粒子系统

● 简述

本次作业的主要内容是完成一个简单的例子系统

● 立场构建

在本次作业中我们需要实现 GravityForceField, ConstantForceField, RadialForceField , VerticalForceField, WindForceField 这四个立场。

首先是重力场，在这个类中我们只需要用构造函数接受外部传入的参数即可，并且在 getAcceleration 函数中根据牛顿第二定律返回加速度即可：

$$\vec{g} = \frac{\overrightarrow{gravity}}{mass}$$

常量立场的实现和重力场基本上相同，所以只需要用下列式子返回加速度即可：

$$\vec{a} = \frac{\overrightarrow{force}}{mass}$$

径向力场（RadialForceField）的作用是将粒子拉向原点，并且力的强度正比于原点的距离，注意到框架在创建径向力场类的时候会传入一个 magnitude 变量，这个变量代表的就是“强度”，即力的强度与原点距离之比。同时力的方向指向原点，所以要记得反向，否则可能会出现意想不到的结果。径向力的加速度的计算公式如下：

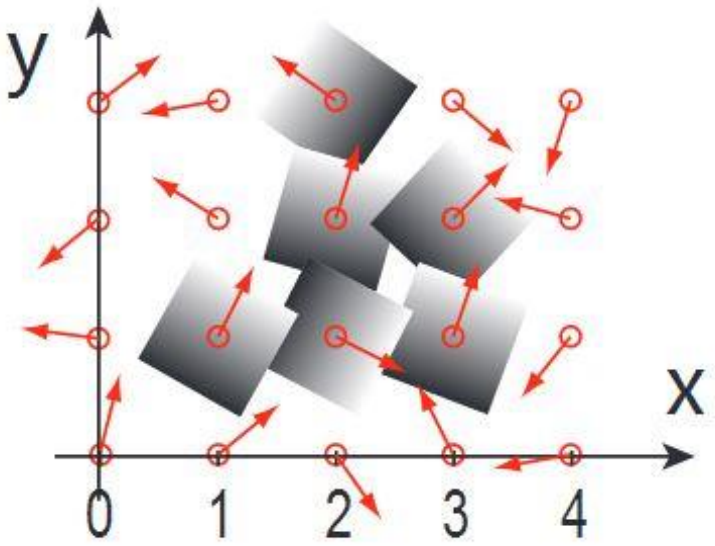
$$\vec{a} = \frac{-\overrightarrow{position}}{mass}$$

垂直立场（VerticalForceField）的作用是将粒子拉向 xOz 平面，和前面相同的是力的强度正比于粒子到平面的距离，这个比值的大小为 magnitude。同样需要注意里的方向，计算公式如下：

$$\overrightarrow{force} = (0, -y * magnitude, 0)$$

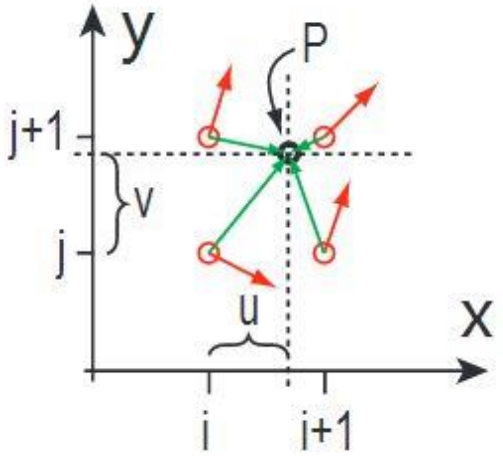
$$\vec{a} = \frac{\overrightarrow{force}}{mass}$$

风力场（WindForceField）的作用是模拟风力，我们这里需要模拟一个平滑随机变化横向风，为此需要一个一维的柏林噪音。首先我们需要在一维坐标轴上，选择等距的无穷个点，将值空间划分为等长的线段（为方便起见，选择坐标值为整数的点），为每个点随机指定一个值和一个梯度（在一维的情况，梯度就是斜率），如下图所示：



对于每一个评估点 P(x,y)，可以取他周围四个点梯度（下图红色部分），并作如下标记：

$$g00 = gradient(i, j)$$
$$g01 = gradient(i, j+1)$$
$$g10 = gradient(i+1, j)$$
$$g11 = gradient(i+1, j+1)$$



如上图所示，我们可以求出图中绿线所代表的四个向量，然后将它和对应点的梯度进行点乘，得到：

$$n00 = g00 \cdot \begin{bmatrix} u \\ v \end{bmatrix}$$
$$n01 = g01 \cdot \begin{bmatrix} u \\ v-1 \end{bmatrix}$$
$$n10 = g10 \cdot \begin{bmatrix} u-1 \\ v \end{bmatrix}$$
$$n11 = g00 \cdot \begin{bmatrix} u-1 \\ v-1 \end{bmatrix}$$

计算完上面后，取一个特别的函数：

$$f(t) = 6t^5 - 15t^4 + 10t^3$$

带入下面的式子进行计算：

$$n_{x0} = n_{00}f(u) + n_{10}(1 - f(u))$$
$$n_{x1} = n_{01}f(u) + n_{11}(1 - f(u))$$
$$n_{xy} = n_{x0}f(v) + n_{x1}(1 - f(v))$$

这样便能通过两个输入值 x 和 y 得到结果值n_{xy}了。这样在实际使用的时候考虑风力的随机因素，我们传入 mass 和 t，然后乘以一个系数，将力施加在 x 轴的方向就能够用更加平滑的随机数来模拟真实的风吹动粒子的效果。

● 积分器构建

积分器的构建这一部没有什么好说的，只需要在 Update 函数里取出当前粒子的位置与速度，然后根据相应的公式对其进行修改，然后调用 increaseAge 方法增加粒子的年龄即可，具体每一个积分器的实现公式在说明中都有详细的描述，这里就不多说明了。

● 产生器构建

产生器构建中一个核心问题就是如何随机化数据。

在水管产生器（HoseGenerator）中，速度和位置都需要混以随机性。我的思路是，在以传入的位置为圆心，传入的随机度为半径的圆内等可能的随机生成向量，公式如下：

$$\vec{\imath} = \left(\frac{\sqrt{2}}{2} * random(-t,t), \frac{\sqrt{2}}{2} * random'(-t,t), 0\right), t = randomness$$

$$\overrightarrow{position} = \overrightarrow{position} + \vec{\imath}$$

$$\overrightarrow{velocity} = \overrightarrow{velocity} + \vec{\imath}'$$

对于如 lifespan、mass 这类的标量，其随机化处理比较简单：

$$mass = mass + random(-t,t)\ , t = randomness$$

$$lifespan = lifespan + random(-t,t)\ , t = randomness$$

对于环形产生器（RingGenerator），我们的目的是在一个环形的带上随机产生质点，并且质点的数量还要随实践增加，对于数量我们只需要在 numNewParticles 函数返回的时候加上当前时间乘以一个系数就好了，而对于位置的随机，我们需要借用圆的参数方程：

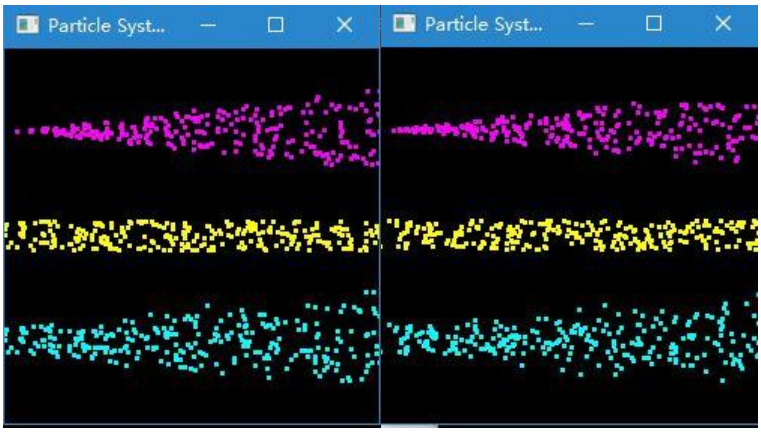
$$\begin{cases} x = r\sin t \\ y = r\cos t \end{cases},\ t \in [0,2\pi]$$

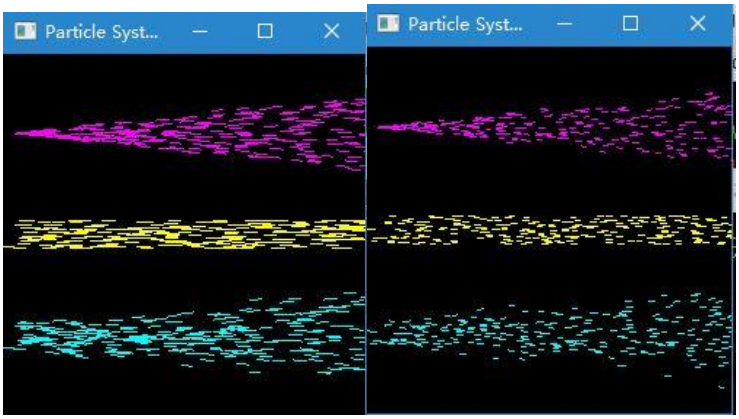
上面这个方程中半径我们使用和质量、声明周期一样的办法给他混入随机性，对于参数 t 只需要让它能够均匀的随机产生即可。

- 试验结果

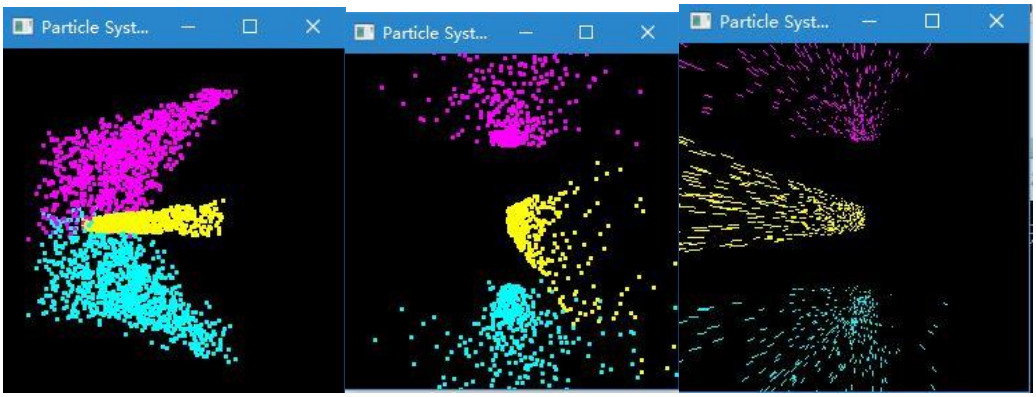
这次的试验只要并且只有完成了立场（ForceField），积分器（integrator）和产生器（generator）才能够进行调试。下面是试验结果：

-input system9_01_hose.txt -refresh 0.1 -dt 0.1
-input system9_01_hose.txt -refresh 0.05 -dt 0.05
-input system9_01_hose.txt -refresh 0.1 -dt 0.1 -motion_blur
-input system9_01_hose.txt -refresh 0.05 -dt 0.05 -motion_blur

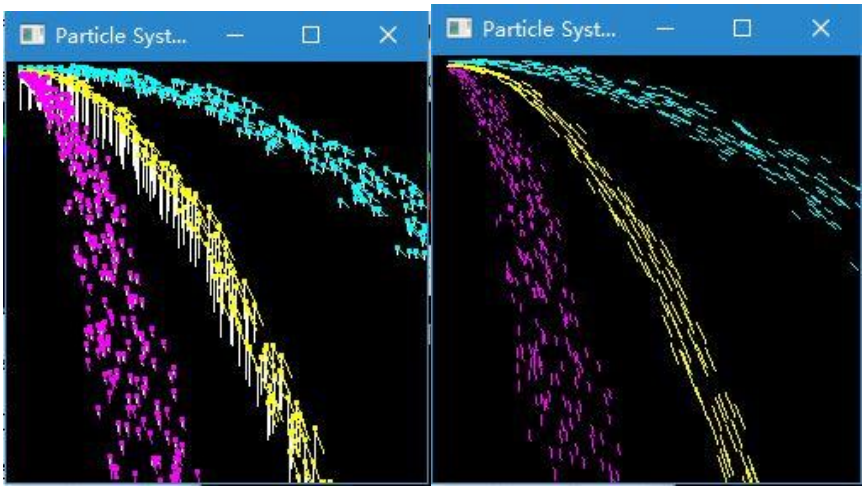




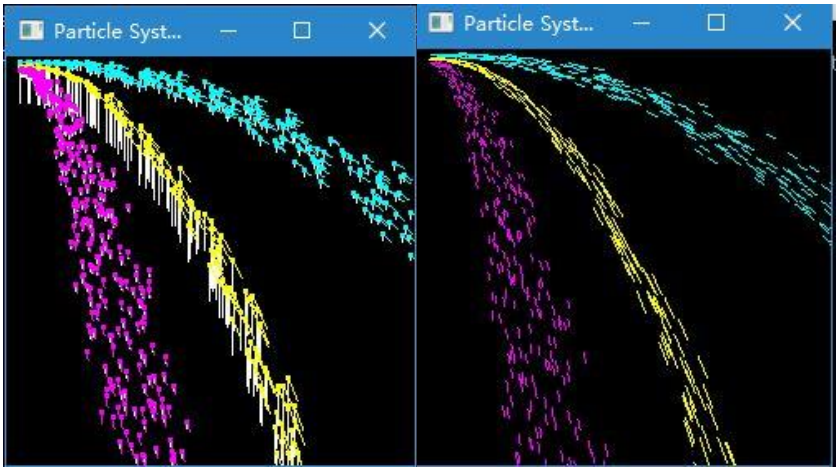
对代码进行修改我们能够得到 3D 效果的水流：



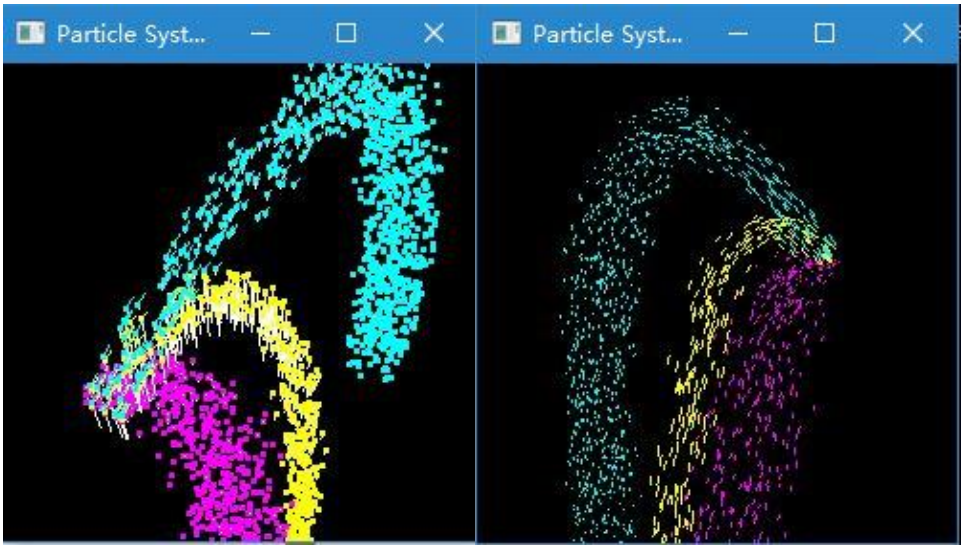
-input system9_02_hose_gravity.txt -refresh 0.05 -dt 0.05 -draw_vectors 0.1
-input system9_02_hose_gravity.txt -refresh 0.05 -dt 0.05 -motion_blur

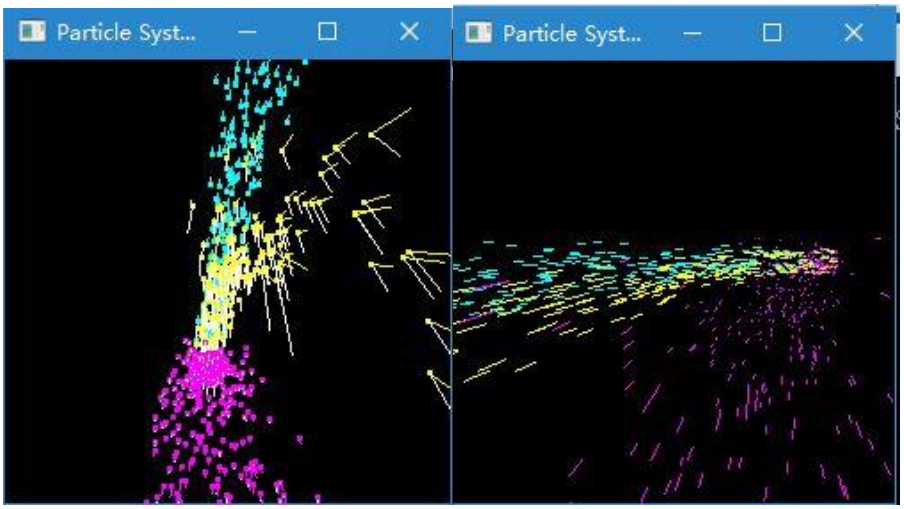


-input system9_03_hose_force.txt -refresh 0.05 -dt 0.05 -draw_vectors 0.1
-input system9_03_hose_force.txt -refresh 0.05 -dt 0.05 -motion_blur



3D 效果的如下图示：

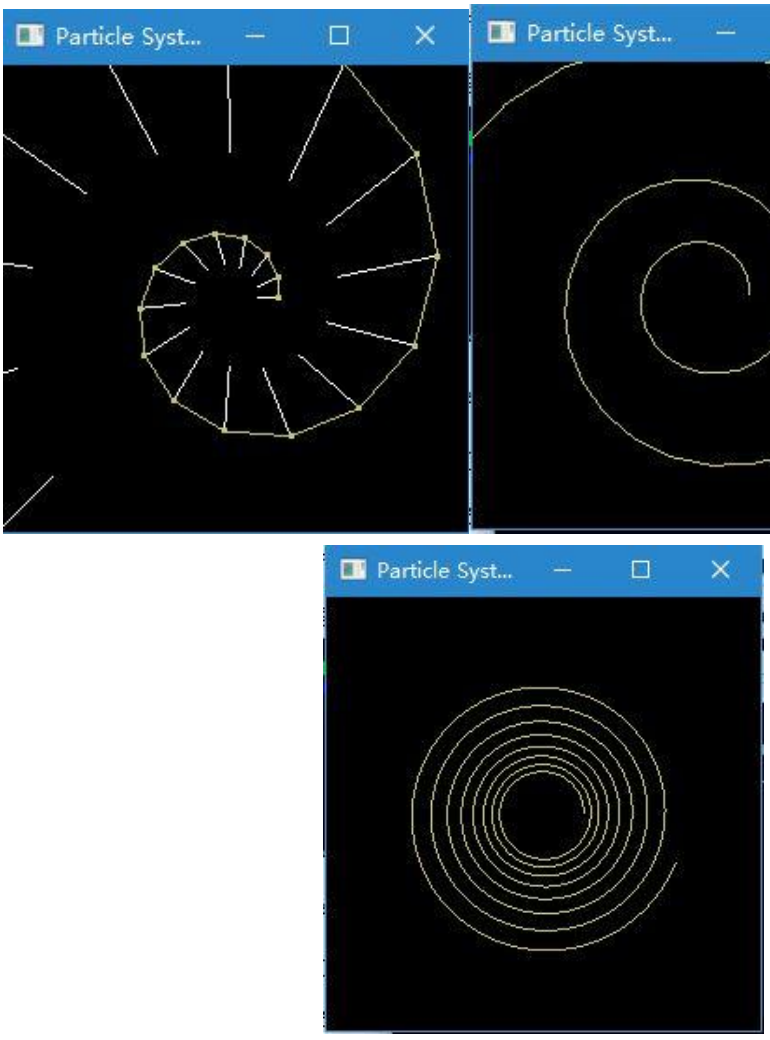




-input system9_04_circle_euler.txt -refresh 0.1 -dt 0.1 -integrator_color -draw_vectors 0.02

-input system9_04_circle_euler.txt -refresh 0.05 -dt 0.05 -integrator_color -motion_blur

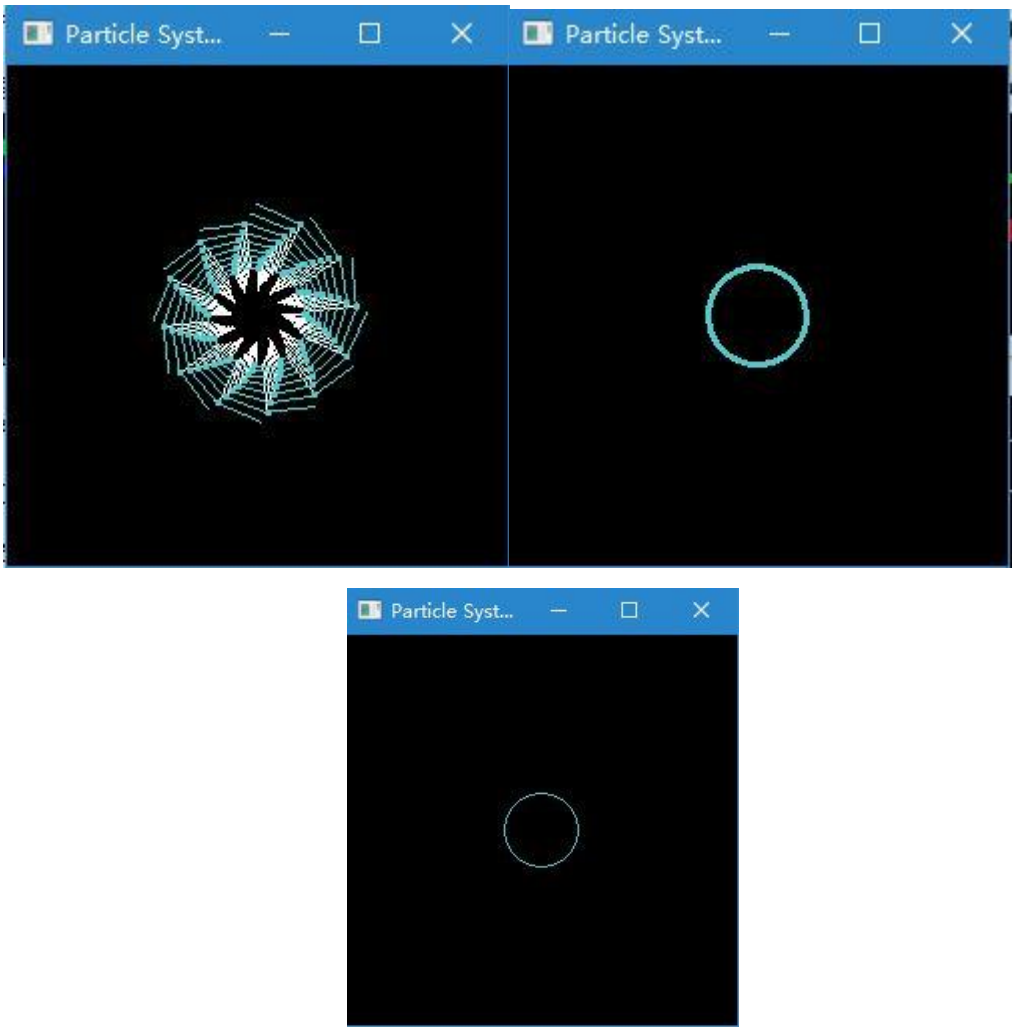
-input system9_04_circle_euler.txt -refresh 0.01 -dt 0.01 -integrator_color -motion_blur



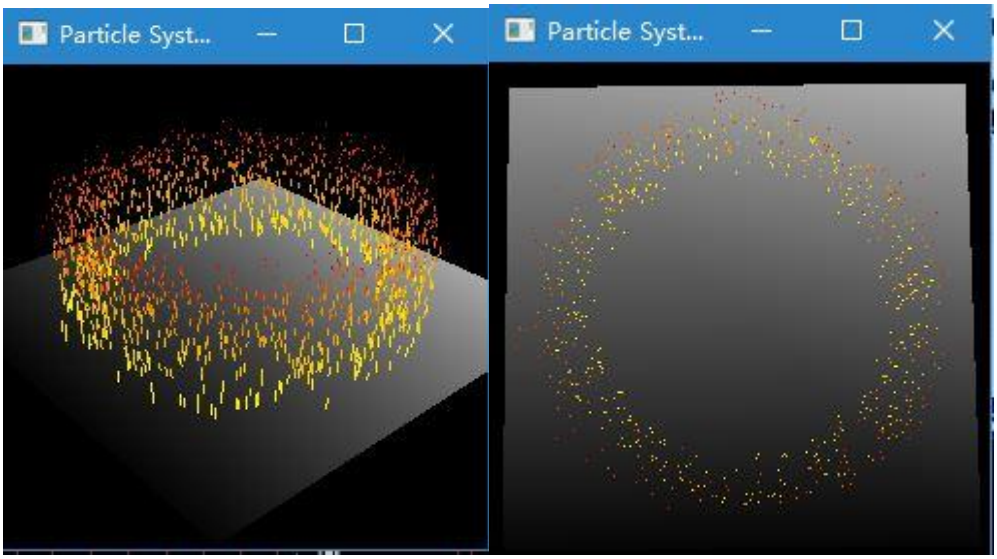
-input system9_05_circle_midpoint.txt -refresh 0.1 -dt 0.1 -integrator_color -draw_vectors 0.02

-input system9_05_circle_midpoint.txt -refresh 0.05 -dt 0.05 -integrator_color -motion_blur

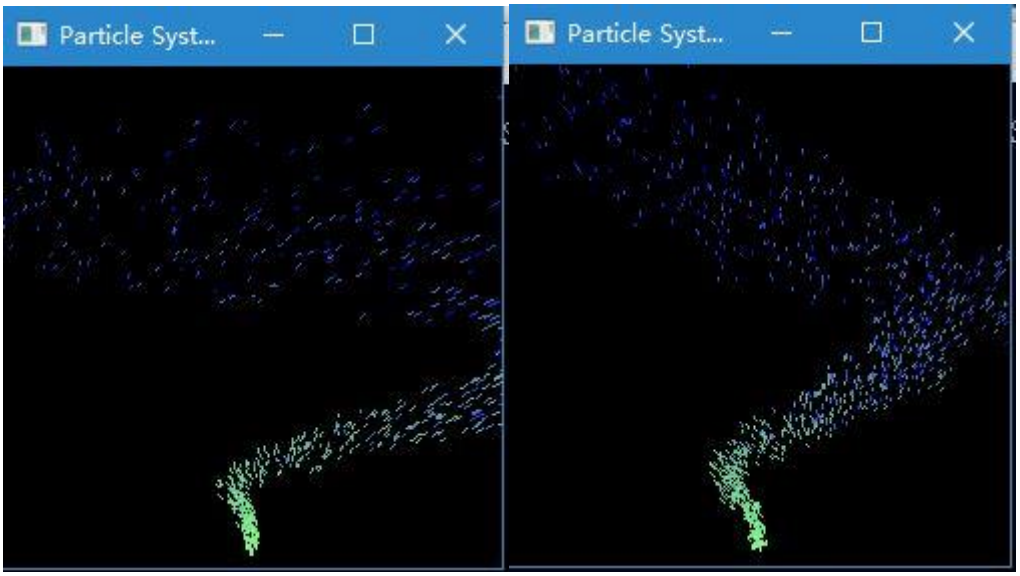
-input system9_05_circle_midpoint.txt -refresh 0.01 -dt 0.01 -integrator_color -motion_blur

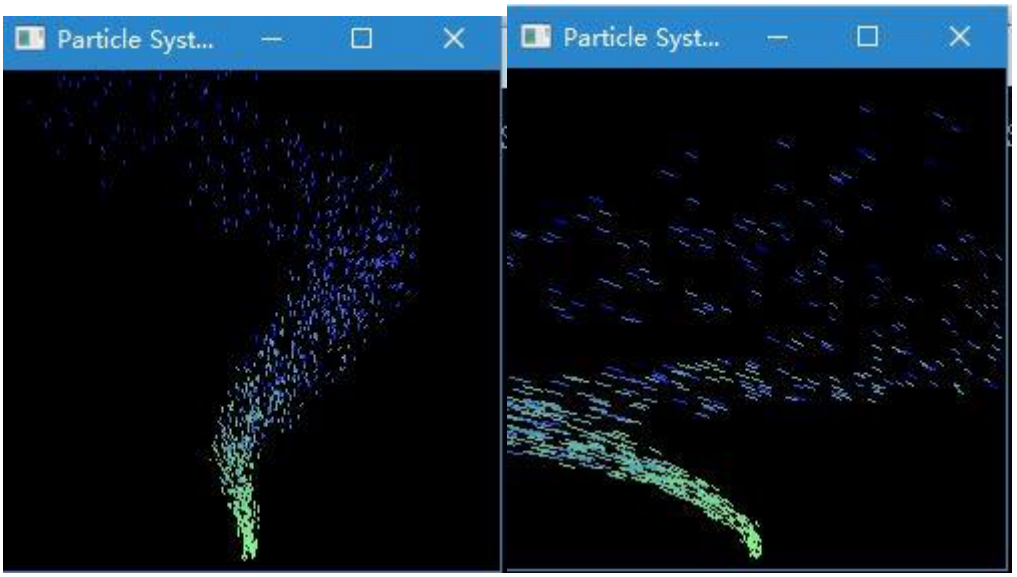


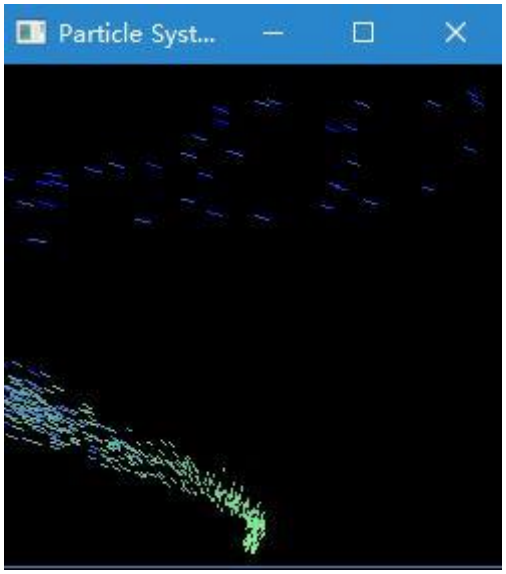
-input system9_08_fire.txt -refresh 0.05 -dt 0.05 -motion_blur



-input system9_09_wind.txt -motion_blur -dt 0.05 -refresh 0.05







- 参考文献

[Simplex noise demystified]:
http://webstaff.itn.liu.se/~stegu/simplexnoise/simplexnoise.pdf