骨骼动画系统学习总结

骨骼动画是进一步的动画类型，原理构成极其简单，但是解决问题极其有优势。将模型分为骨骼Bone和蒙皮Mesh两个部分，其基本的原理可以阐述为：模型的骨骼可分为基本多层父子骨骼，在动画关键帧数据的驱动下，计算出各个父子骨骼的位置，基于骨骼的控制通过顶点混合动态计算出蒙皮网格的顶点。在骨骼动画中，通常包含的是骨骼层次数据，网格Mesh数据， 网格蒙皮数据Skin Info和骨骼的动画关键帧数据。

骨骼动画原理

核心: 通过骨骼带动皮肤运动，也就是通过骨骼的移动动态计算mesh上的点的位置

过程：

1.将mesh上的点转换为骨骼空间上的点。 骨骼空间就是以关节为原点确定的空间，并不是一个实体。 2.通过缩放、旋转、平移将骨骼移动到新的位置。 3.根据骨骼的新位置计算mesh顶点新世界坐标（骨骼移动，但mesh顶点与骨骼的相对位置不变，所以产生了顶点随骨骼移动的感觉），若一个顶点被多个骨骼影响，则要进行顶点混合计算新世界坐标。

蒙皮计算的过程： **顶点在模型坐标系中位置 -> 骨骼空间中位置 -> 变换后的骨骼空间中的位置->模型坐标系中的位置 -> 世界空间中的位置**

Unity3D骨骼动画处理

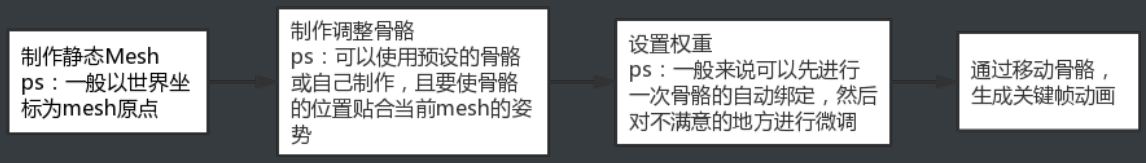
骨骼动画中的骨骼变换，蒙皮的计算，都是在CPU中进行的。在实际的游戏引擎中，这些都是分开处理的，较为通用的处理是将骨骼的动画数据驱动放在CPU中，计算出骨骼的变换矩阵，然后传递给GPU中进行蒙皮计算。在DX10的时候，一般的shader给出的寄存器的大小在128的大小，一个变换矩阵为4x4，如果去除最后一行(0,0,0,1)就可以用3个float表示，那么最多可以表示，嗯，42个左右，如果考虑进行性能优化，不完全占用寄存器的大小，那么一般会限制在30根骨骼的大小上。将这些骨骼的变换矩阵在CPU进行计算后，就可以封装成skin info传递到GPU中。

在GPU的计算中，就会取出这些mesh上的顶点进行对应的位置计算，基于骨骼的转换矩阵和骨骼的权重，得到最新的位置，从而进行一次顶点计算和描绘。之所以将骨骼动画的两个部分分开处理，一个原因就是CPU的处理能力相对而言没有GPU快捷，一般一个模型的骨骼数量是较小的，但是mesh上的顶点数量较大，利用GPU的并行处理能力优势，可以分担CPU的计算压力。

在DX12之后，骨骼变换矩阵的计算结果不再存储在寄存器中，而是存储在一个buffer中，这样的buffer大小基于骨骼数量的大小在第一次计算的时候设定，之后每次骨骼动画数据驱动得到新的变换矩阵，就依次更改对应的buffer中存储的变换矩阵，这样就不再受到寄存器的大小而限制骨骼的根数的大小。但是实际的优化中，都会尽量优化模型的骨骼的数量，毕竟数量越多，特别是影响顶点的骨

骼数量越多，那么计算量就会越大，正常的思维是优化骨骼数量而不是去扩展buffer的大小

制作骨骼动画流程



结果这几步操作后，我们分别得到了这些数据： 1.每个皮肤顶点的初始世界坐标。 2.每个骨骼关节顶点的初始世界坐标。 3.每个顶点被骨骼顶点的影响信息。 4.骨骼如何移动。

Unity3D骨骼动画制作流程

1. 资源下载
2. 资源导入

   直接将资源拖入Unity中即可，可以看到在Unity中生成了一个文件夹和一个预制件。
3. 加入动画

   1.把模型prefab拖入场景中。 2.然后将mixamo.com动画拖到场景中的Samba Dancing中，Unity会自动生成对应的Animator Controller。
4. 运行场景，查看动画效果
5. 直接点击运行即可。