**使用AVFoundation实现视频倒序**

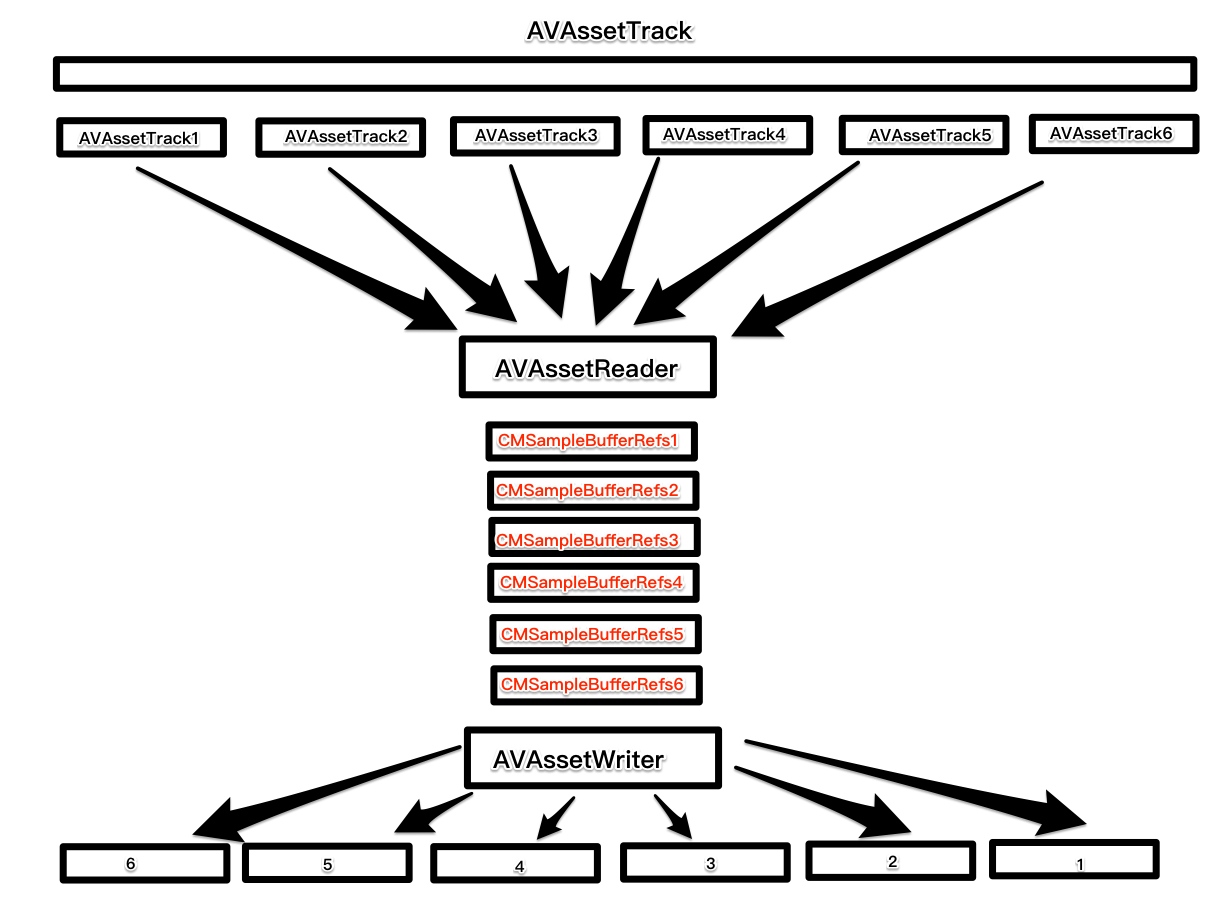
参考：https://blog.csdn.net/chengkaizone/article/details/53670409

https://github.com/KayWong/VideoReverse

#背景 最近在做一个视频类的APP，在视频的编辑模块有一个视频倒序文件的需求，这个和倒序播放的需求不一样，要求的是生成新的倒序视频文件 #研究过程 一开始原本以为这个需求很简单，但是实现的时候遇到了各种麻烦，在最初，本以为可以直接使用AVMutableComposition对Track直接反转的操作就可以实现视频的倒序了（看来我真是Too Naive了），然后发现AVFoundation没有提供这样的接口。后面通过各种Google、StackOverflow，还是让我找到了一些实现的思路，谢谢下面俩哥们的实现思路给了我极大的启发 <https://github.com/mikaelhellqvist/ReverseClip> <https://github.com/whydna/ReverseAVAsset>

**先讲一下视频倒序的实现原理：** 视频文件如果除去哪些杂七杂八的元数据，那剩下的就是一帧帧的视频图片了（视频帧的分类这里不展开讨论），简单的来讲，就是把视频文件里的每一帧按照相反的方向去排列就可以让视频倒着播放了。

大体的实现思路如下：

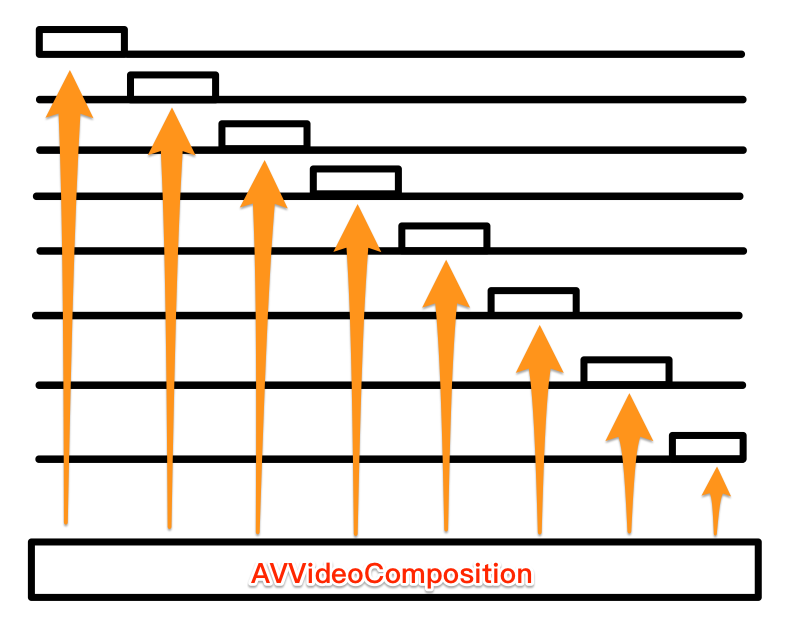


图解：

1. 把一个视频拆分成多个AVAssetTrack，这样做的原因是因为，使用AVAssetReader读取每一帧SampleBuffer的数据是需要把数据加载到内存里面去的，如果直接把整个视频的SampleBuffer加载到内存，会造成闪退
2. 拆分成多个AVAssetTrack之后，使用AVAssetReader**从最后一个AVAssetTrack**读取SampleBuffer，这样我们可以从后往前获取视频的帧数据
3. 获取数据之后通过AVAssetWriterInputPixelBufferAdaptor的- appendPixelBuffer:withPresentationTime:方法来把我们获取到的SampleBuffer倒序写入一个文件

#问题

1. 经过一番折腾，惊喜的发现Demo已经可以Run起来，并且可以输出倒序的视频文件，但是发现最后的时间总是不对，会少了一截，因为加载AVAsset的时候并没有把准确的视频长度信息加载进来，所以我们在加载AVAsset的时候需要调用loadValuesAsynchronouslyForKeys方法来获取准确的视频信息。
2. 但是对于我的需求却并未满足，因为我们在上面说到，我们获取的时候是一个原始的视频文件，这个视频文件没有经过任何的编辑，所以一切看起来都没啥问题，但是有时候我们需要对视频进行一些编辑，比如合成，比如剪切，比如对视频做一些Transform等等的一些编辑操作，水印，动画等等，但是这些操作只是生成了一个新的AVMutableComposition，并没有生成新的视频文件，这个时候如果我们想要直接在经过编辑的文件来生成一个倒序的视频怎么办？在网上找了很久资料，还是没有找到实现的办法，最后翻阅了官方文档之后看到了AVAssetReaderOutput有一个子类AVAssetReaderVideoCompositionOutput，这个子类有一个属性videoComposition，可以赋值一个AVVideoComposition指令容器，这样我们就可以通过这个属性来准确的加载被编辑过的视频的编辑指令了。但是指令有一个timeRange属性，我们直接生成的一组AVAssetTrack和原来的AVVideoComposition并不匹配，这个时候我用到了一个Trick的方法来对应



##解决办法 我在分割AVAsset的时候，在生成一个新的AVComposition的实例时使用insertTimeRange方法插入的时间并非是kCMTimeZero，而是每一个AVAssetTrack在原来的AVAsset对应的开始和结束位置，这样使用AVAssetReaderVideoCompositionOutput来配置videoComposition，Reader就可以准确的读取包含了编辑指令每一帧的SampleBuffer了，然后再按照上面的方法来倒序写入SampleBuffer即可

#经验总结

* appendPixelBuffer:withPresentationTime:方法不能从后往前写，如果你这样做，会写入失败
* 兼顾了编辑指令的实现方法会对性能有一定的浪费，因为我们不在kCMTimeZero时间点生成的AVAssetTrack会在Reader读取的时候获取的都是黑帧，随着视频的分割段数越多，视频的时间越长，倒序生成的时间也就越长
* 目前分割的段数是以每段视频长度最长为1s来分割，如果来操作很长很大的视频，可能需要耗费很长的时间
* 分割视频的时候目前没有对编辑过和没有编辑过的视频作为区分，如果只针对原视频倒序输出，可以把每个AVAssetTrack的开始时间设置为kCMTimeZero，这样性能会有很大的提升

一般对H264编码的视频进行解码，都是从头至尾进行的，因为视频存在I帧、P帧、B帧，解码P帧的时候需要依赖前面最近的I帧或者前一个P帧，解码B帧的时候，不仅要依赖前面的缓存数据还要依赖后面的数据，这就导致了我们没法真正让解码器从后往前解码，只能把视频分成很多足够小的片段，对每一个片段单独进行处理。具体思路如下：我们需要先seek到最后第n个GOP的第一帧-I帧，然后把当前这个点到视频最后的图像都解码出来，存储在一个数组里面。这个n是根据解码数据大小定的，因为如果解码出来的数据太大，内存占用过多，会导致程序被杀掉，我是把视频分成一秒一个小片段，对这些片段，倒过来进行解码，然后把每一段解出来的图像，倒过来编码。使用AVFoundation可以很方便的实现

https://github.com/sagesong/VideoReverse