中国科学技术大学

“科学与社会”研讨课研究报告

报告题目：Simulation Physics Experiment Space

小组组长：罗浩铭

小组成员：罗浩铭　　李璐豪

胡璟凡　　何昕

导师姓名：黄刘生

2022 年 5 月 28 日

一、研究小组成员及其承担的主要工作

学号	姓名	所在学院	在研究和报告撰写中承担的主要工作
PB21030838	罗浩铭	计算机科学与技术	程序框架、模块组装、图形学算法、地震模拟、n 体问题、答辩 PPT、报告撰写
PB21111663	李璐豪	计算机科学与技术	刚体类、报告撰写
PB21111660	胡璟凡	计算机科学与技术	向量与数值运算函数、立项 PPT
PB21111661	何昕	计算机科学与技术	可视化

二、进度安排

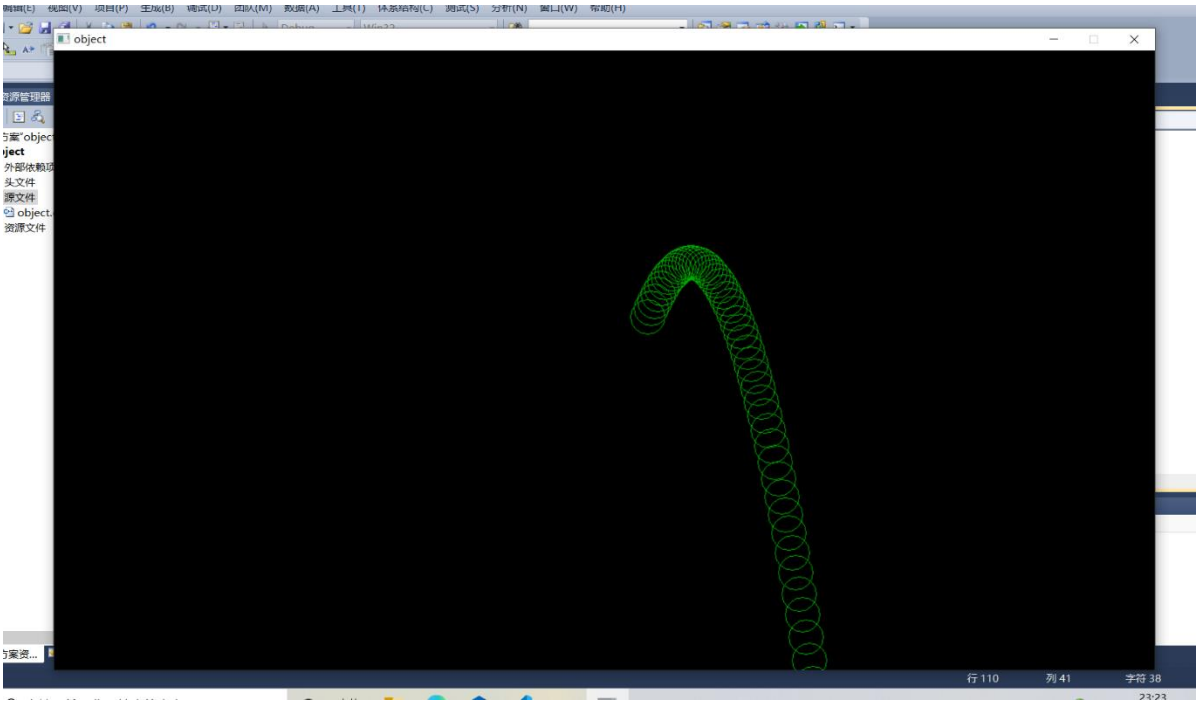
研究工作的具体时间节点和工作进度如下：

2021.11.27

小组第一次会议，确立了虚拟物理实验空间的项目，并大致讨论了实施路线和一些实施细节。

2022.01.22

利用 EasyX，第一次实现可视化模块



2022.01.26
完成程序实现路线的总体规划，明晰了整个程序的实现方案。

SPES实现路线设想

> 初步设想包含了程序中必须要实现的核心内容，以及可能的实现技术

一.程序基础

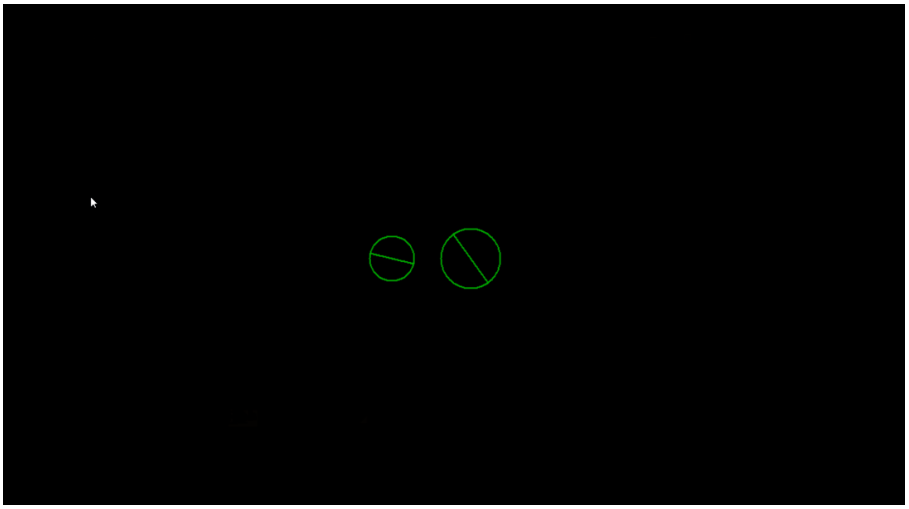
父类-刚体类

变量

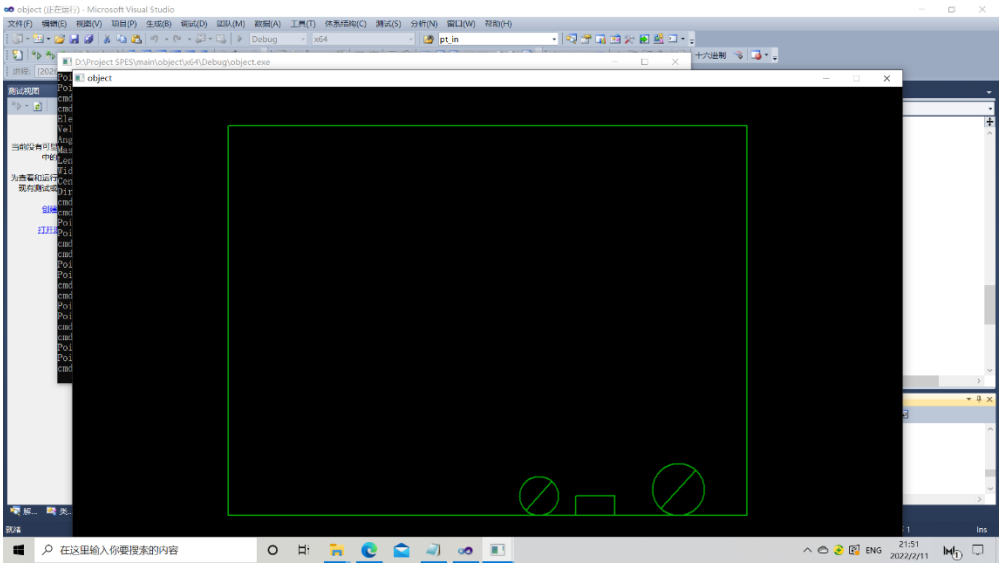
> 刚体会平动（就是高中物理里的正常移动），这直接由物体的质量受合力的大小方向决定 也会转动，直接由物体转动惯量（形状不同还不一样）和力矩决定 积分可以先用欧拉积分，但尽量用RK4。欧拉相当于泰勒展开一阶，RK4相当于四阶。

> 这些是private变量，外界要获取它的值或修改只能通过类里的函数

2022.02.09
底层函数与刚体类开发完成
2022.02.10
第一次成功地在最简单的情况下运行起整个程序，正确地输出了两个球的碰撞结果，标志着程序框架的完成。

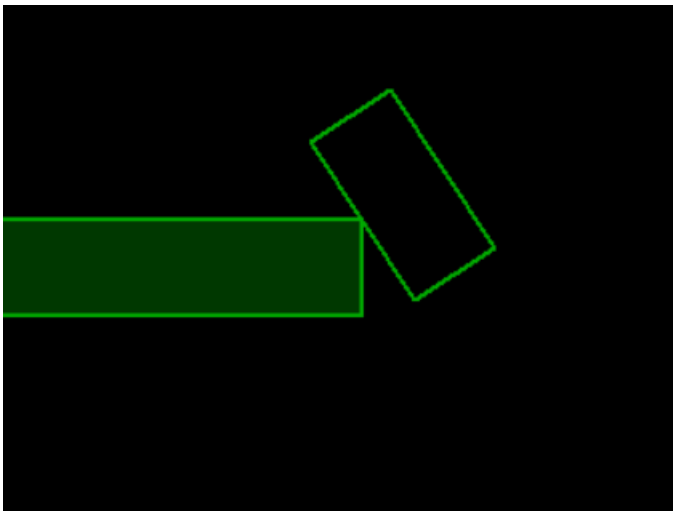


2022.02.11
第一版程序基本完成，可以实现各形状刚体与各种地形在程序里按物理规律正确运行。



2022.02.13

替换了表现不佳的地形类，将原本为一条线的地形换成了用多边形表示。

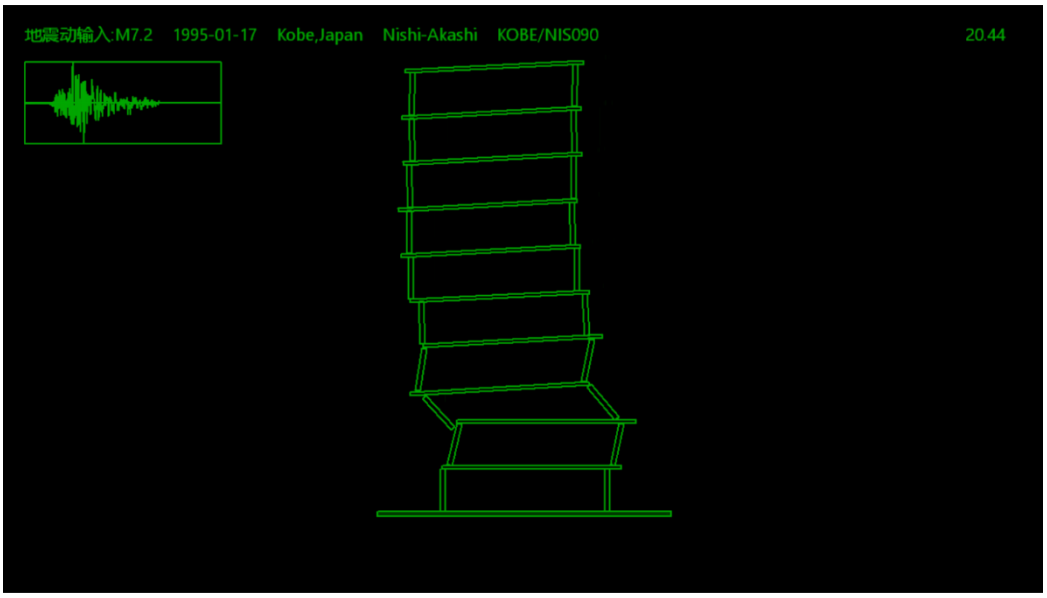


2022.02.14

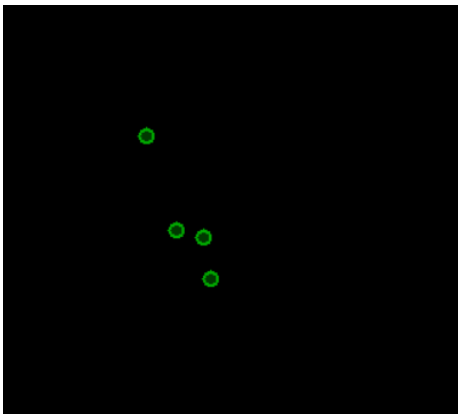
修复所有 BUG，完全替换了多边形碰撞算法，改进了表现不佳的算法，增加了剪枝算法，将程序效率增加约十倍

2022.02.16

地震模拟模块完成



2022.04.01
第一次成果展示
2022.06.05
N 体问题程序完成



2022.06.05
最终答辩

三、摘要、关键词

摘要：
我们的项目目标是创造一个较为真实的、高度泛化的、高可扩展性的物理引擎框架，作为一个仿真物理实验空间，以模拟诸多物理现象，并通过扩展程序将其应用在建筑抗震模拟、n 体问题等诸多实际领域。
我们通过使用 C++对基本的物理原理进行系统的程序化，通过数值积分的方

法，实现对平面刚体力学的仿真，并使用 EasyX 图形库实现仿真结果的可视化。
我们的程序具有高真实性，完全按照物理规律搭建，且使用高精度的 RK4 积分进行计算，非极端情况都可以收敛；具有高自由度，输入的模型可精准调节，可自由定制；具有高度泛化性，可以模拟各种刚体物理现象；作为一个物理引擎框架具有高可扩展性，可以在原程序的基础上扩展出各种特化应用，如地震模拟。

关键词：物理引擎框架　地震动响应模拟　高度泛化　高可扩展性　仿真

四、研究报告

一、 背景和目标

市面上能够自由地模拟物理现象的 APP 或 Web 的数量比较少，且其中贴近物理事实，贴近于现实且功能较为全面的几乎没有。我们的项目目标是创造一个较为真实的、高度泛化的、高可扩展性的物理引擎框架，作为仿真物理实验空间，以各种物理事实为基础，计算这个空间中各种物体之间的相互作用，以模拟诸多物理现象，并通过扩展程序将其应用在建筑抗震模拟、n 体问题等诸多实际领域。

二、 研究报告正文

实现功能

首先，我们实现的是一个高自由度的仿真平台，在消息循环中输入 cir,tri,rect,tercir,tertri,terrect 代表将要添入圆,三角形和矩形的刚体或地形（带前缀 ter）,然后再输入这些刚体或地形的参数，如形状大小参数，刚体还需输入质量、电荷、初始速度、初始角速度等物理参数。而用户可以通过输入数据来自由、精准地定制他们想要的物理模型。

在输完模型之后输入“finish”（若是地震模拟程序还需要输入地震波名称参数），并输入程序的积分时步参数（一般为 10，若积分收敛可以调小，不收敛则需调大），程序将从初始状态开始计算接下来物理空间的变化，并将结果实时地、动态连续地以图形化界面呈现。

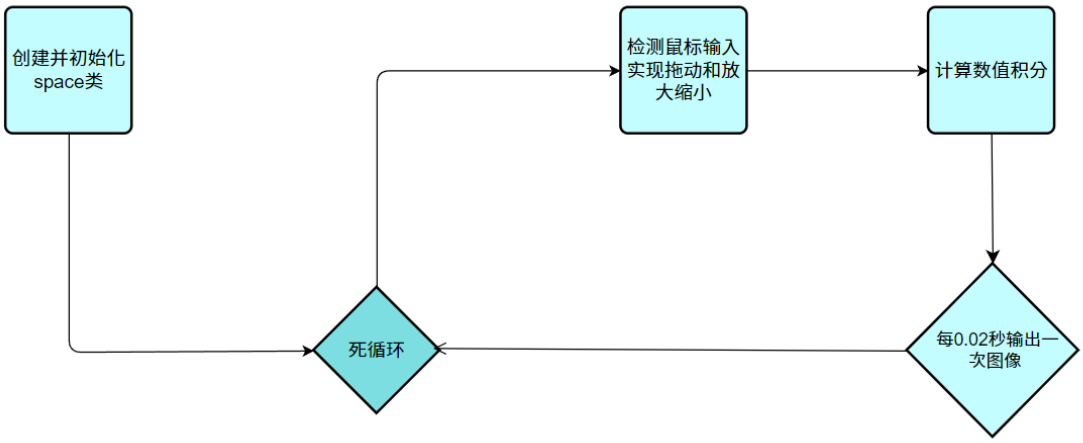
算法流程

我们使用 C++语言和针对 Visual C++的 EasyX 图形库完成整个项目的实现，程序很好地体现了自顶而下的编程思维，分为 main 函数、SPACE 类和刚体类三层，程序流程每个步骤在这三层都有体现。程序运行时只会创建一个空间类，里面装着所有刚体对象。下面是整个程序的结构。

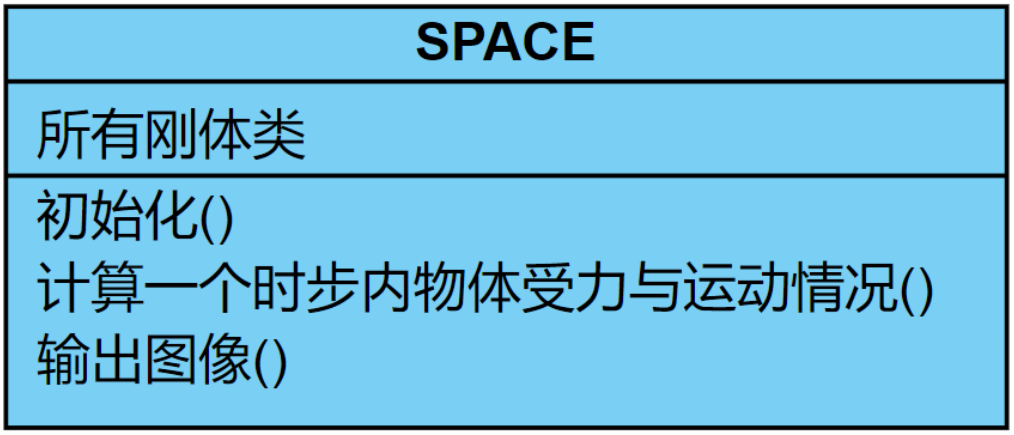
1.main 函数

Main 函数的主要流程如下图所示：

程序将先以消息循环的方式接收用户的输入，并按此输入向 SPACE 类中添加各种刚体和地形。当输入结束后进入死循环，开始计算物理空间之后状态，并实时地、动态连续地以可视化形式反馈，输出画面是 50 帧。每次循环检测鼠标操作，若是有鼠标拖动或是滚轮操作，程序画面将平滑地移动或放大缩小（记录偏移或缩放量，但暂时不体现在画面中，再以指数衰减的形式将偏移或缩放量更新到画面中）。



2. SPACE 空间类



SPACE 类里装着物理空间里所有的刚体和地形，其类函数同样是初始化、计算和输出图像这三个程序流程主要的操作。

计算模块的主要思想是数值积分。为厘清数值积分的概念，我们先从欧拉积分讲起。由于计算机无法表示连续的时间域，我们只好将时间分为一段一段很短的时间 dt，称为时步。欧拉积分即假设这个时步（比如 0.001 秒内）物体的速度加速度不变，得到物体的下一个位置和速度，并由受力情况得到下一个加速度，本质上是泰勒展开一阶近似。而实际用到的是在此基础上的 RK4 积分，它可以理解为泰勒展开四阶近似。这说明其总累计误差阶数为$o(t^4)$，相比于欧拉积分其结果误差小了几个数量级，并尽可能减小数值积分不收敛

的可能。由于 RK4 积分达到了精度和效率的平衡，它成为在工程计算中应用最广泛的数值积分算法。具体实现上，RK4 是进行四次欧拉积分再加权平均，其公式如下图：

$$\mathbf{K}_1 = \mathbf{F}\left(\mathbf{S}^{[i]}\right)$$
$$\mathbf{K}_2 = \mathbf{F}\left(\mathbf{S}^{[i]} + 0.5h\mathbf{K}_1\right)$$
$$\mathbf{K}_3 = \mathbf{F}\left(\mathbf{S}^{[i]} + 0.5h\mathbf{K}_2\right)$$
$$\mathbf{K}_4 = \mathbf{F}\left(\mathbf{S}^{[i]} + h\mathbf{K}_3\right)$$
$$\mathbf{S}^{[i+1]} = \mathbf{S}^{[i]} + \frac{h}{6}\left(\mathbf{K}_1 + 2\mathbf{K}_2 + 2\mathbf{K}_3 + \mathbf{K}_4\right)$$

要得到受力情况，我们先清零刚体类记录力向量的变量，然后要给物体加上重力等各种场力，接着再根据它们的位置判断是否发生碰撞，加上因碰撞而受到的力和力矩，一个时步结束后，每个刚体都将得到所有力和力矩的和，就是物体所受合力和合力矩。最后合力再除以质量，合力矩除以转动惯量就得到加速度和角加速度。

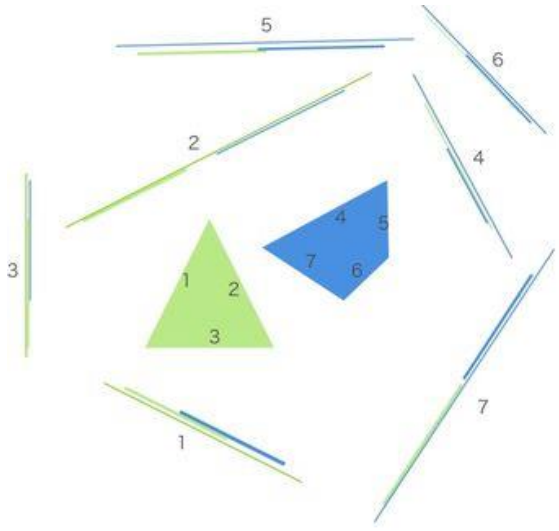
得到加速度和角加速度之后，我们就可以用数值积分来计算下一个时步各物体的速度和位置。

3、碰撞算法

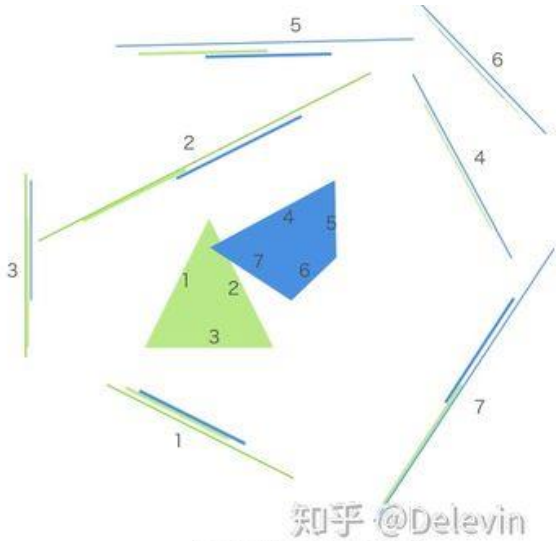
其中，碰撞算法主要是检测物体有无相交。如果两物体相交，则检测碰撞点和碰撞深度，由杨氏模量可得到碰撞力，再由此得到碰撞时的摩擦力，以及作用在碰撞点上的力矩大小，加进受力情况里。

程序使用的多边形检测算法为分离轴算法，其流程如下[2]：

由于我们程序中实现的都是凸多边形刚体，取两个凸边形每条边的法向量，再将两个图形投影到每条法向量上，可以在在每条法向量上得到两条线段。如果每条法向量上的线段都有重合部分（如图），那么就认定凸多边形相互碰撞了。反之，则没有碰撞。



在2和7投影轴上存在阴影间隙



在所有投影轴上均无存在间隙

由于碰撞算法是程序开销最大的部分，可以用简单的筛以筛去明显不可能碰撞的情况。程序中最有效的剪枝是使用碰撞圆算法，每个刚体都存在圆可以包住整个刚体，称为碰撞圆，实际应用时取尽量小的碰撞圆，若两刚体连碰撞圆都不相交，则肯定没有碰撞。这一算法再加上其余各处的剪枝，让程序运行速度提高了约 10 倍。

虽然碰撞算法原理很简单，但因为力加速度这些都有方向，且要判断各种切法向量方向，还要计算碰撞点，所以需要复杂的向量运算。

如下图所示，即使是最简单的圆-圆碰撞算法，也异常复杂，涉及 Schmidt 正交化等诸多线性代数技巧；而两个多边形的碰撞算法代码量更是达到了 210 行。

```
void circle_circle_detect_collision(class Circle &cir1,class Circle &cir2)//改变输入物体的受力
{

    if(normal(cir1.xy-cir2.xy)<cir1.r+cir2.r)
    {


        XY fn1;
        double dep;//深度
        dep=cir1.r+cir2.r-normal(cir1.xy-cir2.xy);
        XY vector_n=(1/(normal(cir1.xy-cir2.xy)))*(cir1.xy-cir2.xy);//单位化后的法向量
        //弹力,法向力矩为0
        XY v1,v2;
        v1=cir1.v+cir1.omega*rotate_anticlockwise(-vector_n*cir1.r);
        v2=cir2.v+cir2.omega*rotate_anticlockwise(vector_n*cir2.r);
        //v1-v2为以v2代表物体为参考系的v1速度,v2-v1为以v1为参考系的v2速度
        double v_relative=normal(v2-v1);
        double a_limit=max(10000,v_relative*v_relative*0.5);//数据基于论文,限制加速度

        fn1=vector_n*((cir1.youngs_modulus*cir2.youngs_modulus/cir1.r/cir2.r)/(cir1.youngs_modulus/cir1.r+cir2.youngs_modulus/cir2.r)*dep*(4*dep*dep*PI));//压强*面积
        if(normal(fn1)>a_limit*cir1.m)
            fn1=a_limit*cir1.m*unit_vector(fn1);
        if(normal(fn1)>a_limit*cir2.m)
            fn1=a_limit*cir2.m*unit_vector(fn1);

        cir1.f=cir1.f+ fn1;
        cir2.f=cir2.f- fn1;//+=没有给向量重载，怕出BUG



        cir1.f=cir1.f-min(cir1.m,cir2.m)*PD_FACTOR* dot(v1-v2,vector_n)*vector_n;//类似于Schmidt正交化的公式，加个速度阻尼
        cir2.f=cir2.f-min(cir1.m,cir2.m)*PD_FACTOR* dot(v2-v1,vector_n)*vector_n;

        //摩擦力，不考虑摩擦力小于最大摩擦力的情况（这种实现不会有问题，否则很难实现）
        double fn=normal(fn1-min(cir1.m,cir2.m)*PD_FACTOR* dot(v1-v2,vector_n)*vector_n);//弹力大小
        XY unit_fric1=unit_vector((v2-v1)-dot(v2-v1,vector_n)*vector_n);//cir1受到的摩擦力的方向向量
        XY unit_fric2=-unit_fric1;
        cir1.f=cir1.f+friction_factor(cir1.material,cir2.material)*fn*unit_fric1;
        cir1.moment+=cross(-vector_n*cir1.r,friction_factor(cir1.material,cir2.material)*fn*unit_fric1);
        cir2.f=cir2.f+friction_factor(cir1.material,cir2.material)*fn*unit_fric2;
        cir2.moment+=cross(vector_n*cir2.r,friction_factor(cir1.material,cir2.material)*fn*unit_fric2);
    }
```

3．刚体与地形类

程序中的刚体类，即 body 类表现的是物理空间中的各种物体。类成员变量主要是分两种：一种是记录物体的各种属性的参数，如形状大小参数，以及质量、电荷、初始速度、初始角速度等物理参数。第二类是记录运动受力情况的参数，如合力、合力矩、位置、速度、加速度。以 body 父类为基础，各种图形分别实现一个子类（因为每种图形的算法都不一样），目前实现了圆形、任意三角形和矩形。

类函数为初始化、输出图像、欧拉积分的底层实现，这里再次体现了程序自顶而下的架构。有部分对不同形状不一样的图形算法如判断点是否在图形内的算法也实现在了刚体的类里。

刚体类的初始化只需输入应有的参数，然后转动惯量等参数会据此自动算出。

而地形类在实现上就是不会发生位移的刚体，其实现除了无需数值积分和外观稍有不同外，其余部分实现与刚体类相同。

4. 数学底层算法与可视化算法

程序底层还实现了大量的向量和几何算法，部分实现在类函数里，部分实现在类外。

程序实现的向量运算有：重载了的加减数乘运算，求模长、法向量，求点积、叉积，旋转向量、单位化向量，判断两线段或线段与圆是否相交，求垂足与计算碰撞嵌入深度等。后来的几何算法几乎每一步都在用上述的函数进行计算。

几何算法主要都在碰撞算法内，主要为判断是否碰撞、计算碰撞点与碰撞深度。同时在类函数内，还实现了判断点是否在图形内的算法。

可视化算法对多边形即是按部就班画出每一条边，对圆形则按照 EasyX 的函数，以圆心和半径为参数画出圆。

程序得以高效编写和正确运行全依赖于这些庞杂的底层函数，这些函数立下了汗马功劳。

```
XY operator-(XY xy1,XY xy2);
XY operator-(XY xy);
XY operator*(double num,XY xy1);
XY operator*(XY xy1,double num);
XY operator*(int num,XY xy1);
XY operator*(XY xy1,int num);


inline double normal(XY vec);//求向量模长
inline XY vertical_vector(XY point1,XY point2);//求过两点直线的法向量
inline XY rotate_vector(XY vec,double rad);//求逆时针旋转rad弧度后的法向量
inline XY rotate_anticlockwise(XY vec);//求逆时针旋转90度后的法向量,计算中用不着顺时针的
inline double cross(XY vec1,XY vec2);//向量叉积的模长
double det_2(XY vec1,XY vec2);//2阶行列式，或考虑正负的向量叉积
inline double dot(XY vec1,XY vec2);//向量点积
inline XY unit_vector(XY vec);//返回单位化的向量

inline double depth(XY pt1,XY pt2, XY pt);//计算点嵌入线的深度,不小于0
inline XY vertical_point(XY pt1,XY pt2, XY pt);//作pt到直线pt1,pt2上的垂直点
XY cross_point(XY pt1,XY pt2, XY pt3, XY pt4);//作直线pt1,pt2与直线pt3,pt4的交点,用此函数请保证有交点

void equation(vector<double> &num,XY pt1,XY pt2);//求过两点直线的方程系数,按Ax+By+C=0顺序
int is_cross(vector<XY> line1,vector<XY> line2);//判断两线段是否相交，是则返回1，否则返回0
inline int is_cross(XY line1_0,XY line1_1,XY line2_0,XY line2_1);//判断两线段是否相交，是则返回1，否则返回0
int is_circle_line_intersect(XY pt1,XY pt2,XY center,double r);//判断线段和圆是否有两个交点，是则返回1
```

三、 结论/总结

我们实现了一个高度泛化的物理引擎框架，能够很好地模拟很多物理现象。程序提供的仿真平台具有高自由度，可以自由、精准地定制输入数据。除此之外，程序还具有高可扩展性，我们可以在此框架的基础上扩展程序，处理 n 体问题、地震模拟。特别是我们的地震模拟程序，可以模拟一个楼房及物体在不同地震波下的响应。程序中，输入真实地震数据（如汶川地震、311 地震），就能得到房屋在真实地震动输入下的响应，可以粗略地从结果中得知地震下不同房屋结构的抗震性的好坏，并可以探究地震对建筑的破坏模式。虽然不如专业课题组用的有限元分析模型，但也算是我们的程序对解决实际工程问题的一个尝试吧。

这个程序也让我们感受到了物理的简洁之美。整个程序只靠计算弹力、摩擦力向量和力矩，就可以完全地模拟各种物理现象。我们没有规定球会滚下斜坡，但摩擦力矩会让它真的转起来；我们没有规定物体的多少部分悬空会翻下边缘，但弹力的力矩让物体只有在超过一半悬空时才会翻落；整个程序没有规定动量守恒和能量守恒，但给出结果却基本符

合这样的规律。当然，这不代表这个程序实现起来简单，因为真正难的部分就在于通过矢量计算和图形学算法来确定弹力、摩擦力向量和力矩，边码代码边在草稿纸上推物理和图形公式可以说是编写任何科学计算程序的常态了。

一个寒假的时间里，我们小组每个人都学习并熟练掌握了一门新语言（C++），看完了大半本《基于物理的建模与动画》[1]，完成了整个程序的开发及测试，各司其职，每个人都做出了相当的贡献，过了个充实的寒假，可谓收获颇丰。最终，我们达成了这个项目最初的目标——实现一个真实的、高度泛化的、高可扩展性的物理引擎框架。

四、 致谢

在论文的结尾，要感谢我们小组每位同学的辛勤付出：感谢我们的组长罗浩铭同学协调、组织并规划好了我们小组的各项工作，将整个程序的框架搭建起来并将组员写的所有模块组装好，且实现了高度复杂的图形化算法，还实现了颇有意义的地震模拟程序，以及 n 体问题程序；感谢胡璟凡同学并实现了高度可靠的向量和数值函数编写，在实际运行过程中，这部分一个 BUG 都没有；感谢李璐豪同学编写的刚体类，虽然刚体类成员变量众多，各项关系繁杂，容易漏更新一些变量，但实际运行过程中这部分 BUG 也很少；感谢何昕同学编写的高度友好的用户界面，平滑的缩放和拖动效果是这个可视化界面的亮点。

我们还要感谢《基于物理的建模与动画》[1]这本书，为我们提供了物理模拟中差分法、图形算法、数值积分算法等诸多算法如的教学，帮助我们克服了程序的难点，顺利地完成了程序。

也感谢这门课程，能给我们一个扎进不熟悉领域的学习机会，并且引导我们做出如地震模拟程序一样与现实问题接轨的程序。

最后，非常感谢我们的导师黄老师，感谢老师对我们项目的指导，感谢老师为我们未来研究之路指点迷津！

五、 附录

（无）

六、 参考文献

[1] Donald, [美]. 基于物理的建模与动画. 电子工业出版社, 2020.

[2] “多边形碰撞检测”, 知乎专栏, https://zhuanlan.zhihu.com/p/86981378, Accessed 14 February 2022.

五、导师评审意见

导师根据报告的整体质量、平时讨论时的表现、答辩的情况及在报告中各学生的贡献程度，着重考察学生的科学探索精神、自主学习能力、独立思考能力、逻辑推理能力、实际动手能力、团队合作精神等要素，给小组成员做出评价。 **导师签字：　　日期：**