## **《崩坏3》画面效果为何惊艳？看米哈游怎么做卡通渲染的你就明白了**

**来自 游戏葡萄 2017-05-16**

[深度](http://youxiputao.com/article/index/id/13)

#### **[ 游戏葡萄原创专稿，未经允许请勿转载 ]**



《崩坏3》上架以来，我们看到了卡通渲染技术在游戏领域的高品质画面表现力，米哈游对这种技术恰到好处的运用也让《崩坏3》受到了广大玩家的喜爱，以及业内诸多从业人士的认可。

事实上，米哈游在制作这款游戏的过程中，也经历了**[反复的尝试和探索](http://mp.weixin.qq.com/s?__biz=MjM5OTc2ODUxMw==&mid=2649704357&idx=1&sn=4e9aee0e86883b7cef1c29b71338e0d0&scene=21" \l "wechat_redirect)**。同时，为了让卡通渲染技术有更好的表现力，他们也利用了Unity 5引擎，探索了次世代卡通渲染技术用法。

2016年11月，米哈游在B站投稿了“琪亚娜·极乐净土”的视频，短时间内成为B站最热门的视频之一，目前已经积累了近230w次的播放量。

视频来源为米哈游创意社区官方网站

在这则视频当中，米哈游尝试了次世代画面品质的角色渲染；全局光照、体积光、面积光源等光效处理；特殊材质的实现以及高级反射材质的效果处理；以及卡通渲染后处理等相关技术。

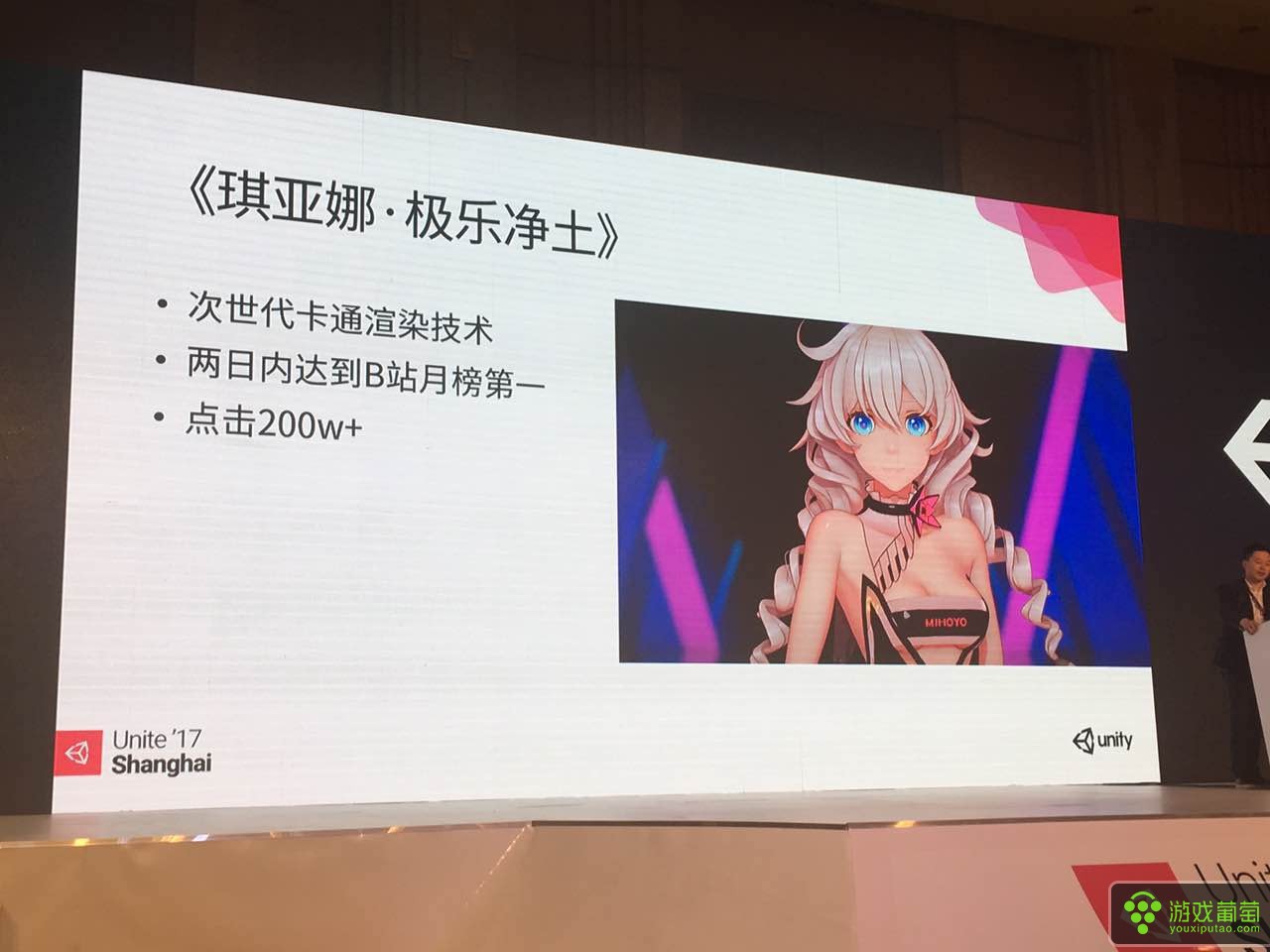


在日前刚刚结束的Unite 2017开发者大会当中，米哈游技术总监贺甲对这些技术进行了逐一解读，并通过自身的实际尝试为例，分享了他们的开发技术经验。

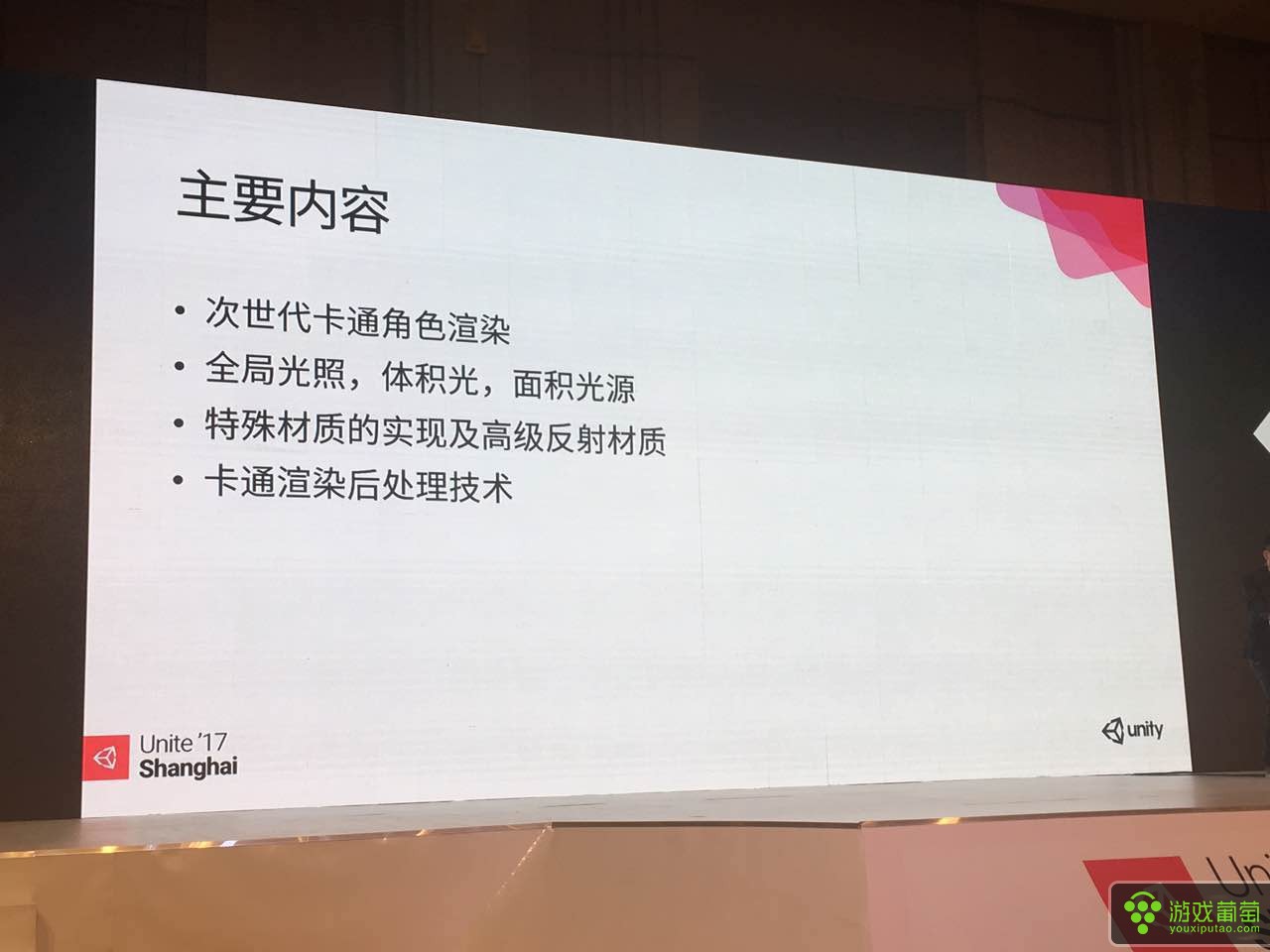
以下内容经游戏葡萄整理发布。

贺甲：大家好，首先自我介绍一下，我叫贺甲，在米哈游担任技术总监和美术指导工作，目前主要关注的方向是非写实渲染以及可交互物理方面的研究。很高兴在这里给大家带来一场有关于次世代卡通渲染的演讲。

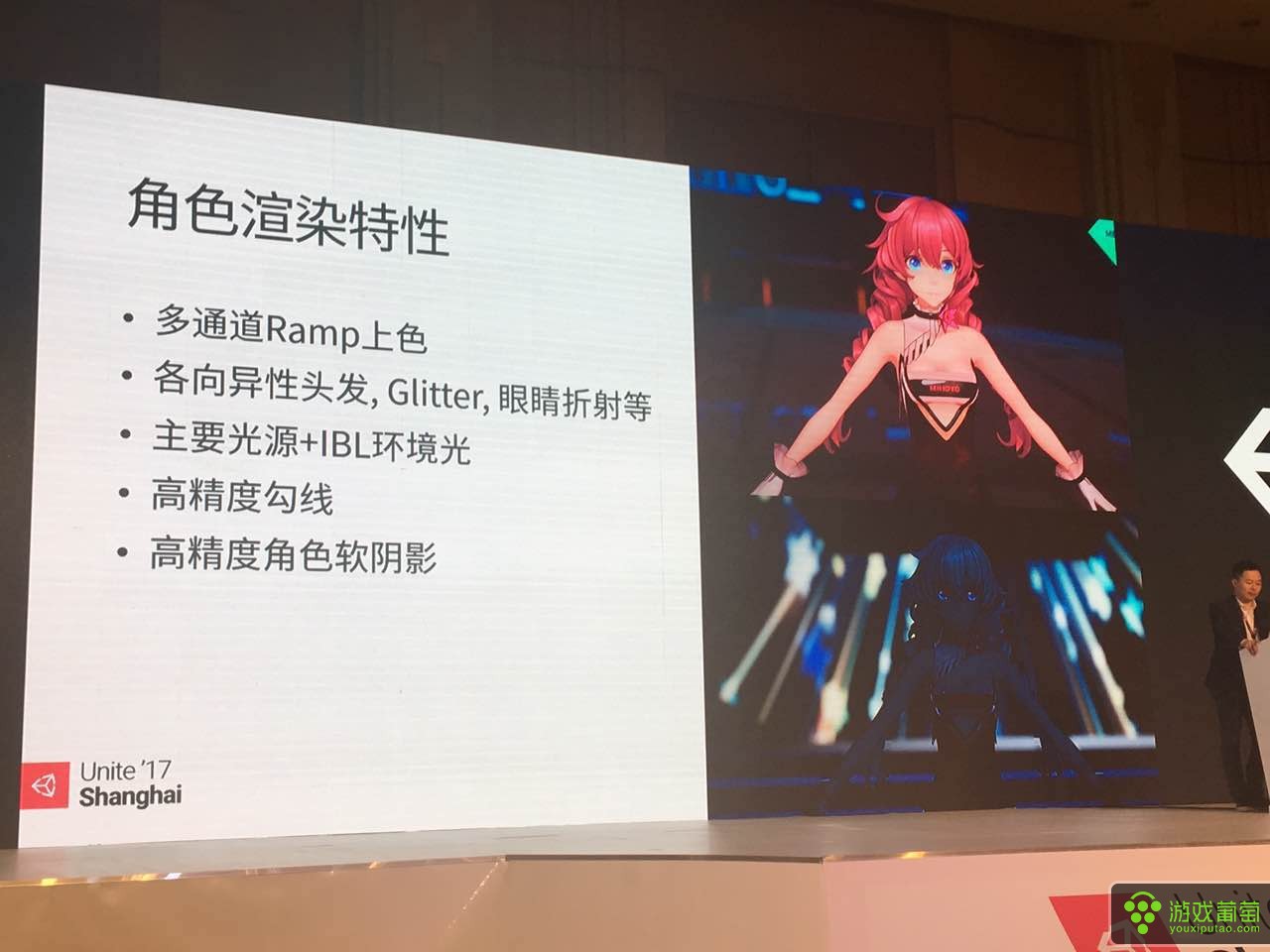
我们在考虑如何运用次世代渲染技术，对当前卡通渲染技术时代进行升级。如何将新的技术和卡通渲染有机结合，也是一个比较大的挑战，我们想还原不仅仅是之前传统上色的质感，还想进一步把插画级细致方面的表现，以60帧每秒帧率表现出来，这种工作量仅靠手绘无法完成，我们选择了极乐净土作为MV来展现新的卡通渲染效果。整个制作过程大概有3到4个月，发布之后只用了2天直接登上B站榜的第一，目前累计有230万点击量。



下面我们来介绍一下这次演讲的主要内容，首先是次世代卡通渲染角色的主要实现方法以及特殊材质的处理，然后是场景方面的渲染，包括全局光照、体积光、面积光源的实现，之后我们介绍一下卡通渲染所涉及的后处理效果。



首先我们来看一下角色卡通渲染的主要特性，第一是我们自己实现的卡通多材质多通道Ramp上色技术，以及一些特殊材质的实现，比如各向异性的头发，Glitter和眼睛折射的效果。然后是光照的构成，我们选择了主要光源和IBL环境光，之后还有高精度的勾线，以及高精度的角色软阴影。



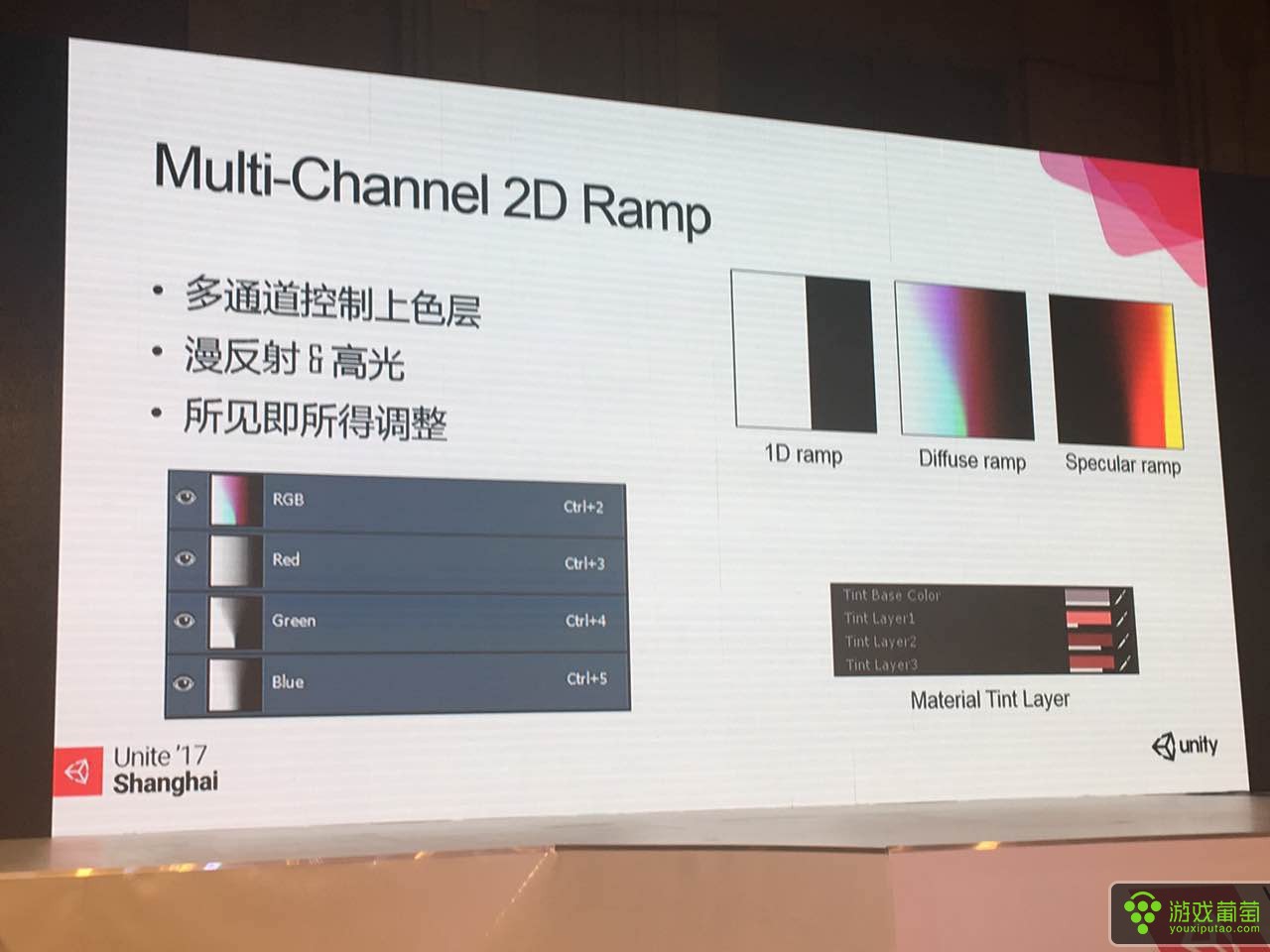
下面几张图展示了上述效果，随着视角的变化，头发高光以及眼睛的折射效果有不同变化。这段是固定相机视角旋转光源之后的动态表现，大家可以看到光影和高光。







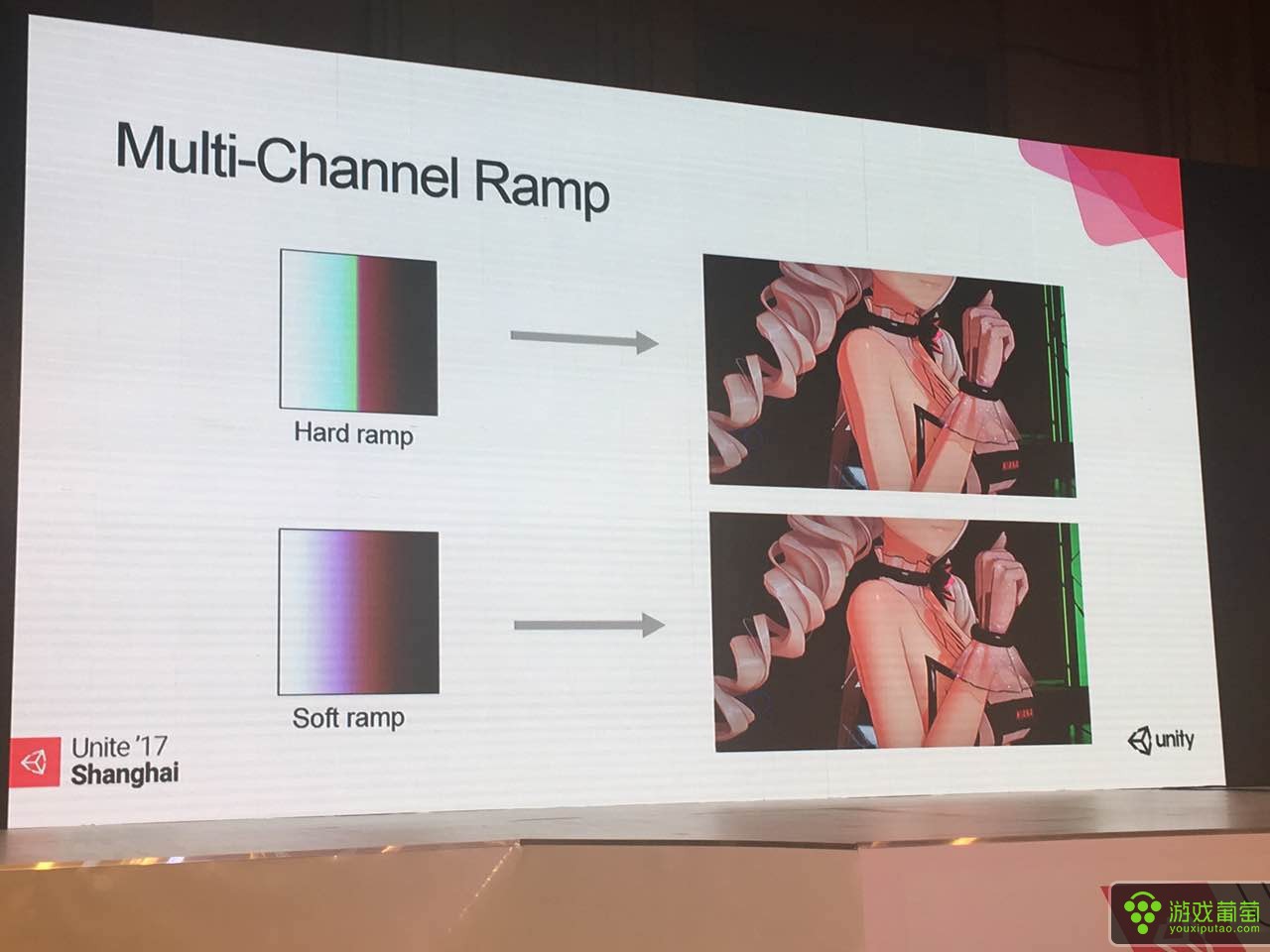
首先我们来介绍一下多通道控制上色方法，我们使用了多通道2D Ramp，使用多通道控制上色层，对于卡通风格画面来说，如果上色只是纯粹的明暗变化，阴影处就会显得比较缺乏表现度，通过调整垂直采样坐标，我们可以实现动态软硬风格的切换。从另一个角度来说，这种方法还间接表现了皮肤的次表面散射效果。



这四幅图展示了多通道上色的叠加效果，大家可以看到通过一层层上色叠加，皮肤细节的阴影变化也变得更加丰富。



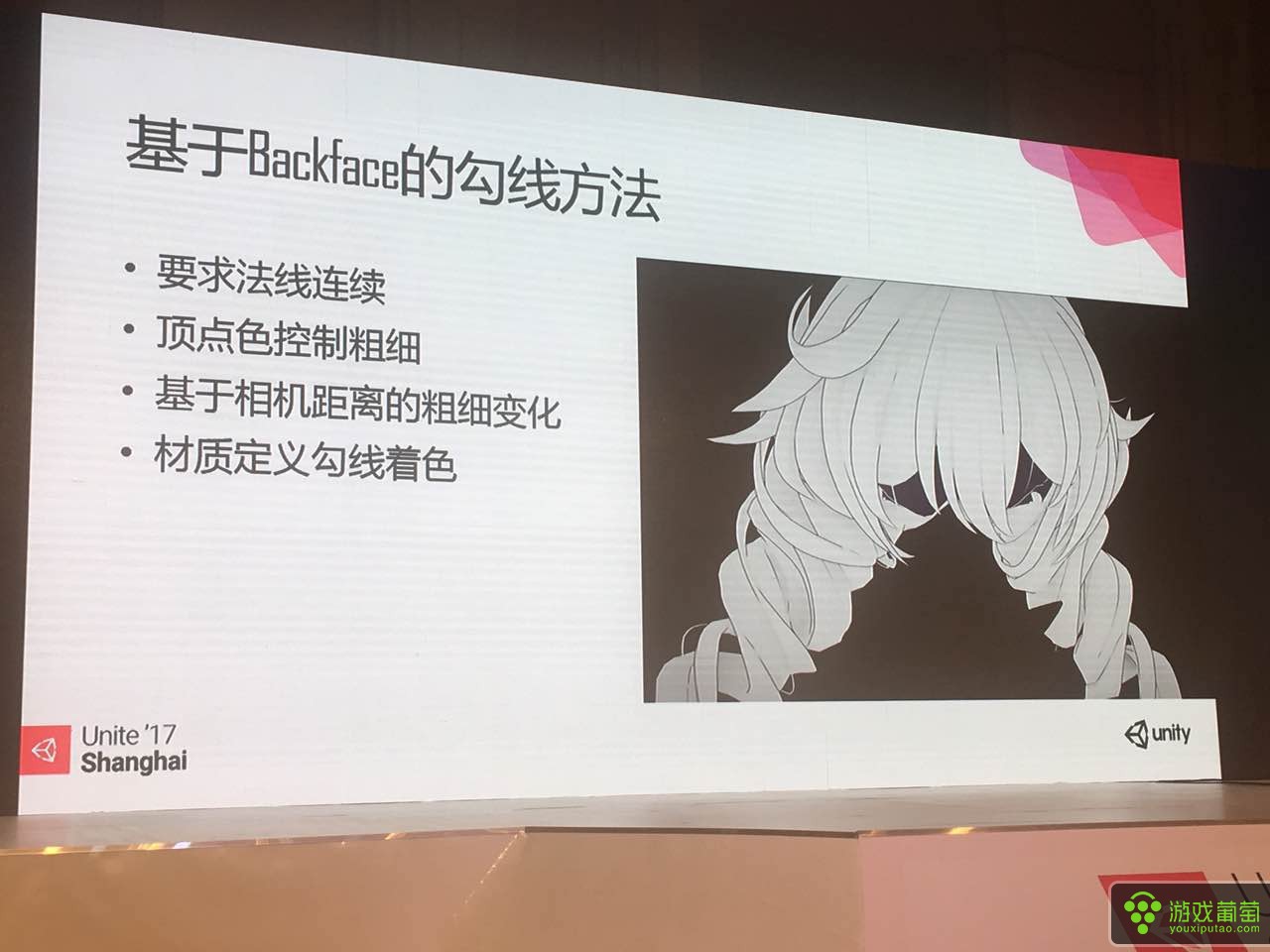
这上下幅图展示了采样位置不同的渲染效果。由于我们使用了2D的纹理，他们之前的变化也可以动态调整，这就方便了我们在想要的结果上做一些精细的调整。



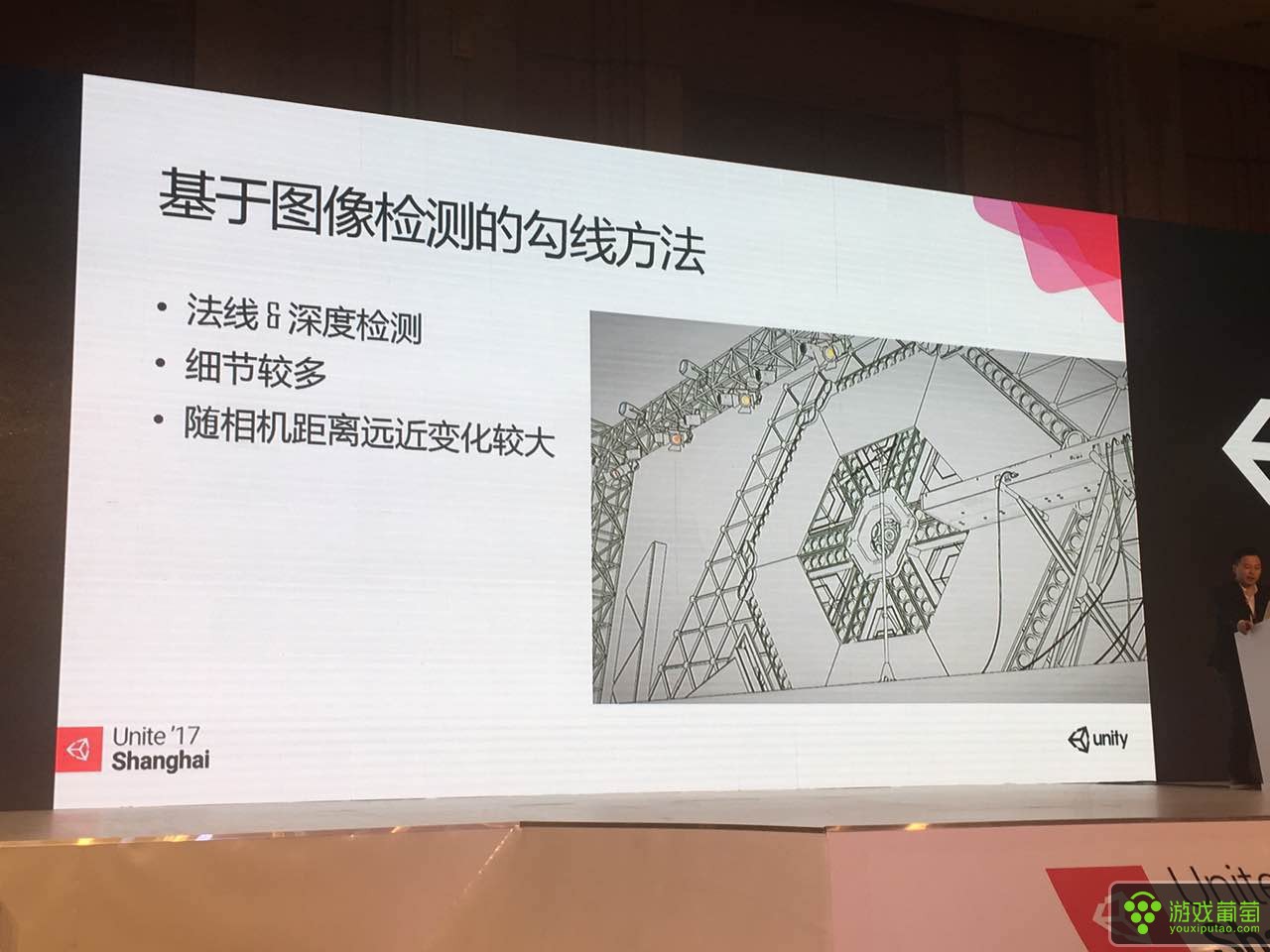
卡通风格对于面部表现一般不会有太多阴影层次的变化，如果我们直接套用这个方法在面部的话，效果就会像右图看起来不自然，为了改善这种情况，我们使用了顶点色通道mask控制明暗强度，通过压低阴影表现达到想要的卡通效果。



下面我介绍一下卡通渲染的常见勾线方法，第一种是Backface方法，是比较高效的方法，而且通过顶点色可以实现各种风格化的控制。比如通常头发丝对勾线逐渐变细，美术就可再在mask来刷线表达想要效果，暗处的色彩饱和度也会提高，这点我们在实现中也有所体现。



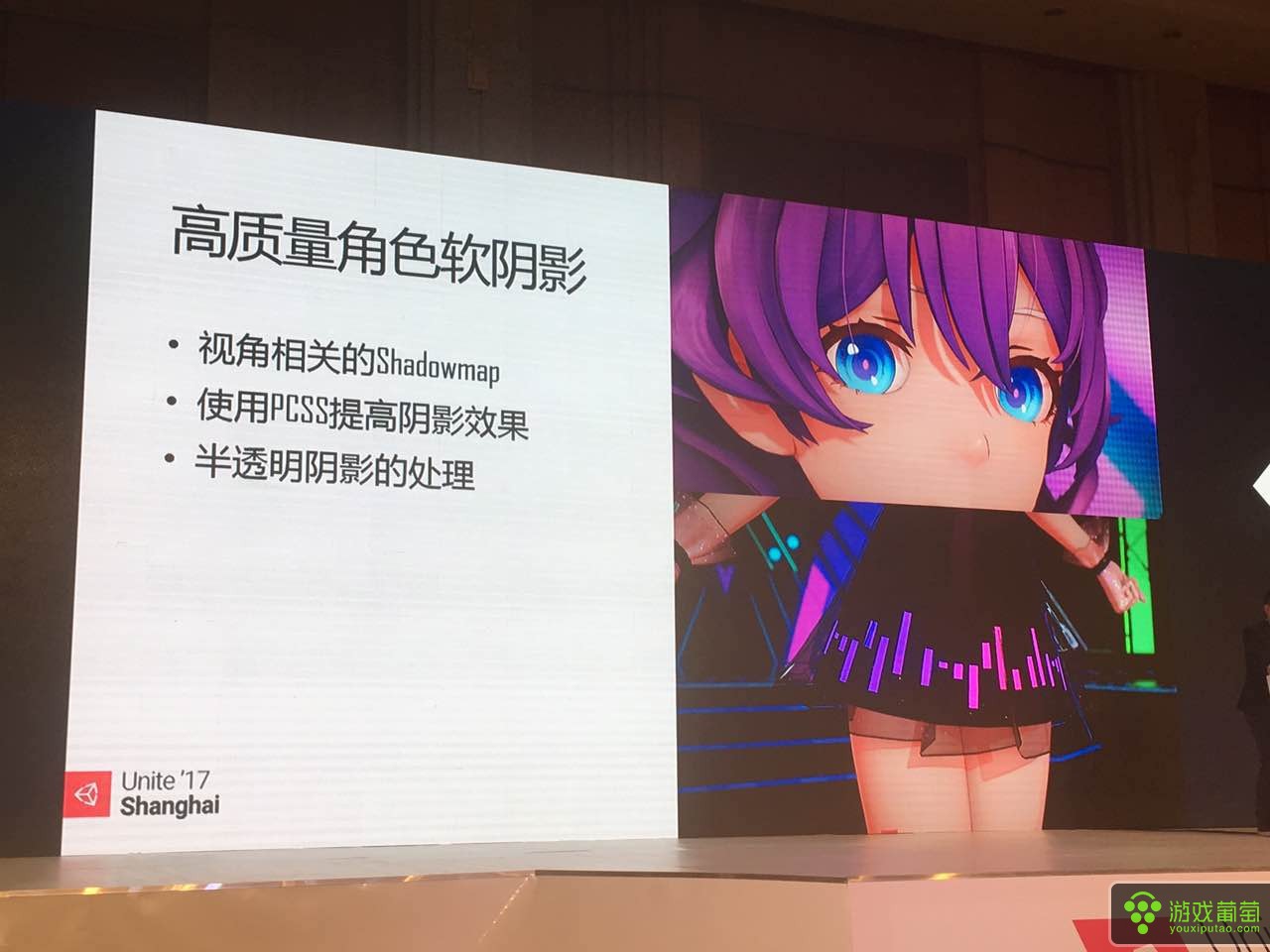
第二种方法是基于图像检测的勾线方法，通过图像检测提取出不良象素标出边界，这样可以标出物体表面的棱角。基于图像处理方法，还可以表现出较多的有勾线细节，随着相机的拉近拉远，勾线就不太能体现具体的细节。 基于图像检测的方法，比较适合用于场景的勾线。这种方法还有一个好处，不会随着细节的复杂度增加，而降低运行效率。



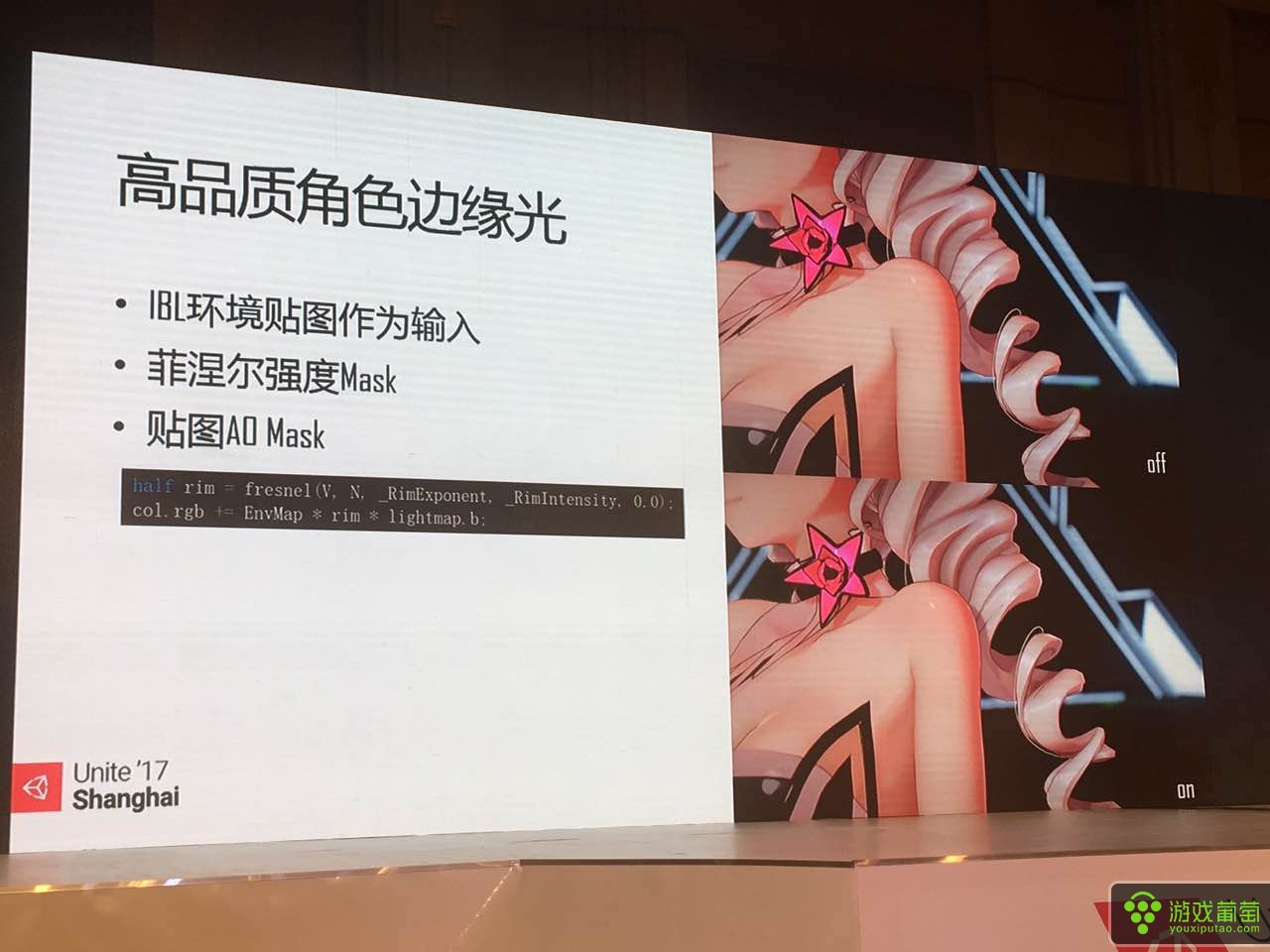
第三种方法是基于笔刷的勾线方法。首先是轮廓线的提取，然后是连接轮廓线，根据模型的关系，将相邻的轮廓边连接成尽可能长的轮廓线，但有可能会出现轮廓线交叉的问题，这点也是需要处理的。然后是轮廓线的分段，在步骤二的基础上，将轮廓线在曲律上分开。然后是笔触，这种方法可以达到更为风格化的表现，笔触会更加明显。



接下来我们讲一下高品质角色阴影的实现，视频中有不少特写镜头，我们想要局部特写也能表现非常细腻的高品质角色软阴影效果。所以我们就单独为角色渲染了一张分辨率为2k的Shadowmap。为了保证相机拉很近的时候仍有较高的阴影品质，需要视角相关的Shadowmap。第二是使用PCSS提高阴影效果，第三使用半透明阴影的处理。大家可以看到，裙子上的阴影是有半透明的效果。



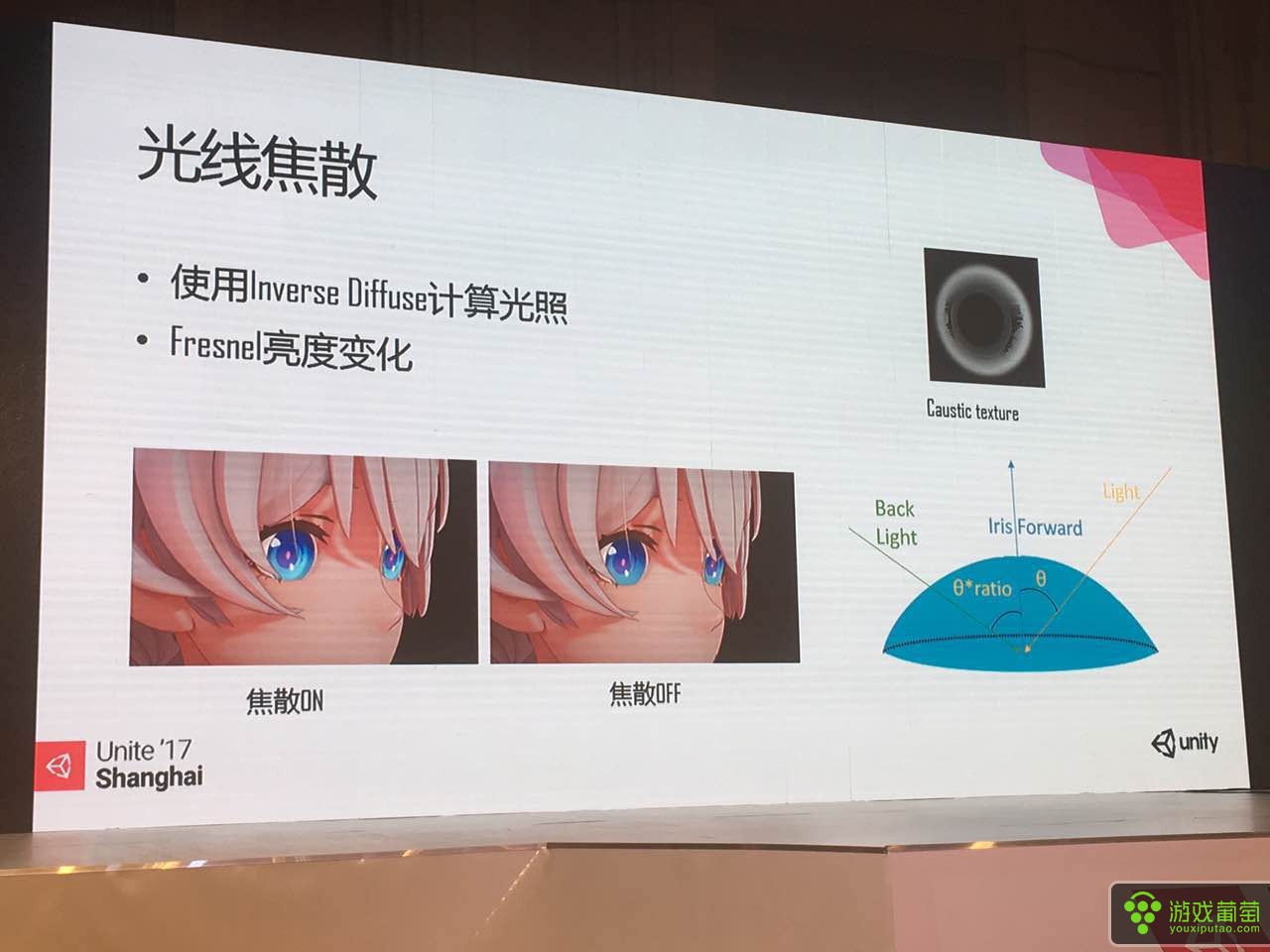
我们再来看一下角色边缘光的处理，由于场景中大部分为面积光源，我们想要做出根据光照环境变化的边缘光一我们就使用IBL环境贴图作为主要输入，使用菲涅尔强度Mask控制光照在边缘出现，最终用贴图AO Mask模拟自遮挡效果。可以看到左边这个对比图。



对于眼睛的处理也使用了基于物理光的折射效果，我们使用了真实的折射算法，眼球本身还是按照球面来做，根据视线角度算出折线系数。这两个对比图显示了有无折射的效果，基于物理的方法，在VR模式下会看起来更有质感，看起来比较类似于像玻璃珠的感觉。



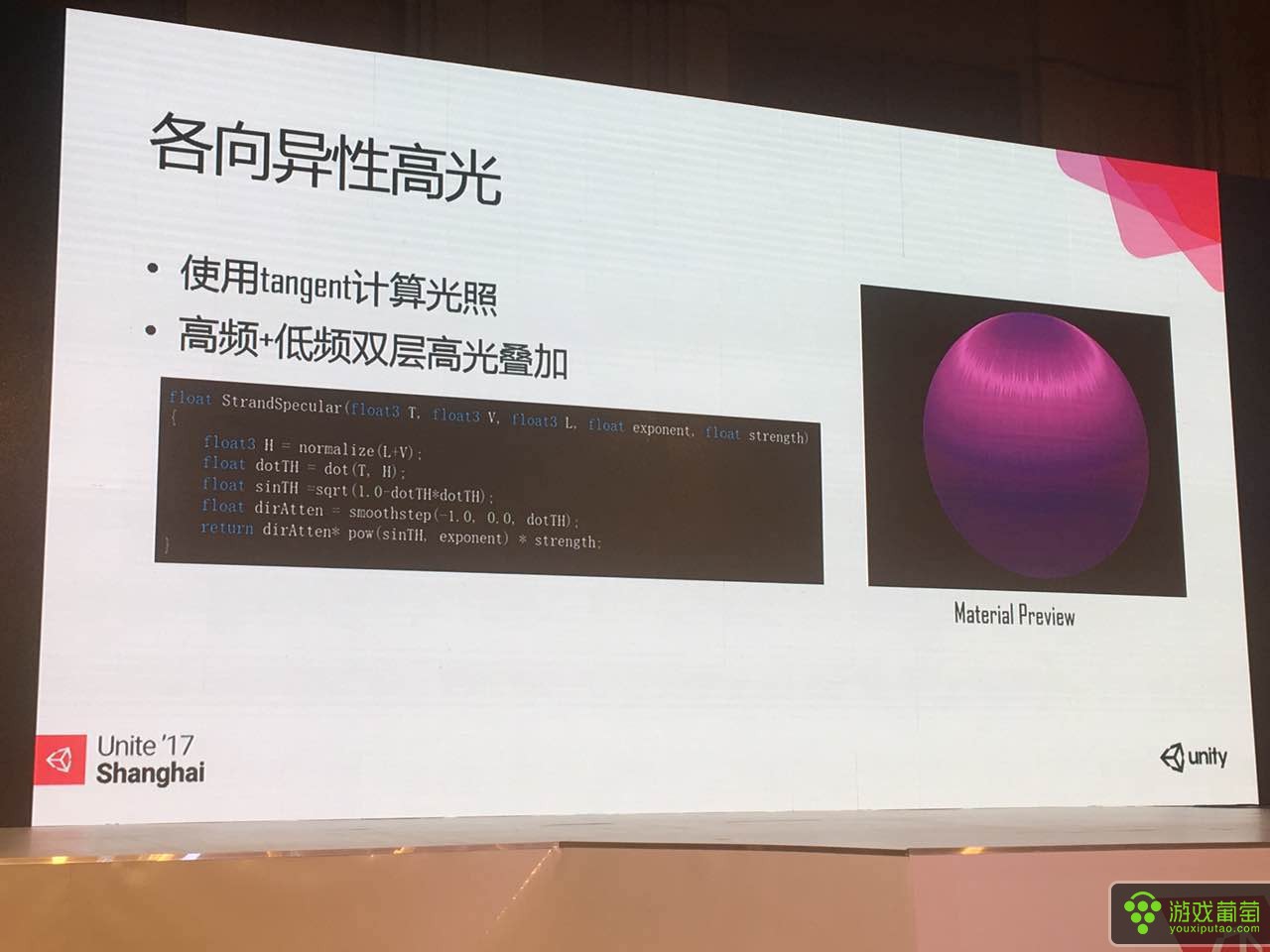
除了眼睛折射效果之外，我们还加入了光线折射后的焦散效果，使得眼睛的材质得到进一步增强。考虑到卡通渲染的特殊情况，我们希望的效果是将散出现在入射光另一侧。实现方法通过入射光和眼球前的夹角，算出入射光的强度，辅助菲涅尔公式，最后得到最终效果。通过对比图我们可以看到，没有焦散的眼睛就显得缺乏质感。



我们看一下头发的渲染，头发渲染是卡通渲染表现的重点之一，这也是和照片写实风格主要区别点之一。常见头发阴影层次一般实现都是贴图上，这样在动态光影变化就不会发生变化。我们的目标之一是将阴影和动态光照随着光源的变化而变化，没有任何内容是直接在贴图上画死的，都是由计算而得到，我们使用了各项异性高光做卡通渲染的头发。通过不同的调整颜色，可以想得到立即的想要的图案。



相比普通的光照我们使用tangent计算光照，可以通过高频和低频双层高光叠加，达到细致的效果。



这张图展示了两层异性高光的效果，左上没有高光效果，左上一层只有低频高光效果，左下一层只有高频光照的效果，右下是最终的渲染结果。我们对两层高光设有不同亮度和摄像，这样使得高光层次更为丰富。



除此之外，我们还使用Jitter map抖动贴图，增强质感，来达到模拟发丝细节的效果，我们还可以做到调节发丝的高光粗细。



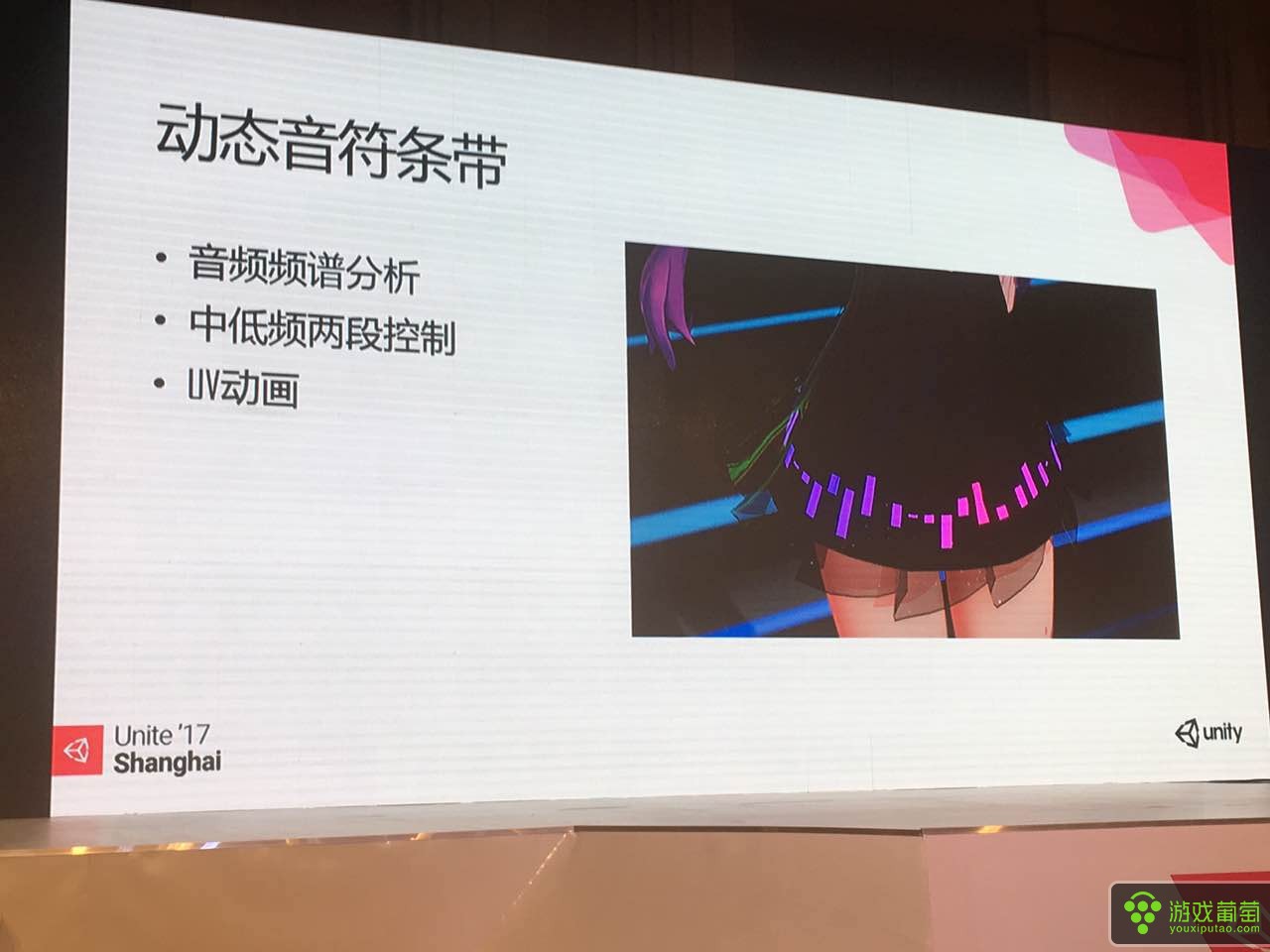
除了头发使用特殊光照材质处理之外，我们的角色材质中，还有一些类似于玻璃水晶等半透明材质，为了使更真实细腻表现这些材质效果，如果我们直接使用半透明混合就不能满足需求，这样的话就需要我们来实现有真实效果的折射。这两个实现，都使用Unity的CommandBuffer实现，RGB通道可以设置为不同折射系数，分别采样三次来模拟色散效果，如果用模糊效果，就生成四张尺寸依次减半的模糊贴图，根据相机距离和材质固有的模糊度，选择对应的来模糊效果。



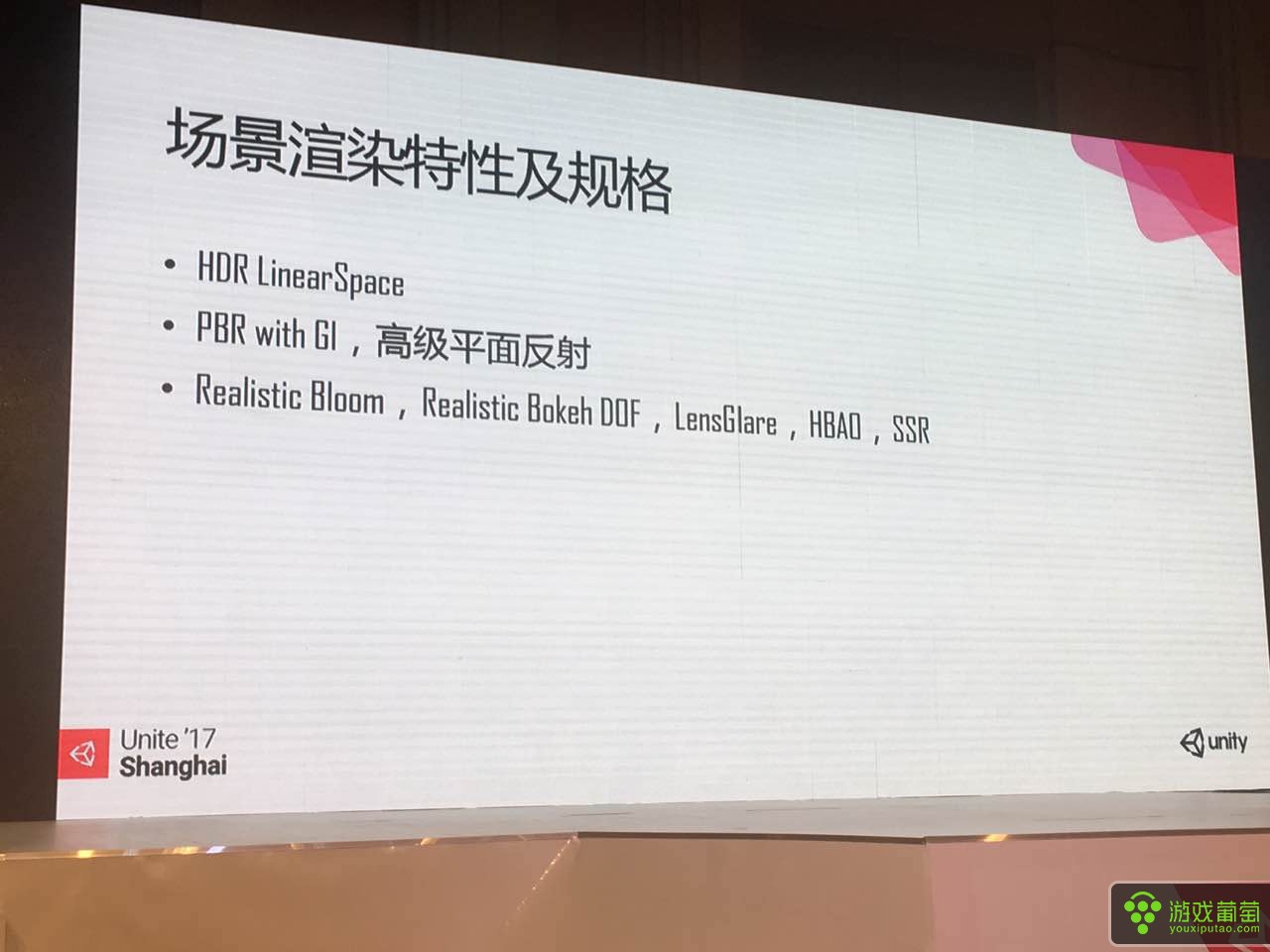
我们还可以看到，还有类似于闪光片的闪烁效果，这个是我们作为第二层材质来处理。首先用specular map表示区域，然后用normal map增加反射不规则性，反射的光源主要是主光源和环境IBL贴图，此外我们使用光图查找表做五彩的闪烁效果。主要方法也是根据入射角度决定色彩偏移来查找光谱图，使用不同光谱图也会有不同的色彩表现。



角色衣服上的动态条带，随着音乐的节奏变化而变化，我们把跳动的音符调在分为两组，分别对应音频文件中的中频和低频信号，然后对音频做频谱分析，每帧得到的中频和低频强度根据强度去伸缩对应层的音符UV，得到动态的效果。



接下来我们介绍一下场景方面的渲染技术规格，整个场景在HDR线性空间下渲染，我们使用修改过的PBR，开启了烘焙enlighten GI。根据需求，实现了平面反射材质，此外还应用了各种后处理效果，比如Bloom、屏幕空间反射等。



这是我们几张舞台的截图特写，大家可以看到，综合应用了之前提到的技术之后，整体感觉可以比较接近离线渲染的效果。舞台中有大量动态视频作为光照来源，并且把强度系数设大于1，作为面积光源。如果想要光照环境实时更新，就需要在脚本里面每帧调用Cache。场景中也设置一些动态更新的Lightprobes，作为角色的光照来源。



我们再来看一下体积光的实现，各种灯光效果和激光效果，也是舞台渲染不可缺少的一部分。使用体积光可以比较好满足上述需求，我们使用常见类型作为体积光的光源定义来源。





我们先来看一下聚光灯的效果，由于想表现空气中散射导致光柱逐渐模糊扩散的效果，光锥体积用过程化参数来控制。为了进一步模拟烟雾效果，我们还使用了3DNoisemap技术，此外配合Cookie map可以定义投射型状，引入了高频的变化，需要对应增加采样数。我们使用两种抖动方式，可以在较低采样实现比较好的平滑体积效果。



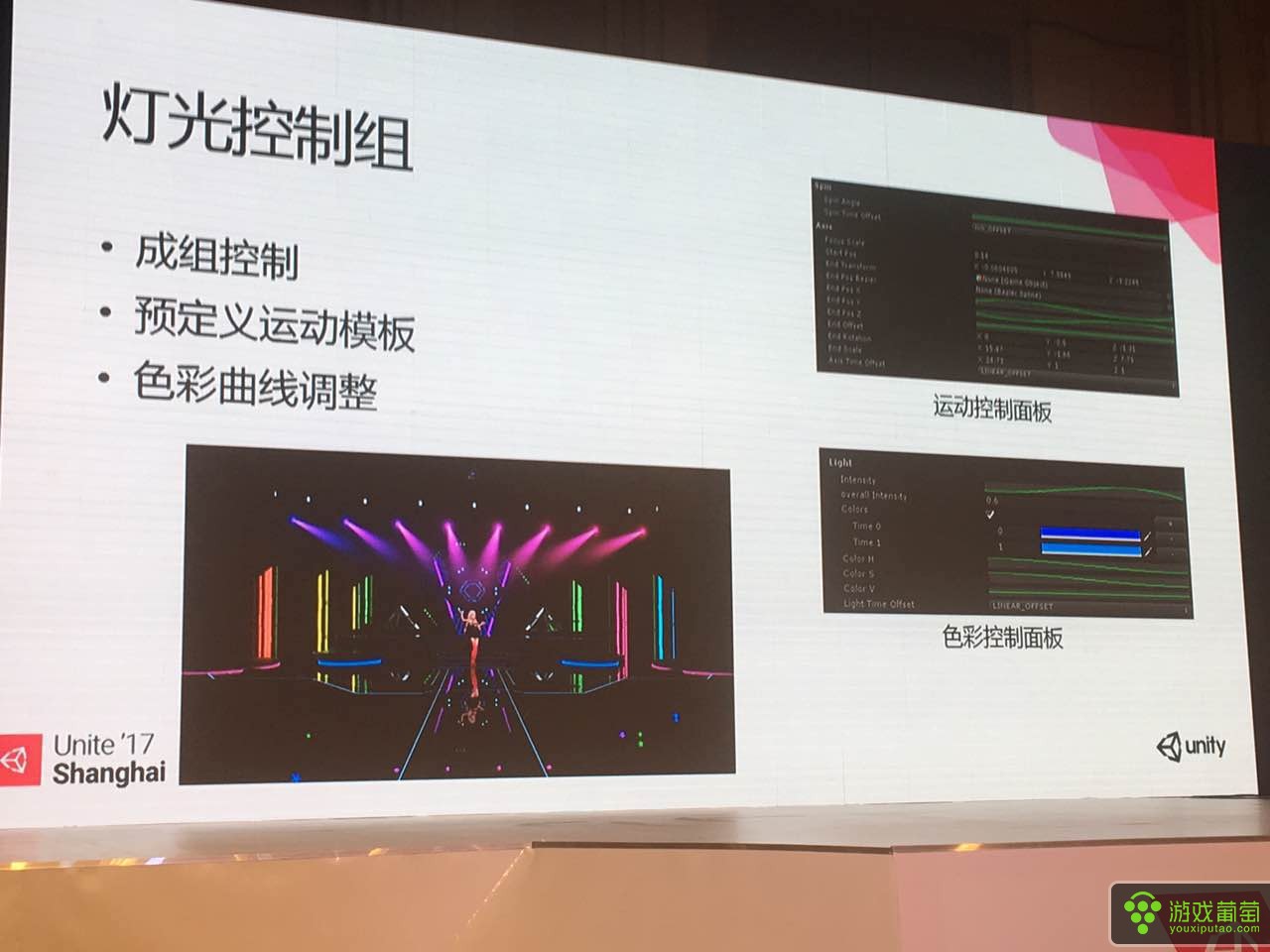
我们还用体积光来表现激光效果，只需使用一个定义激光扫描路径就可以实现，贴图越细，高频程度也越多。



我们的舞台中还有一种常见的大范围镜头平面激光效果，直接使用体积光我们即使把采样数设的很高，也会有一些瑕疵。并且性能下降非常严重，考虑到实现效果和精度，我们又做了新类型prefab来完成这个功能。将单独的激光数作为可以绕中心轴旋转面向摄像机，这样就避免了本身朝向摄像机会变细的情况。此外，激光扫描平面的烟雾效果，也用一个2Dnoisemap可以实现。



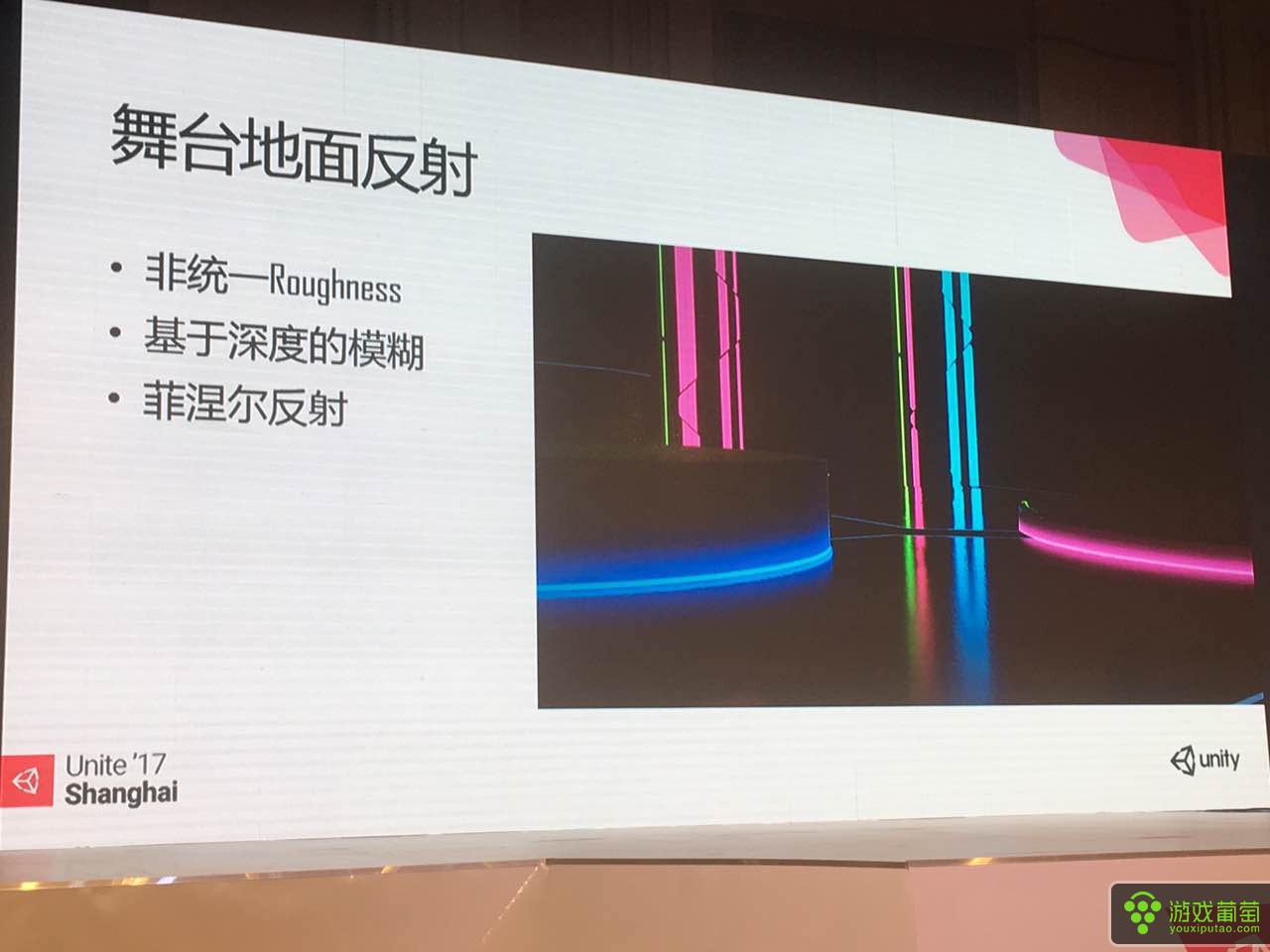
接下来介绍一下灯光组的控制，现实中舞台都有软件和硬件控制灯光变化，我们根据需求写了一个脚本完成类似的功能。首先定义一组灯光，定义一个组的运动和曲线，设置每一个灯光独自变化参数组合不同效果。灯光色彩变化强度也是由曲线控制，我们设了几组灯光配置，这样在播放的时候直接调用需要的配置即可。



下面我们介绍一下地面反射实现，由于我们的舞台一部分是接近于镜面反射材质，如果直接使用SSR从精度就无法满足需求，这就需要我们需要用平面反射实现。平面反射实现大家都不陌生，这种方法也有它的局限性，只能高度统一平面才能使用。为了解决这个问题，在不同落差上就重新渲染，这样效率比较低。我们做了一个平面反射管理器，定义一到多个反射平面，根据当前平面高度查找影射最接近的象素，我们定义了三个主要反射平面，其他的和平面高度接近的平面，查找主要平面的象素来产生。其他一些垂直于地表或者一些细碎的地方，我们则统一使用envmap加SSR。



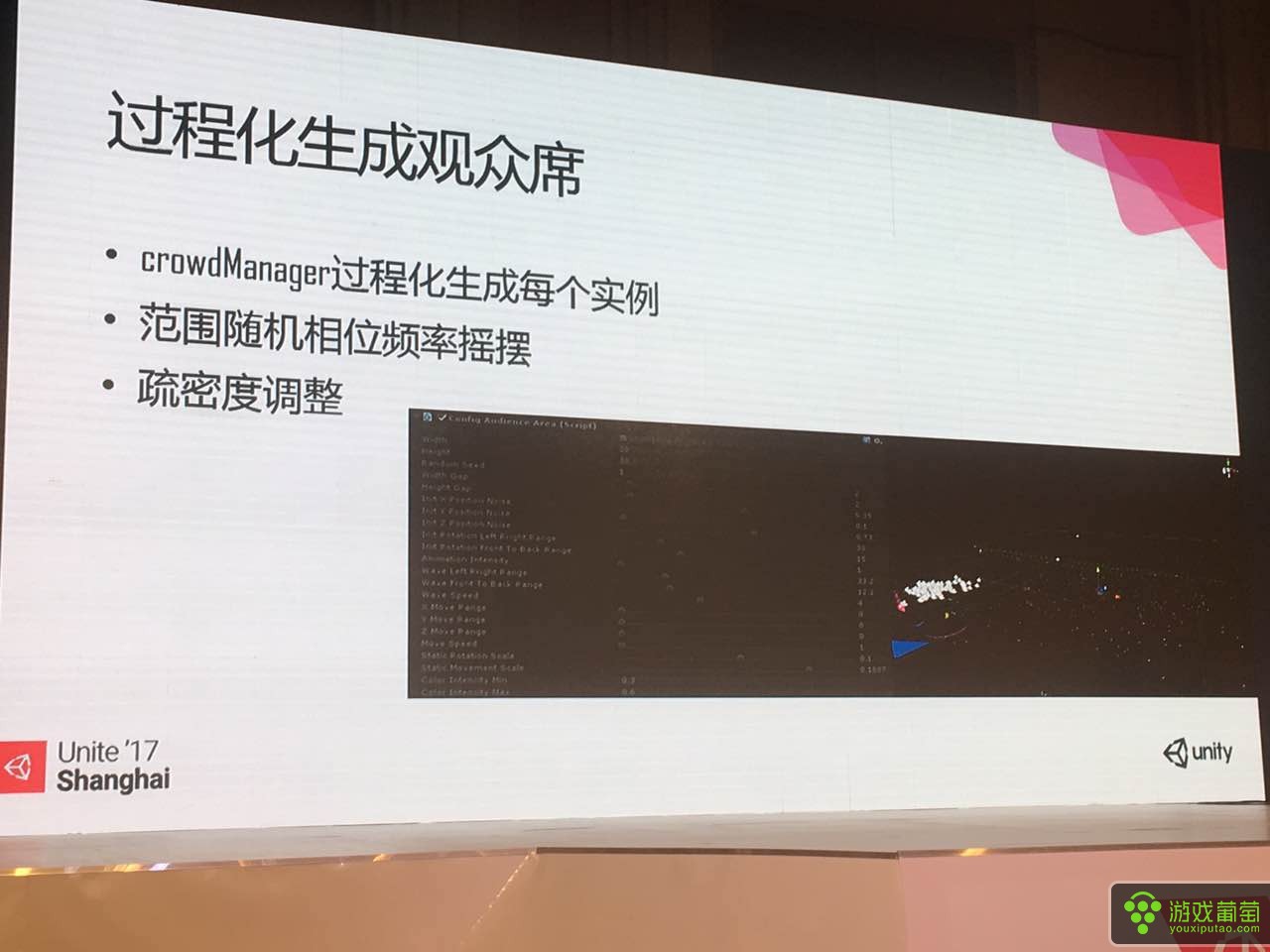
我们想把材质表现的更为细腻，这样对非光滑的地面，我们模拟了基于高度和相机入射角度变化的模糊，基本思路反射纹理生成模糊，根据被反射物体远近选择不同模糊度进行模糊。离反射面越远的物体反射越大，影响模糊度主要因素修改物体到反射点的距离，离距离越大模糊度越高，并受到材质本身Roughness参数和菲涅尔效果的影响。上图是应用模糊后的材质表现，大家可以看到越接近水平角度，反射面越接近镜面。



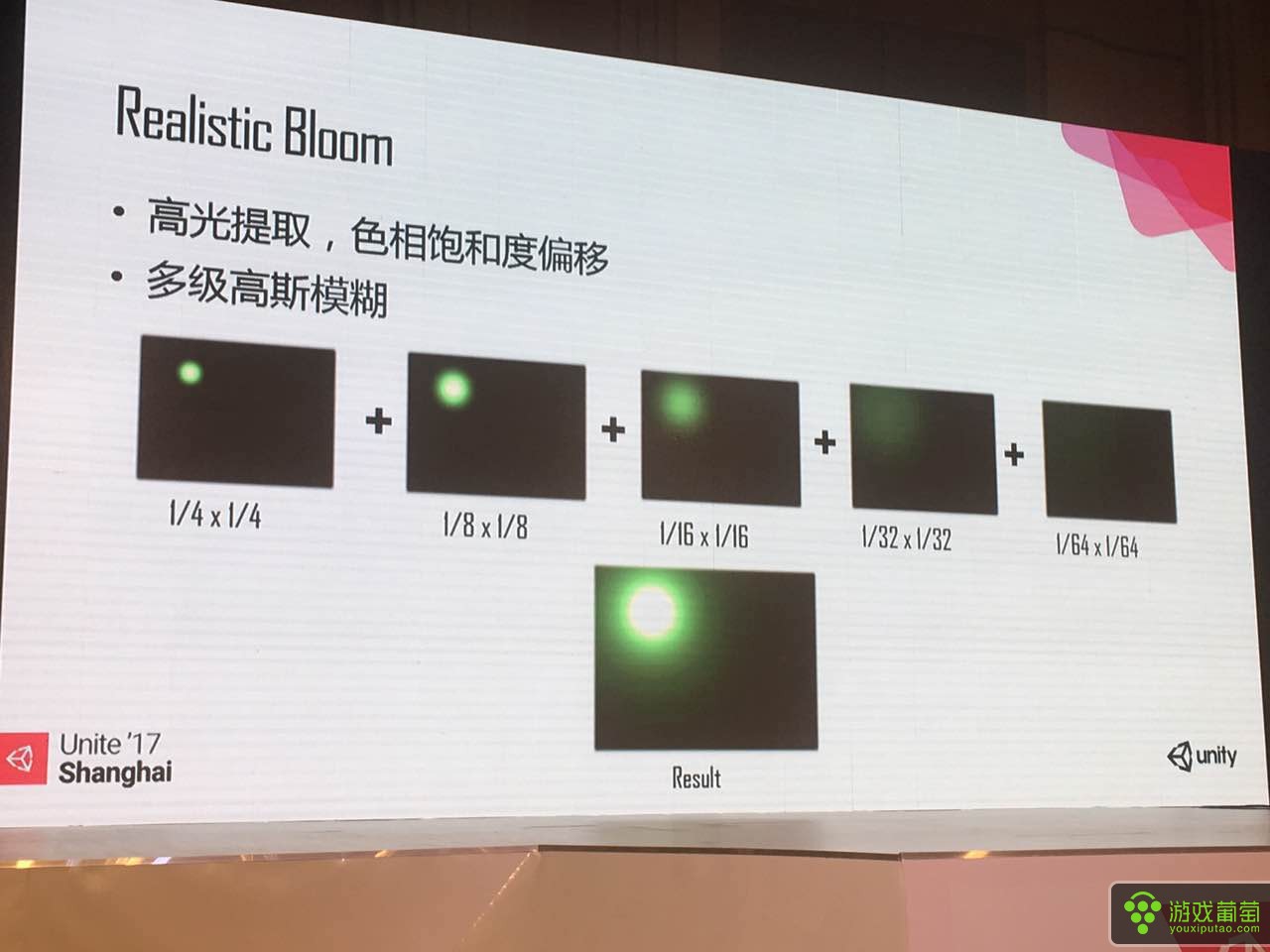
接下来我们说一下过程化生成观众席，大家可以看到挥动的荧光棒完全是由过程化生成。为了表现大量随机群体动作，过程化生成是最高效的方式。这段视频也展示了有500个观众挥舞荧光棒的场面，效率有比较大的提升。实际测试，我们数量从100到400到800，对于帧率并没有明显的变化。



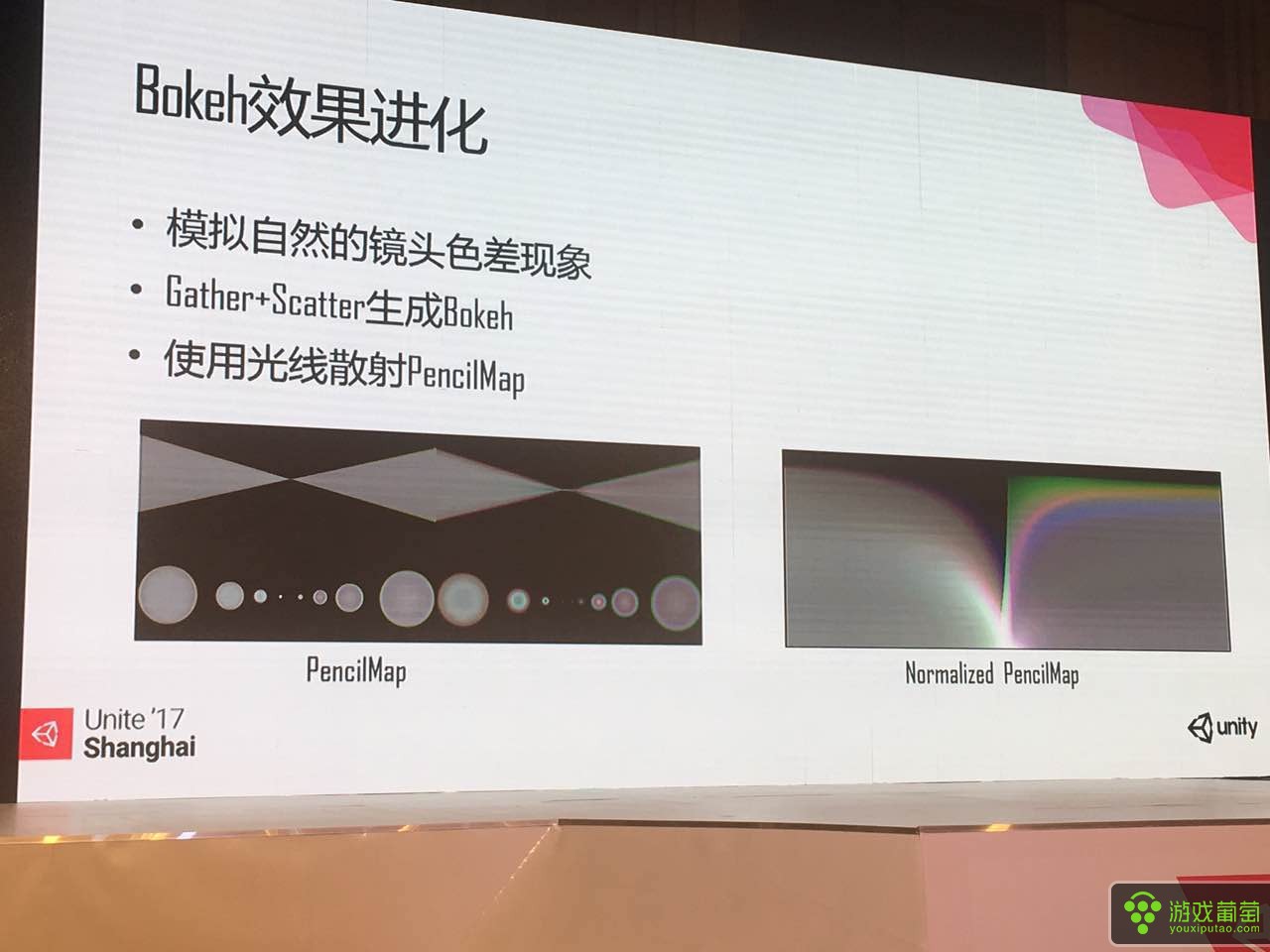
下面我们主要介绍一下实现方式，我们写了一个crowdmanager过程化生成每个实例，观众席每个独立个体表现，由振幅、频率、颜色、位置偏移等随机参数来控制。



接下来我们介绍一下后处理相关的内容，Bloom我们使用高光提取，色相饱和度偏移，这样对卡通渲染来说，色彩看起来就会更加丰富，而且从一定程度上，避免了应用Bloom后，画面会降低饱和度的现象。

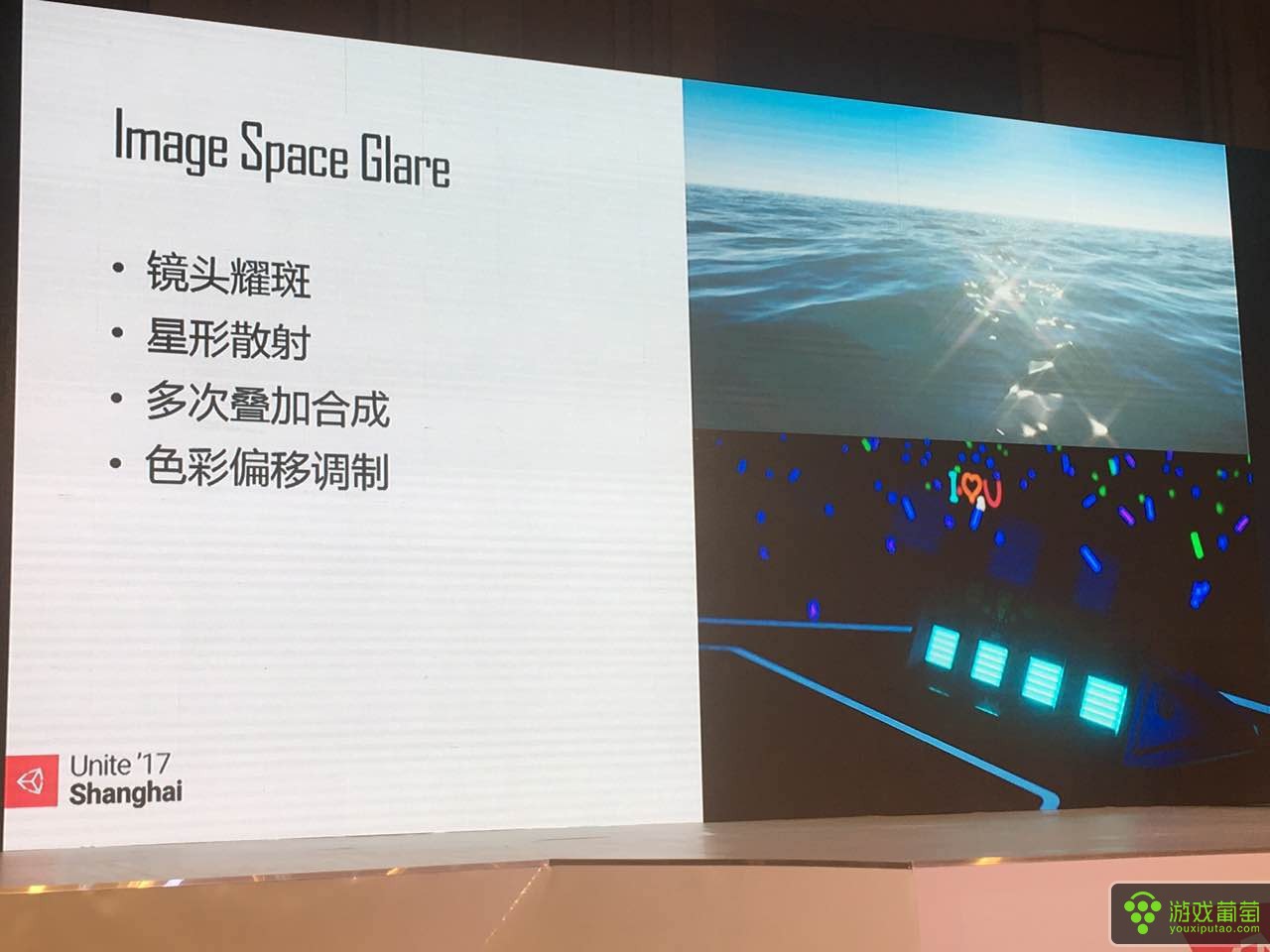


Bokeh效果进化，模拟自然的镜头色差现象，使用不同的pencilmap可以模拟不同的镜片色散效果。左图为对比效果图，我们可以看到应用了之后，Bokeh边缘有蓝色色散现象，比右边更加真实和自然。





我们还加入了加于图像的Glare效果，做法和Bloom一样，通过多次叠加，应用色彩偏移调制以后，得到最终的效果。



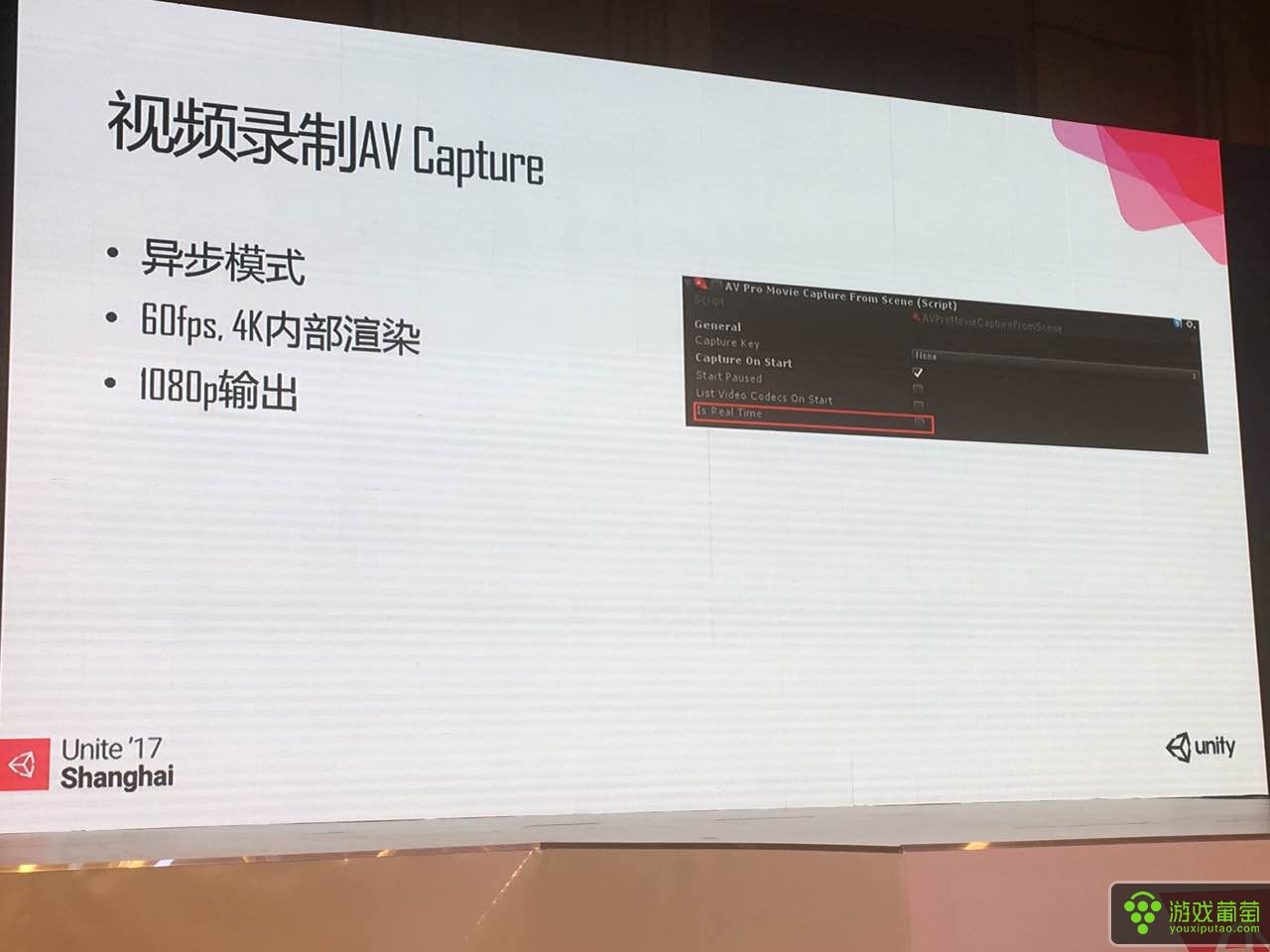
AO方面我们使用了饱和度增强的HBAO，和普通的HBAO相比，它不仅仅在AO地方变暗，而且把色彩饱和度、色相进行变化，这样对于卡通渲染来讲，画面不会变得在暗处比较脏，保持了原有风格，我们可以看到应用AO之后，右图比左图层次更加丰富。



我们在制作的时候，我们使用插件来做分镜编辑，Slate更方便，而且可以自定义需求。我们加入了自定义控制的参数，还有3D渲染相关的汇聚和深度控制参数。除此之外还有灯光控制，大屏幕视频内容切换，这些都由Slate驱动。整体来说，Slate较为应用广泛，也有小漏洞。



最终渲染录制我们使用了AVCapture来做，我们采用异步模式。视频渲染规格为1080p60帧，内部渲染效果使用4k获得更好的效果。视频录制和开始以及结束也是通过Slate触发来控制。



下面我们说一下3D渲染，3D渲染我们使用双向机左右格式分别渲染，有基于分镜的汇聚和深入调整。由于使用两个相机渲染，类似于反射、折射等效果都能做到比较好的支持，没有兼容性的问题。因为我们主要是追求最好的渲染品质，效率并不作为主要权衡点之一。



下一步我们准备支持VR模式，需要做的是优化运行效率，目前来讲，在没有优化的情况下，GTX 980TI 1080p可以接近60帧，优化之后970可以达到120帧的效率，提升空间比较大。另外一方面，由于用到比较多自定义渲染，在VR会出现渲染兼容性问题，这点也是需要解决的。除了VR之外，我们也会继续优化和升级卡通渲染的效果以及运行效率，争取把次世代卡通渲染推向移动平台。随着移动平台性能的持续提升，估计2到3年之后性能接近于PS4。



今天我要做的分享就到这里结束，谢谢大家！