方案总结

使用工具（孙宇康）

包括：动捕文件的转换、动画文件的提取、为动画片段打Tag

工具的开发上，最主要的问题就是使用对象：你对他一无所知，不知道他的技术水平，也不知道他将会以什么方式来使用；所以除了操作流程必须尽量简单直白，符合一般人的使用习惯，最好还要有适当的引导；接受输入的时候，相比起黑名单，使用更加严格的白名单筛查也能减少因为预料之外的输入带来预料之外的错误。

可优化

目前使用界面的提示还不够，可以后续补充

覆写了导入模型的方法，但没有研究出控制的开关，只能对整个代码文件进行注释

为动画打Tag的部分流程设计的不够符合自然使用习惯：可以从重绘Inspector改成EditorWindow的形式，动画改成用ObjectField拖动获取的方法即可

在可视化方面，可以为不同Tag设置不同颜色并实时显示到场景中，优化体验

动画数据预处理（郭畅）

难点

数据预处理部分并没有很多技术上的难点，更多地在于方案的选定以及许多逻辑处理和边界情况的处理。例如：选择什么方案处理动画数据，处理出来的数据以什么方式存储，如何将数据传递给其他模块，预处理过程中的各个部分如何在逻辑上分离，tag是仅给数据打标记还是将数据删除，配置的轨迹点在动画之前或之后如何处理，循环动画如何处理，混合动画如何处理等等。以及，如何获取到更准确的数据，与运行时预测产生的数据更契合。

因此这一部分代码的编写其实更多地是一个调试试错，对比方案和完善逻辑的过程。

可优化部分

对于前面提到的循环动画的处理，如果轨迹点落在循环动画之后（例如动画只有1s，轨迹点落在了1.5s），可以很方便的使用PlayableGraph.Evaluate向后窥探0.5s，记录下0.5s后的状态后再回退回来。但是如果轨迹点落在循环动画之前（例如-0.5s），再使用同样的方法就会产生严重的错误，不仅

会使当前计算的轨迹点得到一个错误的值，而且还会影响其余动画数据的处理。因此目前对于轨迹点为负的情况处理得到的数据并不很精确。

此外，目前对于轨迹点落在动画区间外的估计还仅仅是线性估计，如果是转向动画会有一定的误差。想到的方法是增加角速度的计算。

轨迹预测和高级运动跳跃（王思行）

难点

目前运动模型中阻力的影响导致的边界处理（比如速度降到0以下时反向，速度达到最最大值时的抖动） 注意：边界处理受帧率影响。

轨迹预测要按着动画轨迹来编写。建议先编写显示动画轨迹的测试系统，再根据场景中的动画轨迹（务必确认动画轨迹正确）来确认预测轨迹的编写（匀加速 匀减速 角速度 朝向的变化） 由于转弯等小动作较快，建议逐帧或降低timescale调试。

跳跃过程中跳跃轨迹的自然过渡。

可优化

预测轨迹中朝向变化目前是匀速变化，实际起点到目标点角速度变化是加速匀速再减速。

运动模型可换为临界阻尼模型。

跳跃结束后和后续匹配动作连接过渡不顺畅。建议编写过渡播放动画的debug窗口。

帧匹配（张亮）

难点

关节信息的获取

目前的方案是，从上一次匹配到的动画获取。

优点：不受动画过渡的影响，下一帧的匹配只和预测轨迹相关。

废弃的方案：在场景中记录人物每一帧的关节位置，计算出实时的关节信息。

原因：采用这种方案，会导致在不同过渡方式，不同帧率的影响下，得到的实际关节速度差异很大，同样的参数设置难以适配各种不同的运行环境。

可优化

动画数据处理时最后一帧（长度不足一帧）的处理

当前方案是直接丢弃最后这一部分动画，对于大部分动画来说没有影响。

缺点：对循环动画略有影响。

可优化的部分：如果选择保留，动画播放需要处理播放完后的处理，比如立刻进行匹配。

判断轨迹预测是否符合预期

虽然轨迹点（绿点）可以直观的看出轨迹预测的结果，但最终轨迹预测是要和动捕进行匹配的，如果轨迹预测的结果和动捕的轨迹接近，那么匹配难度就会降低，调参也比较容易。

建议：针对一个动画，对每一帧设计相应的用户输入（移动方向）；播放动捕时在场景中显示对应的预测轨迹与实际轨迹，以直观的查看区别；并且绘制一个曲线图，说明每一帧预测轨迹和实际轨迹的具体差距。

动画过渡和高级动画之翻越（周冰颖）

难点

PlayableGraph动画播放时，settime()函数会使动画从开始到指定时间偏移，从而产生位移，故过渡第一帧要单独处理

现解决方法：使用PlayableBehaviour的继承类，重载PrepareFrame(Playable playable, FrameData info)，在其中设置第一帧权重为0解决。由于多个update存在和帧的时长不一致，单纯在Update中设置权重为0，是会有偏差的。

可以优化

过渡方法只是简单地按比例减少

可以用更加好的过渡函数实现

动画控制器和PlayableGraph的混用

可以调用PlayableGraph的AnimatorControllerPlayable模块实现混合过渡，但是，AnimatorControllerPlayable没有MatchTarget()的函数

对于将被销毁的动画片段，若权重过大，直接撤销播放

可以给各个动画片段，设置不同状态，如设置阻塞状态，或者扩大混合数量