喷泉模拟粒子系统

##### 完成功能：

* 场景建模及渲染，场景添加光照，光照可交互控制；
* 设计实现喷泉粒子系统特效，粒子运动有物理仿真，可实时切换粒子系统中的粒子三维模型；
* 粒子的三维模型的光照、纹理映射；
* 设计实现粒子系统的交互控制。

##### 项目架构：

* 包含Camera.h, CommonHeader.h, Loader.h, Particle.h, ParticleSystem.h, Texture.h, YPS.h 共七个头文件，以及 Loader.cpp, main.cpp, ParticleSystem.cpp, Texture.cpp, YPS.cpp共五个源文件。
* Camera.h 中定义了Camera类，用于实现视角的拖动转换；
* CommonHeader.h 中导入了项目需要的SOIL库及glut库，以方便其他文件导入；
* Loader.h, Loader.cpp 中定义了ObjLoader类及loadObj函数，用于加载.obj格式模型；
* Particle.h 中定义了Particle结构体，表示粒子的各种属性；
* ParticleSystem.h, ParticleSystem.cpp 中定义了一个简单随机生成粒子的喷泉模拟粒子系统，其每一时刻生成粒子的大小、颜色、运动方向都随机，视觉效果不佳，所以在main中没有使用这个类进行渲染；
* Texture.h, Texture.cpp 中定义了 loadTexture函数，实现使用SOIL库加载纹理的功能；
* YPS.h, YPS.cpp 中定义了YPS类(Yet another Particle System), 实现了一个喷泉模拟粒子系统，初始时分为五路固定方向发射粒子，粒子的颜色随时间渐变，用户可以通过键盘控制改变系统发射路数、粒子模型、粒子尺寸以及是否启用旋转发射粒子效果，也可以通过鼠标拖动改变粒子发射方向；
* main.cpp 中定义了程序的初始化函数、OpenGL的主窗口、场景的渲染以及实现交互的函数。

##### 除OpenGL外的环境依赖：

* 项目使用了SOIL（Simple OpenGL Image Library，http://www.lonesock.net/soil.html）以实现加载纹理的功能；
* 项目使用了glm（OpenGL Mathematics, https://glm.g-truc.net/）库以方便计算。

##### 环境配置：

* SOIL作为静态库已经包含在项目文件夹中，无需另外配置；
* 将压缩包中”glm”文件夹加入VC目录下”include”文件夹中。

##### 操作说明

1. 初始状态下，粒子系统加载项目目录下”Data\particle.obj”模型文件作为粒子三维模型进行发射。
2. 鼠标操作：点击左键拖拽切换视角；点击右键拖拽改变粒子发射方向；
3. 键盘操作：

* ESC键：退出程序；
* P键：暂停动画；
* U键：控制是否加载项目目录下的模型文件作为粒子三维模型（如果为否，则使用glut提供的实体对象作为粒子三维模型）
* Z键：继续动画；
* X键：控制是否启用旋转发射粒子效果（默认为否）；
* C键：调整光照明暗；
* V键：（当使用glut提供的实体对象作为粒子三维模型时）切换粒子三维模型；
* A键：控制是否绘制空间直角坐标系（默认为否）；
* S键：控制是否渲染背景（默认为是）；
* D键：增大粒子尺寸；
* F键：减小粒子尺寸；

##### 设计思路

* 设计Particle结构体存储粒子的位置、速度、颜色、尺寸、寿命等属性；
* 设计YPS类，维护一个粒子发射源和存储所有粒子的向量，每间隔一定时间对粒子状态进行更新并渲染；
* 设计Camera类，提供修改视角的接口；
* 复用第一次作业中的ObjLoader类以实现加载obj模型功能；
* 模仿游戏“东方文花帖”中的“境符「波与粒的境界」”，设计旋转发射粒子功能。

##### 技术实现

1. Camera类：这个类的实现较为简单，在setViewTransform函数中调用glRotatef函数，通过对模型坐标轴的旋转模拟视角的旋转；
2. ObjLoader类：在ObjLoader类中定义初始化函数，初始化时读入文件名并逐行读取文件，将文件中定义的顶点、顶点纹理坐标、顶点法向量和三角形面的顶点索引数据存入loader中，在ObjLoader.cpp中提供loadObj函数进行类的创建、初始化操作并且用glNewList创建一个显示列表函数原型，之后绘制时通过glCallList进行绘制。
3. ParticleSystem类：维护一个particles向量，提供update函数，每隔一段时间调用这个函数对向量中的粒子进行更新（每次更新包括对粒子位置、速度以及颜色的alpha值的更改），并且生成随机数目的新粒子；提供emit函数生成颜色、寿命、速度、尺寸均随机的新的粒子；提供render函数渲染粒子。（之后由于这个类的渲染效果不如想象中那么好，在main.cpp中没有使用这个随机生成随机粒子的类作为粒子系统）
4. YPS类：继承ParticleSystem类，实现如下功能：
5. 维护ways属性（初始化为5），通过addWays和cutWays函数修改ways属性，每次调用update函数时调用emit(ways)生成固定数目（ways）的新粒子，调用emit函数时则初始化每个粒子使得它们的速度向量的相邻夹角为2π / ways,以控制粒子发射路数以及发射方向；
6. 维护angle属性（初始化为0），通过pull函数修改angle属性，每次调用emit函数时使用angle属性计算生成的新粒子的速度，以提供拖拽控制粒子发射方向的功能；
7. 维护timeCnt, angularV, spinning属性，其中timeCnt记录时间（初始化为0），angularV为当前旋转发射角速度，spinning表示当前是否启用旋转发射粒子效果，每次调用emit函数时通过spinning判断是否启用旋转发射粒子效果并且增加angularV，如果启用则使用angle属性以及angularV属性计算生成的新粒子的速度，实现粒子匀加速旋转发射效果；
8. 维护rAdd, gAdd, bAdd以及currentColor属性，每次调用update函数时通过对rAdd, gAdd, bAdd判断如何更改currentColor中的RGB属性，并且在更改currentColor后更新rAdd, gAdd, bAdd, 之后调用emit函数时将currentColor赋值给生成的新粒子的颜色属性，以实现粒子颜色渐变效果；
9. 维护size属性，通过changeSize函数修改size属性，每次调用emit函数时将size赋值给生成的新粒子的尺寸属性，以实现控制粒子尺寸的效果；
10. 维护loadModel, modelSize及regHex属性，提供load函数用于加载指定路径的obj模型文件并记录modelSize属性及regHex属性，每次调用render函数时通过loadModel判断是否使用加载的obj文件作为粒子模型，如果是则通过modelSize属性和size属性计算模型需要缩放的倍数（模型可能太大或太小，需要调整到接近粒子大小的尺寸），之后通过调用glCallList(regHex)进行粒子的渲染；
11. 维护shape属性，通过changeShape函数修改shape属性，每次调用render函数时如果使用glut提供的实体对象作为粒子三维模型则根据shape属性渲染对应模型，以实现切换粒子模型的效果；

##### 遇到的问题及解决方法

1. 关于场景建模：

* 问题 1 ：由于决定复用第一次作业中的ObjLoader类，只能加载obj格式的模型，但在网上少有obj格式的并且符合我个人审美观的场景模型。
* 解决方法：下载3dsMax软件，临时自学并且创建了地面、水池模型，另外从网上下载了长椅模型并通过3dsMax重新导出成obj格式；
* 问题 2 ：初次设置好场景后决定每次获取当前粒子系统的颜色属性（currentColor）并且用这个颜色进行水池的渲染，但渲染效果不佳，水池颜色随着粒子颜色变化而不停闪烁；
* 解决方法：设置一个固定的颜色用于渲染水池。

1. 关于粒子的纹理映射：

* 问题：如果是加载obj文件作为粒子的模型，那么可以从文件中获取粒子的纹理坐标；但如果是使用glut提供的实体对象作为粒子三维模型，粒子的纹理坐标难以指定，而如果不指定则渲染效果极差；
* 解决方法：使用OpenGL提供的自动生成纹理坐标功能（参考了网页” OpenGL学习二十二：纹理坐标自动生成” http://tiankefeng0520.iteye.com/blog/2007939）。

1. 关于旋转发射粒子特效：

* 问题：原本的设想是实现右键拖动改变粒子发射方向，粒子上升时x, z轴方向上的速度固定，但下降时x, z轴方向上的速度变化，形成在原点正上方沿y轴负方向观察时粒子运动轨迹呈扇形的功能（大致如下图1，其中粒子沿箭头所示轨迹运动，虚线为拖动前粒子发射方向），但实现时计算模型有问题，导致很长一段没能得到想要的结果；

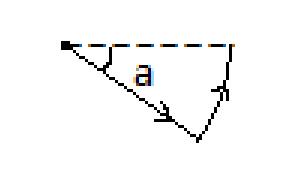


图1

* 解决方法：修改原先的设想，使得右键拖动改变粒子发射方向后粒子做与之前的设想相比较为简单的抛物线运动，并且另外实现了模仿游戏“东方文花帖”中的“境符「波与粒的境界」”的效果的旋转发射粒子功能，（主要通过使得发射粒子的转速匀速加快来实现，参考了网页”「波与粒子的境界」公式” <https://tieba.baidu.com/p/2244198482> ），效果大致如下图2所示，（图中粒子系统ways参数为5，但由于旋转发射粒子的转速在匀速增加，使得粒子构成的图形在视觉上形成了10路发射的效果，并且这个图形也在不断变化），这个效果我个人也比较满意，于是决定采取这种做法。

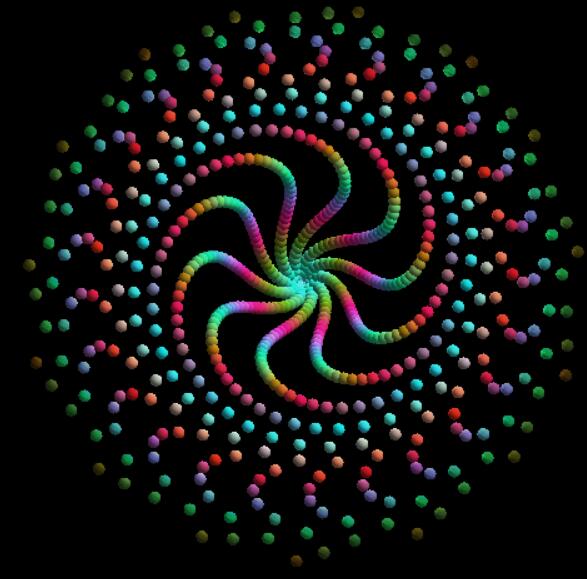


图2

1. 关于粒子颜色渐变的实现：

* 问题：最初使用的实现方式是维护一个colorAdd属性，每次调用update时判断当前颜色rgb值中若有一个或多个大于0.9则设置colorAdd为false,，若有一个或多个小于0.1则设置其为true，之后通过colorAdd判断是对currentColor的rgb值增加一个随机数还是减少一个随机数，这种实现方式导致currentColor不能稳定地变化，有时很长一段时间粒子都保持一个颜色；
* 解决方法：调试过程中观察到currentColor中有时rgb中同时存在一个大于1.0和一个小于0的数，于是修改实现方式，使用三个属性（而不是一个）分别判断当前是否要对R, G或B属性进行增加，并且在设置currentColor后对三个属性都进行判断和重新赋值，使得粒子颜色能稳定渐变，效果如上图2所示。

##### 未能解决的问题

1. 如果加载obj文件作为粒子的三维模型并且模型较为复杂，在运行时会出现明显的掉帧；
2. 由于是初次使用3dsMax，创建的场景模型比较简陋，不够美观；
3. 没有找到较好的图片作为粒子的纹理，导致渲染效果仍然不是很美观。

##### 致谢

感谢肖老师这个学期课堂上下的指导、敦促；感谢助教在微信群里的指导。祝老师和助教新年快乐☺。