完美技术应用实验室

《幻塔》工作室丁同学：UE4开放世界ARPG《幻塔》技术分享

* 2022-02-28

在2020虚幻引擎技术开放日中，来自《幻塔》游戏开发工程师丁同学，为大家带来了《幻塔》游戏的开发技术分享。

**以下是演讲实录：**

**丁同学：**大家好，我现在幻塔工作室任客户端引擎工程师。我今天结合我们游戏《幻塔》来给大家做一些技术分享。我们游戏是基于虚幻引擎4的MMORPG类型的游戏。

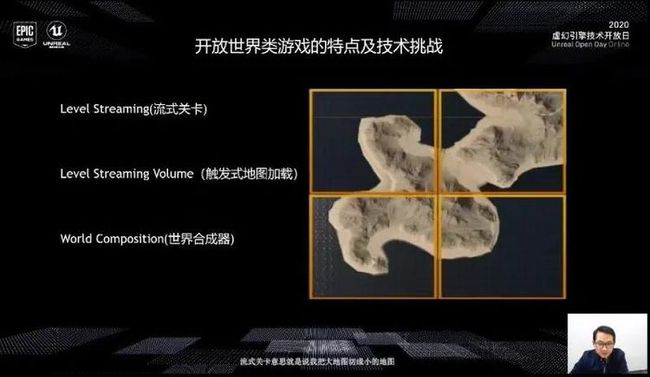
那么我们可以看出，我们游戏主要有两个特点：一个是开放世界，第二个就是高自由度。



那对于在移动平台做这种开放世界类的游戏，我们会面临着很多的技术问题。



第一个技术问题就是大世界，就我们的地图目前有2.5K×2.5K，当然了对于现在的游戏来说2.5Kx2.5K并不算大，但是当你的地图达到一定程度的时候，我们面临的技术问题其实是一样的。



当然Unreal本身对大地图提供了比较好的支持，比如说Unreal提供了流式关卡，流式关卡的意思就是我把大地图切成小的地图，然后按需加载。

**在加载策略方面Unreal也提供了两种：**

第1种：触发式的地图加载，触发式的地图加载适合于，比如说你这个世界并不太连续，两个世界之间有明显的交界，我这个时间我可以在交接处放一个触发盒子，比如说我走到这个地方，这个触发盒子的时候去加载另一块地图。

第2种：世界合成器，就是World Composition，它里边可以分Layer，就是在这Layer里边我可以放不同的Level，这样的话我可以根据Layer定义不同的加载策略，这样的话会更灵活。

这是Unreal做的比较好的地方，当然Unreal也有做的不太好的地方，比如说Unreal的地形系统，那我们来看一下Unreal的地形系统。



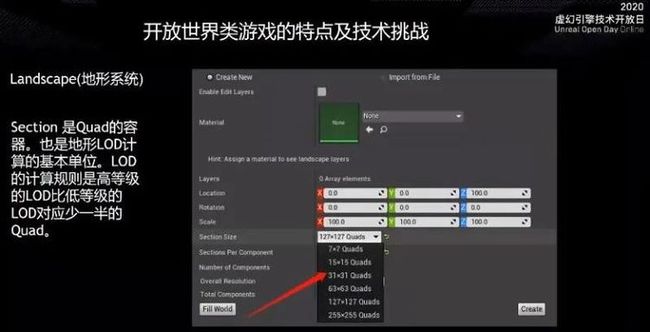
在Unreal中 它的地形系统叫做Landscape，在右边这张图就是它的地形系统的创建界面，比如说我们可以就是调一些参数就可以创建出来，就在点击这里的Create 就可以创建出来一个地形，或者说在这个标签里从外部导入一个高度图来生成地形。

那么跟地形相关的第一个概念就是Quad，那Quad就是Unreal就是地形系统的基本单位，它的大小呢跟Scale相关，那么这个红框圈出的地方就是Scale。



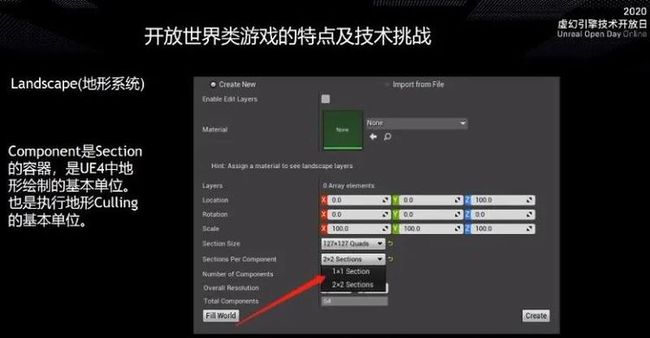
我们看它的单位这里是显示的是100，因为Unreal里边它的单位是厘米，所以说这100是一米，如果你不调这个参数的话，它的一个Quad就是相当于是一米见方的一个块。

第二个跟地形相关的概念就是Section，这Section它是Quad的容器，也是地形LOD的基本单位，比如说地形LOD它计算的规则是就是高阶的，低阶LOD的扩展会减一半。



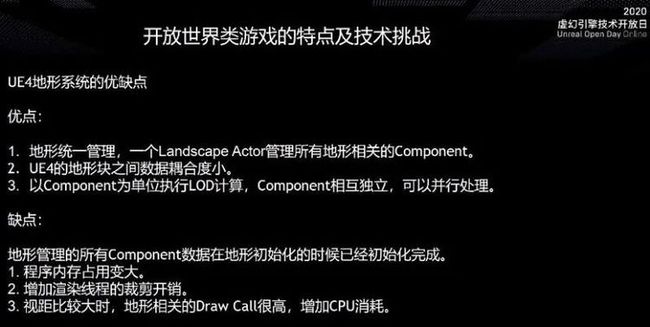
比如说第0级LOD是31×31，那么第1级的LOD就是15×15个Quad，那目前我们可以看出来，每个Section就可以有7×7、5×15、31×31，一直到255×255，一共这6种可选。

那么第三个跟Landscape相关的概念是Component，那Component是Setting的容器，它是UE4中地形绘制和地形裁剪的基本单位，目前它也是只有两种配置：一个是1×1；一个是2×2。



就相当于是说一个Component只有1个Section，或者有4个Section。

那么UE4这是个地形系统的优点，它是地形是统一管理，相当于是我一个Landscape的Actor就管理了所有的Landscape相关的Component。那么它地形、地块之间的数据耦合度比较小，它是以Component为单位就是执行LOD计算，因为它Component是比较独立的，所以说它的计算可以并行起来，多线程来处理。



地形系统的缺点是什么？因为它一个Component一个Actor管理了所有的Component，那这个Actor初始化的时候，所有的Component都已经初始化了，这样会导致三个问题：

第1个问题：内存占用会比较大，地形相关的内存占用会比较大。

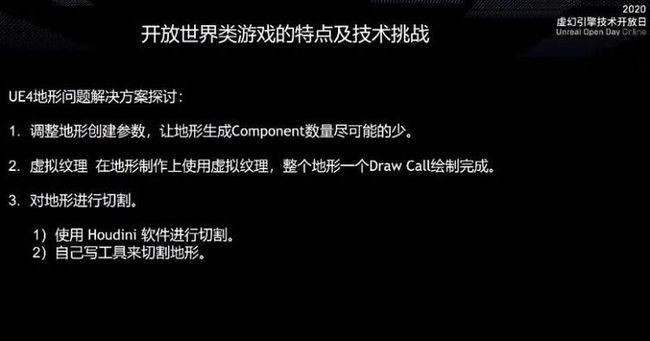
第2个问题：它会增加渲染线程的裁剪开销，因为它这个Component在注册的时候，它这个渲染代理就是SceneProxy都已经创建好了，就相当于是渲染线程在做剔除的时候，有可能你这个地形就是离玩家非常远，它根本是不需要考虑，但是说它现在也需要考虑要不要把它裁掉。

第3个问题：当你视距比较大的时候，地形相关的Draw Call会非常高，增加了CPU的消耗。

这里给大家举个例子，比如说我这一个Component有4个section，那每个Section里面是63×63的Quad组成的，当这个Component离玩家很远的时候，它会退化成2×2的Quad，如果你这个Scale没有改，你相当于是你还是1米×1米，相当于是这一个Component只覆盖了2米×2米的大小，这样会导致你这个视距里面有非常多的Component。

在Unreal中一个Component会包含比如说刚才我们讲的它是2个4×4，2×2 就是4个Section，如果这每个Section它的差别不太大，它会把这个4个Section合并成1个Draw Call，就相当于是1个Component。要么是有1个Draw Call，要么是有4个Draw Call，这样会导致你视距很大的时候，你这个地形相关的Draw Call会非常高。

那么针对这个问题，我们目前有三种解决方案：

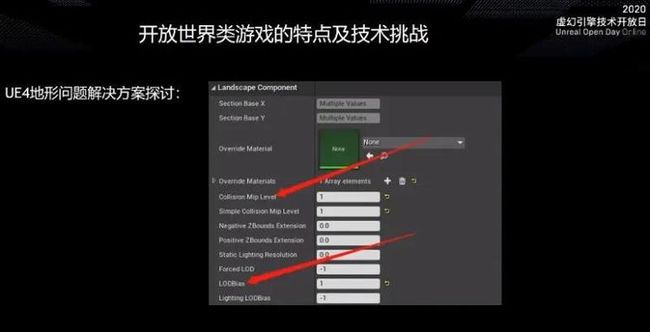


第1种：我们可以调整地形的创建参数，让地形生成的Component数尽可能的少。比如说我可以把Quad调大，然后可以把Section里边的Quad数量调多，让它生成的Component的比较少。当然，这样会带来另一个问题，就是你这个地形Cutting（切割）的力度会变粗。

第2种：可以用虚拟纹理，就是说地形制作上使用虚拟纹理，虚幻引擎官方会在4.26版本的引擎上支持这个功能。相当于是一个Draw Call，就可以把所有的地形画绘制完成。但是虚拟纹理它本身也是有消耗的，所以说它具体在Mobile（移动）平台表现怎么样，还需要进一步的测试。

第3种：对地形进行切割，切割本身也分两种。第一种，用Houdini软件来进行切割。第二种，自己写了个工具来进行切割。

刚才我们所讲到的是优化地形的Draw Call，对于地形的面数，我们目前的做法是用了LODBias。



例如在绘制地形的时候，这个地形是第零级，但我们偏移一级来画，即用第一级来画；那第一级地形，就用第二级来画，以此类推。这样的话很好地控制地形的面数。

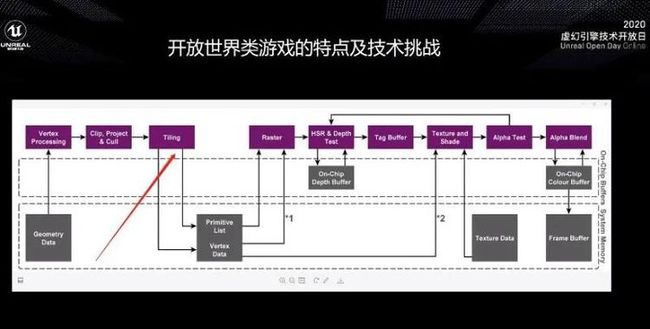
我们看一下这张图：



在这张图中我们可以看到我这个视距是非常远，我们可以看到地图中的大部分的物件。我们在优化之前，我们那个地形的Draw Call有220多个，优化之后我们地形的Draw Call现在只有37个。

对于面数我们使用地形偏移，我们目前地形偏移了一级，偏移的话在这个视角下优化了大概15万面，这个优化其实是相当可观的。

那么在进入第二个主题之前，我们来看一下就是移动平台的渲染管线。



移动平台由于硬件会有两个限制，一个是顶点数受限，一个是Draw Call受限。

我们看一下这是移动平台的渲染管线，其实它这里边会有一个Tiling的过程，Tiling过程就相当于它VS先走一遍，然后决定这个三角形在哪个Tile里边，个时间它会把它临时数据放到这个主存中，如果你的顶点太多，它这边就会形成一个热点。

第二个Draw Call受限，Draw Call提交的时候，CPU要准备大量的数据，而且提交Draw Call的时候会把这个Shader生成，编译成对应平台的编码。

所以说，我们在移动平台主要是这两个限制，我们在做开放世界游戏时，面对超大视距在角色这个位置的时候，我们能够看到的地图非常远。



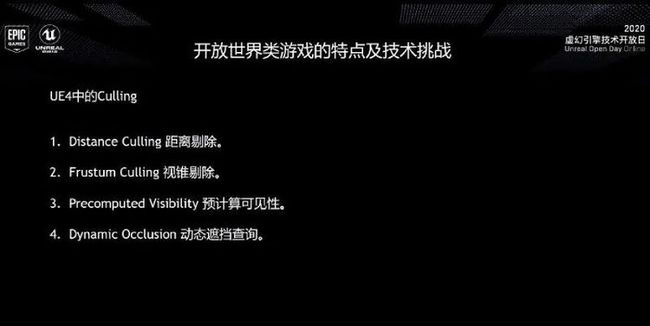
这样会导致两个问题：

第1个：要画的东西非常多

第2个：顶点数也会非常多

而由于移动平台这两块都是受限的，相当于虽然你看到了这么多东西，但是硬件其实是绘制不完的。

这就需要裁剪策略，提前把对视觉不重要的裁掉，Unreal也提供了好几种裁剪方案：

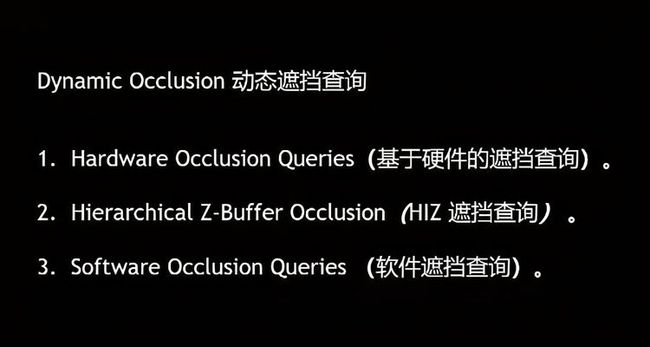


第一种：距离剔除，即Distance Culling。根据你摄像机的远近，来把它剔除掉。在每个X轴上设置一个剔除距离，对玩家交互不强的且视觉不重要的，就剔除掉。

第二种：视锥剔除，即Frustum Culling。把不在摄像机视野范围之内的剔除掉。

第三种：预计算可见性，即Precomputed Visibility。先预先离线计算好，走到这个位置的时候，查询这个计算好的表，哪些东西是可见的，哪些是不可见的。预计算可见性只适合静态场景，主要的应用场景是室内。

第四种：动态遮挡查询，即Dynamic Occlusion。针对于动态遮挡查询Unreal提供了三种方案。



第1种：基于硬件的遮挡查询，即Hardware Occlusion Queries。被查询物会根据它的包围盒提交一个Draw Call，来硬件做Depth Buffer，查一下它有没有被遮挡住。这种方案它的缺点就是Draw Call会比较高。

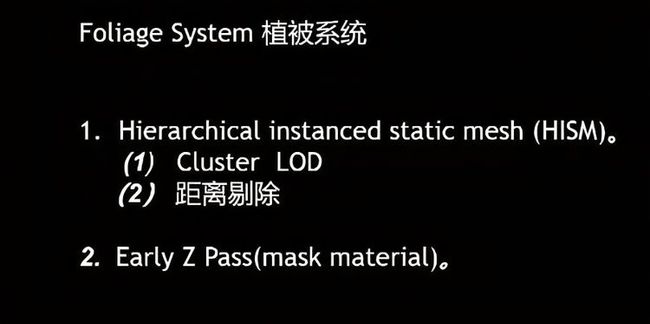
第2种：HIZ的遮挡查询，即Hierarchical Z-Buffer Occlusion。这种方法不需要提交Draw Call，它首先把Depth Buffer做mip，之后根据这个被查询物的这个包围盒再映射到屏幕空间，计算物体最长的边，根据最长的边来算一个它在哪个mip里做，然后在那个mip里做遮挡查询。

这两种都是基于硬件的方法，如果这个裁剪是CPU在做的话，它需要回读到CPU，回读的话会延迟一帧或者几帧，这样就导致了一个问题，因为延迟了所以裁剪是不准的。Unreal还提供了一种基于软件的遮挡查询。

第3种：软件遮挡查询，即Software Occlusion Queries。被查询物先用CPU进行软光栅化，作为软光栅化来生成这个Depth Buffer，然后再来做遮挡查询，这样的好处就是它不需要回读了。它全是用CPU来计算的，它具体性能表现怎么样，还需要一定的测试。

那么对于开放世界的游戏，另一个不得不提的主题就是植被系统了。如果你这个开放世界很大，但是植被很少的话，那你这个世界的细节不丰富。以前的植被系统都是实例化绘制的，就是Instance Static Mesh即ISM。

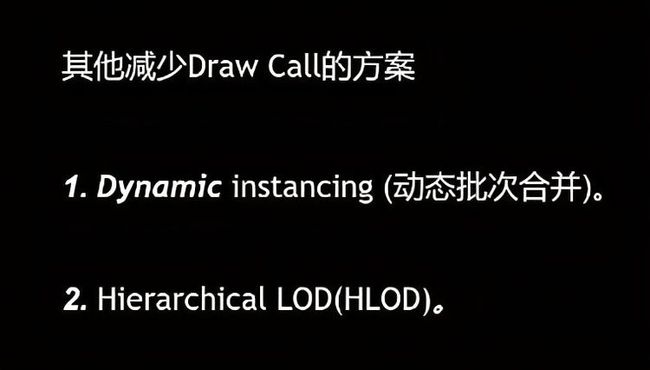
ISM的裁剪是它的主要问题，比如1000个物体，人物视野只看到了1个，但是这1000个物体都是要提交的。



所以说Unreal针对这个问题做了个Hierarchical的Instanced Static Mesh即HISM。它的实现是就是根据树形结构来做了Cluster来分组，可以把裁剪粒度更精细，还可以做Cluster LOD，同时我还实现了距离剔除。

这些都是针对CPU的粗粒度剔除，对于GPU优化方面，Unreal提供了Early Z Pass。植被系统中树、草都是Mask材质，Mask Material需要做Alpha test，所以Unreal目前的做法是把Mask材质先在Early Z Pass中深度画一遍，利用这段时间已经完成了Alpha test，在Early Z Pass把Mask材质的Pixel过滤掉，这样就优化了Mask材质。

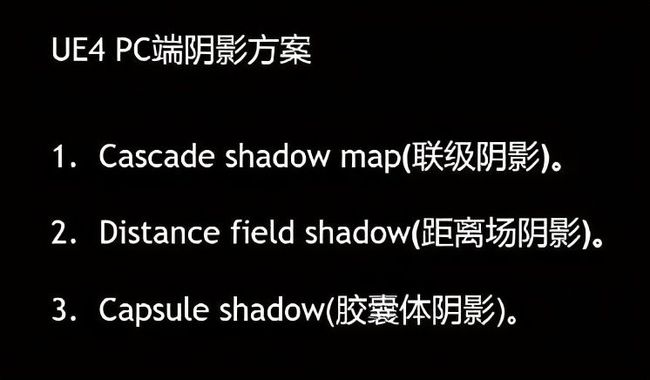
其他减少Draw Call的方法：



第1个：动态批次合并，即Dynamic Instancing，这是4.2版本加入的功能，对于我们这个游戏来说，开启了态批次合并之后，优化了150个Draw Call左右。

第2个：Hierarchical LOD，即HLOD。例如一个大的关卡，可以生成一个代理，一个Draw Call把它画出来。HLOD也有他的问题，它会导致你的包体跟内存变大。

对于开放世界游戏，还有一个问题就是阴影，我们看到的东西很多，如果阴影绘制的距离很近，这样就导致了场景很平，没有层次感。Unreal在PC平台上提供了很多应用方案。



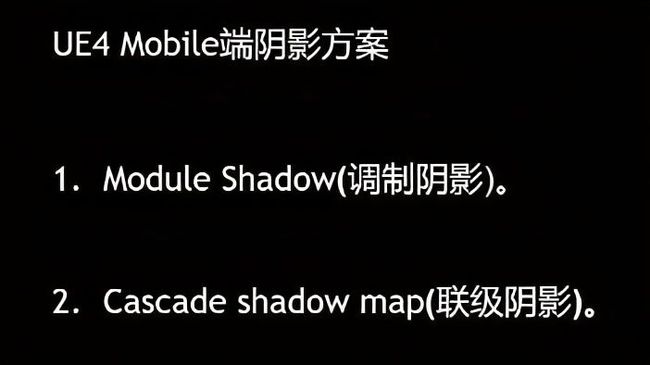
第1种：联级阴影，即Cascade shadow map。

第2种：距离场阴影，即Distance field shadow。

Unreal官方建议，在近处用联级阴影，在远处用距离场阴影。

第3种：胶囊体阴影，即Capsule shadow。利用物理资产Physical Asset来生成阴影，这种实现有一个好处就是它支持间接光照。例如一个场景里全是烘焙的这种Lighting，联级阴影是生成不了这种阴影的，但是胶囊体阴影可以对这种间接光也产生阴影，但是你要做的非常细的话，你需要很多胶囊体，这是它的一个缺点。

目前在移动平台上，UE4的阴影方案主要是两种：



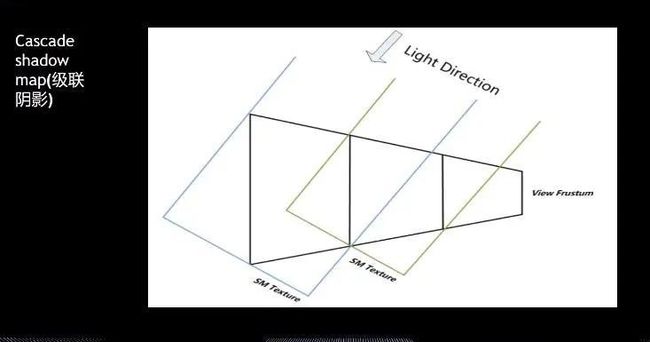
第1种：调制阴影，即Module Shadow。

第2种：级联阴影，即Cascade shadow map

先来看调制阴影，先看这张图：

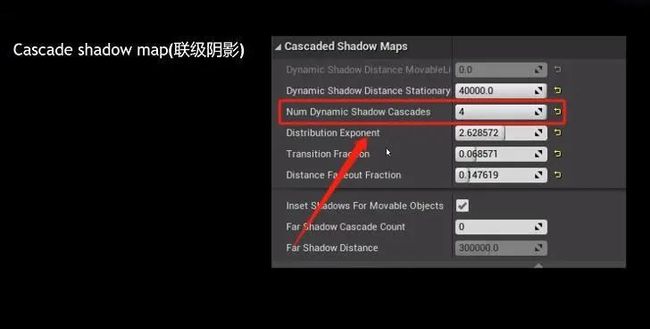


调制阴影它其实它主要是跟场景做了简单的乘积，它跟其他阴影混合的时候，是明显的叠加效果，所以说这种基本满足不了现在游戏的品质要求。所以UE4在移动平台上只能选择级联阴影。级联阴影最初是为了解决阴影贴图的精度不够而导致的透视堆叠。



根据这个图说明，把视锥分为三级，再根据太阳方向，把小视锥包围住，生成一个正交投影的矩阵，然后每一个级生成一个ShadowMap，这样也更符合人类的这个视觉特点。

在UE4里级联阴影的设置界面



设置界面包括了设置级数的、每一级结束了怎么切换等等。

**对于级联阴影，它主要的问题是什么？**

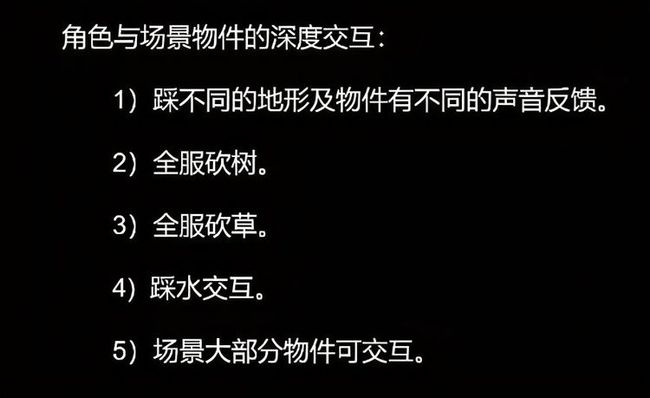
主要的问题是因为，级联阴影每一级都是一个视锥，它需要单独做跟场景做裁剪，就是CPU内的裁剪消耗比较高；另一个是它级数很多，需要绘制的阴影的物件会很多，导致你的场景的Draw Call会比较高。

在移动平台上，由于带宽受限，阴影贴图做成1024这种分辨率是一个比较合适的选择，这样就需要更多的级数去提高它的精度，但是由于之前讲的几点限制，所以在移动平台做不了这么多级数。

针对级联阴影的问题，我们现在的做法是用4级，前2级是全动态，后面2级是只画静态物体，前2级做分帧刷新，即不是每帧都提交，后面2级做低频刷新，即隔几帧才刷新一次。

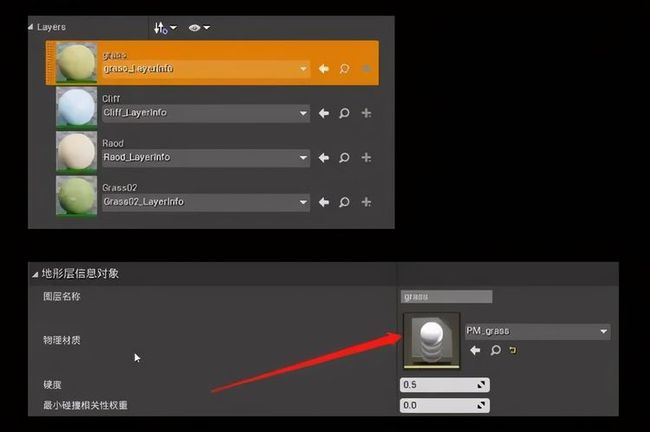
通过这样修改CSM，可以做到400M距离的阴影效果。

我们游戏的第二个特点就是高自由度，在以前的游戏中，人物与场景交互是很少的，当然随着Unreal对物理引擎就是良好的封装，为我们创建这种更高自由度的交互就是提供了可能。



我们游戏中做了踩不同的地形和不同的物件，会有不同的声音反馈，这样更真实。还有不同的砍树，踩水等交互，另外场景中大部分物体都是可以被推走或者破坏掉的。

那么这种根据不同的交互产生不同声音是怎么做的呢？



我们地形分了4层，在草这一层，通过匹配对应的物理材质的声音来实现。



例如砍树，我们是全物理模拟的，而并不是简单的播一个动画。



例如砍草，整个世界大部分草是可以砍掉的，而且别的玩家是可以看到的，而不是简简单单的一个客户端表现。

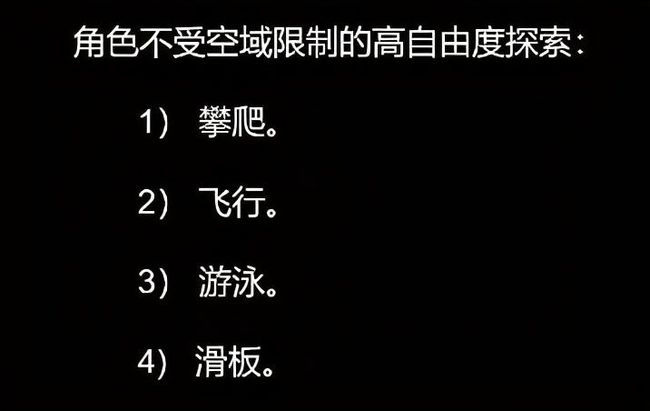


这是踩水，角色走在水里边会产生这种涟漪效果。



游戏中大部分物件是可以被破坏掉，或者挪走等等。

为了制作出高自由度的游戏，也因为UE4的Movement的组件本身提供了非常多的Movement mode，所以在飞行、游泳等之上，设计扩展出了攀爬滑板等模式。



攀爬



游泳

滑板

我们游戏立项的时候，拿到的是虚幻4引擎4.20版本，目前我们升到了4.25，那随着版本的迭代UE4，对移动平台的支持越来越好，比如说它添加了动态批次合并、骨骼实例化等等一些新功能。我们相信随着UE4对移动平台支持越来越好，以后会涌现出越来越多的基于UE4的精品游戏，谢谢大家。