计算机图形学大作业报告

张荣城 515030910413

1. 简介

此次作业主要实现了一个利用SPH算法计算的流体，交互方式主要采用鼠标键盘，另利用安卓手机的加速度传感器使用socket通信来控制。

1. 操作说明

F1切换粒子显示模式（用point绘制，带纹理绘制，模型绘制），F2、F3控制镜头远近，F4切换控制流体容器转动模式（鼠标拖拽，手机加速度传感器控制）F11、F12控制光照强度，wasd控制场景移动，上下左右键控制流体容器的移动。

1. 思路

在刚看到题目要求的粒子系统时，由于没有大的限制，一时无从下手。后来在考虑一段时间后突然想到可以实现一个水壶倒水，水落入水盆的场景，这样可以使用手机的陀螺仪来控制水壶的倾角，会很有意思，尤其是如果把水盆的盆面正对屏幕，则可以实现adobe宣传片中手机模仿洒水，电脑屏幕上出现反应的效果。

因此在考虑过后，预计实现需要考虑三块内容：

1. 水的粒子系统实现方式。
2. 安卓手机的交互。
3. 水的渲染。

在实现的过程中发现，由于sph算法非常精细复杂，对于算力的要求很高，在我的电脑上，没有加载场景模型的情况下，512个水粒子可以无卡顿，而加载后已经出现卡顿，如果按照开始的设想去设计，最终效果很差，因此最后没有实现倒水，而是仅仅将流体放在一个立方体容器中，进行控制。

1. 难点及实现
2. 流体的算法

在确定一开始的设想后，我首先实现了一个简单的粒子系统，接着查询了相关论文和博客，包括动态的水面模拟[1]，OpenGL中基于粒子系统的喷泉模拟实现[2]，但都不能实现设想的效果。

接着去问了船建学院的同学，想知道如何计算倒水时的轨迹，初步设想要考虑到空气阻力，张力等，但询问无果，都是说太复杂无法算，后来找到一个模拟软件，想看看模拟需要的因素，再倒推，但依然无果。最后询问助教得知了SPH算法，于是上网查询，进行实现。主要参考这篇博客：APH算法简介[3]。计算公式可参见此博客。

sph\_ParticlePool用于粒子的存储和管理

sph\_neighbour\_table用于管理粒子相邻的粒子，用于计算相邻粒子对该粒子产生的影响。

Sph\_brid\_container用于计算容器对流体的影响。

Sph\_fluid\_system负责所有粒子的光滑核函数的计算，粒子属性的更新。

Sph\_particle不实现绘制方法，具体如何绘制放到源.cpp去做，这样glut与sph系统就解耦了。

2、模型的加载

一开始仅仅在加载粒子模型时需要使用，因此使用了在作业一中写的objloader，之后需要加载大的模型，发现自己写的不够用了。

首先想采用assimp库加载场景模型，但网上大多数教程都是glew，而我用的是glut，因此最后采用OpenMesh。此处有个坑，OpenMesh里应该有ws2def.h，与win2sock.h有冲突，而后者是使用安卓进行连接时用到的库，在尝试使用命名空间分隔无果后，发现include顺序调换之后不再报错。

3、渲染

我设想的流体是透明的，发现使用glBlendFunc后可以实现透明，立方体容器同样使用。

纹理的加载使用了一个texture.h完成。

1. 交互

首先实现鼠标键盘交互，像控制远近，控制上下左右移动都是为了调整模型到合适的位置方便，在实现过程中加入的。场景模型也是可以变换角度的，但是最终实现版本中没有让其随流体的容器一起变换角度。视角的变化是采用gltranslate去变换物体位置实现的。此处遇到一个坑，由于glrotate是渲染的时候做的旋转，也就是说在流体看来，重力方向还是原本那样，在渲染的时候就跟着旋转了，而不是在世界坐标系中的(0,-9.8,0)，因此需要在每次rotate之前把新的重力向量传入fluid\_system。新的重力向量是采用数学方法推出的。以x轴转yRotate度，以y轴转xRotate度，最后的位置是(-9.8 \* sin(xRotate / 180 \* PI) \* sin(yRotate / 180 \* PI), -9.8 \* cos(xRotate/180 \* PI), 9.8 \* sin(xRotate / 180 \* PI) \* cos(yRotate / 180 \* PI))。此处遇到一个坑，math.h中采用弧度制，glrotate采用角度制，导致一开始怎么调也不对。

接着是实现安卓手机与之的交互。开始时设想采用陀螺仪获取角加速度，让流体的容器跟着旋转，可是实现完成后发现这个实现无法让流体容器与手机的角度同步，很难控制。后来发现加速度传感器的xyz数值就是重力在三个轴上的分量，也就是是说，只要解上面那个公式的方程即可得到xRotate、yRotate的数值，这样就可让容器的转动与手机同步，即

X = -9.8 \* sin(xRotate / 180 \* PI) \* sin(yRotate / 180 \* PI)

Y = -9.8 \* cos(xRotate/180 \* PI)

Z = 9.8 \* sin(xRotate / 180 \* PI) \* cos(yRotate / 180 \* PI).

最终实现成功了，不过此处的一个坑是asin、acos在遇到不合理的参数比如大于1时会报nan，出错，因此需要检查一下再输入。

但最终实现效果也没有预期的好，因为涉及到了网络，无法做到实时流畅的响应，还是会有延迟，考虑更流畅的方式可能是利用计算机摄像头或者其它设备来实现操控。

1. 总结

这次大作业虽然没有实现预期那么好的效果，但是在利用安卓手机进行交互这一点还是增添了一点趣味性，并且在实现的过程中让我意识到了adobe宣传片的效果的大致实现思路，同时也意识到，利用网络进行实时的交互这种方式不可靠，不够快也不够稳定安全，我觉得这也是需要特定交互硬件的原因之一；在利用SPH算法实现流体的过程中，我发现这个算法过于精细，一定有近似算法，在较低计算量的情况下实现不差于它的效果。

总的来说，此次大作业的完成过程中，学到了很多，同时也收获了一定的成就感，后续的改善空间也很大，将来如果要实现adobe宣传片中那种酷炫的效果，此次作业也能提供很多参考。

代码量：pc端约1400 安卓端约200

参考博客/论文：

1. 动态的水面模拟，http://blog.csdn.net/zju\_fish1996/article/details/52317363
2. OpenGL中基于粒子系统的喷泉模拟实现，https://wenku.baidu.com/view/c79b56d476eeaeaad1f33068.html
3. SPH算法简介，https://thecodeway.com/blog/?p=83