计算机图形学

作业3

README

1. 时间积分器

实现欧拉法（Euler），梯形法更新质子系统的每个质子当前的状态。两种方法实现的步骤相同，不过使用的公式不同。

实现步骤为：

获得质子系统中的各质子的状态，以数组形式返回。

以质子状态为参数传入质子系统中的evalF（）求导，获得速度f，乘以步长得到▲x加上初状态获得当前状态。

欧拉法

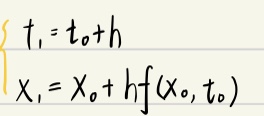
实现ForwardEuler类的takeStep方法。

首先使用质子系统类内置函数getState获得质子状态，以数组形式保存。

将状态数据传入质子系统内置函数evalF中求导获得速度。

用for循环对每个状态质子进行积分，更新状态。

欧拉法使用的公式为：



代码表示为：

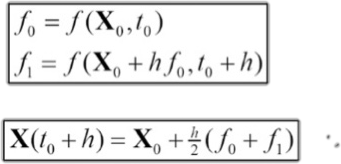
state[i] = state[i] + stepSize \* f[i]

梯形法

与欧拉法过程相似，但是梯形法需要进行两次积分。

第一次用欧拉法进行积分之后，第二层用积分结果作为参数放入质子系统中求导获得新状态下的速度，之后与初速度求均值，再加上初状态。

公式表示为：



代码表示为：

第一次积分

state[i] = state[i] + stepSize \* f[i]

得到当前状态速度

vector<Vector3f> f1 = particleSystem->evalF(state1);

第二次积分求均值

state[i] = state[i] + stepSize \* (f0[i]+f1[i])/2;

1. 简单例子

这里继承了抽象类ParticleSystem，实现它的evalF（求导）、draw（画图）方法。

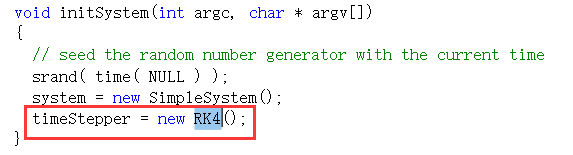
evalF方法

此方法主要实现的就是对传入的状态值进行求导，并且返回相应求导后的数组。在网课中已经介绍了简单一维质子系统求导后的f值为：。所以传入的状态值进行遍历，相应的状态值对应的导数为（-state.y,state.x,0），然后返回结果。

Draw方法

在简单例子中就是遍历数组每个state的位置信息，进行画图即可。

Main.cpp修改使用的微分方程求解方法，修改出下图给出，new后修改RK4、ForwardEuler或Trapzoidal。



使用RK4、Trapzoidal都不会让粒子向外飞出；而ForwardEuler会让粒子向外飞出。

要看到粒子还要对类simpleSystem进行初始化，这里初始化一个半径为1的粒子，代码Vector3f（1，0，0），添加到系统的粒子数组m\_vVecState中，直接push\_back。

1. 物理仿真

粒子状态初始化

需要对类PendulumSystem进行初始化，和简单系统不同，存储状态的数据不仅要存储粒子位置，还要存储粒子速度，采用的策略是，其中位置存储在偶数下标中，而速度存储在奇数下标中。

evalF的实现

for a given state, evaluate f(X,t)，可知，速度可知保存在状态数组下奇数下标下，只需要求出加速度即可。加速度由合力除以质量得到。

力的表示

因为系统中的状态量中只存储了位置和速度，要统计收到的重力、弹力和摩擦力，所以另外新增了一个类，保存了粒子的质量、弹性系数、摩檫力的系数、连接绳子静态长度和下标、还要判断粒子是否静止。

设置完粒子属性类之后，使用保存的属性进行计算各分量力的数值。

需要注意的是，当物体静止时，合力为0，加速度也为0，就不需要计算合力了。

重力

直接将质量\*10即可得到重力，再使用Vector3f(0,-g,0)转化成向量形式即可。

弹力

利用公式。计算弹力。

摩擦力

利用公式计算。

合力

将上述计算出的各分力进行相加后除以质量即可，因为三个力分量都是向量，就不需要再进一步转换了。

单摆

为了区分是否为单摆或多粒子链，在类中添加了成员变量isSingle，并初始化为true，即为单摆。

并且在Main函数中新增了函数drawSingleString，即画出弹簧。实现原理很简单，固定一个端点，另一个端点为粒子当前的坐标即可，此处固定的端点为原点。

多粒子链

将变量isSingle设为false，并且新增函数drawString，画出多粒子链的函数。与单摆的函数基本原理相同，只不过将固定的原点换成了与当前转入参数下标的粒子相连的粒子，也是画出两个粒子的坐标。

需要注意的是固定最顶端的粒子，该粒子没有初速度，并且设置该粒子的isStatic参数为true，即该粒子保持静止，所受的合力为零，跳过合力的计算。

Draw

实现的是画出粒子位置。具体实现为，遍历状态数组，因为位置数据保存在偶数下标，直接输出偶数下标数据，取出粒子坐标，进行画图即可。

1. 粒子系统布料

evalF的实现

将类ClothSystem继承于类PendulumSystem，并且类ClothSystem的eval函数在类PendulumSystem实现速度存储和求合力加速度的基础上实现加风。具体实现过程为，增加一个方向上的力并且除以每个粒子的质量求合加速度即可。

Draw的实现

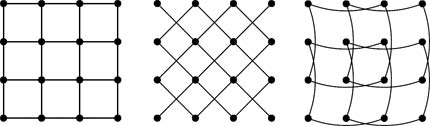
类ClothSystem的draw函数只要实现画布料和画结构弹簧。

首先画出一个半径为1的圆球。

布料渲染过程通过遍历整个粒子数组，取出当前点和右侧，下方右下方的四个点，四个点两两组合形成四个向量，两两配对，叉乘之后，生成两个法向量，画两个三角形。

画结构弹簧通过遍历粒子数组，每个粒子数组都保存着对应下标相邻粒子的弹簧静态长度，只需要取出结构弹簧长度的相邻粒子的下标，得到点数据，与球心距离比较小于设定值就将两个点连接起来即可。

通过指导书中给出的三种弹簧的大致分布图可知具体弹簧各粒子连接情况如下：



根据上图的每个点相邻点的下标分布规律进行判断和保存下标。

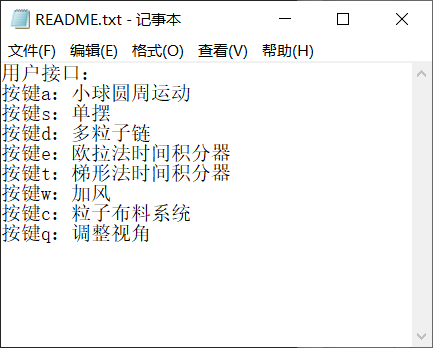
接下来就是添加用户接口。

5.附加分

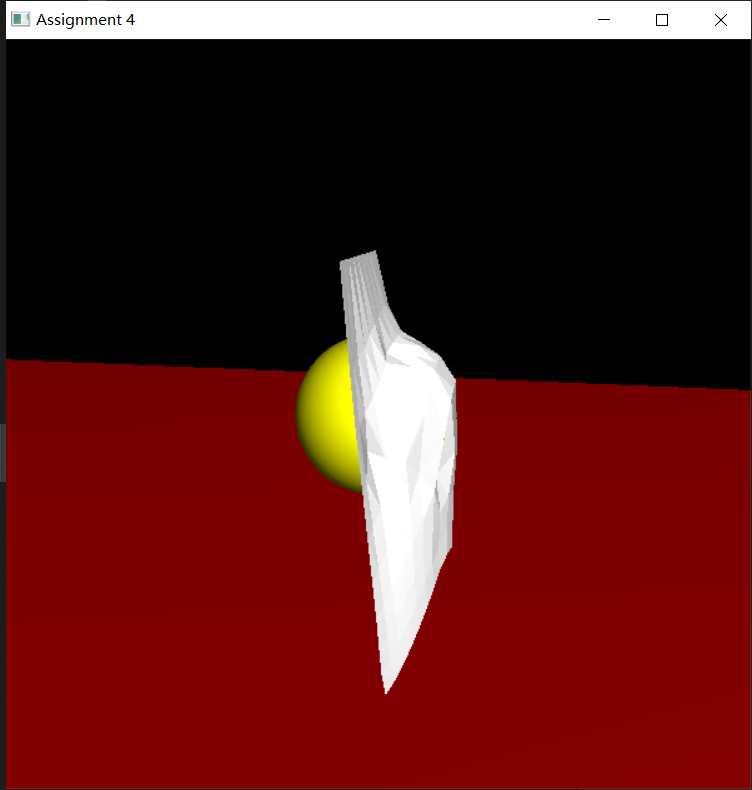
实现了加风的操作，实质上是在x的正方向加上一个值为6的向量。

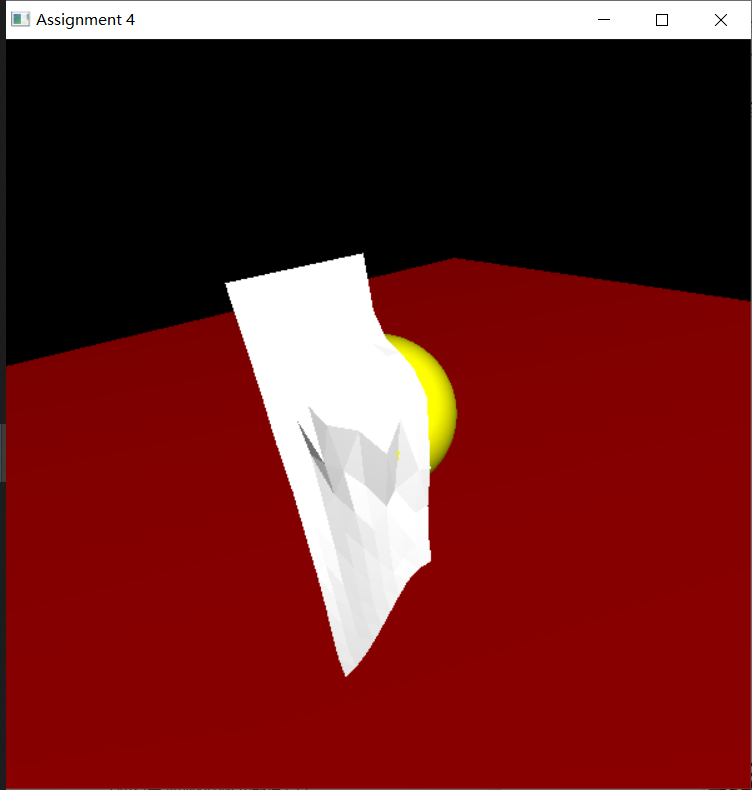
6.实习成果展示

用户接口

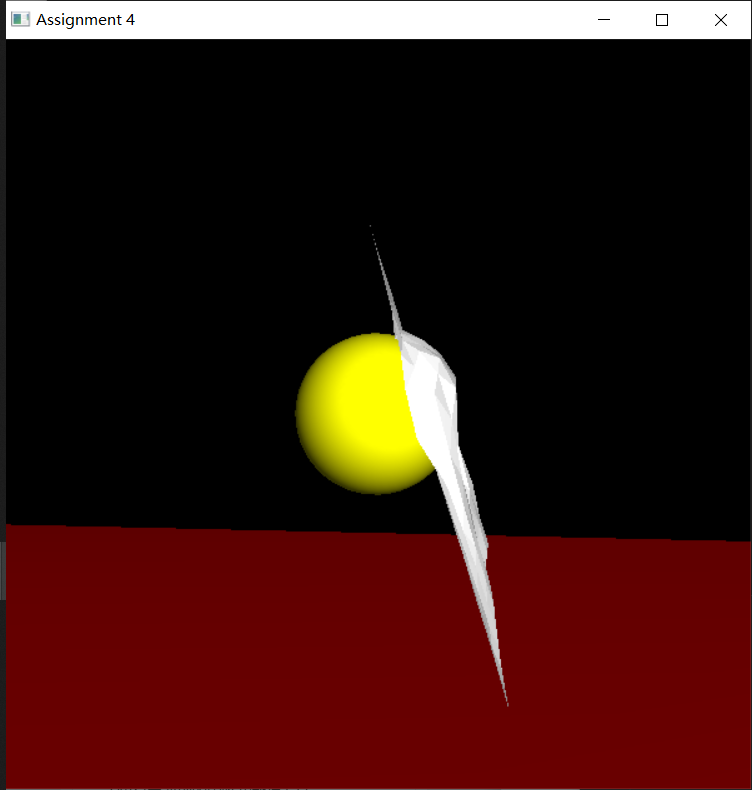


粒子系统布料





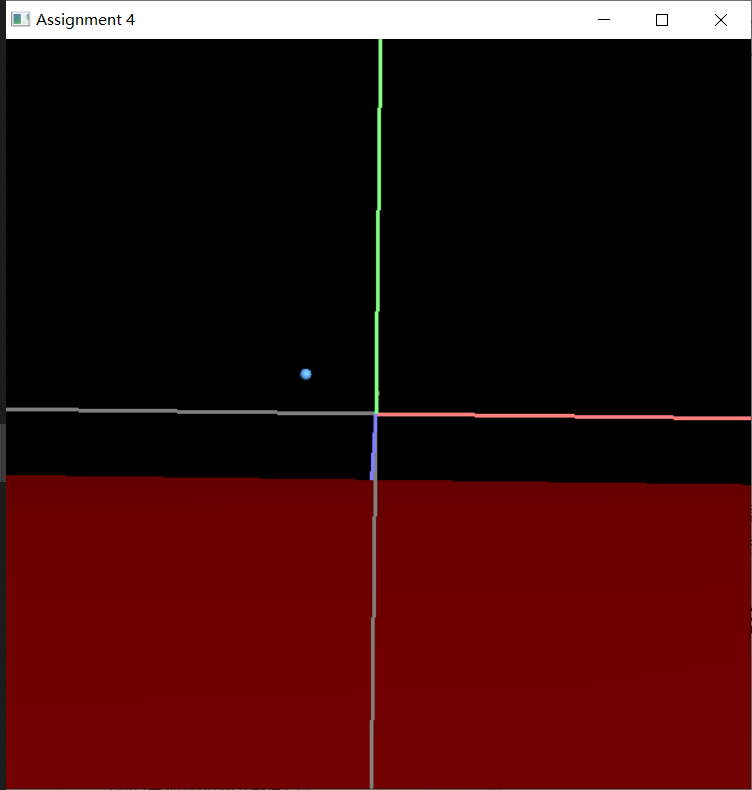
加风后



小球圆周运动

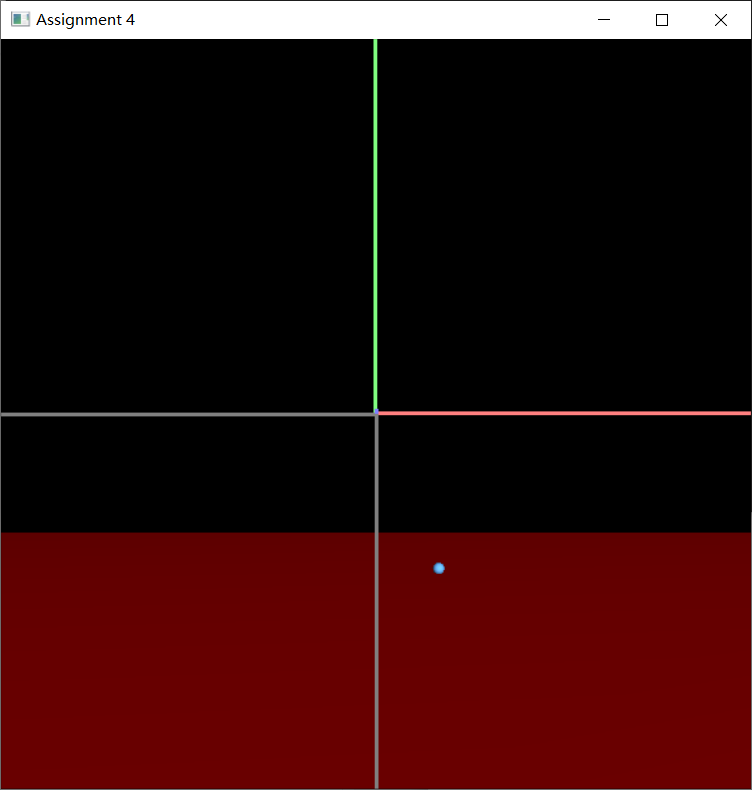
此时是梯形法时间积分器

可以看到半径不会变大



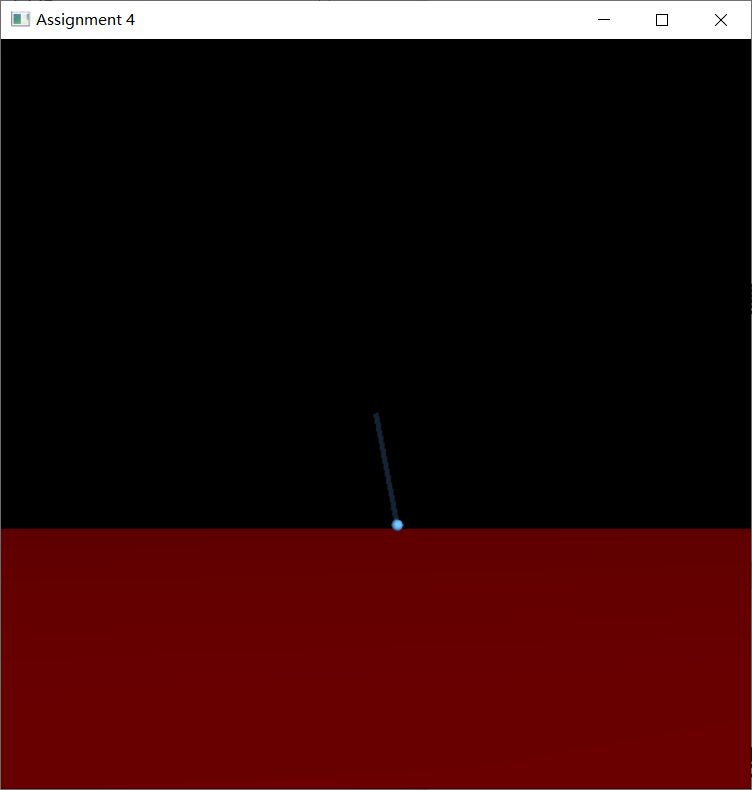
换成欧拉法时间积分器

可以发现小球逐渐向外发散

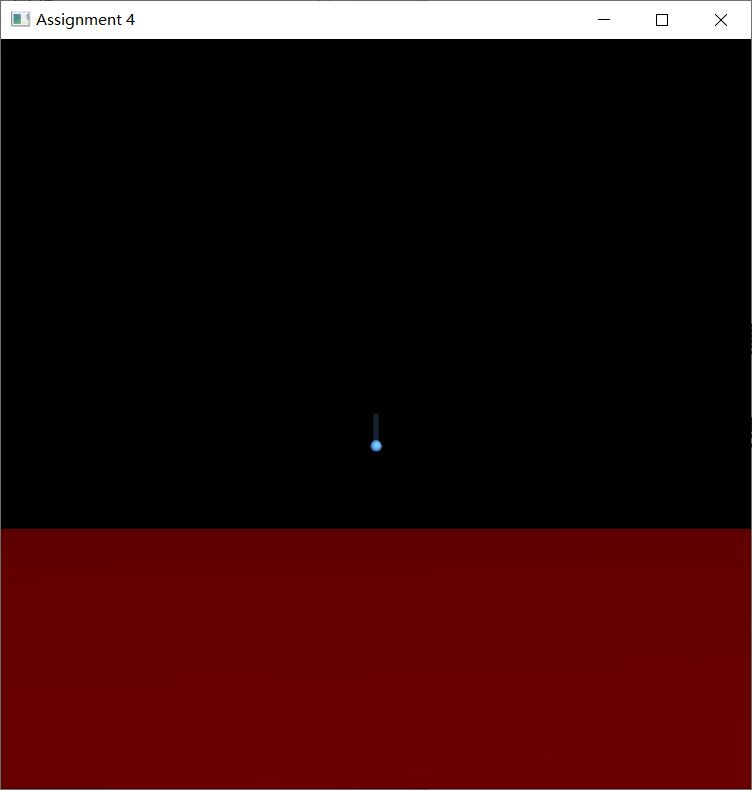


单摆

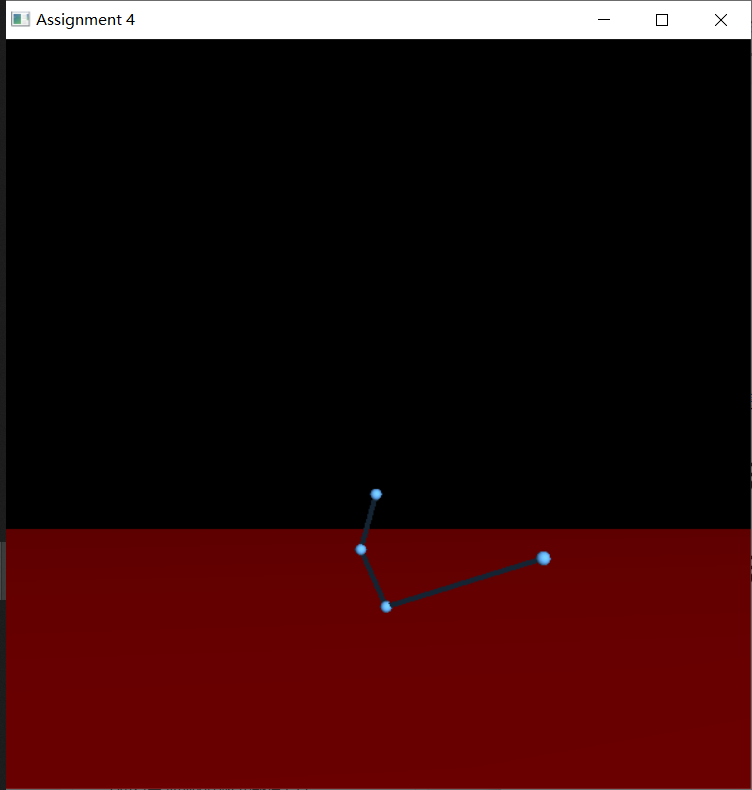
被拉伸时



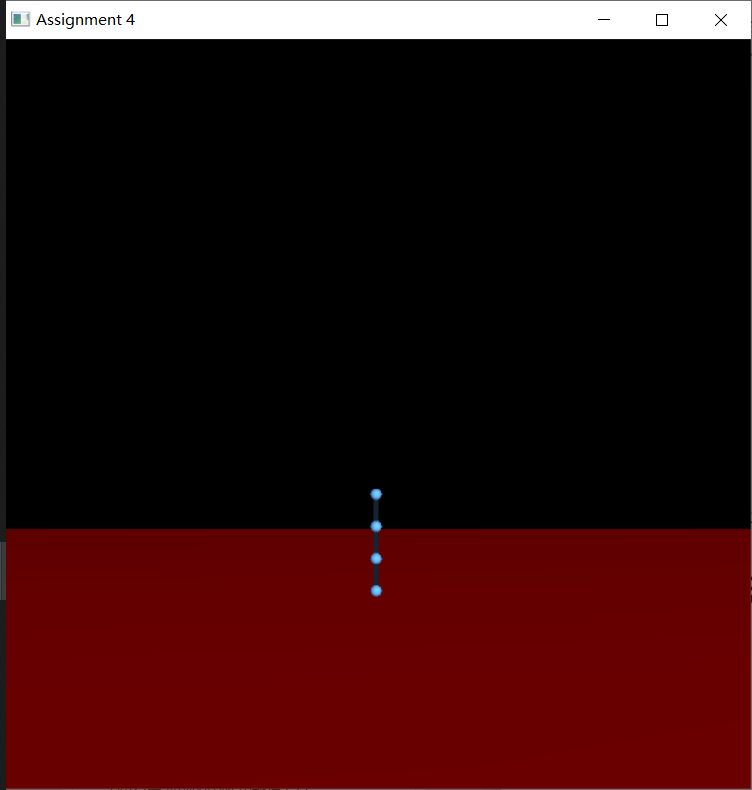
自然伸长时



多粒子链



自然伸长时



7.实习中遇到的困难和解决方法

1.在作业开始前对所需要的相关知识陌生

通过网课，ppt进行学习，记录笔记。

2.无论是单摆还是多粒子链，还是布料，运行程序后，整个模型都下落消失在屏幕中。

通过设定单摆还是多粒子链设定顶层粒子静止。布料粒子的第一行所有粒子都静止。

3.弹性系数的设定

通过多次的设定，尝试。

4.多种显示造成系统冲突报错

因为本作业需要实现用户接口显示单摆，多粒子链，布料。在显示一种模型之后，参数随之改变，而导致其他模型的显示出错或报错。之后使用多个bool变量进行控制。

8.实习总结

本次的实现非常有趣，将物理知识可视化，我从中收获了很多。学习了如何通过法向量进行布料的渲染，复习了之前学习的物理知识（弹力，重力，摩擦力，合力和加速度的计算），让我深刻感受到了物理、数学知识在计算机图形学中的重要性。

但是在本次作业中没有实现布料的左右移动，只是在刚开始布料生成时停留在小球上，如果未来有能力、有时间，一定进行钻研，努力解决这个问题。