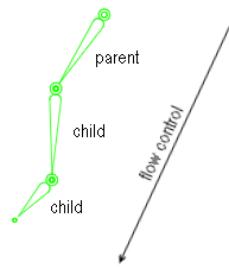
From wikipedia

根据维基百科的描述以及我自己在实验过程中的体会，我对骨骼动画的实现做了以下的总结：

1. Rigging, 即骨架的构建，为网格建立骨关节骨架层级的过程。通常使用层次结构将骨骼连接起来，完成骨架。

Hierarchy (层次结构):

影响骨骼也会影响其所有子骨骼。因此，在设置骨架时，第一个关节称为父骨骼。每个后续骨骼将直接或通过另一个骨骼间接连接到父骨骼。当父骨骼移动时，它也会移动它的所有子项，但是当子骨骼移动时它不会移动它的父项（我们的手指可以在不移动手的情况下移动，但是当手移动时它会移动它的所有手指）。

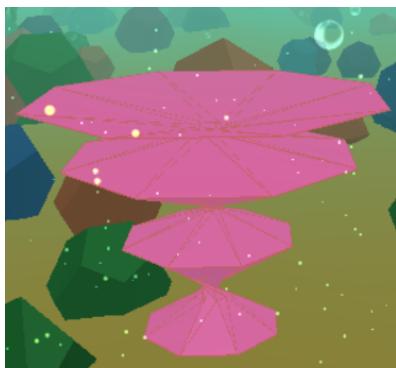


2. Skinning, 即蒙皮，将骨关节绑定到角色Mesh的过程，其中Mesh表示对象的皮肤，并且骨骼用于以模仿现实世界中的实际移动的方式移动网格。将顶点指定给骨骼时，会定义一个权重，用于确定骨骼移动时骨骼对顶点的影响程度，通常的做法是使所有权重1（每个顶点）的总和。
3. Keyframes, 即关键帧。一系列关键帧定义观察者将在最终动画中看到的一系列运动。要获得角色的某个姿势，您需要更改骨骼的位置。这可以通过使用几何变换（例如平移，旋转和缩放）来完成。所有骨架动画都有关键帧。谈到游戏动画时，他们通常会有动画周期。一个动画循环，包含了最后和第一关键帧之间的平滑过渡所有重要的关键帧。

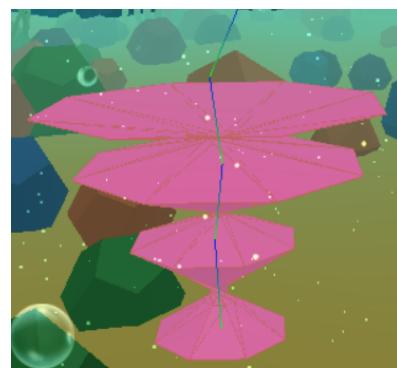
具体实现:

1. 创建了一个圆柱体的几何体，并且可以通过设置可以设置长度和节数，并设置每一个顶点受到骨骼下标的影响及权重：
 - a. 我们利用了数学方法(正弦函数+对数函数)完成圆柱体的半径变化，实现了类似海洋植物的效果。
 - b. 同时利用正弦函数周期性地控制圆柱体各个骨骼位置、旋转角度的变化，实现模型的周期性动画。
2. 实例化一组骨骼，并且设置好相关的父子级别关系。
3. 使用geometry创建THREE.SkinnedMesh模型
4. 通过three.js自带的类 THREE.Skeleton那一组骨骼创建模型骨架，将骨骼添加到上一步创建的模型里面，使得模型和骨架互相绑定。

实验截图:



原始效果



添加骨骼辅助线

4.2.7 Particle System

实验原理:

在用粒子系统模拟气泡时，一共实现了两类气泡：

- 利用THREE中point Material实现的普通气泡，彼此之间没有作用力，只做加速上升和随机左右的浮动，由四种不同大小，不同贴图的气泡进行模拟，显得更加真实
- 利用THREE中point Material实现的特殊气泡，即下图中红色框内的一簇气泡，该簇气泡是想模拟海底某处的孔洞偶尔会涌出一股气流，产生一簇气泡，这些气泡有相似的初始位置，在向上运动的过程中彼此之间有互相牵制的微弱的力，即群集效应，这样就不会因随机运动导致彼此集体太分散，且气泡在上升过程会不断变大，变得更透明，到海面会消失不见，由此来更加真实地模拟物理世界

实验截图:



4.2.8 Explosion Effect

实验原理:

一个上升的色块到达一定的高度后，发生爆炸，向预设的方向发射出更多的色块来模拟烟花爆竹的效果。

实验截图:



5 遇到的问题和解决方案

5.1 模板测试功能的bug

模板测试功能由于使用透视投影摄影机渲染，当摄影机位置或旋转角度变化时会导致可视区域发生形变和位移；

解决方案：先使用正交投影摄影机来渲染可视区域，写入模板缓冲区后再使用透视投影摄影机来渲染海底场景。

5.2 碰撞检测性能问题

使用 RayCaster 判断交点个数的方法来检测碰撞对性能要求很高，组员电脑普遍帧数在 10 帧左右

解决方案：通过使用判断下一帧位移的方式来检测碰撞。这样能够达到类似的效果而可以在普通设备上正常运行（保持20帧以上帧率）

6 小组成员分工

15331200 – 林景仰: Skeletal Animation、Complex Lighting -- 法线贴图、代码整合

15331212 – 刘博: 基础框架搭建、海面场景搭建、Complex Lighting -- 法线贴图

15331222 – 刘钟涛: Skeletal Model Use、阴影修正

15331251 – 彭彩莹: SkyBox、Display Text、Stencil Test、海底场景搭建

15331238 – 马晓鹤: Gravity System、Texture Mapping、Cloth Simulation、Particle System

15331253 – 彭高: Collision Detection、性能改进