Metahuman解析

[UnrealCircle成都]MetaHuman原理及流程浅析 - Epic Games China｜孙丹璐_哔哩哔哩_bilibili

大家好，我是EpicGames中国的技术美术孙丹璐，今天为大家带来的分享是Metahuman原理及流程浅析。

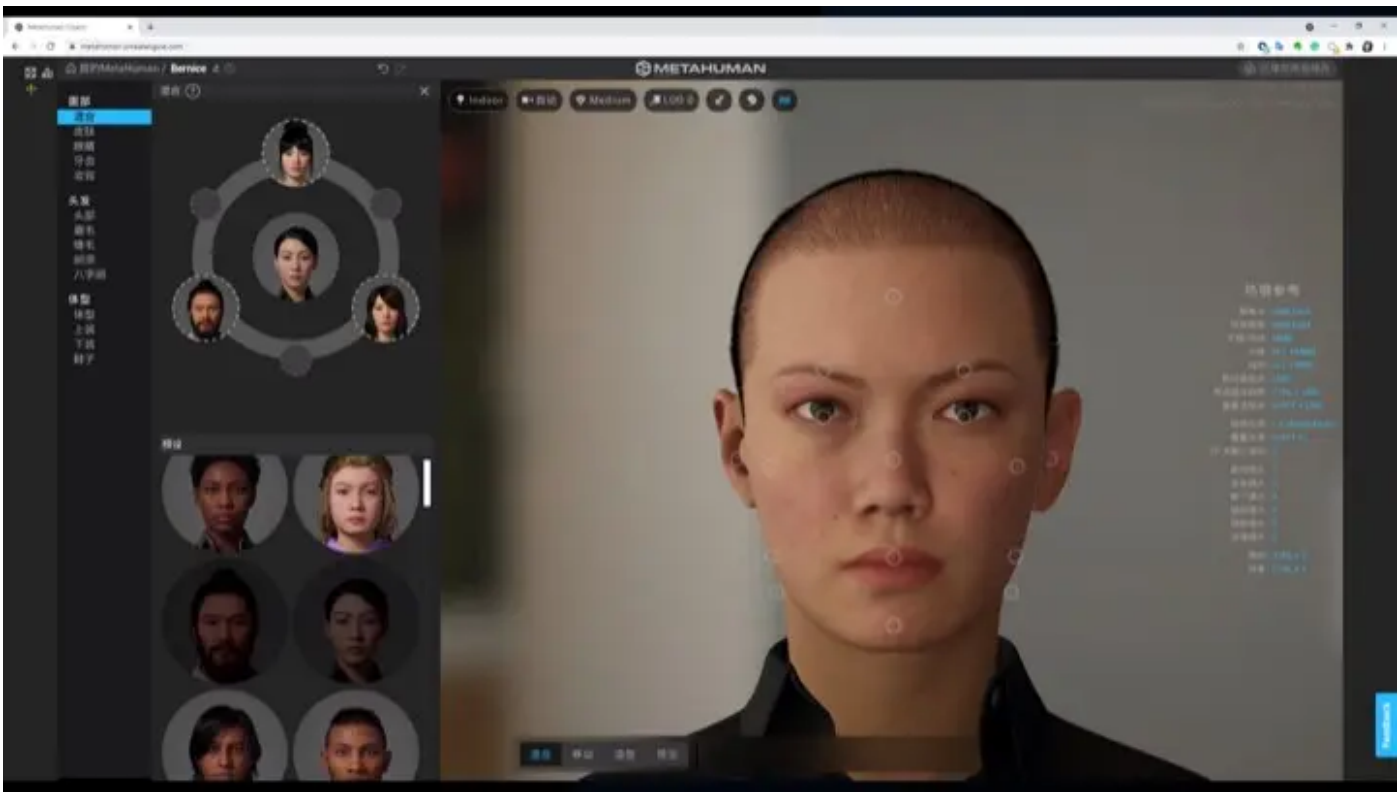
现在数字人是个非常火热的题材，不管是游戏、影视、直播、工业、广告等领域都有着大量需求。但是传统数字人的制作在设备，人力和时间成本上都非常高昂的，每个流程，每一个环节比如扫描角色、绑定，动捕和面捕进行驱动，也都需要一定的技术门槛才能达到的。所以在市面上的大多数案例中，比如大家看到的AAA游戏中，基于成本，一般只有有限的主角会采用完整的高精度数字人效果，其他角色的精度都相对低很多。

基于此，我们提供了一套方案叫做MetaHuman Creator，快速和容易地创建独特高保真的数字人类的工具，你可以直接操纵面部特征，调整肤色，并且从预设的身体类型、发型、服饰等范围中选择，在metahuman中甚至可以编辑角色的牙齿，在角色制作完成后，你的角色会包含完整的绑定并可以直接在虚幻引擎或者maya中制作动画。

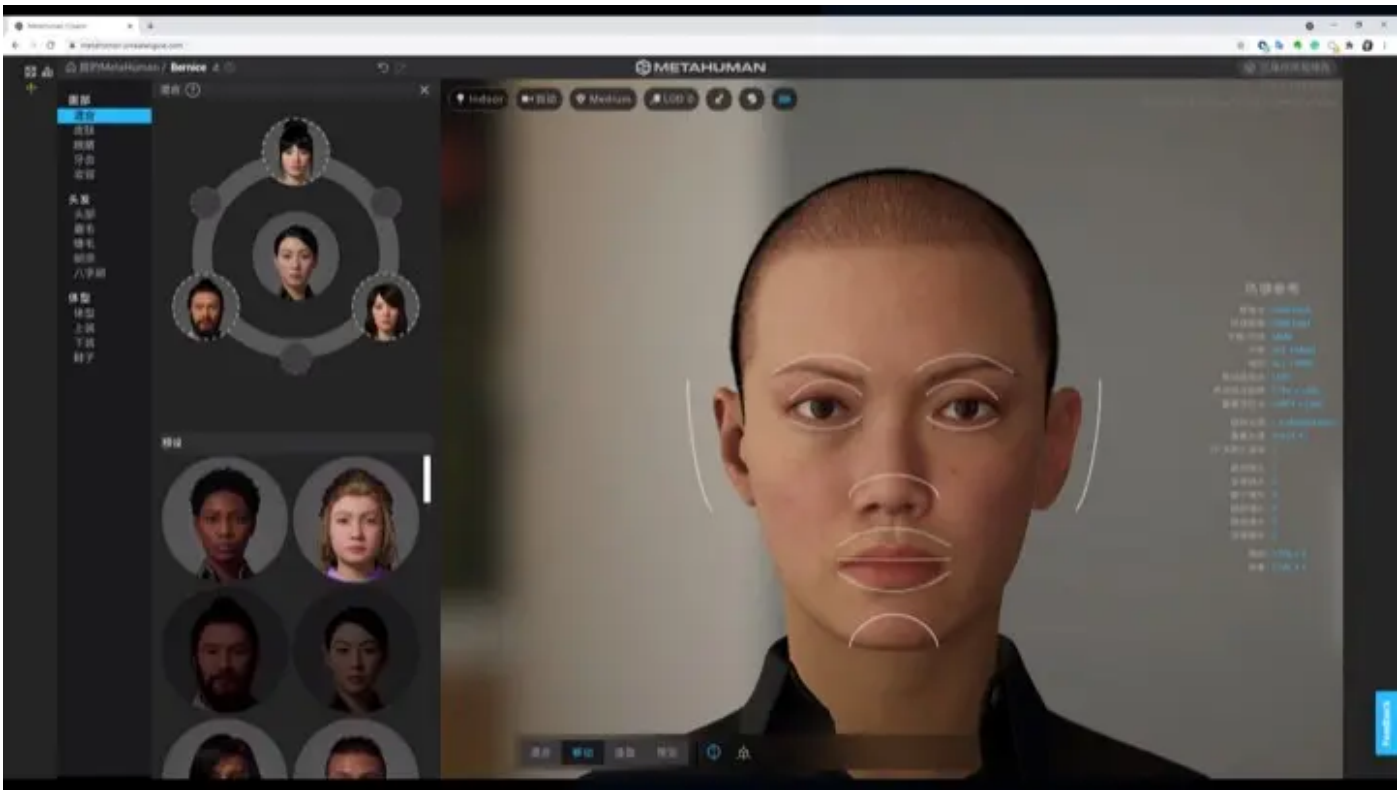
为了达到零门槛创建数字人，可见即所得的目标，我们提供了基于Epic生态的三个套件进行协同工作，通过MHC快速在线制作数字人，利用Quixel Bridge导出数字人，在虚幻引擎中应用到游戏或者影视当中。

https://v.qq.com/x/page/s3312...

我想先为大家展示下通过MHC进行捏人，MHC依托于网络，通过Pixel Streaming在网页端交互使用，在这个视频演示中，我捏出一个全新的数字角色实际只用了4分钟，大家可以看到操作也很直观易懂，上手门槛非常低。在捏脸的部分，面部混合提供三种模式 。

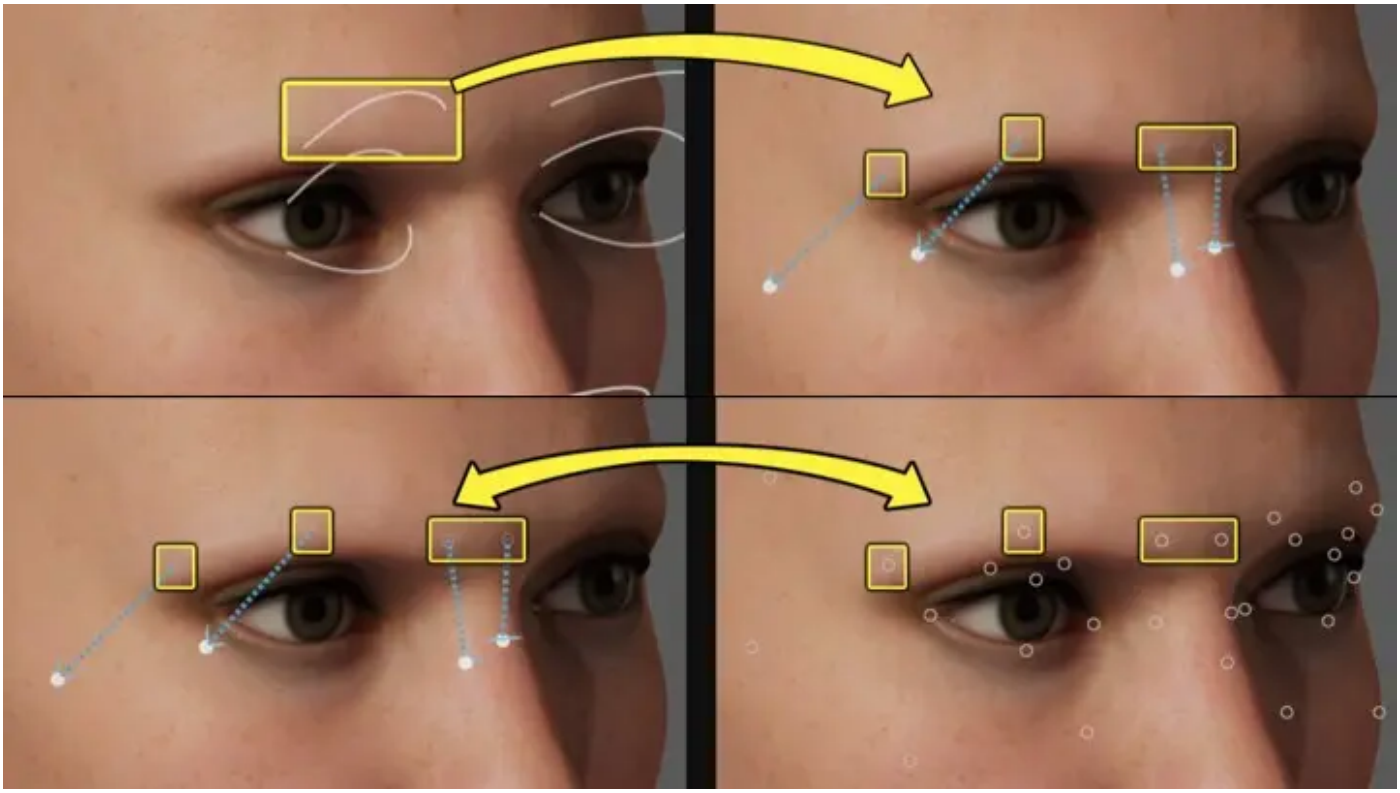


第一种是混合模式，最粗粒度的捏脸，通过左上方的混合圈，添加3~6个角色区进行插值，你每控制一个特征点进行捏脸，就是采样这几个预设角色的特征点去做插值。

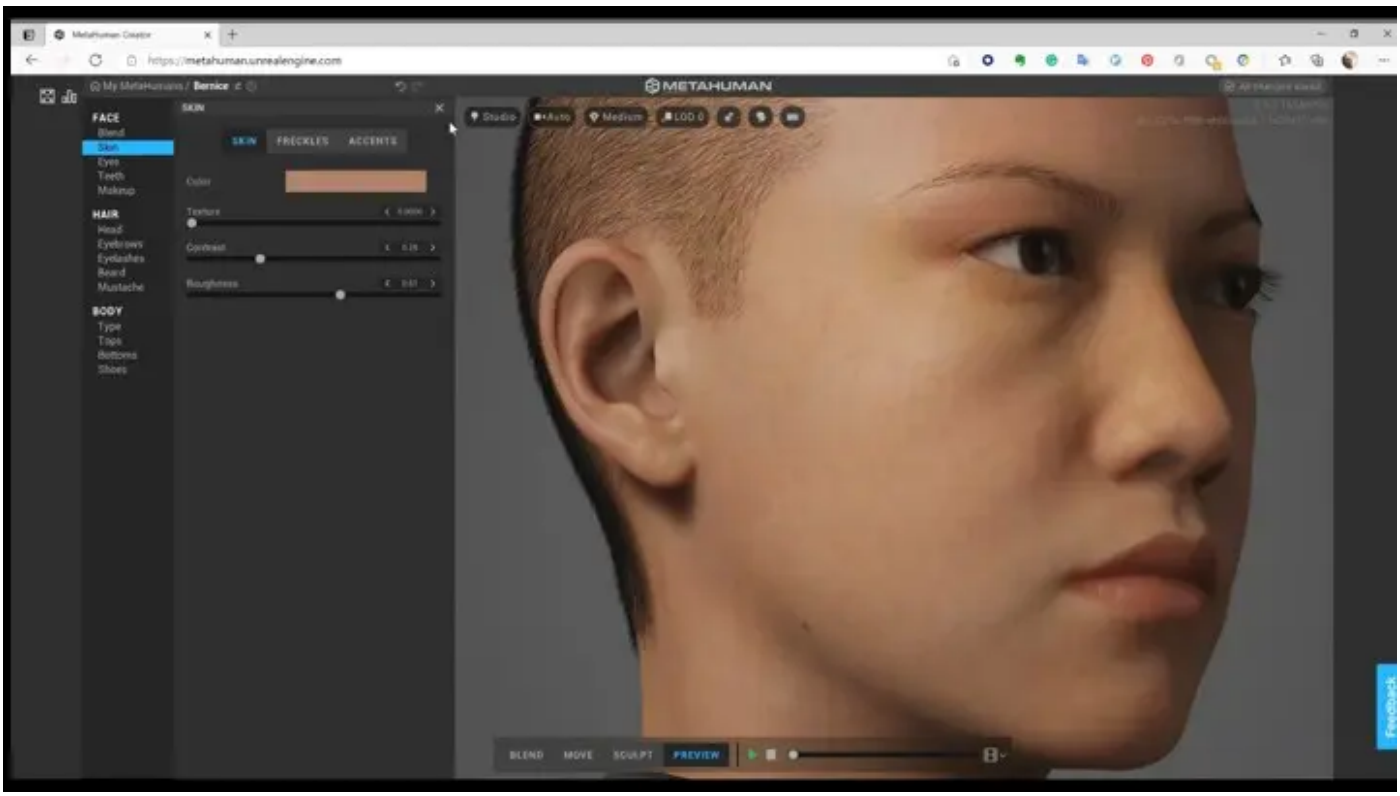


第二种模式是移动模式，按照五官划分了特征点，在这个模式下可以快速的修改五官。

第三种模式是雕刻工具，是Creator提供的最细粒度的工具，可以针对最小特征点，比如鼻尖，来区进行细节调整，Creator中大家需要注意到这里的雕刻和Zbrush中的雕刻是存在一些不同，MHC中的雕刻, 是通过你期望雕刻的形态，在基因库中索引一组面部特征点再进行融合。另外Creator中常见的一个误区是，移动和雕刻工具，实际上与你左上角预设好的几个角色是无关的。它移动和雕刻都是在整个基因库中进行索引，并不依赖于左上角预设好的角色。

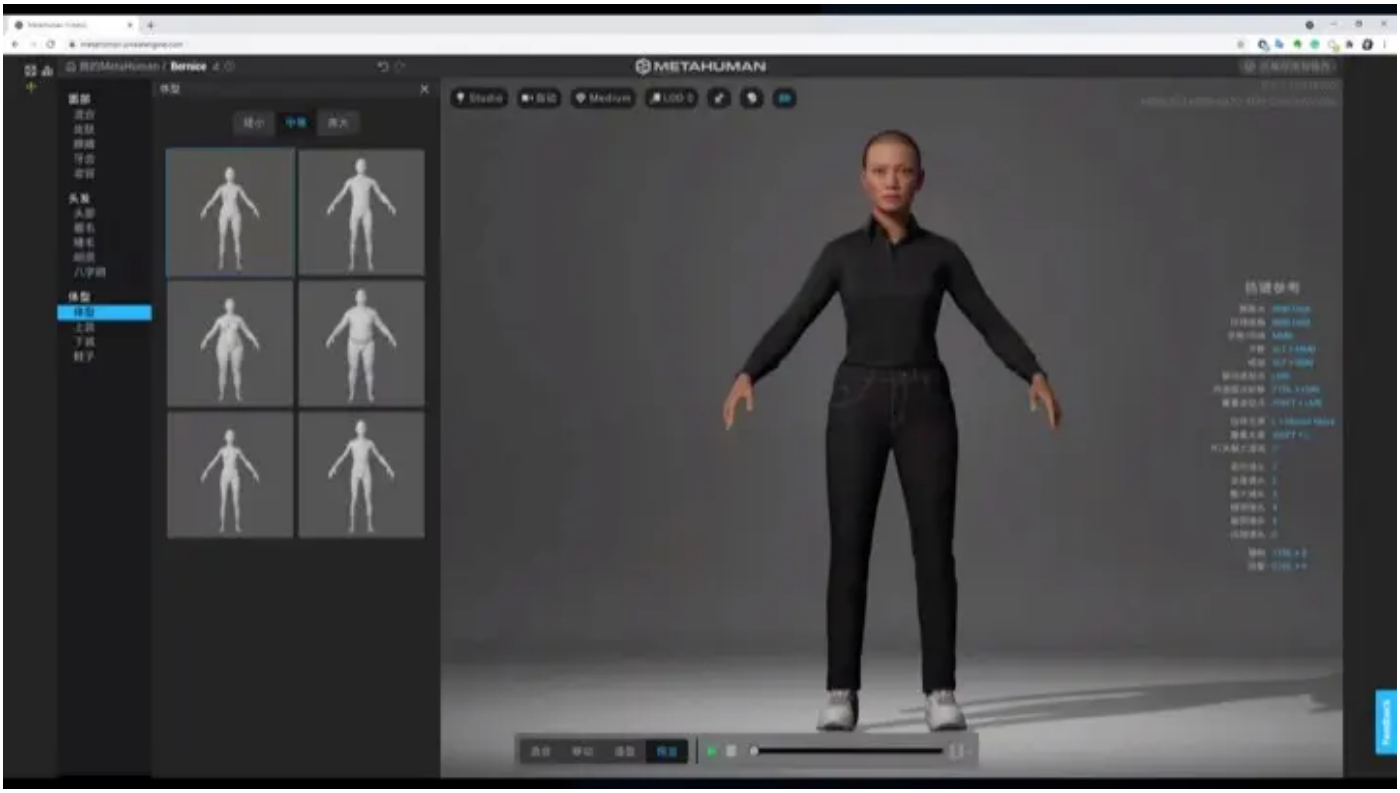


另外，其实移动工具和雕刻工具，是可以一一对应起来的，比如移动模式的眉弓处控件，对应的是雕刻工具下 ，眉心、眉头、眉峰、眉尾、这四个特征点。



另外在Creator中也可以调整肤色和纹理细节，皮肤的纹理选项不但可以增加脸部表面的细节，也会对面部的形态产生影响，比如皱纹对于面部形态其实是有一个形变的。

Creator中也提供了相对灵活的妆容设置，比如粉底，眼影和眼线，各部位的腮红、口红，能够进一步减少用户对于数字角色进行二次修改的需求和开销。



在身体的塑造上，MHC提供了预设好的18种体型进行选择，头部目前并不会随着身体的高度或体型选择自动的发生改变。所以其实有的时候你选择体型之后，可能看起来身体和头的比例有些奇怪，所以我们提供了头部缩放功能。

头部的缩放并不是完全等比缩放，而是会更符合人体特征，你可以看到，你现在缩放头部的时候，你的脖子并不会随之无限的增粗，这样缩放也能较好的处理头颈和身体的衔接。

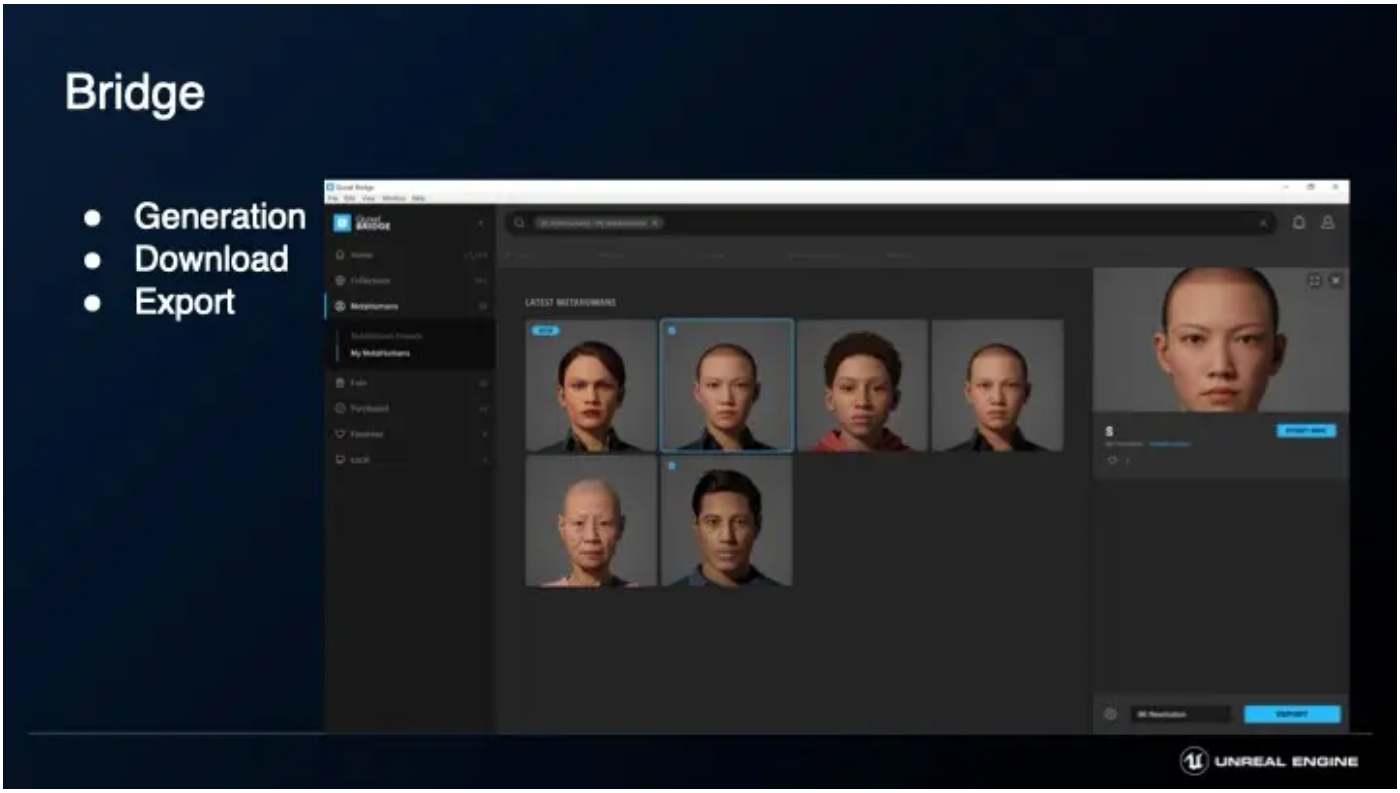
我们来看下MHC背后的技术，首先我们会通过4D扫描大量的现实人类面部，通过人工和ML处理数据，提取出每个角色的特征点信息，存入一个叫做DNA的数据格式中，而DNA用于描述角色的外观、骨骼绑定，最后我们把这些信息存放到GenePool中，GenePool是我们的一个数据库，用户每

一下捏脸操作，实际上就是在GenePool的数据库中进行索引，混合，最后基于你捏好的数字角色，也会生成一份独一无二的DNA数据。

在体型方面，采用了传统流程制作的体型。主要是目前我们还没有较为完善的身体数据库，所以没有办法和头部一样可以融合不同体型的特征点来捏体型，另外现在我们也不提供利用缩放进行任意的体型调整的设置，主要原因是整体效果不能达标，比如说局部调整过体型之后，你的身体和脖子的接缝处并不是一个很好处理，另外服饰也是很大的问题，局部放过的身体很难做到布料贴合。

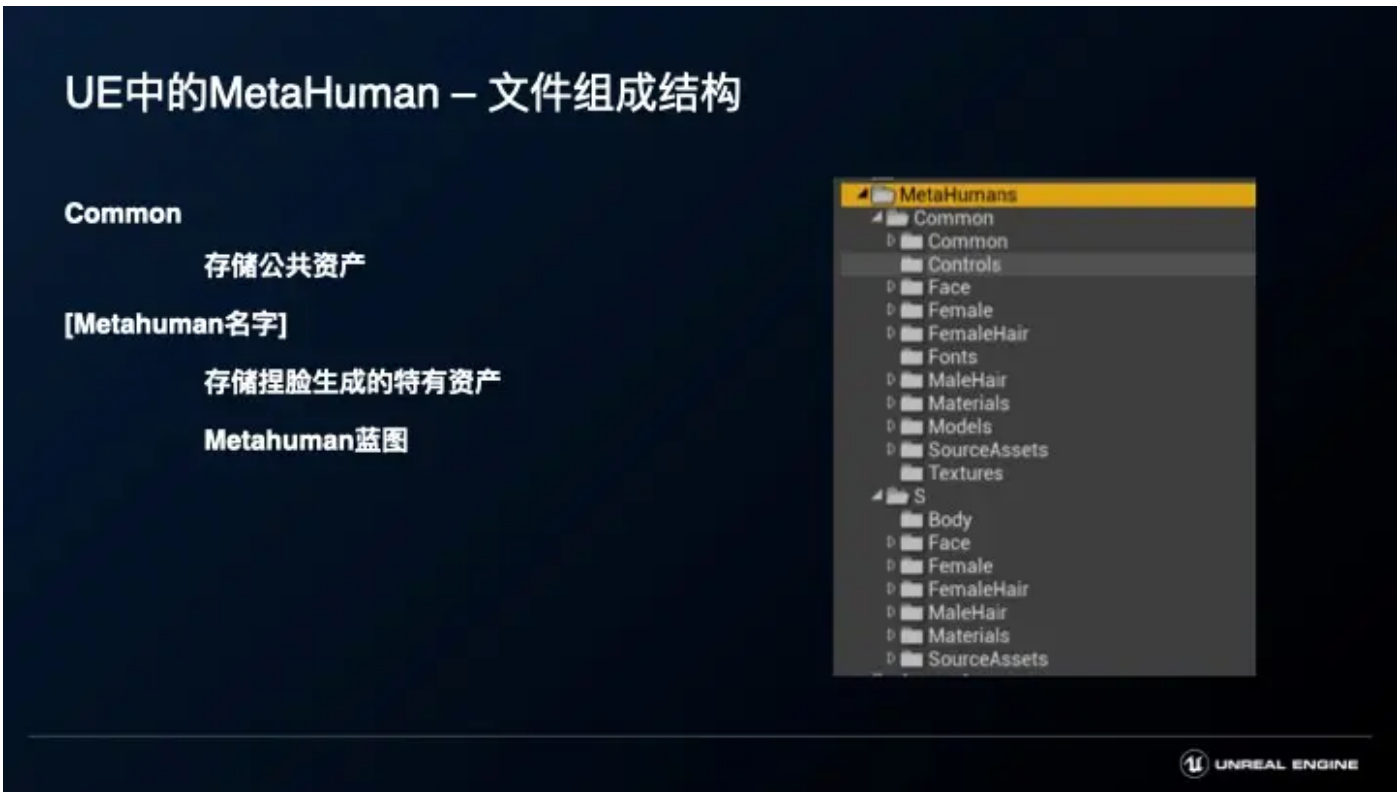
在纹理方面，皮肤的纹理是通过扫描数据中的纹理进行合成来生成制作，以确保达到更真实的效果，扫描的纹理分离成低、中、高频率的细节纹理，用于在Creator中自定义的混合，而其他地方使用的纹理，则使用更传统的管道去进行制作。

另外MHC是一个基于云端的APP，我们把MHC放在了云端。主要是因为以上数据量，尤其是面部GenePool和纹理数据的数据量非常非常巨大的，而且未来会不停扩充，单就数据量和运算压力恐怕不太适合放在个人电脑上进行运算，所以我们采用了基于云端的方案，同时云端还有个好处，更新迭代很快，很多问题和改善可以直接后台完成，不需要每一位用户都进行频繁的更新。为了把云端的MHC输送到浏览器，我们采用了Pixel streaming，Pixel streaming是一个基于虚幻引擎的流送技术，在云端服务器上运行虚幻引擎应用程序，通过WebRTC将渲染的帧和音频流送到浏览器和移动设备，不知道大家有没有意识到，使用了pixel streaming意味着，后台跑的也是虚幻引擎，在云端的虚幻引擎中制作数字人，导出到本地的虚幻引擎里进行使用，所有参数，模型，毛发，布料，纹理，材质，LOD参数都可以一一匹配，这是真正的所见即所得。



从MHC捏好之后，就可以通过Bridge下载你制作的数字角色了，打开Bridge可以看到Metahuman的分栏，里面会实时更新你捏的MH，在Bridge需要配置你希望下载的纹理分辨率。在点击download的时候，会有一个自动步骤是生成，生成是根据捏人时候设置的数据，比如皮肤，以及download设置的参数，基于这些数据生成纹理和LOD，因为后台跑的是虚幻引擎，所以模型以及骨骼的LOD，都是基于规则，使用引擎内LOD减面和骨骼LOD剔除达成的，而对于毛发的LOD，为了保证效果，是手动生成的插片和模型LOD，MetaHuman的资产生成可能需要一点时间来处理，这主要取决于所选择的资产是什么，它们的纹理分辨率以及你现在的网速，还有就是会考虑到你现在的Bridge队列中有多少资产需要生成，平均来说，如果你下载的是一个有1K纹理的MetaHuman，需要大约20分钟来生成和下载。

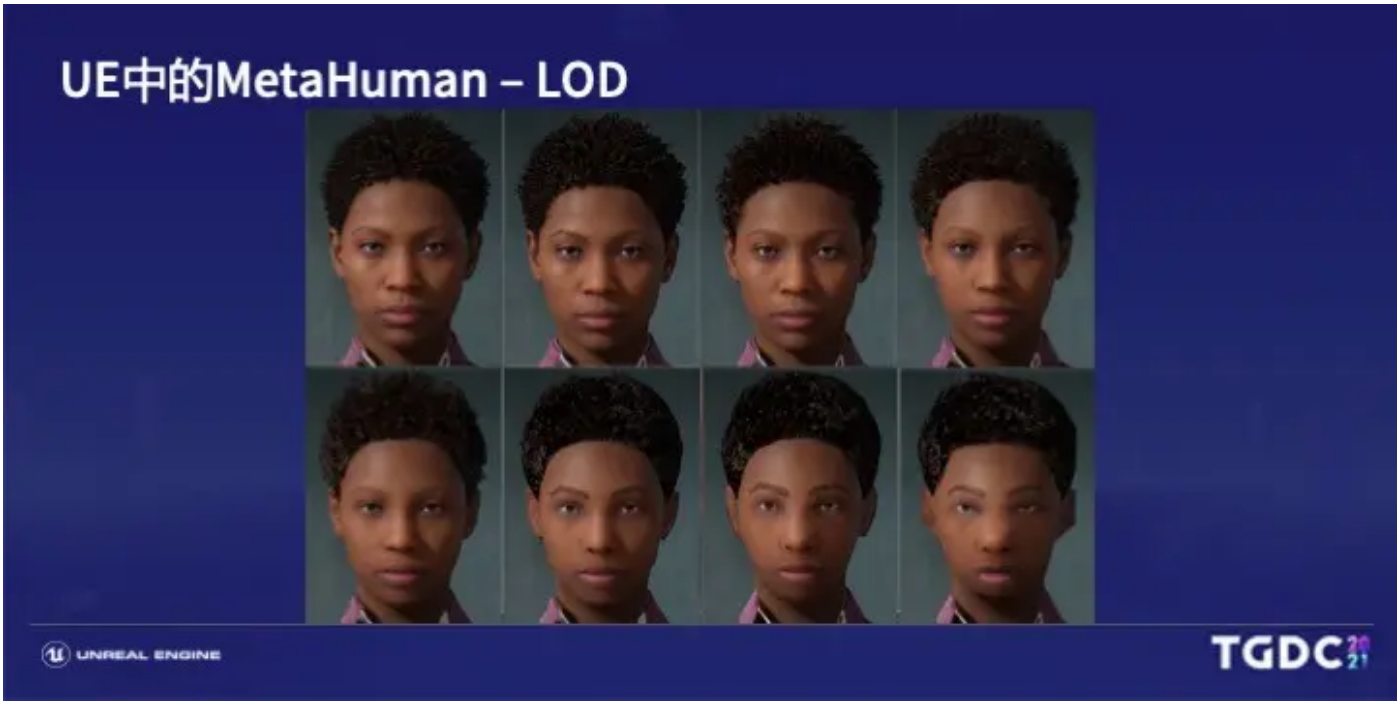
在下载的时候选择的纹理分辨率，并不是一个绝对分辨率，而是一个指引的分辨率，也就是说如果选择下载8K的，实际上只有贡献最大的normal和cavity纹理是8K的，而皮肤albedo是2K的. 这其实和皮肤本身的特性有关，一般SSS皮肤散射的效果接近于低频的滤波，效果有点类似于blur之后的纹理，所以albedo的高频细节本身就会被SSS的shading特性抹掉，我们对albedo的贡献做了效果和效率上的评估，发现8K 4K的albedo相对于2K的效果改善几乎是肉眼不可区分的，所以在这里使用了2K的分辨率。而roughness纹理，是使用虚幻中一个自带流程将法线转换为粗糙度，通过Bridge导出可以导出到maya和虚幻引擎中，接下来我们来分析在虚幻引擎中的MH。



导入引擎后，会有一个专门的Metahuman文件夹，对于所有Metahuman生成的数字角色来说，无论是男性还是女性，有些资产内容都是共通的，我们把这些存储在MetaHuman/Common文件夹中，而每个人特有的资产则存储在相应名字的文件夹下。

位于Metahuman文件夹下，Metahuman的身体各部件会以一个蓝图组合到一起，包含躯干，面部，面部各种毛发，出于性能和效果的平衡，并且为了适配不同的平台，不管躯干还是毛发，都有自己的LOD信息。

比如头部我们设置了8层LOD，我们为每层LOD设置了一定规则，每层LOD都有独立的，顶点量、blendshape,、关节数、动画纹理、蒙皮影响等区别，blendshape因为性能开销比较高，所以从LOD1开始就没有在使用，而纹理动画则是因为需要读入的纹理非常多，所以LOD2开始没有在用，当然从屏幕上看这些头部，可能LOD3或者LOD4开始 ，之后的效果会显得有点粗糙，但是LOD的选择一定是要综合考虑到屏占比

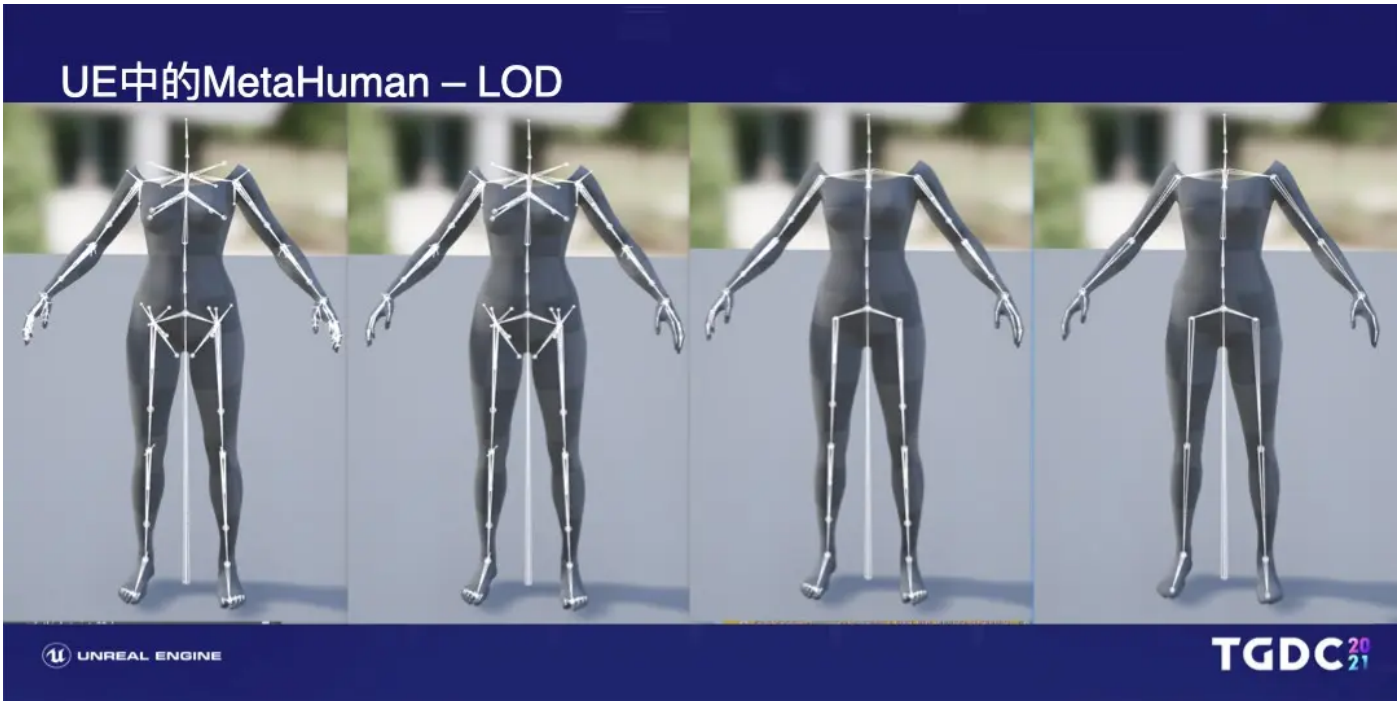


来去衡量效果的。

比如我在对应LOD的屏占比上去截了这三张图，分别是拉到最近时候的LOD0，中距离看到全身时候会切换到的LOD3，以及远距离切换到的LOD7。

我在相同距离下，把模型强制按LOD0渲染作为对比，也就是图上现在新增的LOD3和LOD7旁边的对应的头部和全身，可以看到两者的区别并不是特别大，也就是在这样的距离下，在这样的屏占比下，LOD切换对于画面的效果并不会影响太大，LOD其实不光是对于性能有提升，对于画面效果也是会有一定改善的，比如大家常见的画面失真摩尔纹，引入LOD就可以改善这种情况。

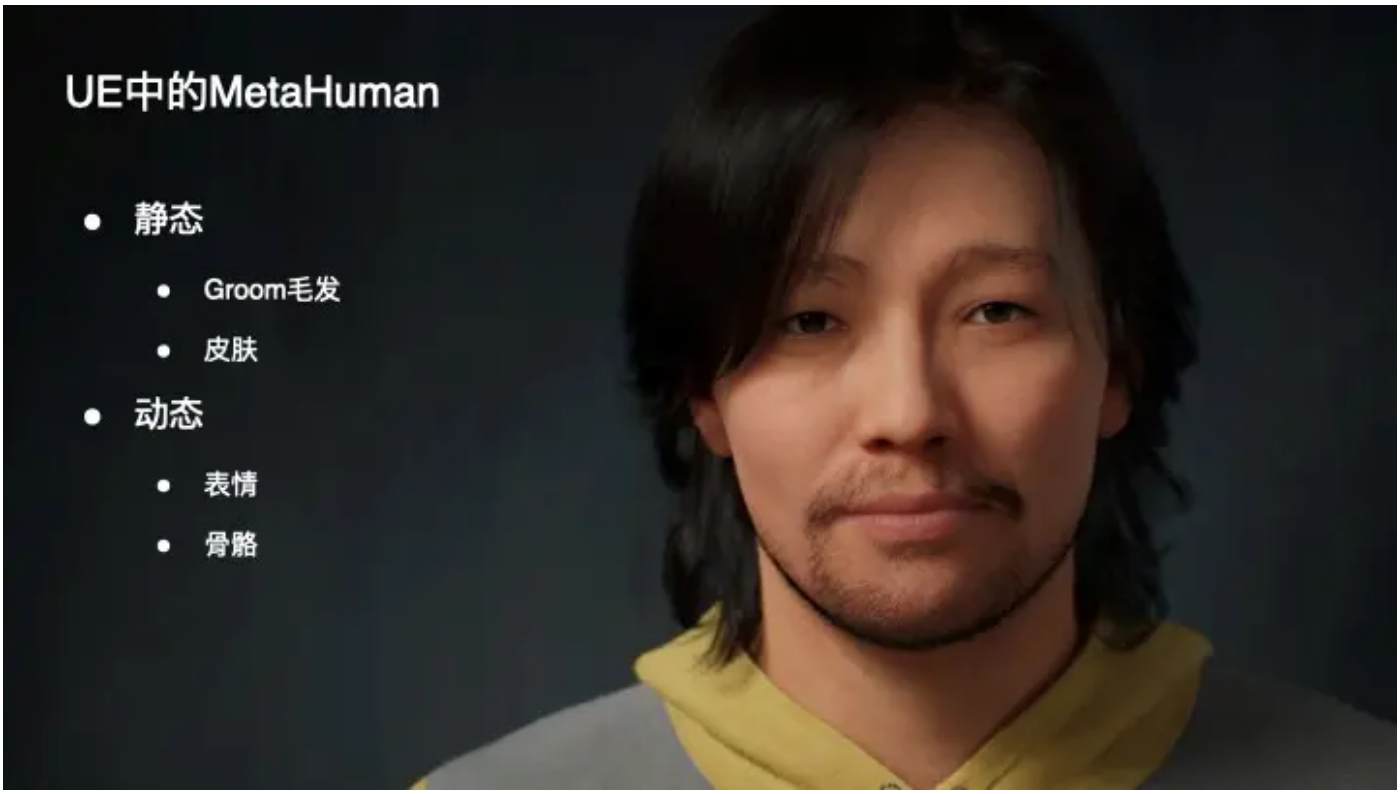
这是身体模型的LOD数据，身体只有4层LOD，以画面上的裤子为例LOD0是有一万五千面左右，而LOD4 大概是一千五百面左右。



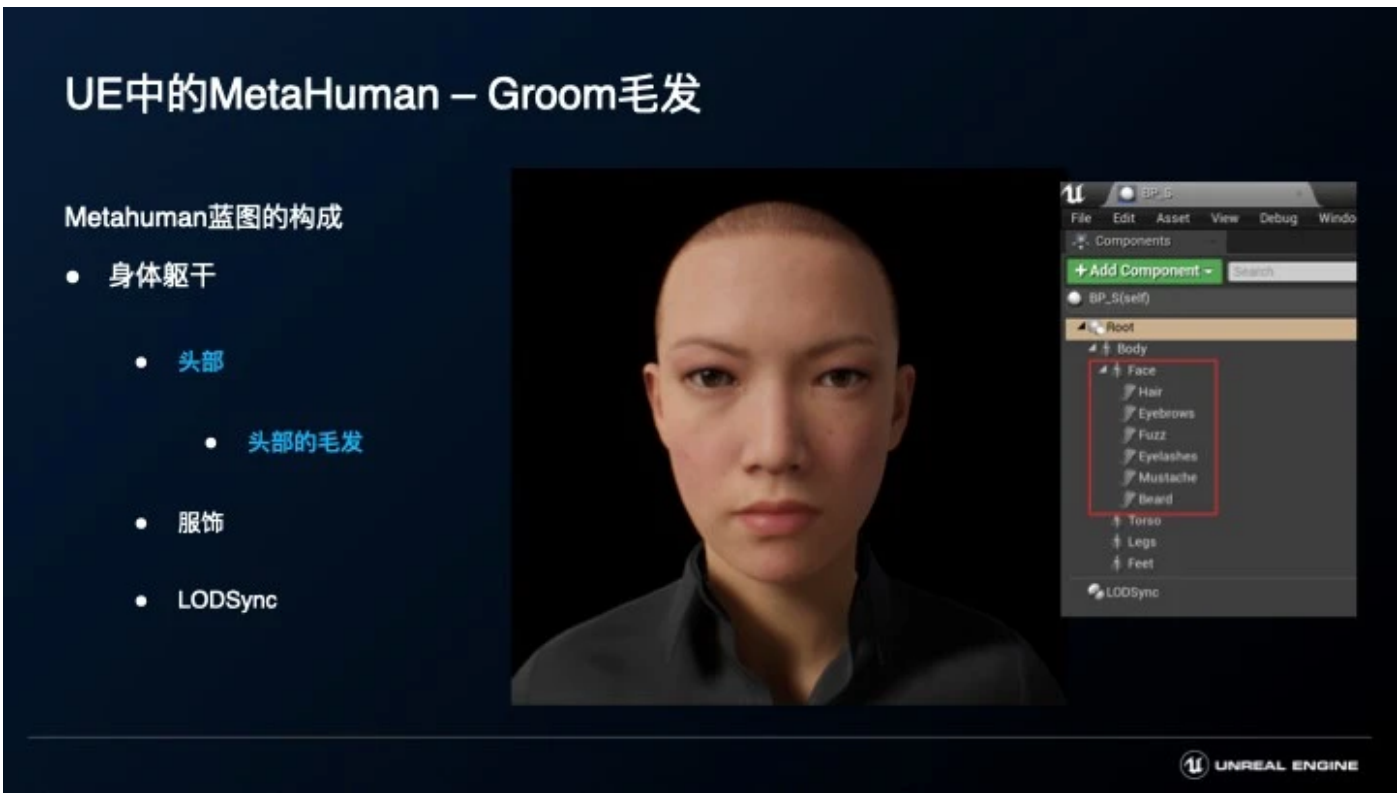
对于骨骼而言，骨骼模型的LOD可以大量减少skinning的开销，但是他还需要骨骼的LOD，骨骼的LOD可以很大减少动画Evaluation中采样、插值、混合的开销。那么身体骨骼LOD0是150根骨骼，LOD4是55根骨骼。

大家其实注意到，每个不同的部分可能LOD层数不一致，比如说刚才的头部是8层LOD，而身体的LOD只有4层，同时要考虑到，因为身体和头部的屏占比不一致，所以计算出来的LOD也会不一致。那就会导致效果上出现一些不匹配的问题，比如身体和头部的接缝处没有办法完全匹配，或者说，我们考虑毛发的话，屏占比很小的眉毛groom可能永远没有办法渲染到LOD0，而只有很近处才需要的汗毛groom，却因为包围盒是整个面部，所以会在中距离处也会渲染到LOD0，这是一个性能上的浪

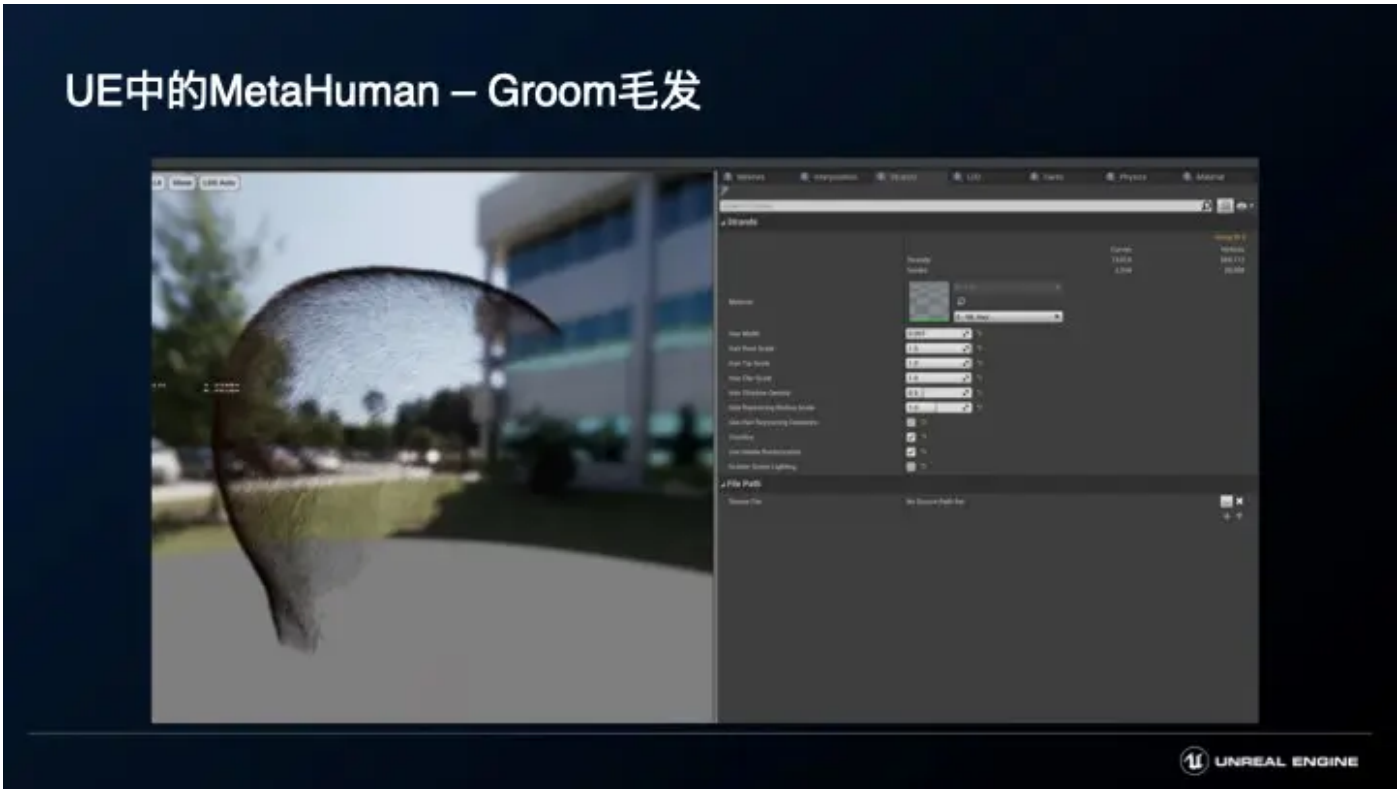
费。那么为了解决这个情况，引擎加入了LODSync component，通过配置LODSync component在这里面配置不同部位的贡献和计算方式，可以让不同部位也可以做到共享一套LOD。



因为时间关系，可能没有办法分析引擎中metahuman每一个细节，下面我会以相对静态和动态资产为区分，结合可伸缩性，主要介绍下面几个主要的部分。



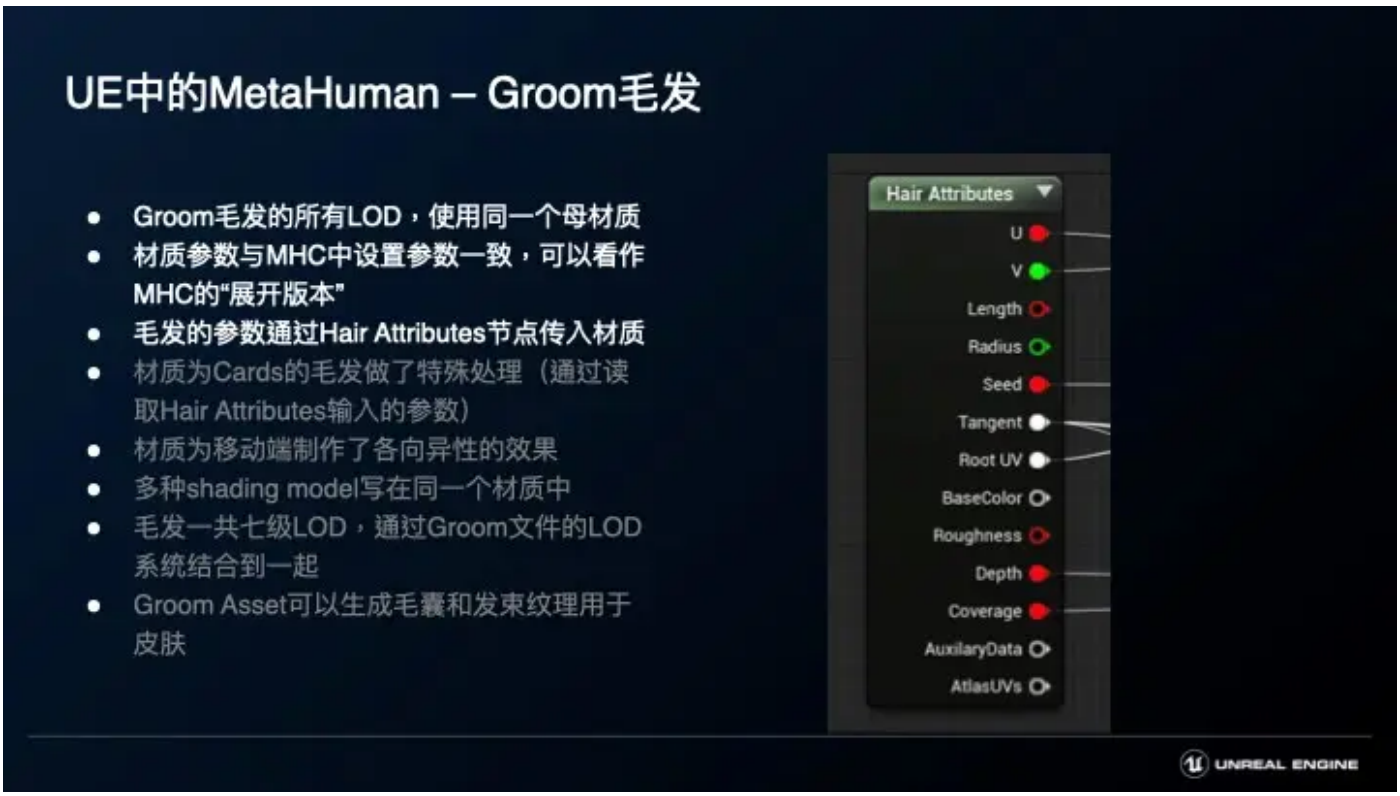
首先是毛发，毛发包括：头发，眉毛，面部的绒毛，睫毛以及胡子。



毛发采用了引擎的Groom毛发系统，这里不会细展开groom本身，而是介绍Groom对于Metahuman的一些处理，以及值得注意的地方。

Groom毛发的所有LOD，不管是插片还是mesh，都使用同一个母材质。

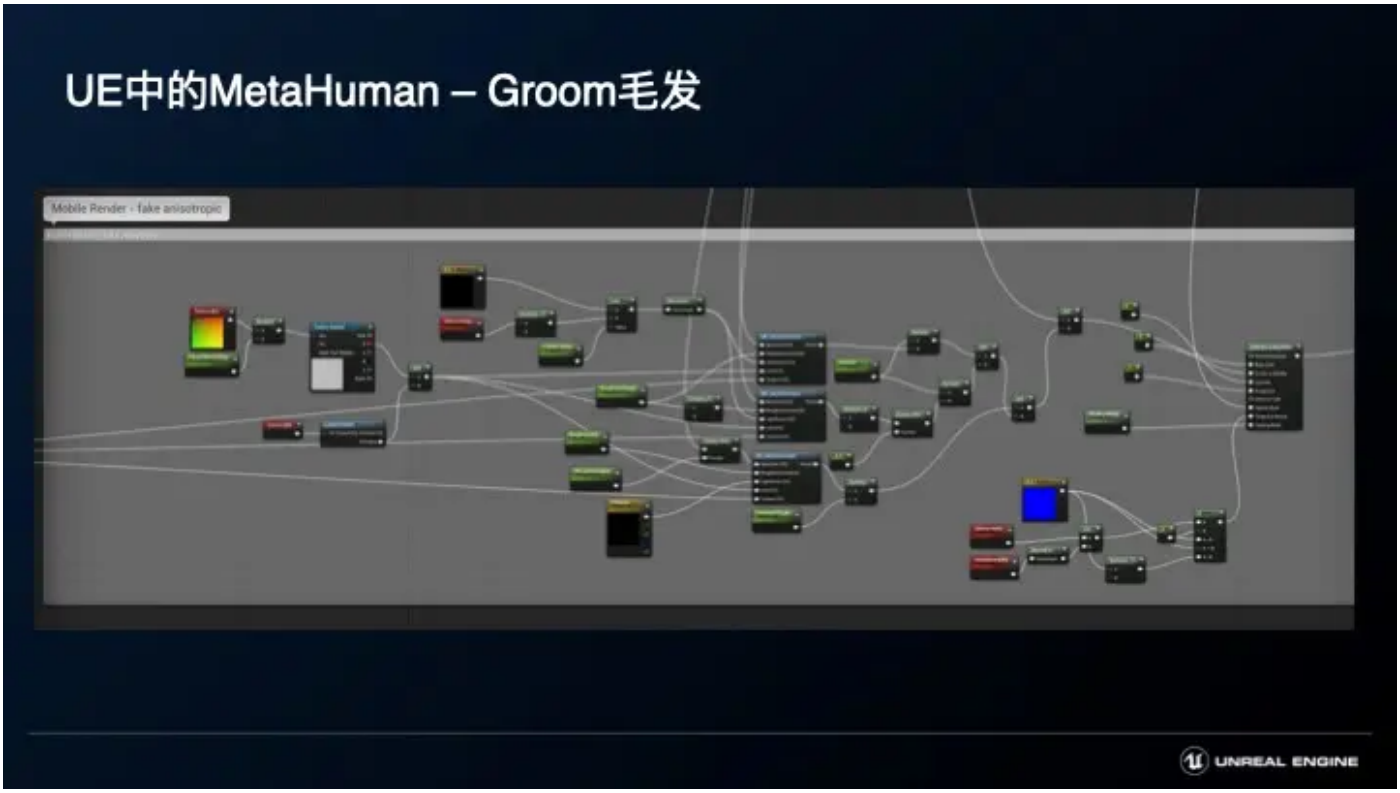
毛发的材质参数与MHC中的参数是一一对应，可以看作MHC的“展开版本”。比如可以看到材质中，有红色素，黑色素，如果你在MHC调整头发颜色的时候注意过，这些参数就是MHC中的调整参数及数值。



另外Groom毛发的参数是通过Hair Attributes传入到毛发材质中的。

将深度纹理输入到hair attributes的方式，是在groom asset中，cards分页下指定对应纹理，比如说深度纹理，一般输送到插片毛发中是会用于、pixel、depth、offset，他是改善头发与头皮交叉的效果，另外是Coverage用于毛发的opacity，里面比较特殊的是Attribute纹理，Attribute纹理对应的是Root UV和Seed，Seed也就是毛发的unique ID，一般是用于在材质中加强头发的发丝感。

大家知道，后续我们会为移动端增加多种shading model,但是当前移动端还不支持，所以当前只能fall back到default lit。



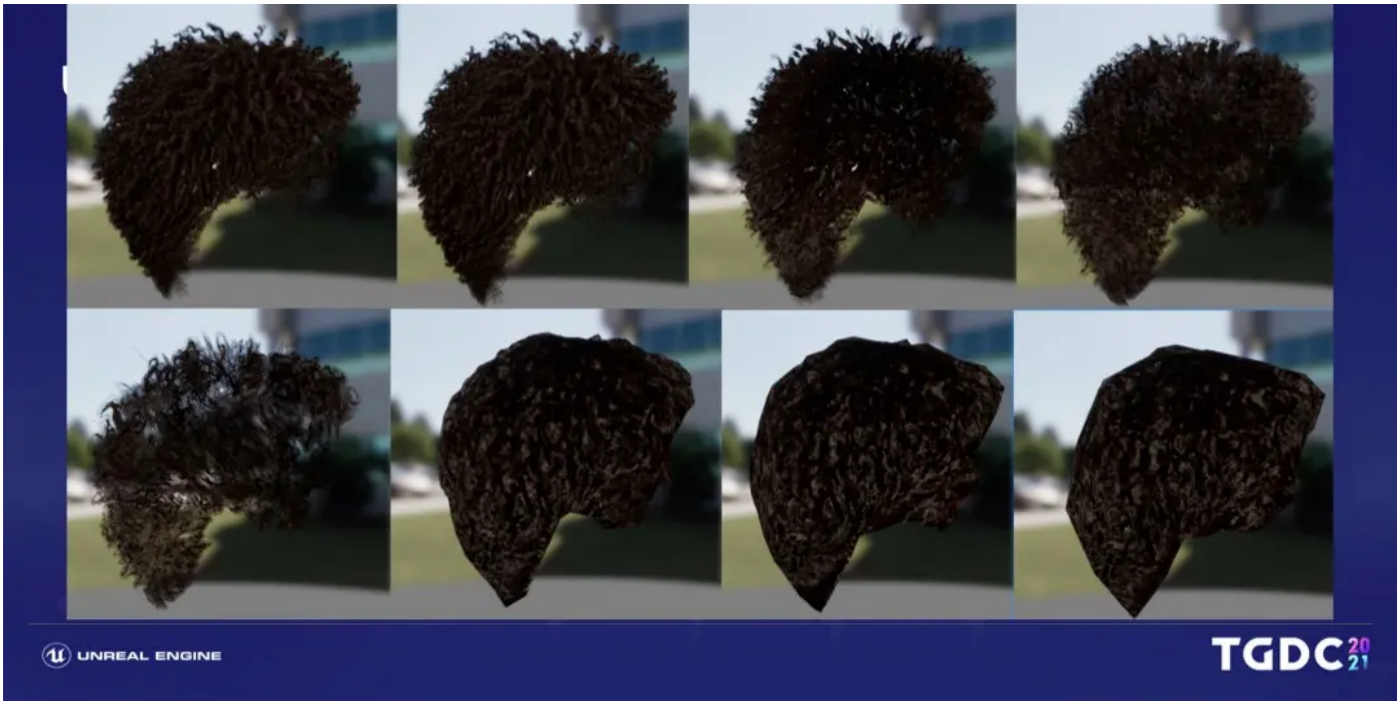
毛发在效果上是有一定特殊性的，他基本上来自于两个大的功能，一方面来源于毛发的各向异性特性，另外一个是层层毛发会具有一定的透射散射，那为了改善移动端的效果，毛发材质中为移动端单独增加了各向异性的效果，就是现在截图中的这一段，至于毛发透射散射的部分，因为移动端只采用了一级LOD采用插片毛发，其余都是Mesh毛发，在mesh毛发上比较难以表现毛发间细腻的透射散射，所以综合考量，我们并没有在移动端去处理这块。

刚才其实是多种shading model，但是都写在同一个材质中，这其实是源于引擎最近几个版本提供的新机制，材质系统并不锁定整个材质的shading model，而是把shading model作为一个参数去进行

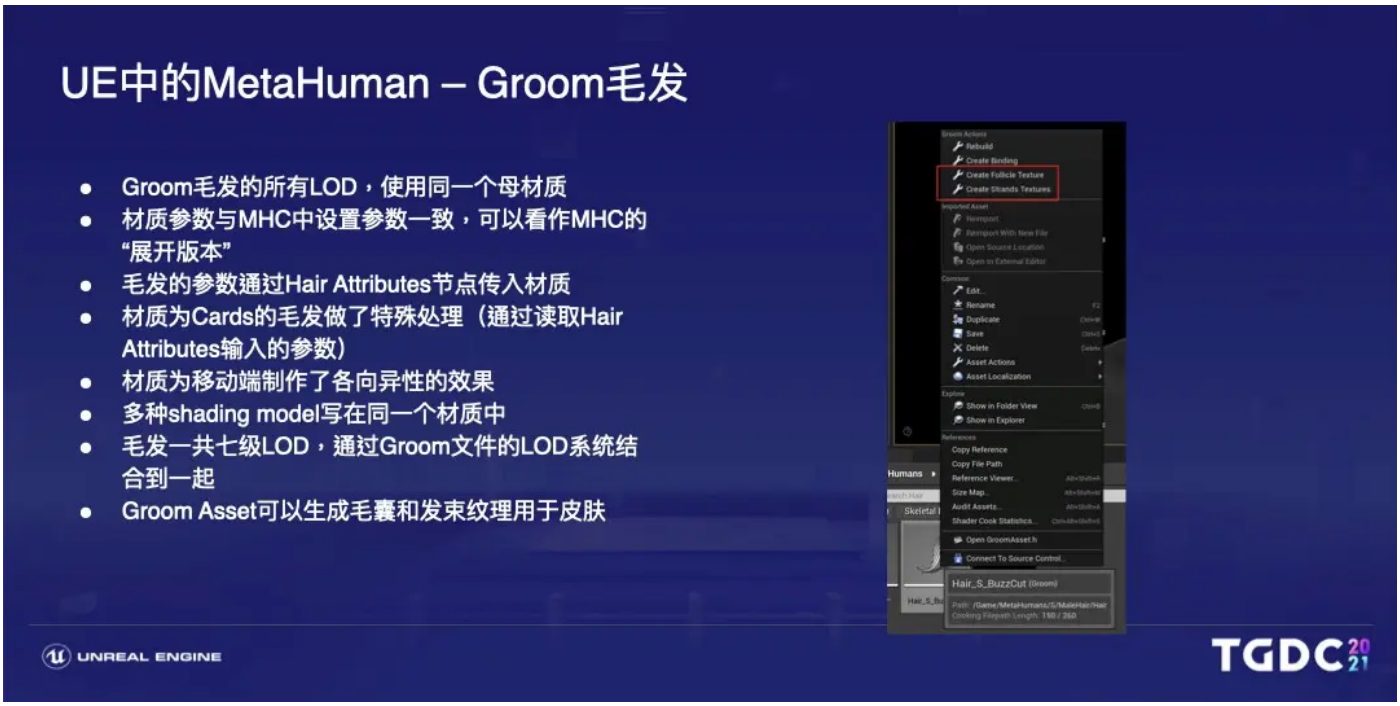
一个输入，然后不同的material attribute输入到不同的feature level下，这样子就可以做到一个材质实现针对不同平台，按照不同的shading model去进行渲染。

毛发的多级LOD，Groom, 插片，mesh，这些不同形态的资产是通过Groom文件中的LOD系统结合到一起，通过LOD分页，可以对这些LOD进行统一管理。

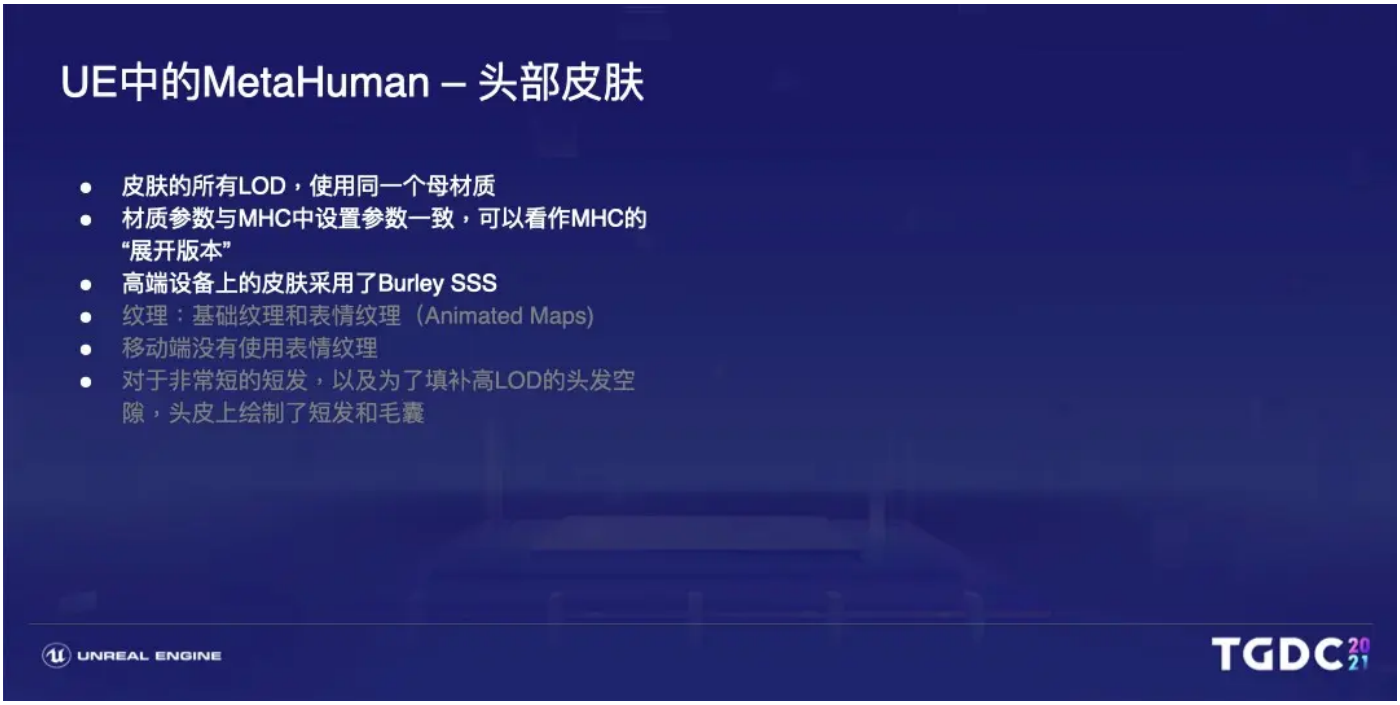
这是MH毛发的LOD数据，LOD 0–1 使用strand based hair，一般用于高端平台，比如说次世代主机和高端PC，LOD 2–4 基于插片毛发，LOD 5-7 是基于简化的mesh毛发。



这是MH中一个角色的毛发LOD展示，我们可以看到其实有几层因为面数比较少，所以会变得稀疏。



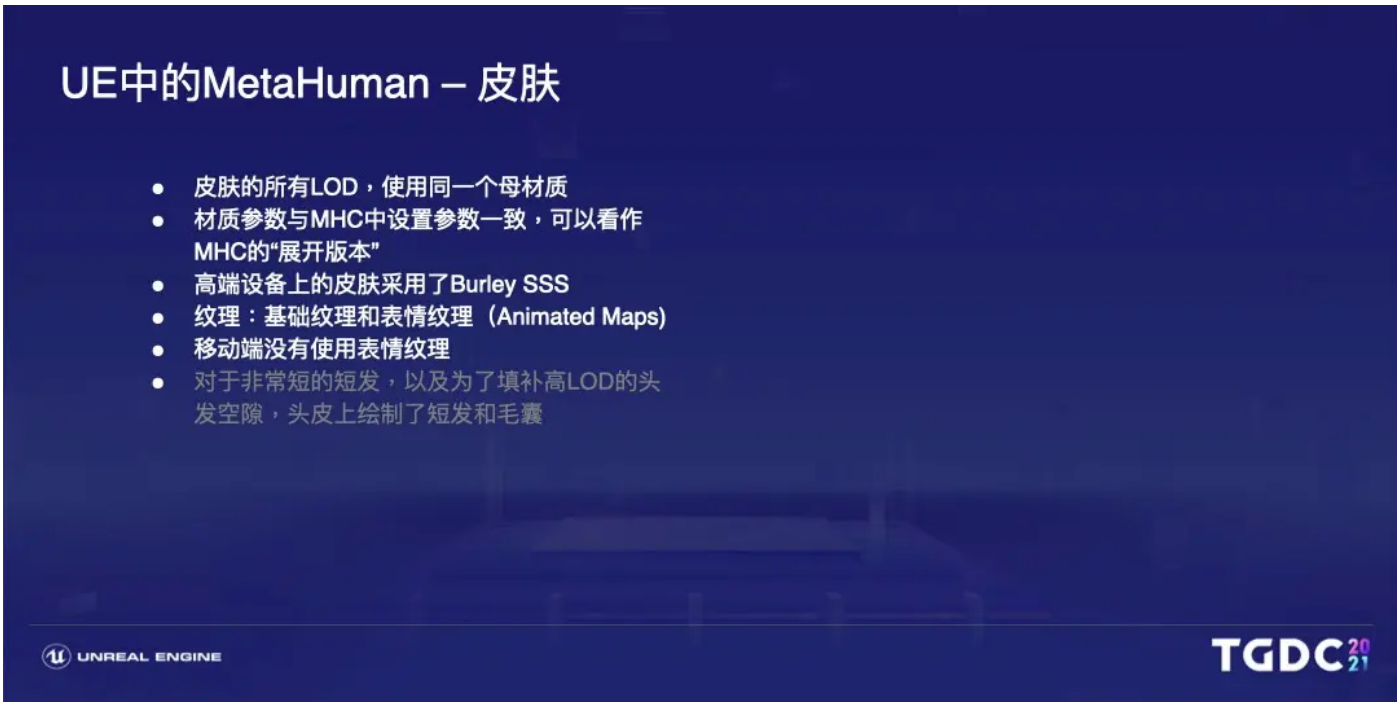
这个解决方案之一是使用遮罩纹理放在头顶皮肤，引擎也提供了对应工具，就是说右键groom asset, 可以生成毛囊纹理，此外，对于非常短的短发，其实并不适合创建插片或者mesh来做为LOD，这个时候也可以通过groom asset生成发束纹理，等会讲到皮肤，我们也会提到这张发束纹理。



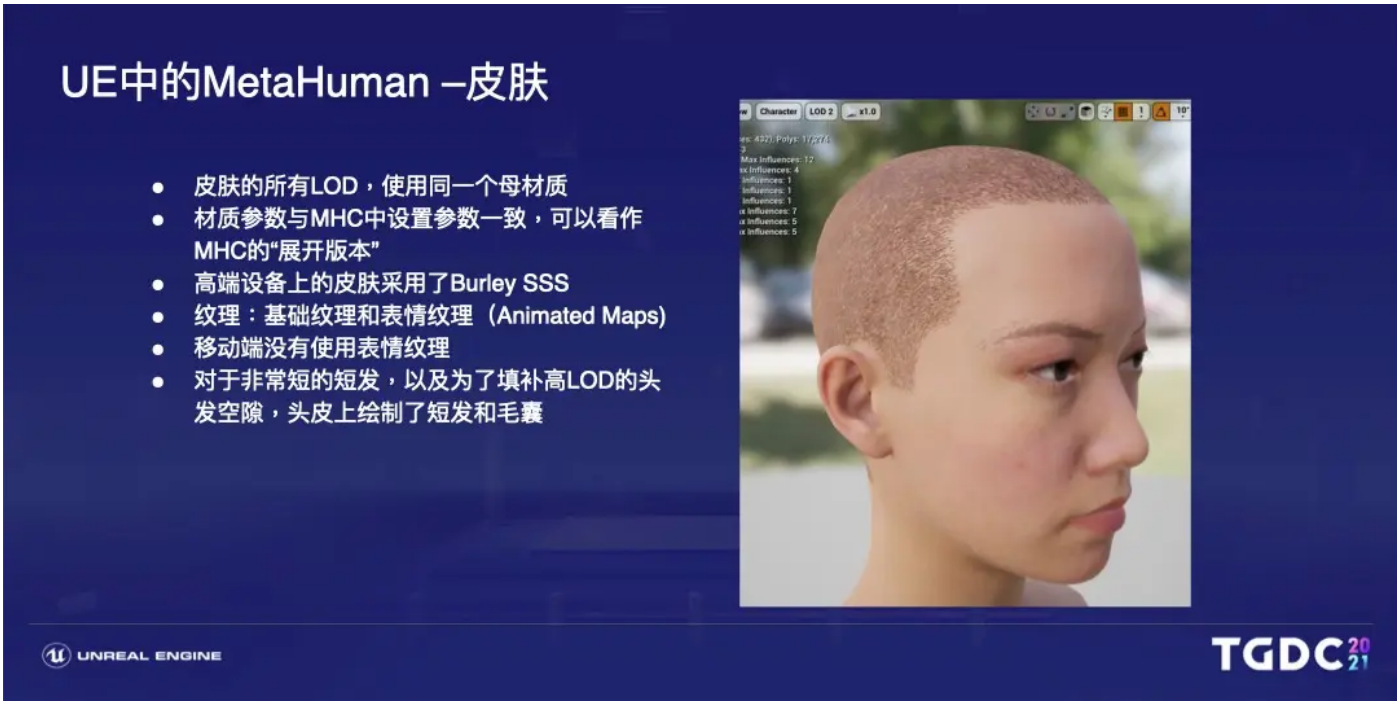
皮肤和头发的设计是有一些类似的：比如只有一个材质资产。材质属性与MHC中是一一对应的，在中高端设备上，皮肤的渲染模式采用了Burley SSS，移动端因为没有这样的shading model，所以会自动回退到Default Lit。

皮肤的纹理方面，特殊点在于有一套表情纹理，是一个基础姿态和三个表情姿态，通过在做表情时融合表情纹理，它可以提高整体的细节效果。

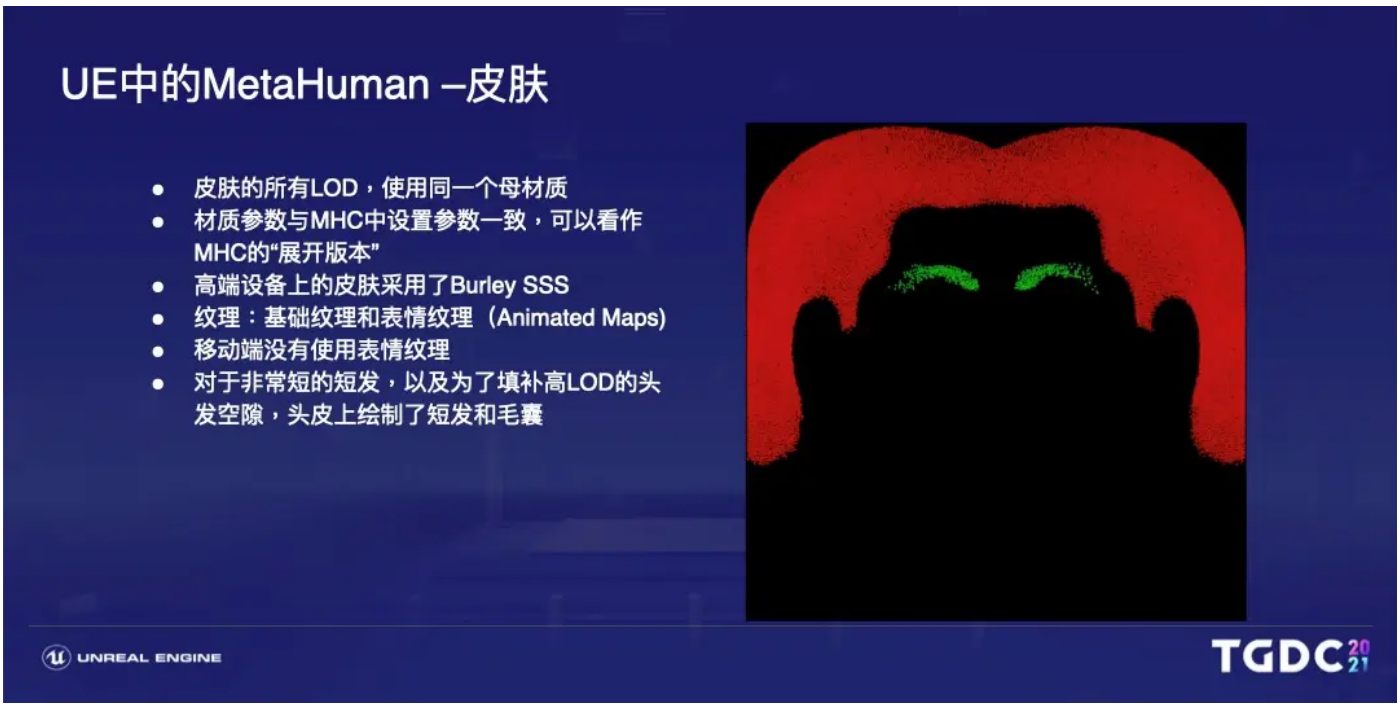
表情纹理主要用于描述皮肤在表情比较大、皮肤褶皱比较深的时候的形态，比如左图是基础姿态,右图是努力抬眉毛或者额头，有明显抬头纹时的形态。



可以看到，表情纹理会用到大量的纹理数，因此对于带宽和渲染都会造成一定压力，移动端是没有使用表情纹理的。



对于皮肤，除了绘制皮肤本身的效果和妆容信息，有时候还需要绘制毛囊或者发束，比如说演示的这个角色，头发非常短，头发是没有制作插片毛发或者Mesh毛发而是把短发的发束纹理，绘制到了皮肤的头皮部位。

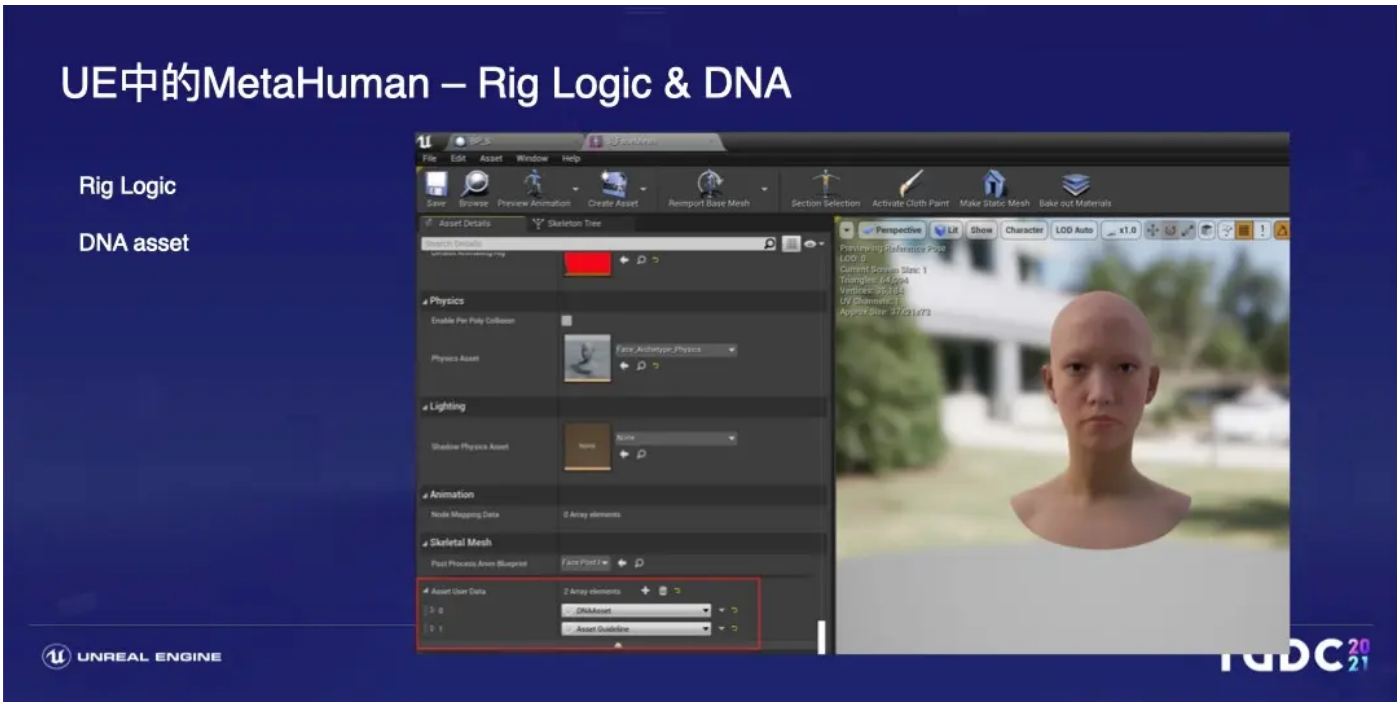


头皮上绘制短发和毛囊的这张纹理，就是通过刚才提到的Groom asset生成的发束纹理。

上面介绍的groom毛发，还有皮肤都是引擎中已有的机制，可能很多用过虚拟引擎开发的同学都已经非常熟悉了，下面着重介绍下表情。

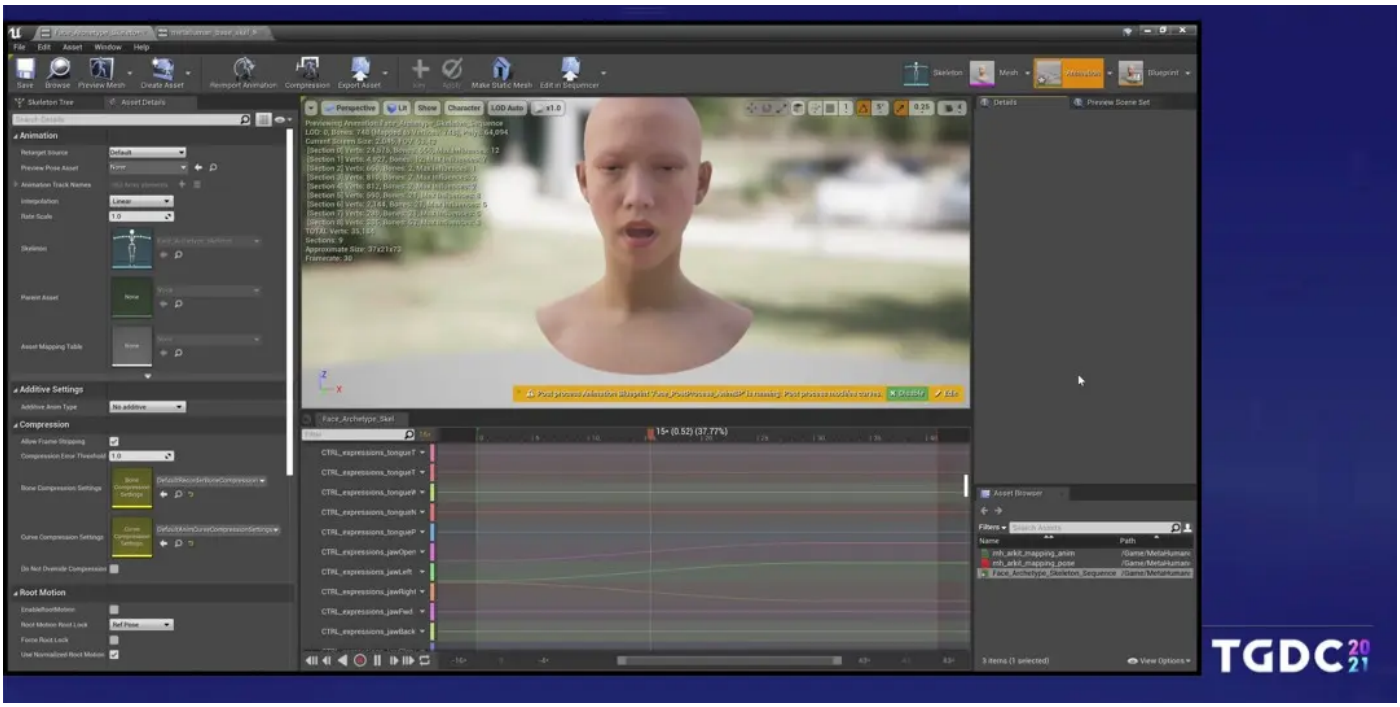
为了实现数字角色真实可信的表情，metahuman采用了骨骼形变，blendshape材质中加入表情纹理的混合方案，表情本身来自于骨骼形变，而blendshape用于补充细微肌肉细节，表情纹理主要用于加强皮肤出现褶皱时候的表现，大家可能还记得，在metahuman面部LOD0有600多根骨骼，同时又要保证不同LOD的表现基本一致，还要控制这三套参数，整体加起来是非常复杂繁琐的。所以我们采用了来自3Lateral的Rig Logic系统，Rig Logic是一个非常非常复杂的规则集，可以从简单的输入来驱动数以千计的Blendshape和骨骼，表情纹理，Rig Logic运作需要一些模型的额外信息，比如说模型的特征数据，以及模型的rigging行为，提供这个数据的，就是我们在一开始提到的每个模型独有的DNA文件，DNA文件用于描述角色的外观和骨骼绑定，基于这些，即使是不同的metahuman面

部的骨骼，蒙皮，面数等信息有所差异。也可以通过使用这样一套的RigLogic，用一套驱动数据驱动所有metahuman的表情，你不需要对每一个角色都去重新的制作。



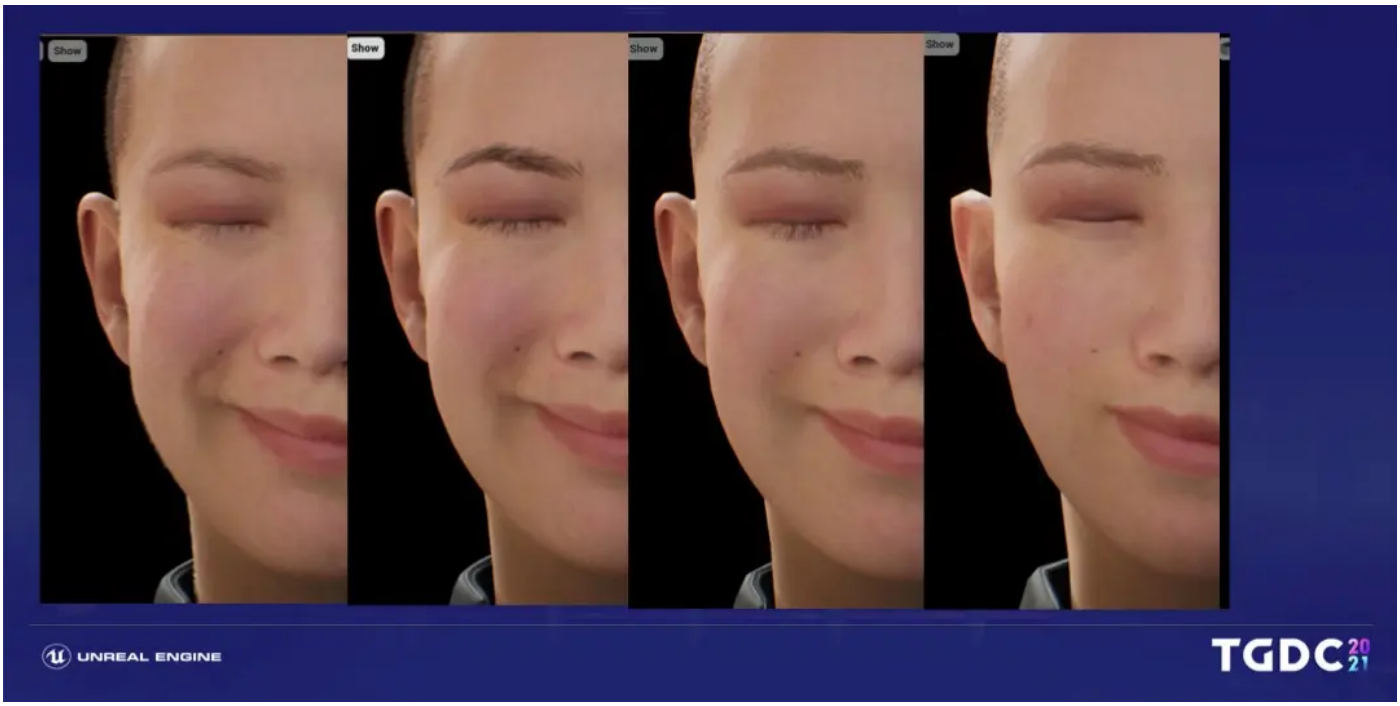
如果你在引擎中打开模型头部，可以看到details中有指定了一个DNA asset文件，这个文件来源于你在捏脸的时候，根据捏脸的数据生成的，DNA 文件是与捏的MH一一对应的关系。另外如果你把metahuman导出到Maya中，也会看到提示，是否要安装RigLogic，包括指定这套DNA。

RigLogic驱动表情采用的是RigLogic表情曲线，每个RigLogic表情曲线, 对应一组骨骼和blendshape还有表情纹理。

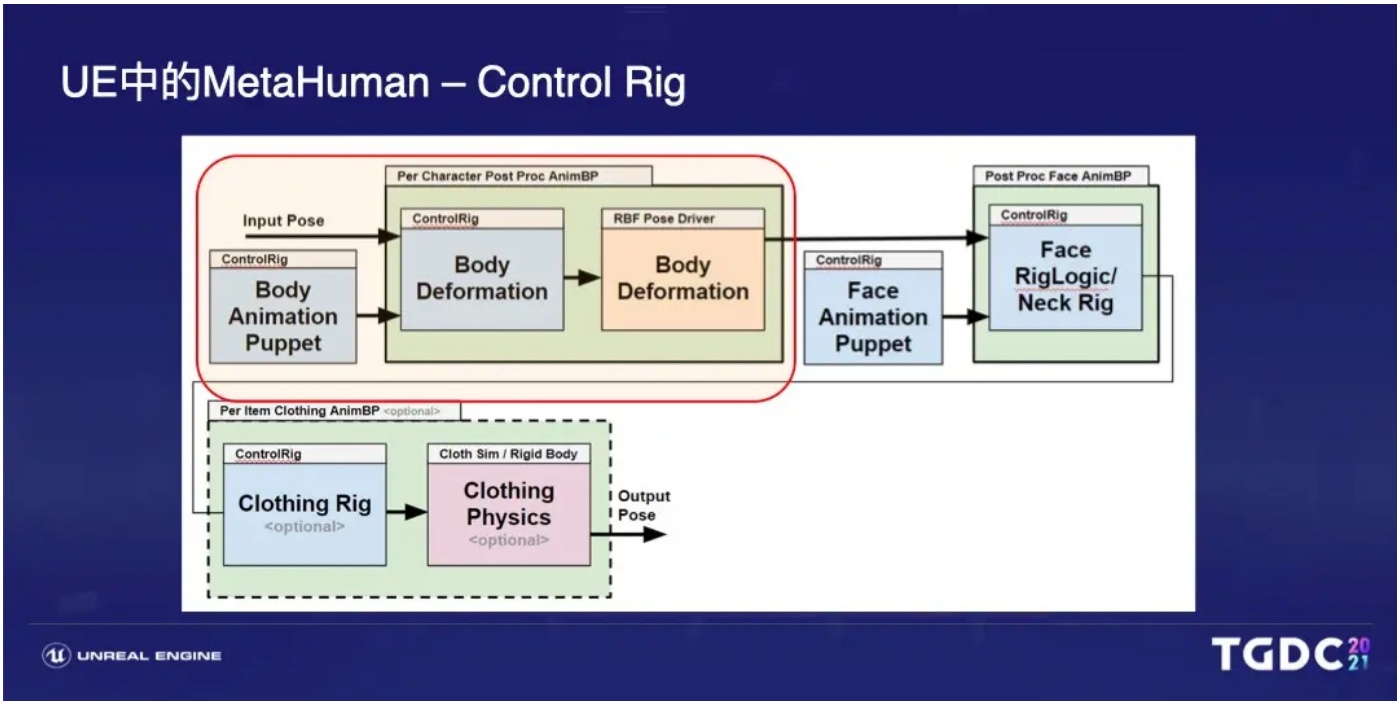


比如我通过sequence生成这样一个打哈气的表情，生成好之后是基于你K的数据，生成了RigLogic表情曲线，而RigLogic表情曲线再进一步驱动骨骼，blendshape，以及你的表情纹理，并且这套曲

线可以在动画中二次调整，你可以看到我刚才改变这条曲线，模型会随之发送改变，另外Rig logic表情曲线在runtime下也可以做出调整。

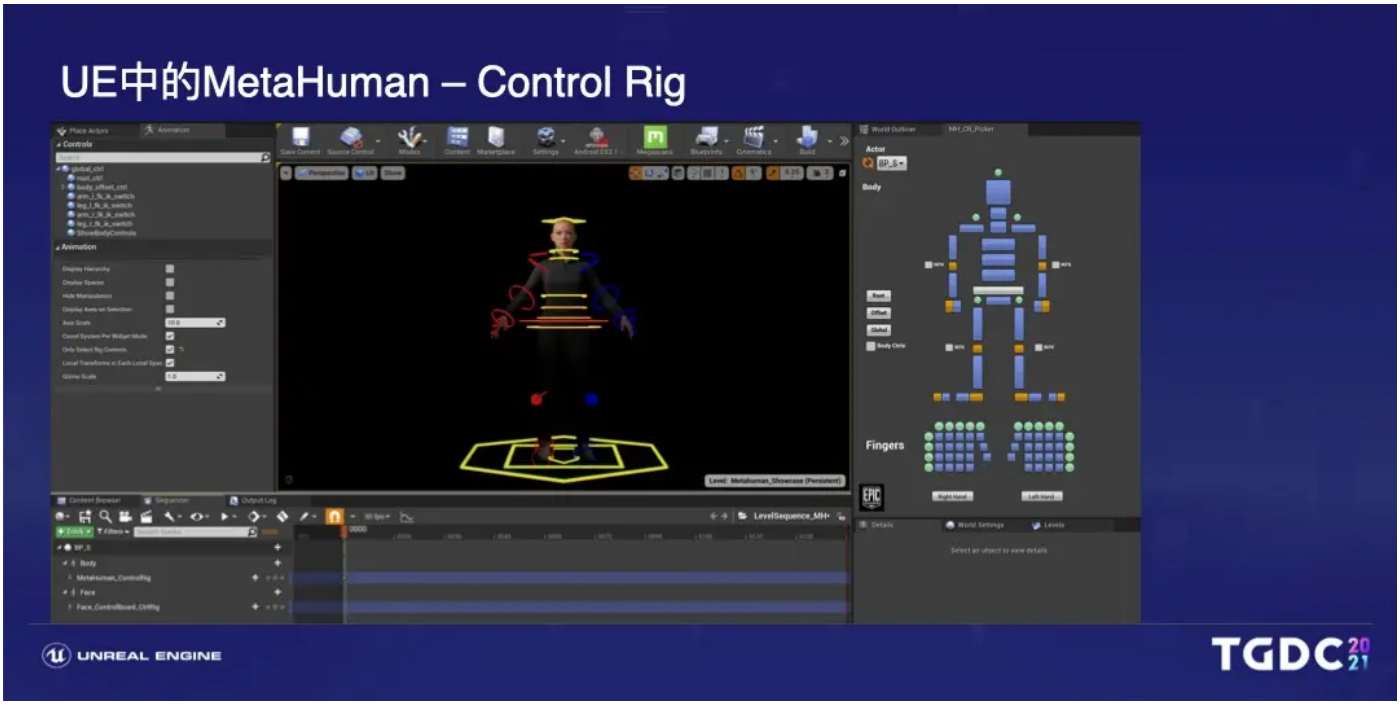


刚才提到面部表情是由骨骼、blendshape、面部表情纹理共同构成，对于基础的面部表情都是基于骨骼的blendshape贡献细微的肌肉变化。表情纹理贡献做表情时皮肤的褶皱，对于面部表情的品质，这里做了个简单的对比，从左到右分别是LOD0，1，2，4，除了面数和纹理本身精度的降低导致的影响外，表情方面，最左边是包含所有信息的表情，也就是说骨骼、blendshape、表情纹理这些所有信息的表情，第二张去除了blendshape的贡献，第三张去除了表情纹理的贡献。那么可以看到，对比起来，第二张和第一张的区别不是那么大，只在眼周有一点点微小的变化，而对于第三张和第二张的对比其实差别是蛮大的，她的鱼尾纹都会消失掉，也就是说对最终品质而言，大多时候表情纹理的贡献是大于blendshape的。第四张是相对于第三张做了进一步的减面，本身的面数只有1400面，同时也没有表情纹理的改善，在表情的表现上会弱一些。那么在引擎中制作表情，不管是通过录制或者是手调，都是通过Control Rig操作RigLogic表情曲线来做到的，我们来看下Metahuman中的Control Rig。



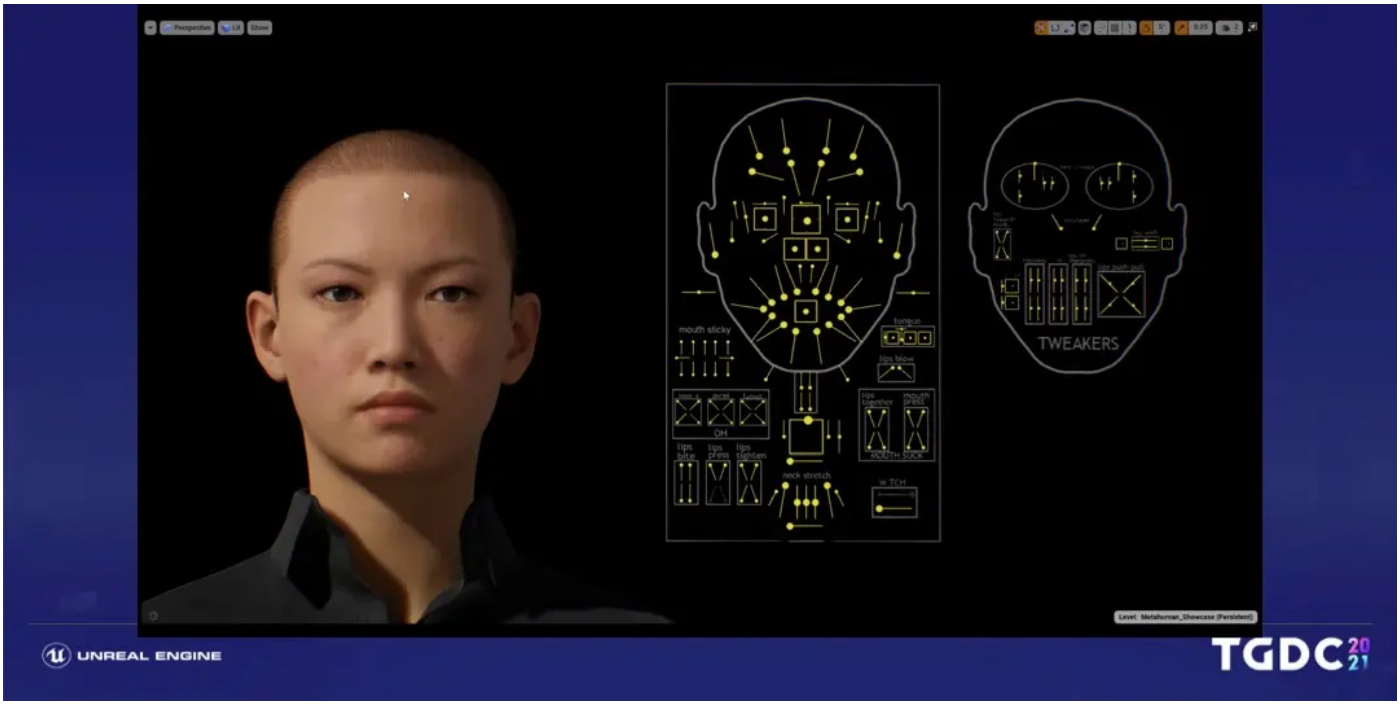
Metahuman中，Control Rig可以认为是分为了3个绑定分层，首先是木偶绑定，也就是MetaHuman的第一个绑定分层，用于动画师通过Control Rig来创造动画和表情的。

身体和面部的Control Rig都位于Common文件夹下，打开可以看到完整的ControlRig实现，身体Control Rig是比较常规，是直接驱动骨骼，而面部的Control Rig则是驱动Rig Logic 表情曲线。

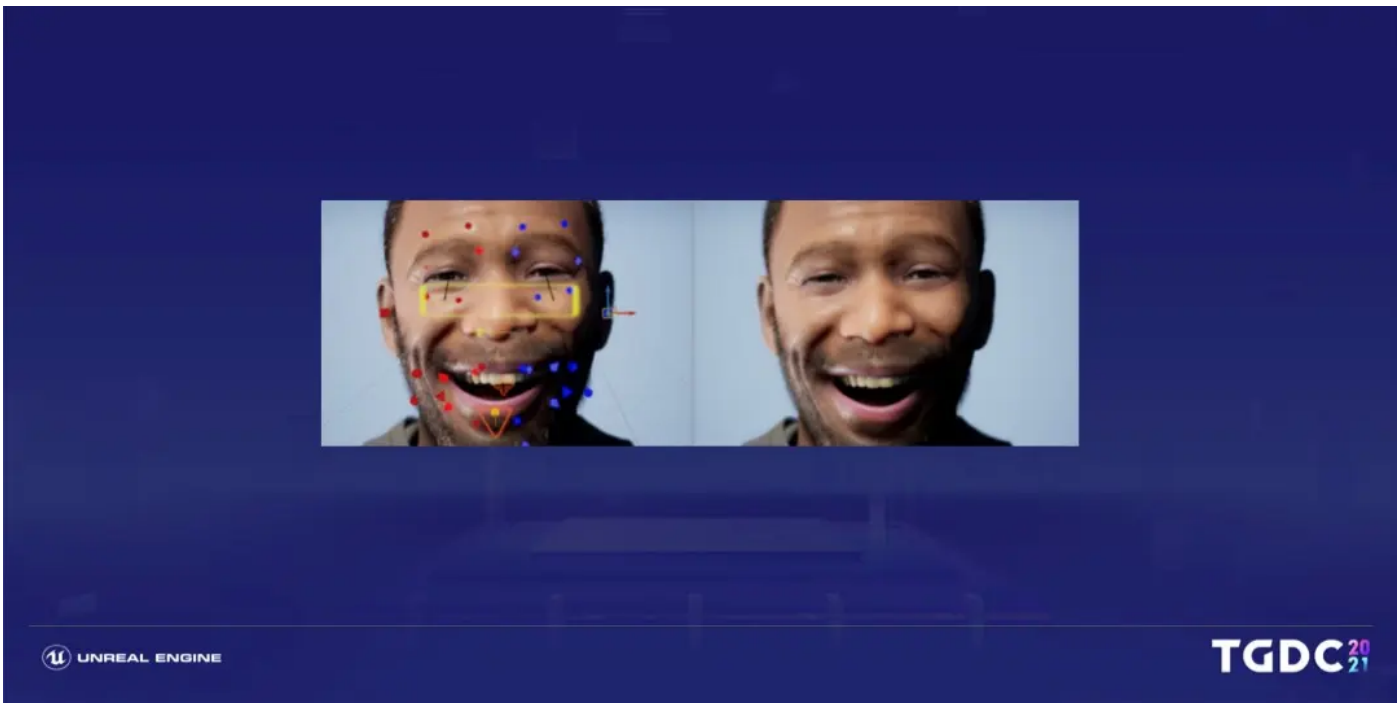


当然对于动画师而言，不太需要关注Control Rig的实现，实际需要的还是一套好用的K动画工具，对于动画师在引擎中K动画，使用方法也很简单，就是直接把角色蓝图或者角色拖到sequencer中，因为模型身上有Control Rig信息，就会自动出现身体和面部的Control Rig界面，对于身体而言，你还可以用Metahuman中预设好的Editor Utility Widget，快速选择Control Rig以及切换 IK FK。

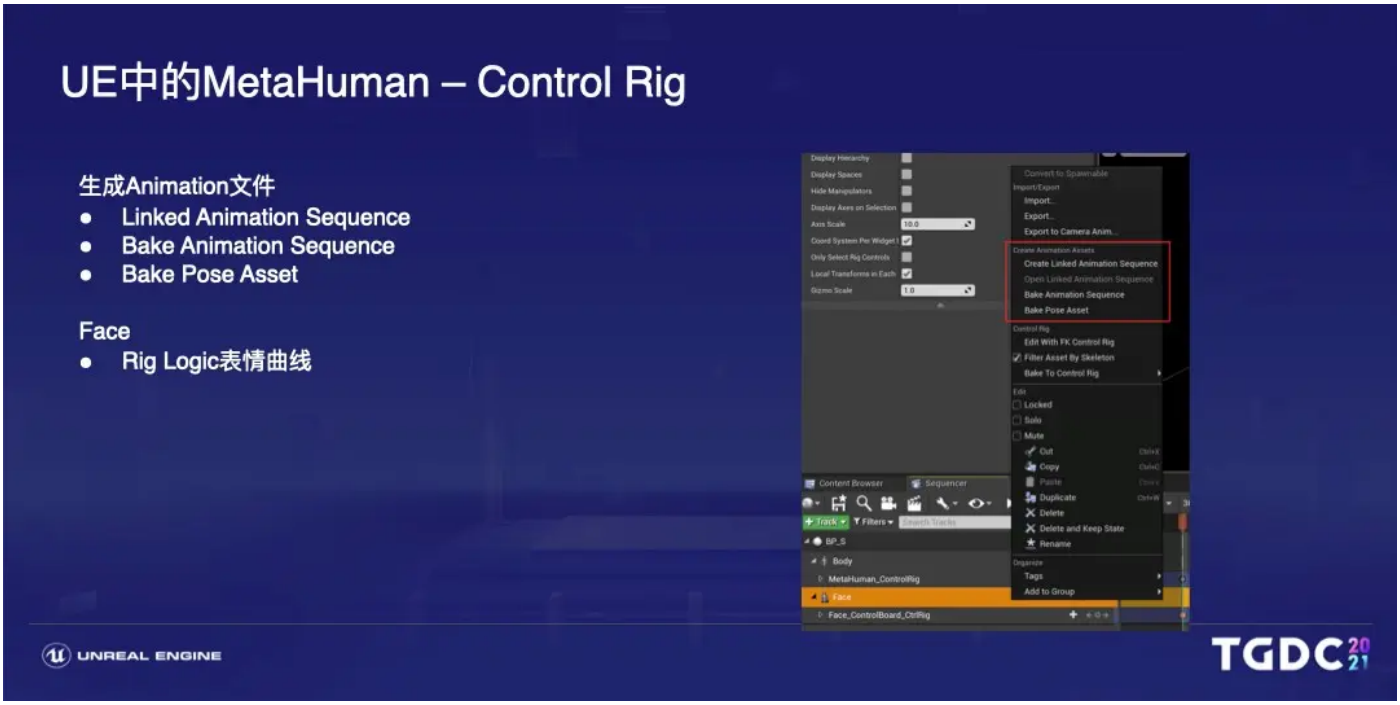
对于面部Control Rig，Face Control Rig中分为三块，一块是用于面部捕捉的backwards solve，一块是forwards solve，也就是用于我们在sequencer中K动画，另外还又一个比较小的模块是setup event中添加了一个眼部关注点的模块，因为刚才介绍了面部表情是非常复杂的，面部的Control Rig是驱动Rig Logic 表情曲线，Rig Logic这套系统会根据模型对应的DNA asset信息，把Rig Logic 表情曲线映射到blendshape和面部骨骼以及动画纹理上。



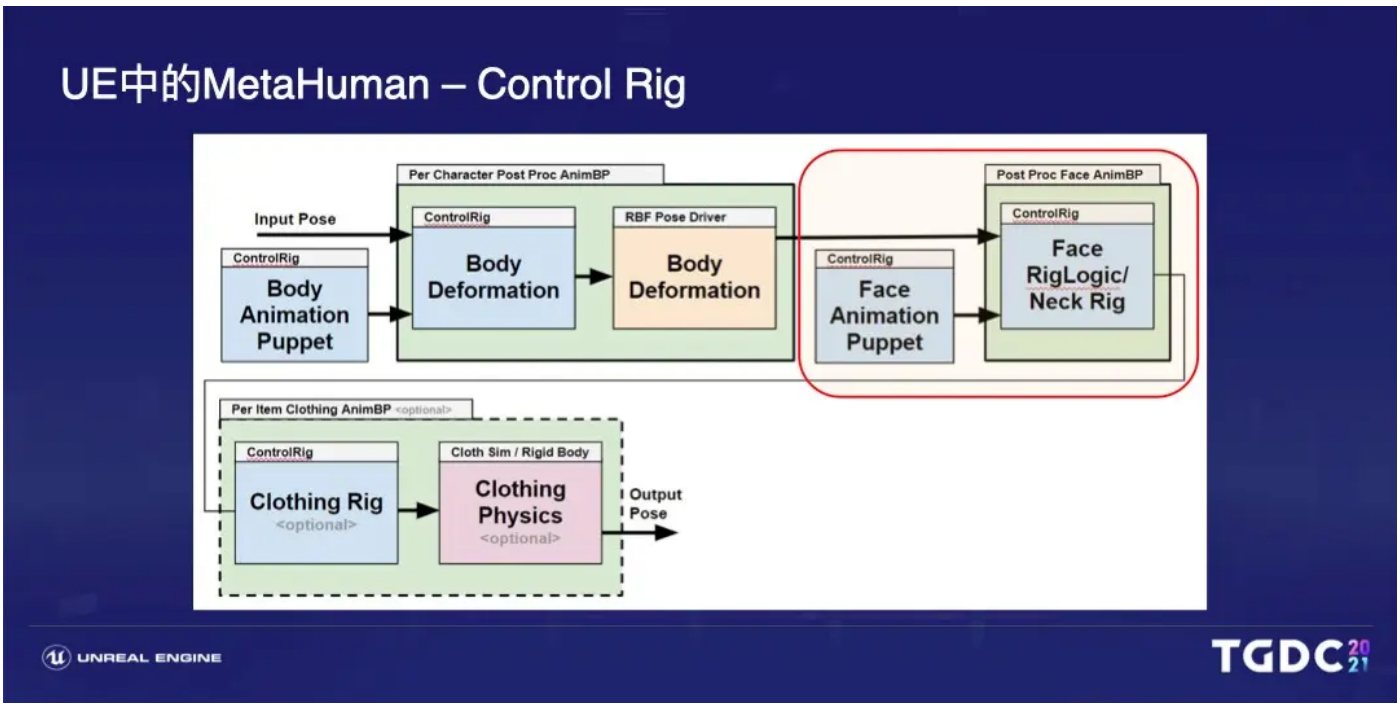
对于面部我们提供了一套预设好的Control Rig面板，这里做个简单的操作演示，在这里每个控制器对应一组肌肉群，另外面部Control Rig提供了watch的小模块，你的眼睛会始终盯着这个模块，在模块移动时不止眼球会不断发生转变，眼周的肌肉也会做出正确的改变。



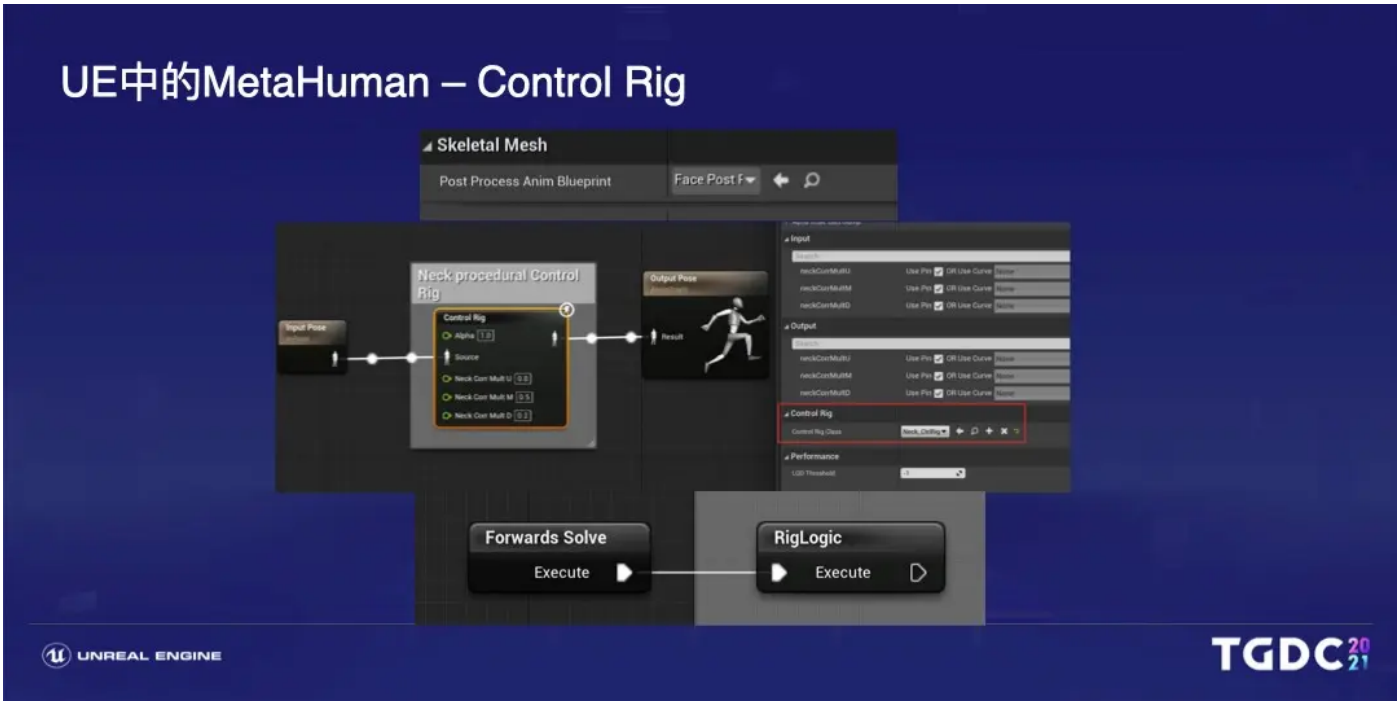
另外我们也提供了一套简易的面部Control Rig，直接依附在面部上，这样可以更直观的进行表情制作。



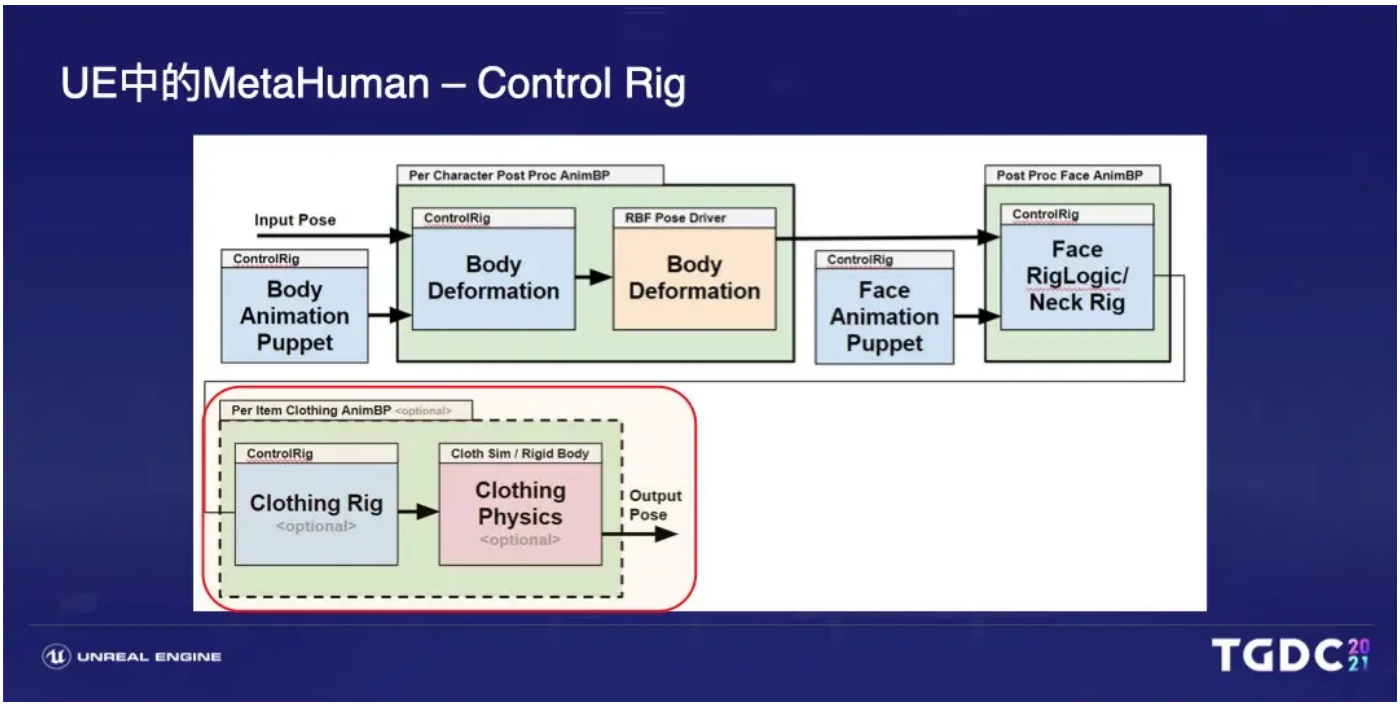
如果你是为了在游戏中使用，在Sequencer用Control Rig作好动画后，为了性能考虑，最好把K好的数据以动画文件生成出来，引擎默认提供了这样的工具，右键模型，可以生成animation文件，身体的animation保存是骨骼的transform信息，而面部的animation, 保存的是rig logic表情曲线。



接下来是Control Rig的第二个绑定分层，变形绑定： 变形绑定用于驱动所有的旋转和结构关节，模型身上会指定post process animation blueprint，运行一套用于runtime的Control Rig。



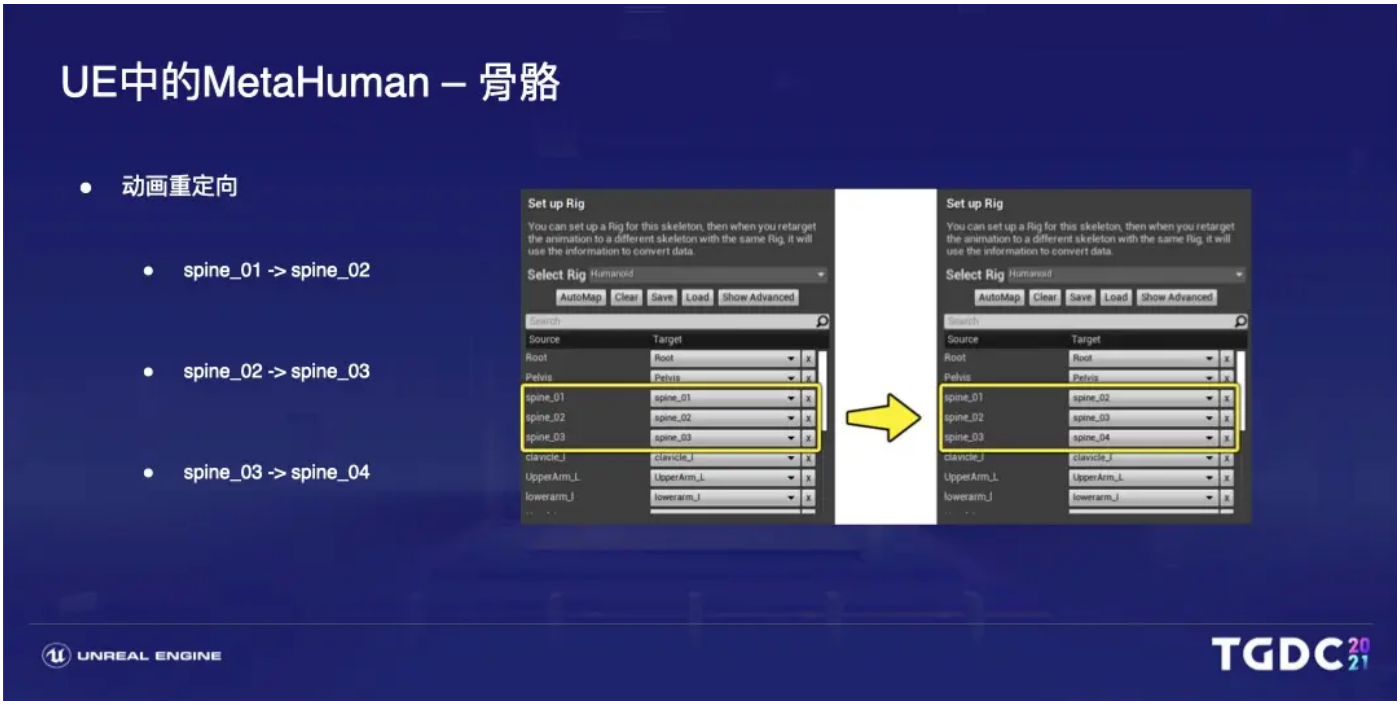
以头部为例，使用了RigLogic在runtime运行，这里允许各种动画接口驱动或者运行RigLogic表情曲线，形成最终的面部表情动画。



最后一个绑定分层是服饰，服饰主要是采用Control Rig来去修正表现，但也不是每一个服饰表现都会用Control Rig，比如说连帽衫的帽线和脖子后方的帽子褶皱，其实是采用了Rigid Body作为物理模拟方案。

在身体骨骼方面，为了表现更真实可信，相对于之前大家比较熟悉的小白人，MetaHuman的身体骨骼方面增加了一些额外的关节，增加的这些关节大多数是肢体末端的叶子关节或是辅助关节，基本只会影响到表现，但是需要额外注意的是增加了额外的脊椎关节，这样会导致骨骼层级有所变化，所以如果要复用小白人制作的动画资产的话，是需要做一个重定向的操作。

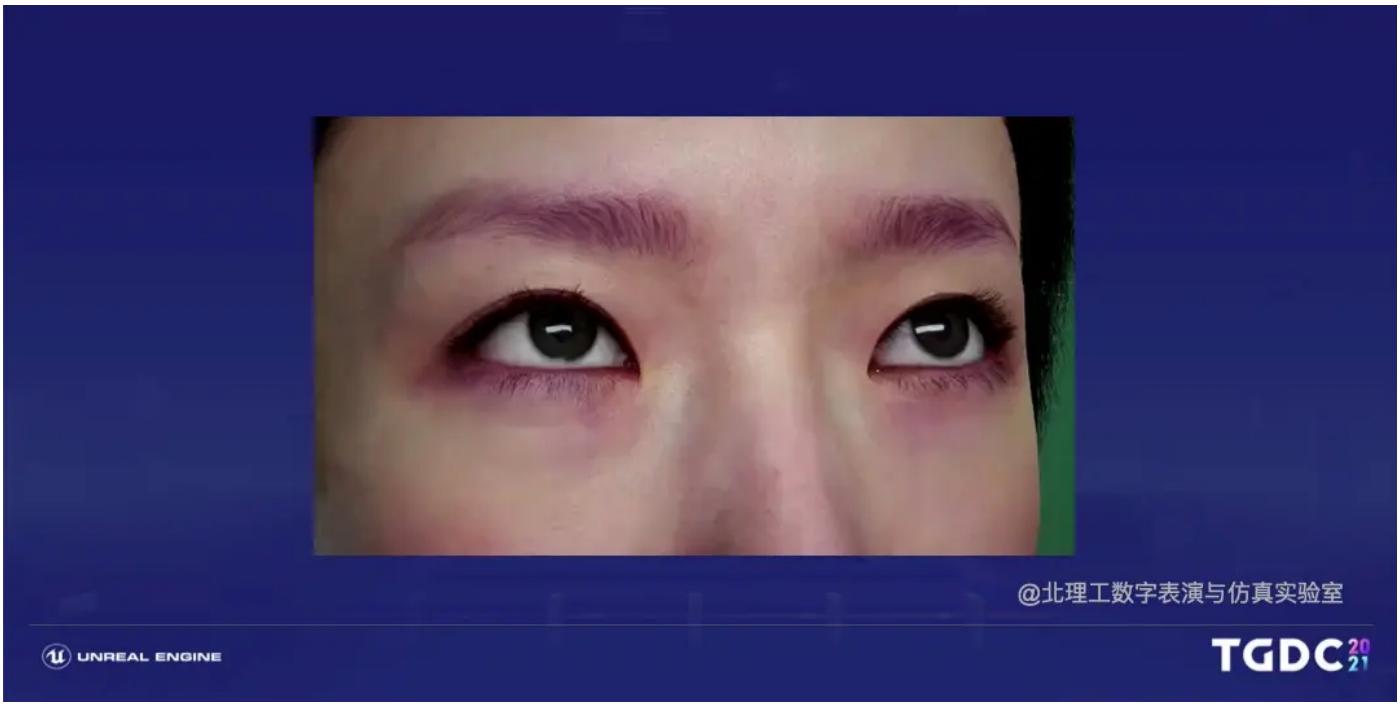
对于肢体末端关节，我们对于手指的骨骼进行了一个细化，另外增加了新的脚趾关节。



刚才提到如果你想要让metahuman复用商场中基于小白人制作的动画，你需要重定向的操作，重定向的话，你是需要重新制定脊椎骨骼的映射关系，对于其他的新增骨骼部分，因为小白人没有对应骨骼，就会跳过这节骨骼的计算，所以并不会受到影响。

最后还有两点补充，表情DCC只能使用maya，而不能使用motion builder和3Dmax，这其实是因为3Lateral自己内部一直使用的maya，也就只有针对maya的RigLogic插件，所以如果需要在DCC中制作表情，因为依赖于rig logic，所以只能使用有rig logic插件的Maya。其次是我们建议把Maya作为动画制作工具，而不是修改工具，我知道，其实很多用户希望能去二次修改metahuman,我们实际上是不推荐这么做的，原因在之前也提到过，修改模型可能会让你模型的DNA文件是没有办法做到完全匹配的，从而rig logic驱动的表情依赖于DNA，但是他通过这样的驱动的话，会出现一些不那么匹配的情况，所以在有一些表情下边会看起来有些假，所以我们推荐的流程是只有Maya作为动画制作

工具。当然我也知道现在社区中已经有很多人做过尝试，用RigLogic驱动自己的模型，可以看到真人的局部和非真人的效果还是很棒的，这里挑选了两个来自中国开发者的分享。



这是来自北理工数字表演与仿真实验室的尝试，他们有一套自己的扫描角色库。

这是David Jiang的分享，他使用的是自己创作的兽人形象。但是这里依然要强调，其实我们并不推荐这样的流程，可以给大家分享的是我们其实也在内部感知到客户的这样的需求，所以我们也在积极的尝试这件事，不远的未来会给大家提供一套更灵活的捏脸包括二次修改的一个机制。

谢谢大家，今天我的分享就到这里。

Q & A

Q：为什么Metahuman Creator捏不出明星脸的亚裔女性？

孙丹璐：因为Metahuman Creator中的捏脸，取决于数据库中扫描的人脸数据，目前虽然数据库中亚裔是占了四分之一，但是因为疫情原因，整体数据库的总量并不是特别大，所以有些脸型或者面部特征并没有收录到库中，那么这些脸型和面部特征就比较难捏出来了，那么基于此，其实在应用方面，尤其是游戏行业的应用，我们更建议把Metahuman Creator用于捏大众脸的NPC上面，目前 我们第一优先级就是扩充数据库，并且我们会征集这些数据来源覆盖各个行业与民族，争取添加更多类型的数据特征到这个库中。

Q：Metahuman Creator为什么不建立本地库？

孙丹璐：其实这个问题在刚才的演讲中已经简单提到过了，先是这个数据库 数据库中的数据是非常非常庞大的，你在捏脸的时候需要处理的数据量也非常大，那么 考虑到你的硬盘或者是内存 cpu和gpu的运算压力，它都是非常大的 对于电脑的配置要求是很高的，我们的期望是让每个用户都能比较低门槛的去打造数字人，这个低门槛当然也包括了硬件设备，另外就是Metahuman还是一个相对早期的状态，它的更新迭代是比较快的，直接放在云端可以避免每个用户去频繁的进行更新，主要是出于以上这两个考虑，我们最终把Metahuman Creator放在了云端。