1、特效区分：具体场景特效（主城，战斗）、UI特效

2、依据特效区分，做贴图合并，共用材质，使用Texture Sheet Animation切分图集，动态批处理

3、使用animation代替animator

4、特效渲染顺序确定（地裂、爆炸...）

5、使用Start Color和Color over Lifetime来改变粒子颜色（它们都是顶点色），而不是使用shader上的tint color

6、尽量使用Billboard代替mesh粒子，Billboard抛弃了顶点位置，UV，顶点颜色等熟悉，更轻易的进行批处理合并

7、使用alpha blend（垫底色）和additive（曝光）两shader，两shader均需要支持顶点色，不需要带tint color

8、不同材质的发射器赋予不同Order in layer值，相同材质的发射器使用相同的Order in layer值，有助于避免渲染打断（即避免出现“ABA”的情况）

9、同屏幕粒子发射个数需要限制

10、开启动态批处理

11、命名要规范，拼音也可以

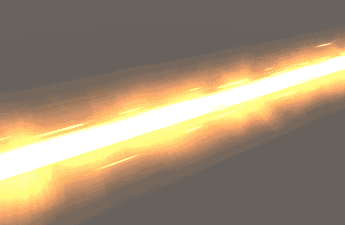
参考：

1、https://www.cgjoy.com/thread-188808-1-1.html

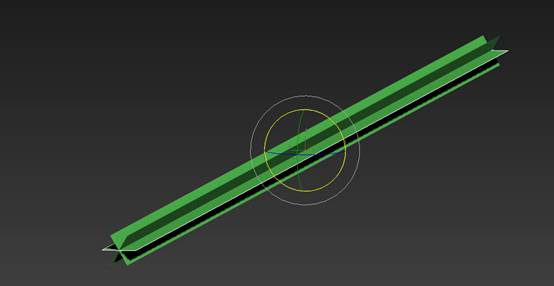
2、https://www.cgjoy.com/thread-211517-1-1.html

在制作U3D特效过程中，如何尽可能用少的DC和资源做出丰富的效果，是特效师常常需要考虑和注意的问题。今天灵灵奇就和大家分享一个小技巧，如何利用模型和UV来减少DC和贴图资源。

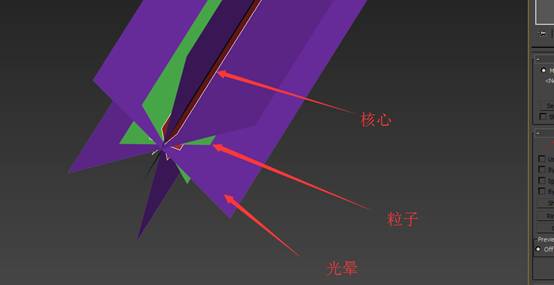
我们将以一个简单的激光案例进行讲解，用一个DC和一张128贴图进行制作。了解它的原理后，就可以把这个技巧用到其他效果上啦~



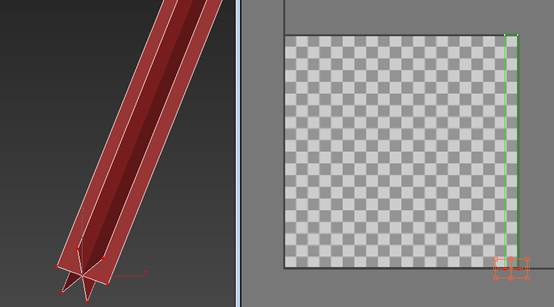
**1、在MAX中，创建一组交叉面，添加“editPoly“”修改器，将它们“Attach”在一起**

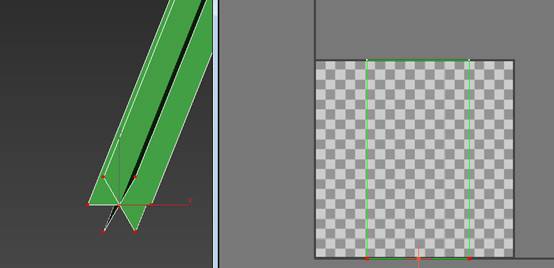


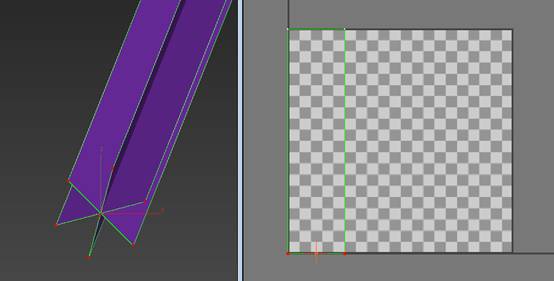
**2、复制出另外两组，并进行大小和旋转的差异**



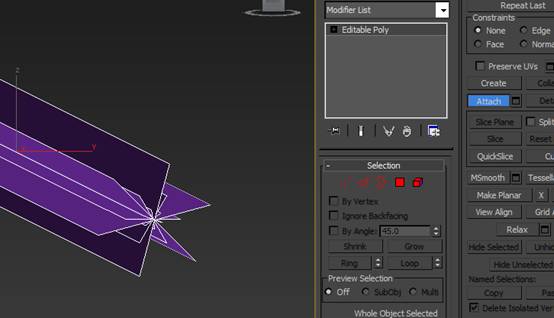
**3、分别添加“Unwrap UVW”修改器，将UV进行缩放错位，以便区分。**



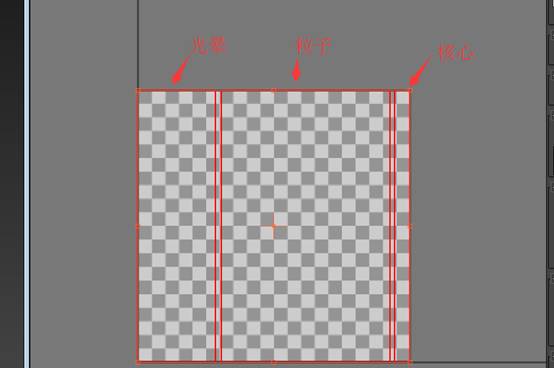




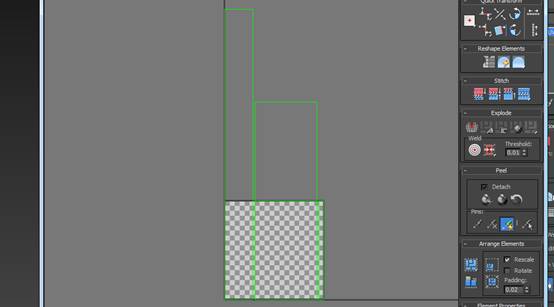
**4、将三个模型转成“editable poly”，并“Attach”成一个模型**



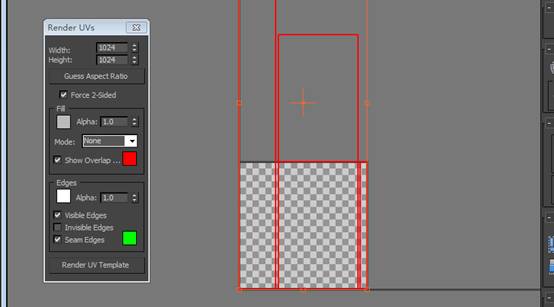
**5、再次添加“Unwrap UVW”修改器，整体调节他们的UV分布**



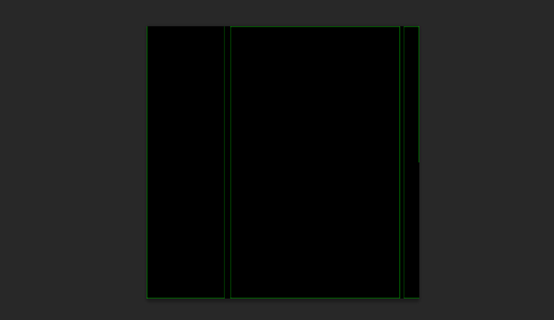
**为了实现速度上的差异性，调整UV的拉伸比例**

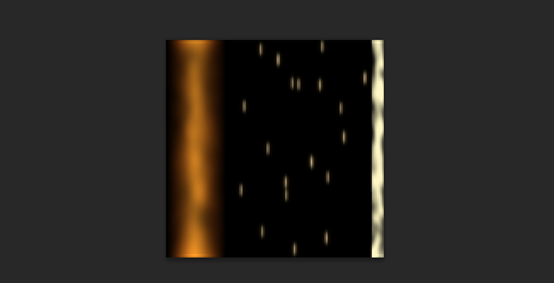


**6、渲染UV图**

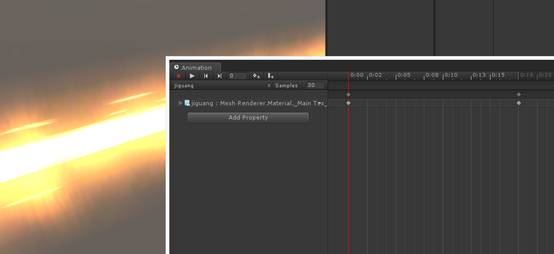


**7、将UV图导入PS中，根据UV分布绘制贴图，将核心、粒子、光晕都绘制在同一张图上。**

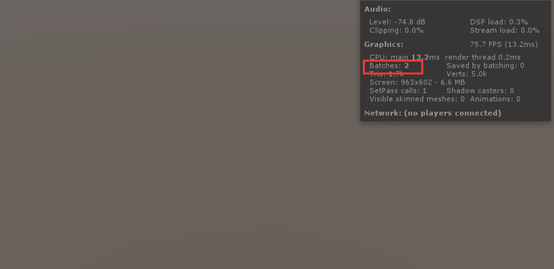




**8、将模型和贴图导入Unity，并设置UV动画。这个操作很简单，在这里就不一一演示了。通过上面的操作，我们就得到了一个简单的激光效果。**



**由于我们只用了一个模型和一个材质球，所以整个特效只占用一个DC。**





本案例，主要基于以下几点来控制DC：

1、两个不同的模型，即使用的是相同的材质球，也会产生两个DC

2、两个相同的模型，使用同一个材质球(这里指系统自带的particle Additive 和particle ALphaBLend ， 其他shader 可自行测试一下 )且材质球参数完全相同（如果有动态，则动画曲线也要完全相同），则只产生一个DC

3、两个相同的模型，使用同一个材质球但材质球参数不完全相同(即使是动画曲线不同)，则产生两个DC

**主页面Main**

--------------粒子出生时------------------



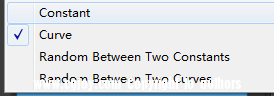
**Duration（持续）：**粒子发射的持续时间。值是秒。

**Looping**（循环）：让粒子发射时间循环起来，一直发射粒子。

**Prewarm**（预热）：开始播放粒子是已经是发射了一段时间的。只在looping循环时才有效。

**StartDelay**（开始延时）：粒子系统创建后隔多久后才开始发射粒子。如果有Prewarm则没法设置延时。值是秒。

初始值大部分可以设置它的四种变换。



**Constant**（常量）：固定的值，不会变化。

**Curve**（曲线）：用曲线代表Duration时间内的变化。竖轴是这个值的变化、横轴是Duration时间。但注意的是初始的值，出生时的不同，一旦出生了，是其他值控制。

**Random Between Two Constant**（随机在两个常量间变化）：随机值

**Random Between Two Curve**（随机在两个曲线间变化）：随机曲线

**StartLifetime**（初始生命）：决定它将持续多少秒消失。

**StartSpeed**（初始速度）：运动快慢。它的单位是米/秒。（米是一格子的长度，并不是真的米，就是一个单位长度，说米是便于理解。）

**3DStart Size**（3D初始尺寸）：可以让粒子在x、y、z三个轴上有不同的尺寸。当粒子是billboard模式时，z轴调节无意义。在粒子发是mesh且Render Alignment不朝摄像机时调节这个值，很有用。

**StratSize**（初始尺寸）：总体设置初始的尺寸。Xyz三轴同步大小。值米。

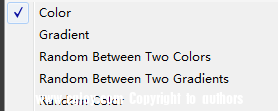
**3DStart Rotation**（3D初始旋转）：粒子可以绕着XYZ三个设定不同的角度。

**StartRotation**（初始旋转）：默认是只设置ｚ轴方向旋转角度。值是角度。

**FlipRotation** （反跳旋转）：主要针对设定的Start Rotation值进行反方向的变化。假如旋转是10。这个值设置为1，就会变成-10。设为0.5就会变成一个从-10~10的随机。

**StartColor**（初始颜色）：出生时的颜色。

颜色上经常有这几种变换。



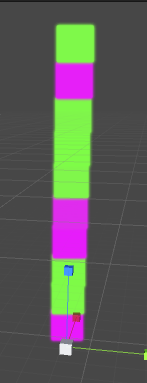
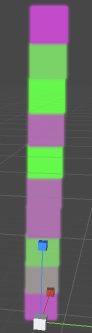
**Color**（颜色）：就是设置一个颜色。

**Gradient（**渐变）：可以设置一个渐变色。从左到右是Duration时间。

**Random Between Two Color**（随机在两个颜色变化）：随机颜色。

**RandomBetween Two Gradient**（随机在两个渐变变化）：随机渐变。

**Random Color**（随机颜色）：它跟RandomBetween Two Color最大的不同是可以控制随机颜色的数量。

**GravityModifier**（重力修正器）：粒子施加一个向下的力。负值是向上的。注意这个值是一个外部的力，他不随粒子系统缩放而改变。

**SimulationSpace**（模拟空间）：就是使用的空间坐标。粒子的运动所使用的空间坐标。

**Local**（自身）： 使用自身粒子系统的游戏物体的坐标。

**World**（世界）：使用世界坐标，就是场景坐标。

**Custom**（制定）：使用另一个物体的坐标。当这个物体坐标变化时，粒子也会跟着变化的，比如物体的位移会带动粒子位移。

                   Customsimulation Space（制定模拟空间）：制定这个物体。

**SimulationSpeed**（模拟速度）：整体改变粒子的运动快慢，注意不只是粒子的速度。而是粒子总体属性，在空间位置中的变化，整体放慢或加快。

**Delta Time**（变量时间）：Delta Time是一个函数。确切意思是上一帧的时间情况。因为设备刷新率的不同，所以一帧消耗的时间也是不一样的。这个Scaled是调节使用时间变化而非帧的变化。每帧走1米和每0.1秒走一米是不一样的。默认应该是Scaled（缩放的）是正确的。

**Scaling Mode**（缩放模式）：缩放粒子系统时的状况。

         Hierarchy（阶层）：？？默认值，不清楚含义。

**Local**（自身）：效果上是和hierarchy一样，如果粒子系统缩放、粒子的运动距离和大小同时被缩放。Gravity Modifier的力不会被缩放。

         Shape（形状）：只会缩放发射器的大小。

**Play On Awake**（唤醒就播放）：勾掉时，运行时出现粒子系统时并不会播放，直到在程序中够上这个勾，才会播放，便于程序控制。

**Emitter Velocity**（发射速率）:是关于粒子移动方式。不太明白？？transform能理解。Rigidbody啥含义？物理系统。

**Max Particles**（最大粒子数）：为了节省开支设定粒子数量的上限。

**AutoRandom Seed**（自动随机种子）：粒子呈现的状态是由seed种子绝对的，相同的seed产生的结果是相同的。而seed数十分庞大，如果用了随机种子，你可能永远不会碰到数值一样，粒子状态一样的情况。

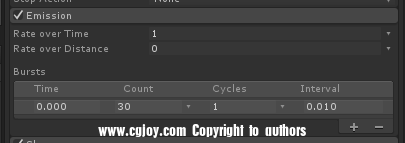
**Random seed**（任意种子）：可以设置种子。Reseed（播种）可自动填写一个种子数。粒子不是每次产生的效果都是好看的，这里确保这个好看的粒子效果一直存在。

**Stop Action**（停止行动）：当粒子系统发射完了所有粒子之后的程序的处理方式。跟消耗有关。

**None**（无）：就是无处理。内存被动释放。

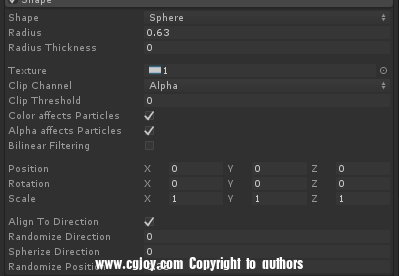
**Disable**（禁用）：就是被废弃状态，不会在出现，但是还在内存中。

**Destroy**（摧毁）：被摧毁掉。内存会直接释放掉。

   
**Rateover Time**（发射频率按时间）：1秒发射的数量。曲线时间轴是Duration时间。  
**Rateover Distance**（发射频率按距离）：1米发射的数量。曲线时间轴是Duration时间。  
**Bursts**（迸发）：  
从第几（Time）秒开始。发射多少个（Count）。这次发射循环几次（Cycles）。每次循环间隔多久（Interval）。  
如果使用一个效果隔多久迸发一次，只需要一个Bursts设置Cycles和Interval就行。  
Count的曲线时间轴是Duration时间。  
CyCles可以选择infinite（无限）模式，会一直循环。

**发射器形状Shape**

-------------------十样杂耍-----------------------



**Shape**（形状）：**是Sphere（球）和Hemisphere（半球）时**。

**Radius**（半径）:单位米。

**Radius Thickness**（半径厚度）：球体由壳至核心的厚度。0是在壳上发射，1是整个球体发射。

**Texture**（图片）：设置一张图片在发射器上，根据这图片的颜色和透明区域，可影响粒子的颜色和透明度。

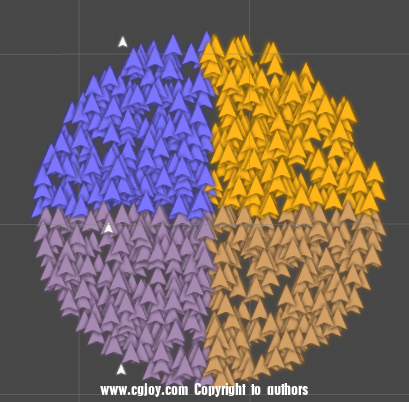
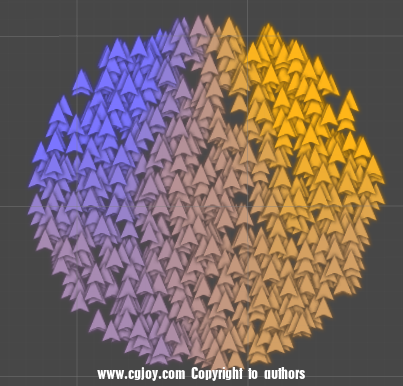
Clip Channel（裁剪通道）：clip对图片进行裁剪，控制粒子发射的范围。设置 R G B A 四个通道作为裁剪的起始，黑色始，白色末。

**Clip Threshold**（裁剪起始）：0是不裁剪，1是最大裁剪。

**Color affects** Particles（颜色影响粒子）：图片的颜色。

**Alpha affects Particles**（透明度影响粒子）：图片的alpha。

**Bilinear Filtering**（双线过滤）：对图片进行优化处理。双线处理主要是让像素比较少的图片进行一下类似模糊的效果，从而“感觉”会好一点。具体的是从临近的四个像素点进行双线插值计算，得到像素颜色。像素游戏经常有这个选项。

 不勾  勾

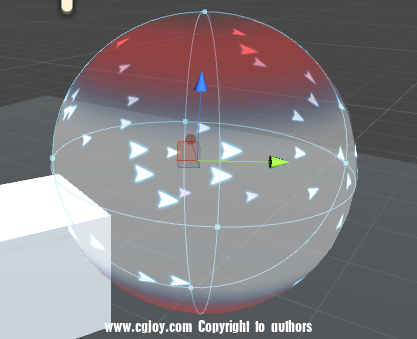
**Position、Rotation、Scale**。注意区别于其GameObject的transform。这些相当于一个子物体了。

**Align To Direction**（趋向排列）：使粒子的方向（y轴）都趋向一个趋势（粒子z轴不再朝着摄像机，而是跟发射器形状而定，失去公告版功能）。观测，这个趋势是物体的z方向。圆形显示就是贴到球面上，但并不是真正的贴上，只是值是碰巧贴上的。注意趋向，不是指向。是一个接一个的传递。

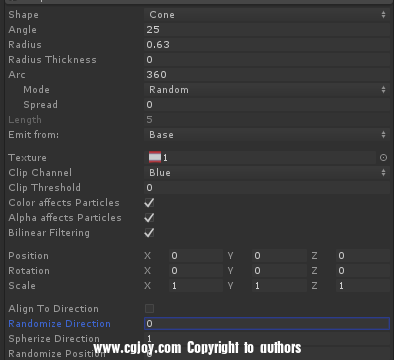
**Randomize Direction（**随机趋向）：使粒子的y轴随机。0是不随机。1是最大随机。

**Spherize Direction**（球面化趋向）：使粒子以一个球体的形态趋向于物体z方向。圆形就会贴到发射器上。如果是有厚度的，相当于每个厚度都是一个圆的形状，一层一层的。

**Randomize position**（随机位置）：距离发射器原有发射点的位置远近随机值。



**Shape是cone（圆锥）时。**



**Angel**（角度）：角度。

**Arc**（弧度）：对应圆形状的发射器的有效作用的弧度。一周是360度。要注意一点是圆的形状，当是cone是底部的圆。

**Mode**（模式）：提供了粒子发射情况形成的动画效果。

**Random**：默认是随机。整个区域内都发射粒子。

**Loop**：发射源变成圆上的一个点，不断在区域内做圆周循环发射粒子。（效果类似于用动画k一个圆周运动的粒子系统以world为轴运动，形成的拖尾效果）

**Ping-pong：**乒乓。运动到终点在返回，往复。

**Burst Spread：**迸发散布。当**bursts**发射粒子时，时数量平均分布。



**Spread**（散布）：在弧度内分布情况。值设置的逻辑很有趣。是上面的Arc（假设360）乘以这里设置的值，就是每隔多少度才发射粒子，0\*360=0，不存在间隔，每个地方都会发射粒子。0.1\*360=36。每间隔36度才会发射（圈内有10点发射粒子）。0.5\*360=180。每间隔180度才会发射（圈内有2点发射粒子）。在bursts发射粒子时，能使粒子数平均分布。

**Speed**（速度）：在loop和ping-pong时。控制这个点运动速度的。

**Emit from**（发射来自）:

**Base**：基础。只有一个底部发射粒子。

**Volume**:体积。形成一个真正的圆锥体发射粒子。

**Length**（高度）：只有在Volume模式下能才有高度。

**其他值参上方。**

**Shape是Dount（甜甜圈）时。**



先定义，整个圈的圆是大圆。围成这个圈的圆柱的圆是小圆。

**Radius：**是大圆半径。

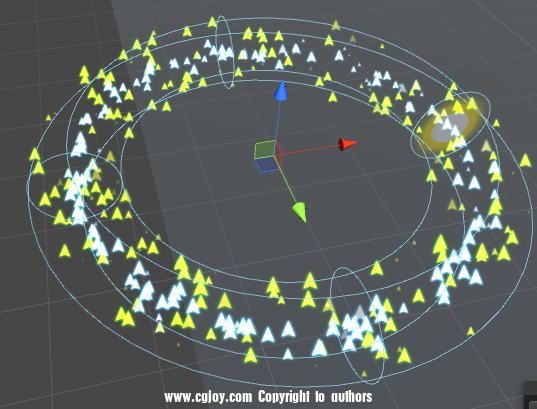
**Dount Radius**（甜甜圈半径）：小圆的半径，单位米。

**Arc**（弧度）：所对应的弧度是大圆的。

**Mode** 下Burst Spread的散布变的不太有效。Burst Spread是对应的大圆，平均分布的。但是分布在小圆的却是随机的。所以视觉上仍是比较混乱。除非小圆的半径很小，才能看的出来。

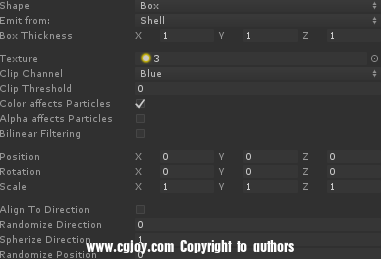
Texture的图分布在小圆的横截面上。

Spherize Direction是以大圆趋势指向的。



**其他值参上方。**

**Shape是box（盒子）时。**



**Emit from**（发射来自）:

**Volume**（体积）：整个体积内发射粒子。

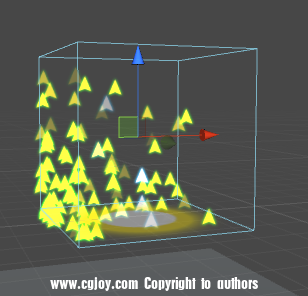
**Shell**（壳）：只在表面发射粒子。

**edge**（边缘）：只在变上发射粒子。

**Box Thickness（**盒子厚度）：设置了三个轴，和上面不同的是，这个厚度的0是一个面或边，1是对应的另一个面或边。所以调整三个值都为1时，粒子会缩到一个脚发射。怀疑是程序员偷懒，1应该是盒子中心位置才对。

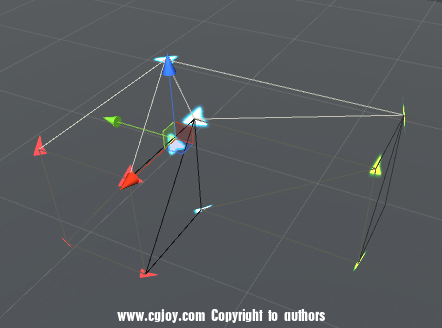
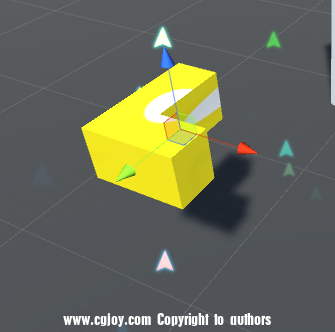
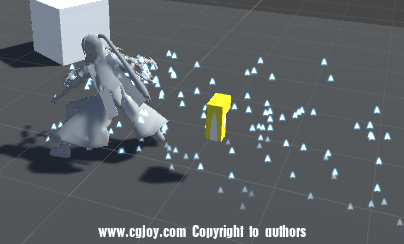
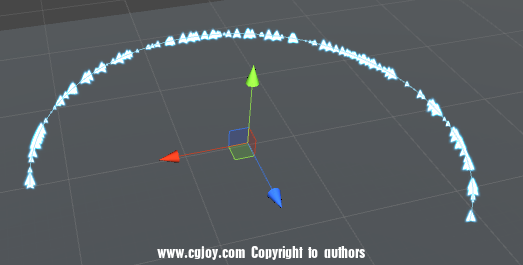
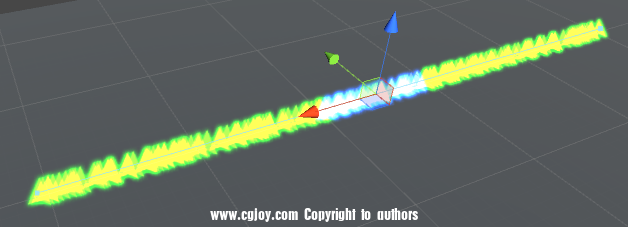
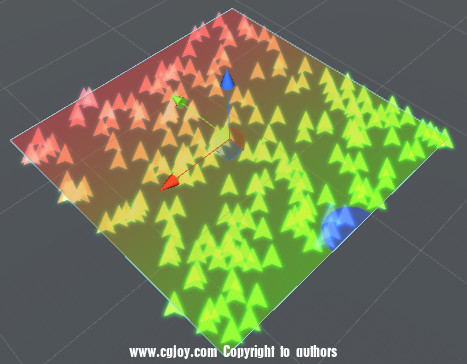
Texture的图片是在底面上。

Spherize Direction是box的切面球体。



**其他值.....参上方。**

**本帖隐藏的内容**

Shape是**mesh**（网格）时。  
  
   
**Type**（样式）：  
         **Vertex**（顶点）：只在mesh顶点发射。  
         **Edge**（边缘）：只在mesh边上发射。  
         **Triangle**（三角面）：在模型的三角面上发射，实质上就是模型的表面了。  
**Single Material**（独立材质）：可以在程序中指定由特定的索引号的mesh发射。特效用不着。  
**Use Mesh Color**（使用网格颜色）：就是mesh的顶点色会影响到粒子颜色。  
**Normal offset**（法线偏移）：根据法线往外偏移。数值单位是米。应该是和法线的计算方式有关。注意这里和Randomize position的区别，Randomize position是根据xyz轴进行扩散偏移的。而这里是法线的位置，偏移后发生截面不会扩大，而且模型的法线是可以更改的。  
**UV channel**（UV通道）：可以让图片使用不同的UV通道，一个模型可以设置多个UV存在不同的通道中，一般场景模型都有2套UV。  
   
其他值.....参上方。  
  
  
Shape是**mesh  Renderer**（网格渲染）时。  
属性和mesh是一样。但是可以指定场景中的物体的。发射的位置也会变成场景中物体发射。Texture相关功能仍会起作用，但是图片无法被显示了。  
  
   
  
  
Shape是**Skinned  Mesh Renderer**（骨骼网格渲染）时。  
含义和mesh Renderer差不多，但是可以指定一个绑定骨骼的mesh，随动作摆动。  
   
  
  
  
Shape是Circle（线圈）时。  
发射器是一个圆圈。功能和用法与Cone差不多，只是只有一个底部的操作而已。Texture在圆心平铺。个人感觉完全没存在的必要。  
   
  
  
  
  
  
Shape是**Edge**（边缘是）时。  
发射器是一条线。功能参上。Texture因为是一条线读取的是图片x值为0的y上一条线，就是图片最底部的线。存在意义不大。  
  
   
  
  
Shape是**Rectangle**（矩形）时。  
发射器是一个长方形。Texture就是平铺，其他功能参上。存在意义也不大。 

**生命时间内的速率 Velocity over Lifetime**

---------------无垠宇宙。



**Linear（**线性）：粒子朝X，Y，Z轴的方向匀速移动。曲线时间轴是Duration时间。

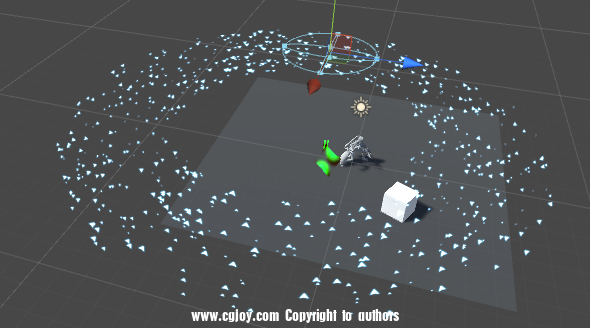
**Space**（空间）：可设置为是世界空间world或自身空间local。

**Orbital**（轨道）：可以使粒子绕着某个轴自行旋转。数值是可以快旋转的速度。观察到，值为1时，粒子绕行一周所需粒子生命大致为6.28s。是pi\*2。

**Offset**（偏移）：在Orbital旋转的过程中在施加一个轴方向的速度。能使粒子不在绕着中心自转，而是形成一个圆环，适合模拟星系。这个速度和上面linear施加其上，效果看是差不多的，但是计算方式是完全不同，这里不做深究，只要注意Orbital所旋转的轴，这个轴方向是无法施力的就行。增加值，可以是圆环范围扩大。值为1时，半径是1m。

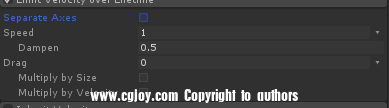
**Radial**（半径）：这个值增加整个粒子的运动形成一个螺旋线。值越大，范围越大。默认是0，就是一个圆。还是深究下，这里的Orbital、Offset其实用到的是一个渐开线，而半径是指基圆的半径。

**SpeedModifier**（速度修正）：如果上面的各项值都调好了，忽然想在增加下粒子运动速度时，可以修正性的调节。效果上是和Orbital轴上的值是一样的。



———————————————————————————————————————————————————————————————————————————— **生命时间内限制速度Limit Velocity over Lifetime**

----------------爆炸必点项



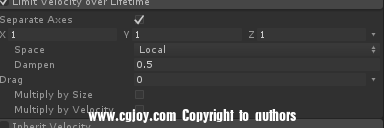
**Speed**（速度）：限制后的速度。时间轴是duration时间。

**Dampen**（抑制）：抑制的含义是在粒子生命期间的速度由初始速度改为当前设置的速度，值越小抑制的越晚。值越大抑制的越早。值0是整个粒子生命时间内没有抑制。1是整个时间内完全抑制。抑制速度不会大于初始速度。

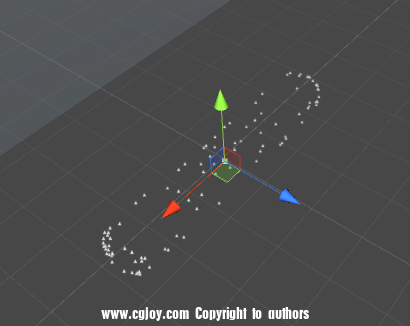
**Drag**（拖拉）：类似于抑制的功能。是另外的一个控制粒子运动速度的功能。类似于一个跟粒子速度反向的一种力。设置的值和初始速度有关，初始速度越大，能拖住粒子所需的力就约大。值和粒子初始速度一样时，达到一个临界值，粒子运动到终点时，速度为0。

**Multiply by Size**（乘以尺寸）：和粒子尺寸关联起来。用Drag这个值乘以粒子的Size值。当粒子大小小于1时，这个值会被缩小，大于1会被加强。

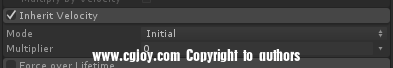
**Multiply by velocity**（乘以速率）：让他在和速率（包含初始速度）关联起来。用drag乘以速率。就可以实现一个效果：设定一固定的drag值，速度增加时，拖拽的力也跟着增加。



**SeparateAxes**（分离轴）：当勾选时。出现可设置的xyz轴和空间坐标选择。可设定xyz三轴方向上的抑制后的速度。会更改发射器影响的粒子方向。

   
———————————————————————————————————————————————————————————————————————————— **继承速率Inherit Velocity**

----------粒子甩出去



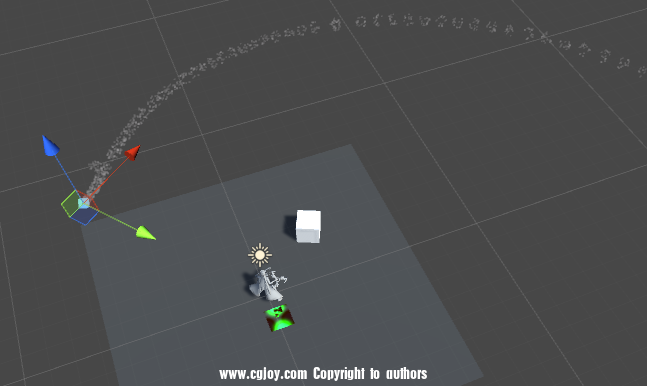
继承速度的含义是粒子继承了粒子系统这个物体位移速度。前提是粒子使用的空间坐标是world才行。

**Mode**（模式）：提供了两种initial（初始）current（当前的）。

**Multiplier**（乘值）：是速度要乘的数，速度是粒子系统在（世界）空间坐标中的速度，

         initial（初始）：计算粒子出生时的粒子系统速度。1就是原始速度，2，就是2倍速度。小数，速度会变慢。

         current（当前的）：计算的是实时的粒子系统速度。粒子系统愈快，粒子的运行速度就越快。但值为1的时候，粒子移动速度和粒子系统速度一致，所以视觉上看不出粒子的位移，大于1或小于1时才有意义。



———————————————————————————————————————————————————————————————————————————— **生命时间内受到的力Force over Lifetime**

------------------加速度移动。



给粒子施加力，力的含义参考重力，一个物体收到重力影响是一个加速运行的。这里也的力也一样。粒子持续收到力，是加速度移动的。本质上是和Gravity Modifier是一样的，但GravityModifier可以看做是一个外部施加的力。而这个则是内部的力。

给予了xyz三个轴方向的力，可设置自身空间或是世界空间。

**Randomize**只在两个值随机时有效，含义不太明确。

————————————————————————————————————————————————————————————————————————————

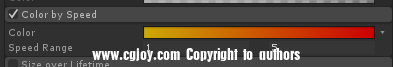
**生命时间内的颜色color over Lifetime**

略。

————————————————————————————————————————————————————————————————————————————

**颜色受到速度影响color by speed**

---------------------万年不用



含义很简单就是粒子速度的快慢会获得不同的颜色影响。首先颜色是个变色才有意义。因为粒子快慢都是一个颜色也看不出来。

如果所示，粒子速度是1的时候是黄色，是5的时候是红色，中间值就是中间颜色。

**SpeedRange**（速度范围）：设定受到颜色影响最小速度和最大速度。

至于粒子速度可以在初始速度的时候给个随机获得不同，也可以用Velocity over Lifetime、Limit Velocity over Lifetime、 Force over Lifetime等功能改变粒子速度。好像是废话。

————————————————————————————————————————————————————————————————————————————

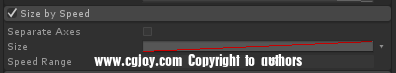
**生命时间内的尺寸Size over Lifetime**

略。

————————————————————————————————————————————————————————————————————————————

**尺寸受到速度影响Size by speed**

---------------------万年不用2



含义和 color over lifetime类似，Size必须是个变化的才有意义，这里是个曲线。

如图当速度为1时，尺寸为最小。当速度是5时，尺寸为最大。

**SeparateAxes**（分离轴），可以单独设定每个轴方向的尺寸变化。

————————————————————————————————————————————————————————————————————————————

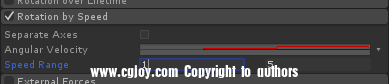
**生命时间内的旋转Rotation over Lifetime**

略。

————————————————————————————————————————————————————————————————————————————

**旋转受到速度影响Rotation by speed**

---------------------万年不用3



含义和以上color over lifetime类似，Angular Velocity（角度速率）必须是个变化的才有意义。

**AngularVelocity**（角度速率）：角度旋转的速度。是转速，容易和角度混淆。

如图当速度为1时，旋转速率为最小。当速度是5时，旋转速率为最大。

**SeparateAxes**（分离轴），可以单独设定每个轴方向的尺寸变化。

————————————————————————————————————————————————————————————————————————————

**外部的力External  Forces**

---------------------把粒子吹跑

https://attach.cgjoy.com/attachment/forum/201806/05/200003mu5s7s675pcpcp6x.png

值很简单，就一个乘值。

外部力是指粒子系统以外的，施加在空间中的力。比如Unity自带的Wind Zone就是一种力，勾选这里就会让粒子收到风的影响，默认是1就是正常值，要加强影响就增加值。

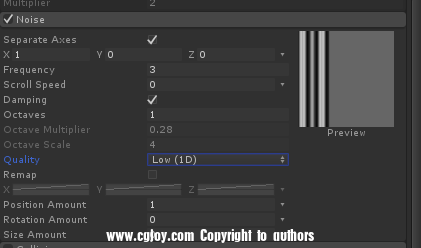
————————————————————————————————————————————————————————————————————————————

**噪波 noise**

---------------------花絮乱飞。

Noise被翻译或变化成噪波、噪音、杂波、云彩、噪点等五花八门的名字。本质上就是一张黑白波动的可调节的图。利用这张图做遮照从而影响图片通道、力、时间、模型等等你能想到的变化。

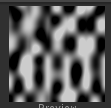
首先理解下这个功能的运作原理：是以一张噪波形式的信息作用于粒子。用黑白强弱影响粒子的速率、转速和方向、和尺寸。这张图可以是一个1维的一条线上影响、也可以是个2维一个平面上影响、还可以是个3维、一个立体的体积影响。



**Preview**（预览）：只是预览噪波信息形成的图像，并不是真的这个用到的这张图，只是便于让你看到数值调整到一个什么程度。这张图其实是平均竖分成三分的，左中右分别代表了x、y、z轴上的信息。



这张图是可以在Quality中设置一维、二维、三维、三种级别的噪波图。其中二维和三维从图形上是看不出区别的，因为就是在一个2d界面上的(⊙﹏⊙)b。

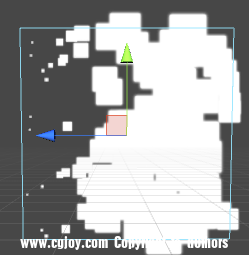
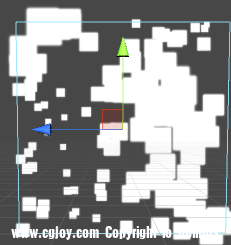
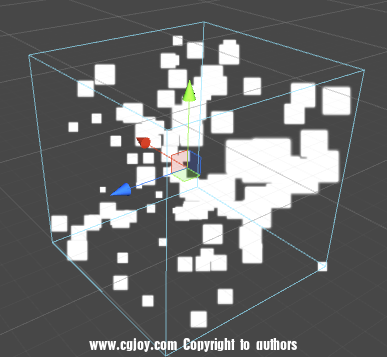
 1D  2D  3D

**Strength**（影响力）：

**从噪波图理解**：就是黑白的程度，值越大对比度越强，黑的更黑，白的更白。值是可以超出黑白色溢出的。

**从影响粒子的尺寸理解**：黑的会缩小粒子尺寸、白的会放大粒子尺寸。越黑缩的越小、越白放的越大。注意当Separate Axes分开轴时，粒子如果是公告版实质上产生效果的只有x轴。

       1D图时只会在一条线上影响，图上蓝轴方向。2D图时会在一个平面上影响，会在蓝轴和绿轴方向影响。3D图是一个体积，三个轴都会有影响。

 1  2  3

**从影响粒子的转速理解**：黑的会逆时针旋转、白的会顺时针旋转。越黑逆针转的越快、越顺针转的越快。当粒子是公告板时，轴起作用的只有z轴，z一直朝向摄像机才是公告板。

**从影响粒子的速率理解：**白的会朝一个轴的正方向移动、黑的会朝一个轴的反方向移动、越白正方向速度越快、越黑反方向速度越快。

当是1D图时，就是朝三个轴的两个方向移动同时随机快慢发射，三个轴同时发射，就会形成一个随机紊乱的粒子走向了，原理类似于Velocity over Lifetime三个轴给正负随机的效果。当是2D图时就变的有趣了，正常来说只一个平面的值，但是当粒子移动后后会得到一个抑制的力，变成一个减速运动，直至速度变成零，一旦尺寸和转速同时附加了速度的变化，获得了这个抑制力，转速和尺寸也会变化趋于一个中间的初始值。3D图的情况和2D一样。

**Separate Axes**（分离轴）时，可以单个调节每个轴。

**Frequency**（频率）：

**从噪波图理解**：就是缩放整个噪波图。值越大噪波图缩的越小。值小大噪波图放的越大。

**从影响粒子的尺寸理解：**和噪波图的情况一样，影响的更加细分了。

**从影响粒子的转速理解：**和噪波图的情况一样，影响的更加细分了。

**从影响粒子的速率理解：**速率就和其它情况不一样了。首先要理解一个概念：噪波图影响的信息并不是真正影响粒子速度的。而是粒子在多大的空间内位移，strength越强,空间越大，                粒子速度自然移动的越快了。而这里的frequency变大了，是指上是空间被拉长了，类似于粒子系统的scale。而strength相当于发射器的大小。效果上看他和strength调大是差不多的。同时在       2D图以上时，它的抑制的力会随着frequency增加。

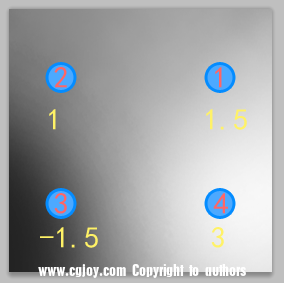
**Scroll Speed**（滚动速度）：

**从噪波图理解**：就是滚动起来形成一个循环，值大就滚动的越快。

**从影响粒子的尺寸理解：**粒子受到的尺寸随着时间会变化。效果上就是一个粒子会变大变小。

**从影响粒子的速率理解：**粒子受到的转速方向随着时间会变化。效果上就是一个粒子忽左忽右的转。

**从影响粒子的速率理解：**粒子受到的速度和方向随着时间会变化。效果上就是一个粒子。忽然朝一个方向，然后转向反方向。如图假如有粒子2在0s受到了-2的值影响；在1s后手打-1的影            响。那在0s的时候以-2会朝反方向位移，到1s后又以1的速度朝正方向移动。

 0s     1s

**Damping**（抑制）：个人理解是提供一种抑制力，解决一些过小的影响产生的抖动。

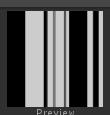
**Octaves**（八度音阶）：奇特的名字其实就是噪波的分层或者叫复杂度。就在在原有的黑白图的变化上在增加一层细微的变化，八度音阶是CDEFGABC八个音级，通常说的上8度下8度就是这个意思，这里只是一个比喻。值是代表分多少次。

**Octaves multiplier**（分层乘值）：就是和原始图的混合程度。0是没有混合，随值增加颜色越来越深。1是完全和原始图一个程度了。

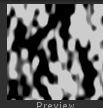
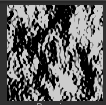
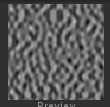
**Octaves scale**（分层尺寸）：值1是和原始图一样大。值越越大细节越小。

       整体这个分层的意义不并不大，首先消耗会很大。而直接效果和Frequency增大值的效果很类似，除非在一些非常非常精细的情况下会有区别。所以建议忽略这个功能。

       1D噪波在Octaves值1、2、3的情况和增加Frequency值的情况

 1  2  3    Frequency增加

2D噪波在Octaves值1、2、3的情况和增加Frequency值的情况。3D和2D类似。

 1  2  3     Frequency增加

**Quality**（质量）：就是提供了1D的噪波、2D的噪波和3D选择的地方。

**Remap**（重置图）：重新调节噪波图的黑白。类似于曲线的功能。可以从新定义粒子收到的影响。

**Position Amount**（位置合计）：粒子受到到的位移影响、某种层面说就和strength值一样。

**rotation Amount**（旋转合计）：粒子受到到的旋转影响、值是每秒多少度。

**Size Amount**（尺寸合计）：粒子受到到的尺寸的影响。

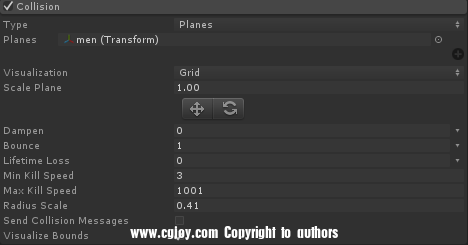
    这三个值可以设置为0，选择性的让噪波影响粒子。



————————————————————————————————————————————————————————————————————————————

**碰撞collision**

---------------------有碰撞就会有结果。



**Type**（类型）：提供了两种plan和world类型。World主要针对场景中有collider组件的物体（包含terrain）会产生碰撞。Plan是指定一个的物体的transform，作为一个面产生碰撞。

**Planes**（面）：指定物体的transform为面。这个物体可以是场景中的也可以是资源中的。主要是用他的变换值。一版是和场景中的物体互动是有意义的，如果只是想得到一个碰撞可以使用空物体，会剩一点。

**Visualization**（形象）：有Grid（网格）和Solid（固体）两种。只是便于查看的显示方式。

**ScalePlane**（缩放片）:就是缩放Visualization这个片，还是为了便于观察。

**两个按钮**：可以直接在这里对指定的物体反向操作。

**Dampen**（抑制力）：碰撞后，粒子的速度被抑制多少，1是抑制到速度为0，粒子会停留在片上，0是没有抑制。

**Bounce**（弹力）：碰撞后弹起的情况。1跟初始速度一样的速度反弹。0是不反弹，粒子会朝平面方向运动

**LifetimeLoss**（生命损耗）：碰撞后，粒子生命所短的数量。1是缩短到最大。碰撞后粒子生命结束。0是没有损耗。

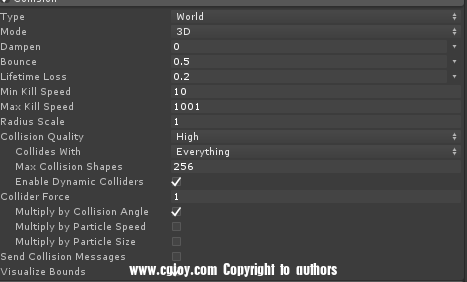
**MinKill Speed**（速度小于值消灭粒子）：碰撞后的粒子速度小于这个值被摧毁。

**MaxKill Speed**（速度大于值消灭粒子）：碰撞后的粒子速度大于这个值被摧毁。

**RadiusScale**（半径的尺寸）：其实粒子的碰撞是和场景中其他碰撞是一个原理的，每个粒子都有一个圆的碰撞组件。计算碰撞就是计算这个组件和其他物体的碰撞的。这里调节半径尺寸是调的每个粒子碰撞大小。

**SendCollision Messages**（发送碰撞信息）：程序相关，程序会得到相关信息作出处理。

**VisualizeBounds**（显示界限）：就是显示出碰撞框。



Type是World时。

**Mode**（模式）：是2d碰撞组件还是3d碰撞组件。

**CollisionQuality**（碰撞品质）：设定了三种级别。Low和Medium使用是静态碰撞。什么是静态碰撞？凡是直接记录在内存中的东西通常都是静态的，记录在内存中的碰撞叫静态碰撞。为什么放到内存中？计算小节约资源不用每次都实例化。但会有个问题，就是不能移动，一旦移动这样的碰撞就会耗时重新计算，直接导致粒子无法产生碰撞。所以根据情况如果粒子系统经常被移动那就使用high。如果总是固定不变的可以使用后两种。这个移动是相对于世界坐标的，就是就算你在父物体中没动，但父物体动了也算。

**Collides With**（发射碰撞和..）:设置什么层会发生碰撞、什么层不会。方便操作。

**Max Collision Shapes**（最大碰撞体数量）：粒子最多和多少个物体（每个物体1个碰撞组件）产生碰撞。

**Enable Dynamic Colliders**（启用动态碰撞组件）：这个不明白，动态碰撞是个啥玩意儿，提示说：这是让粒子碰撞到会动的物体上？居然打问号，无语.......

**CollierForce（碰撞组件的力）：**程序相关。

         Multiplyby Collision Angle（乘法受碰撞角度影响）

         Multiplyby Collision particle Speed（乘法受碰撞粒子速度影响）

         Multiply byCollision particle Size（乘法受碰撞粒子大小影响）



————————————————————————————————————————————————————————————————————————————

**事件Triggers**

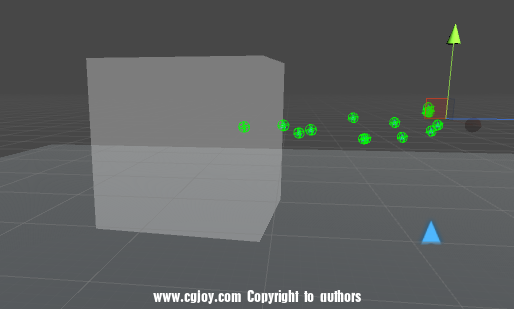
---------------------还是一个碰撞。



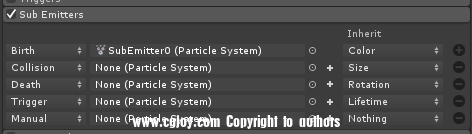
设置了**Inside**（内部）**Outside**（外部）**Enter**（进入）**Exit**（出去）四种状态的**Ignore**（忽略）、**kill**（消灭）、**callback**（回调）三种情况。回调的含义是程序的回调函数。

**Colliders**（碰撞的物体）：应该是个模型才有意义。真正起作用的是物体上面的collider组件。

**RadiusScale**（半径的尺寸）和**Visualize Bounds**（显示界限）和碰撞功能里一样。



————————————————————————————————————————————————————————————————————————————

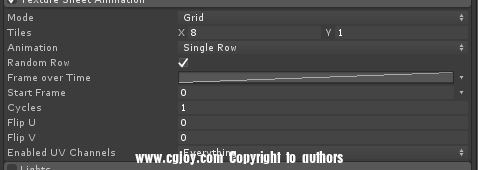
**子发射器 Sub Emitters**  


设置了粒子在五种情况下产生子粒子的情况：**Birth**（出生）、**Collision**（发射碰撞）**Death**（死亡）、**Trigger**（触发时间）。  
**Inherit**（继承）：设定了**Color**（颜色）、**Size**（尺寸）、**Rotation**（旋转）、**Lifetime**（生命）**Nothing**（没有任何）几种继承其粒子属性的值。

**图片小片动画Texture Sheet Animation**

**---------------你要的序列图**

**Mode**（模式）：提供了Grid（格子）和sprite（雪碧）两种模式。



**Tiles**（平铺数）：一张图标被切割的数量。x是横轴的数量，y是纵轴数量。

**Animation**（动画）：有两种形式，

**Whole Sheet**（全部小片）：从左至右、从上至下播放整个图片里的小方框。

**Single Row**（单排）：只播一横排，因为可能会有很多排。当勾选了Random Row（随机行）时，随机选中一行播放。不够时，有一个Row值供手选使用哪一行。

**Frameover Time**（时间内的帧数）：通常设置曲线是有意义的，设置数值只会显示固定的一帧（并非是帧，而是第几个小图）。注意的是小图片的第一张的序号是0。

**StartFrame**（开始帧）：从第几帧开始播放，仍会完整读取序列。

**Cycles**（循环）：正常图片的整个循环是在粒子的生命时间内循环一次。这里可以改变循环几次。

**Flip U**（U跳转）:每个小图片在U也就横轴方向会产生左右翻转、1是全部翻转。0是没有翻转、中间值是随机翻转。

**FlipV**（V跳转）：每个小图片在V也就纵轴方向会产生上下翻转、1是全部翻转。0是没有翻转、中间值是随机翻转。

**EnabledUV channels**（激活UV通道）：选择要使用模型的UV通道，通常在发射模型时有意义。



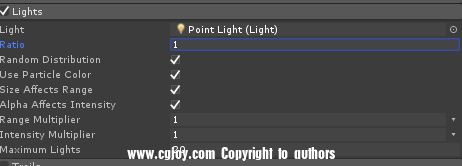
**Sprites**（雪碧）模式使用一个图片的sprite循环播放。这个sprite应该是一个被切分好的图片，才有意义。其次，将sprite指定到粒子系统后，会替代粒子系统使用的材质球，你看到的图是sprite的序列图。指定方式是一个图片被切成很多个小块，需要将每一个小块都放到一个新建接口中。



————————————————————————————————————————————————————————————————————————————

**灯光Light**

**---------------粒子产生灯**



**Light**（灯）：设置一盏灯，使粒子出生时同时有实时光照的效果，实质上就一个粒子一盏灯。

**Ratio**（比率）：产生的粒子会有多大比例有灯照，0是一个也没有，1是全部粒子都有。

**RandomDistribution**（随机分布）：当Ratio不是1时，设置产生灯照的粒子的分布情况，随机就是随机分布，勾掉则会根据粒子情况每隔多少秒出现一个灯照。

**UseParticle Color**（使用粒子颜色）：灯照颜色使用粒子的颜色。

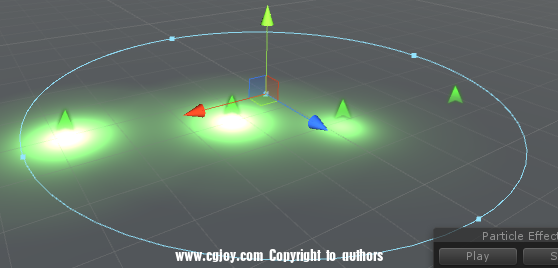
**SizeAffect Range**（尺寸作用与范围）：粒子的尺寸会影响灯照的范围。

**AlphaAffect Intensity**（透明度作用于亮度）：粒子的透明度会影响灯照的亮度，调整粒子的生命颜色时就可以实现控制灯照的亮度了。

**Rangemultiplier**（范围乘值）：增大灯照的范围。

**IntensityMultiplier（**亮度乘值）：增大灯照的亮度。

**Maximum Lights**（最大灯数）：因为是实时光照。灯的数量可能会根据粒子数量产生非常多，消耗会非常巨大。这里的值控制最多可以有多少个灯照被创建。



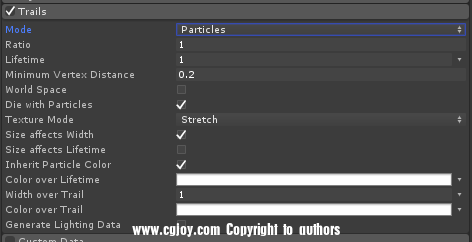
————————————————————————————————————————————————————————————————————————————

**拖尾 Trails**

**---------------流星划过**

理解下拖尾的含义，一个物体从A点到B点，跟随物体每隔多长时间（Minimum Vertex Distance）产生一次网格，是串联起来的。这个网格开始消失的时间是（lifetime）多久。其次还有颜色、宽度等信息。

**Mode**（模式）：提供了Particle和Ribbon模式。



**Particle**（粒子）：使用这个模式使粒子产生一个拖尾。

**Ratio**（比例）：粒子会产生拖尾的比例。1是全部粒子都会有拖尾。

**Lifetime**（生命）：拖尾的生命值。就是拖尾开始消失的时间。1是和粒子生命一样长。注意的是，默认Die with particle勾上时，粒子消失后拖尾也会消失，这个消失跟拖尾的Lifetime没有关系，只是被摧毁了。拖尾的生命时间就是开始消失的时间。

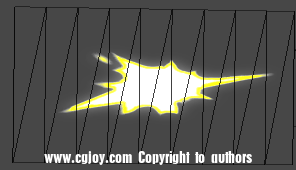
**Minimum Vertex Distance**(最小顶点距离)：mesh的两个顶点的距离。也就是细节程度，值越小，效果越好。实质上是产生一次mesh的时间。当然时间越小整个生命里产生的网格越多了。早先版本trail的产生是按每多少帧产生mesh的数量算的，所以会造成一旦卡了，trail就完全没法看了，这里是按时间算的，不用担心掉帧了。

**WorldSpace**（世界空间）：勾上时，这个拖尾会受到空间位移的影响。不勾是，只会相对于自身粒子系统进行拖尾。

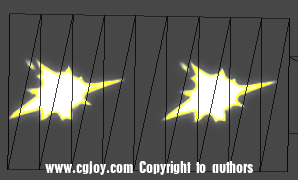
**Diewith particle（**跟随粒子消亡）：勾上时，粒子消失后，拖尾一起消失。

**TextureMode(**图片模式)：

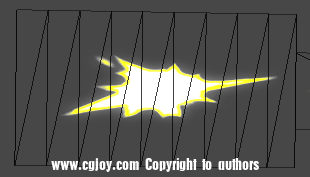
**Stretch**(拉伸):在整个mesh上拉伸整张图的U。



**Tile**（平铺）：U不会被拉伸，超出部分是重复平铺。

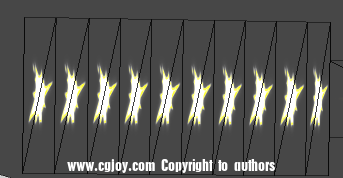


**Distribute Per Segment**（每段都分配）：和Stretch效果差不多，但计算的切入点不一样，



Stretch是先计算出trail的长度，在拉伸图片。这个是先计算又多少个网格，每个网格上应该分配多少图片的信息。理论上应该更精细。

        Repeat Per Segment（每段重复）：每一个网格上分配一个图片。



**Sizeaffects width**（尺寸影响宽度）：勾上时，默认宽度和粒子大小一致。而且宽度也随粒子大小变化了。

**Sizeaffects lifetime**（尺寸影响寿命）：勾上时，生命时间（也就是开始消失的时间）会受到尺寸的影响。当粒子变的巨大和很小时，智能的缩减拖尾时间很有必要。

**Inherit particle color**（继承粒子颜色）：勾上时，拖尾颜色会受到粒子颜色的影响。

**Colorover lifetime**（生命期内颜色）：拖尾生命时间内的颜色情况，是时间层面的，使用Gradient模式可以设置随时间变化颜色，开始是什么色，后面是什么色。跟粒子生命时间内颜色类似。

**Colorover trail**（拖尾颜色的情况）：拖尾自身的颜色变化情况。是空间层面。使用Gradient模式可以设前段是什么色，后段是什么色。注意和Color over lifetime区别。

**Widthover trail**（拖尾的宽度情况）：重要调节，来真正影响拖尾的宽度的值。曲线模式下可调节拖尾的前后宽度，横轴是整体长度，竖轴是宽度。

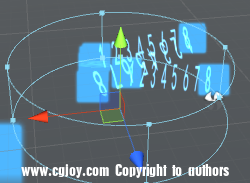
**GenerateLighting Data**（生成光照信息）：让trail可以收到光照。



**Ribbon**（丝带）：让产生的粒子链接起来。目前来说功能很匮乏，效果很尴尬。

**Ribboncount**（丝带数量）：就是使用几条丝带来将所有粒子链接起来。1就是只有一条。当丝带数大于场景中的粒子数时，就不在会产生粒子。多说下粒子的事情，每个粒子系统的粒子都是有序号的，假设一个序号为1的粒子2秒后消失，内存上说但他并不是真的消失了，它会从新回到发射器原点，模拟一个新的粒子出生，拥有新的生命和速度大小等。而这里的丝带链接目标是和序号有紧密关系的（坑深不挖）。所以当你看到条带数是粒子数的一半时，是两两链接的。

**SplitSub Emitter Ribbons**（分离子发射器的丝带）：就是如果当期粒子系统是一个子发射器。让每个粒子产生的新的发射器和其他产生的发射器不关联。



————————————————————————————————————————————————————————————————————————————

**自定义数据Custom Data**

**---------------请关闭**

要用到脚本才能起作用

————————————————————————————————————————————————————————————————————————————

**渲染器Renderer**

**---------------最后一步**

简单说下unity里面的坐标轴。

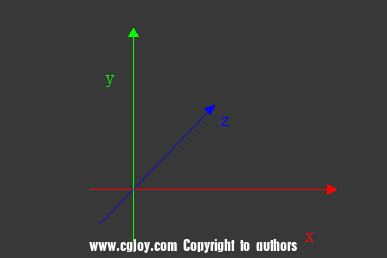
回忆下数学。

在纸上一个两个方向的坐标。一个横轴、一个竖轴。横的是x轴、竖的是y轴。

如果这个坐标是3d坐标。还有一个轴，它是从纸上指向你的一个轴。这就是z轴。

将这三个轴放到unity中就是unity的xyz。Unity就是继承了这样的概念从纸里指向你的轴是z轴。而y轴则朝上的。

Z轴通常也是一个游戏物体的正方向或前进方向。所以一个粒子系统发射的正方向也是z方向，这就是我们用创建一个Cone发射器时需要x轴倒转90°才能冲着天发射，是让Z轴朝上。

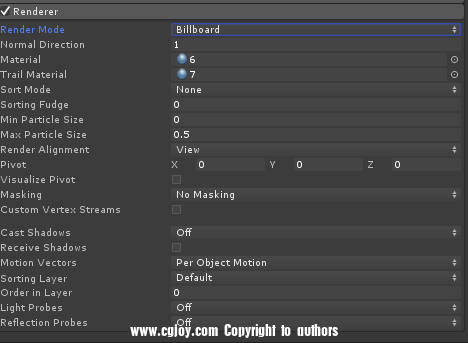
     

数学3d坐标和unity坐标。

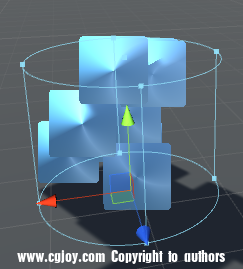
而类似于max中坐标系统是x和y是在地面上，z轴则是向上的。

**RenderMode**（渲染模式）：提供了几种粒子的显示模式。

**Billboard**（公告板）：永远朝向摄像机。粒子其实是一个plane，只是他的z轴（反方向）每时每刻都朝向摄像机而已。



**NormalDirection**（法线方向）：改变粒子片的法线方向。当粒子受到光照时（关键是shader要支持），将一个片模拟成一个受光照的体积。



**Material**（材质）：粒子的材质。图片通过shader生成一个材质。

**TrailMaterial**（拖尾材质）：拖尾材质。

**Sort Mode**（排序模式）：给粒子排列先后顺序，它是针对粒子系统自身发出的粒子进行排序。

而且粒子材质球是可排序的才有意义。就是材质要是透明材质才行。

         None（无）：不排序，混乱的。

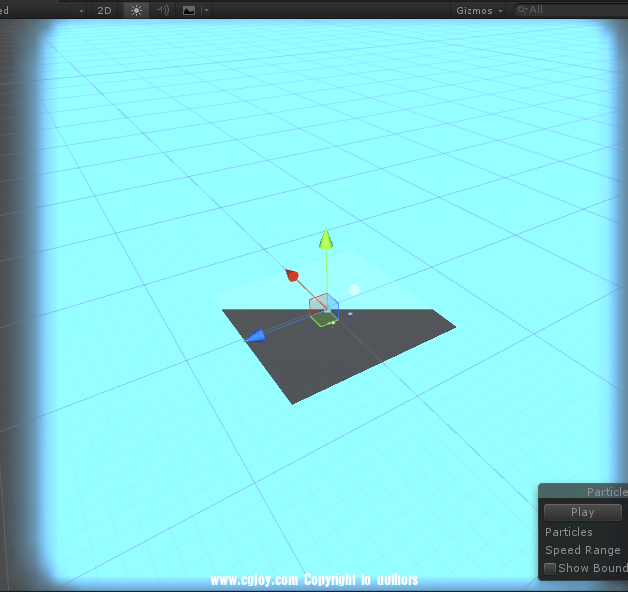
         ByDistance（依据距离）：根据粒子距离摄像机的远近排序。比较常用。

         Oldestin Front（最早在前）：谁先出生排列在前，后出生排列早后。平面上可能有用。

         Youngestin Front（最新的在前）：谁后出生排列在前、后出生排列在后。

**SortingFudge**（排序欺骗）：作者的意思大概是说这是一个欺骗性的排序，它让粒子假装往前或者往后移动的多少距离。负值是原理镜头、正值是靠近镜头，靠近镜头的肯定会在前面。当然材质球也必须是可排序的材质球才行。例如相加、相乘、透明混合等。

**Min particle Siz**e（最小粒子尺寸）：简单说小于这个值的粒子都会被放大到这个值这么大。这个值是屏幕大小，值1就是一个平面那么大。做屏幕特效时有用，不必去把粒子设的很大，用这个就能准确设定大小。



**Max particle Size**（最大粒子尺寸）：这个值同样只屏幕大小。当粒子显示大于这个值时，会被缩小。避免细小的粒子冲向摄像机时变的巨大。

**Render Alignment**（显示队列）：指定粒子朝向方向，也就是z轴朝向。

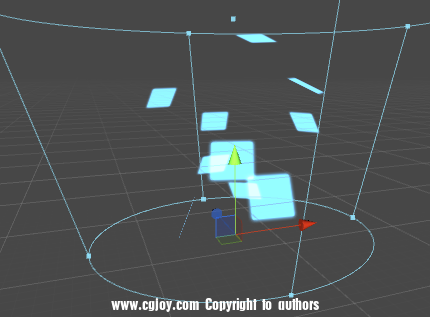
         View（视野）：粒子片z轴朝向你观察的方向。

         World（时间）：粒子片z轴朝向世界的z轴。

         Local（自身）：粒子片z轴朝向粒子系统自身的z轴。这时就可以旋转它了，就是真实粒子片的状态。

         Facing（面朝）：粒子片z轴朝向摄像机位置。和view效果很像，但是却不并不一样。这里朝向时，会去掉透视关系，只是拿z这个面朝向你。Mesh状态下能看的更清楚。

         Velocity（速率）：粒子片z轴朝向自身运动的方向。



**Pivot**（中心点）：设置了三个方向可以调节中心点。在Stretched Billboard会有大用。

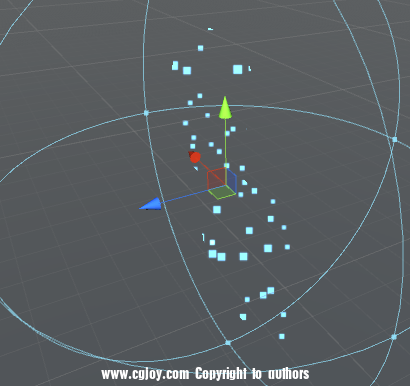
**Visualizepivot**（可视化中心点）：使粒子的中心点显示出来。粒子真正的运行位置，调节pivot时有用。

**Masking（**遮挡）：是关于sprite遮挡的功能。简而言之，sprite有个组件可以把图片做成一个遮照放到场景中。处理超出这个遮照范围外就会消失，或者只在范围外显示。通常用于平面类型特效上面。

         No Masking（无遮挡）：不会有遮挡。

         VisibleInside Mask（遮照内部可见）：只在Sprite Mask内部显示粒子。超出不显示。

         VisibleOutside Mask（遮照外部可见）：只在Sprite Mask外部显示粒子。内部不显示。



**CustomVertex Streams**（自定义顶点流）：程序相关，脚本调用。主要处理顶点色。

**CastShadows**（投射阴影）：可以投射阴影。但材质必须是不透明类型的。

         Off（关）On（开）Two Sided（双面）shadowsOnly（只有阴影。-）

**ReceiveShadow**（接收阴影）：公共版模式下意义不大因为是一个片接收到阴影并不舒服，建议mesh模型下使用。

**MotionVectors**（移动向量）：和摄像机的运动向量有关。不太明白意义是什么。可以确定的是和特效制作没关系了。

**Sorting Layer**（排序层）：设置一个层处理前后关系，越早创建的层越靠后，越新创建的层越靠前。

**Orderin Layer**（层中排序）：在这里可以在同一个Sorting Layer层里设置前后关系。值越大越靠前。

要注意这两个功能和Sorting Fudge是有区别的。Sorting Fudge是假设他在场景中的位置，负值是离镜头较远、正值是离镜头较近。近的肯定是会覆盖掉远的，从而调节先后关系。而这里是真正更改渲染顺序的。在unity中整个排序是这样的，先计算shader上面的顺序，然后是Sorting layer的前后，然后是Order in layer大小，最后才是Sorting Fudge的影响。

**LightProbes**（光探测）：

**ReflectionProbes**（反射探测）：

这两项和其拓展出的功能都和灯光探测有关，坑深不挖。特效制作基本不用。

**Stretched Billboard**（拉伸公告板）：公共板会朝一个方向延伸。