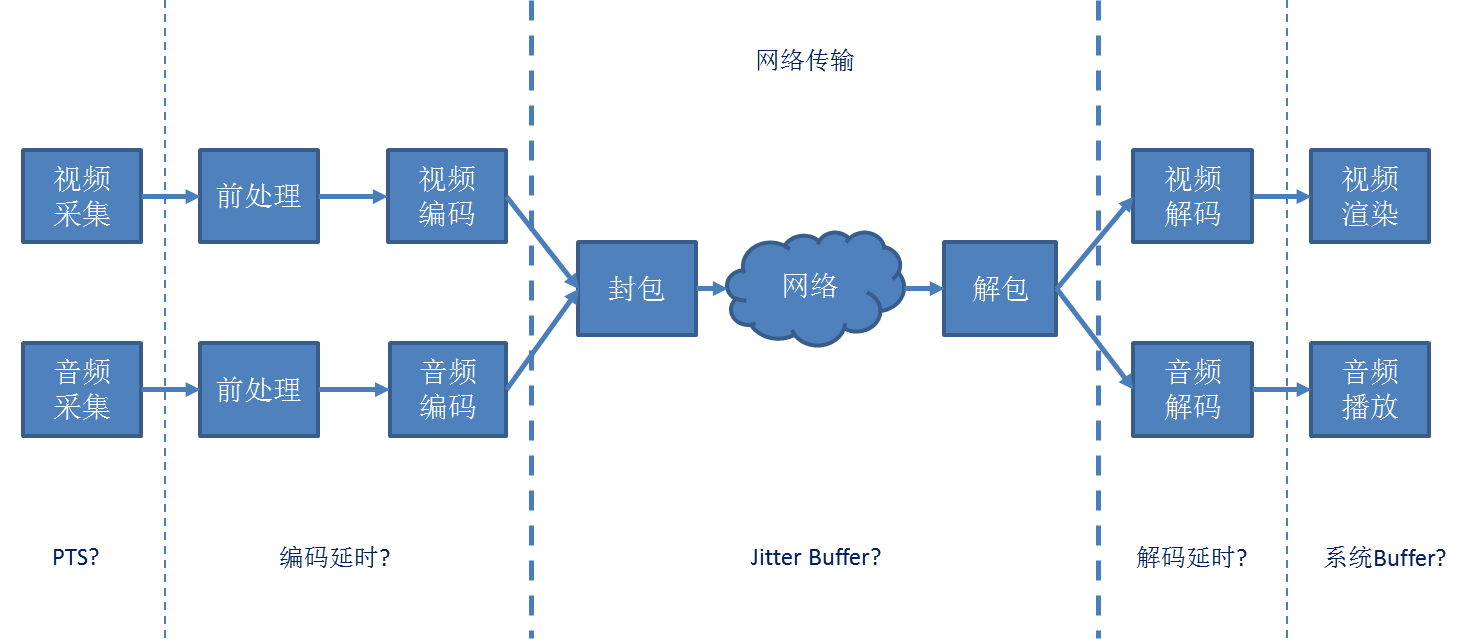
# 问题1 实验部分的设计和实现的难度

打算用本地播放和网络拉流播放两种方式进行实验，主要工具是FFmpeg。在实验之前需要先熟悉用FFmpeg进行音频和视频播放的原理。可以参考的源代码有：ffplay(Visual Studio 2010版本)，video.js(WEB端开源播放器)等等。

### 1.1 本地播放

大概看了一下音画同步的原理，关键是获取帧的PTS和DTS，然后以音频为基准，让视频去同步音频，如果视频慢了就增加音频播放延迟，如果视频快了就增加视频播放延迟。实现的时候会给音频和视频各分配一个缓冲区。PC端实现难度不大，打算用FFmpeg开发，因为有过PC客户端的开发经验；安卓端由于不同硬件的解码速度不一样，并且要用到不太熟悉的Java语言，所以可能比较麻烦；WEB端的话，对JS语言还不太熟悉，实现难度也比较大。

### 1.2 网络拉流播放



不同步的基本原因有硬件固有特性、人为引入。人为引入主要在以下环节：采集、编码、传输、解码、同步算法、渲染。

拉流打算用RTMP流，由于网络传输的影响，为了在播放出来能够不让用户感知到音画不同步，需要故意设置一个Jitter Buffer进行延迟播放，Jitter Buffer的大小需要随网络状况改变自适应。目前打算先用OBS或者某些其他推流软件试试，然后根据这些软件自定义的流协议获取需要的信息，比如帧PTS、DTS等等。然后播放原理和1.1中的方法类似，现在PC端播放出来，然后尝试移动端、WEB端。如果都比较顺利的话，再试试自己用FFmpeg推流。总体难度较大。

# 问题2 方案的理论科学性，实验条件是否满足

### 2.1 实验环境

主要用PC端进行本地播放和网络播放的推流拉流。如果实验顺利，可以用Android、iOS移动端、WEB端作为接收端多进行几组实验。

### 2.2 实验数据

主要是音画播放出来的时间戳数据，用肉眼无法观测，可以用公司这边的音视频同步设备进行测试。测试视频采用标准的测试视频，视频会定期发出Beep声，并在发出Beep声的同时画面闪烁，因此用声敏和光敏设备转化为电信号，即可进行比较准确的实验数据分析。除此之外也可以在播放端进行日志反馈，把音画播放的时间戳记录到日志中，与电信号得到的数据进行比较。

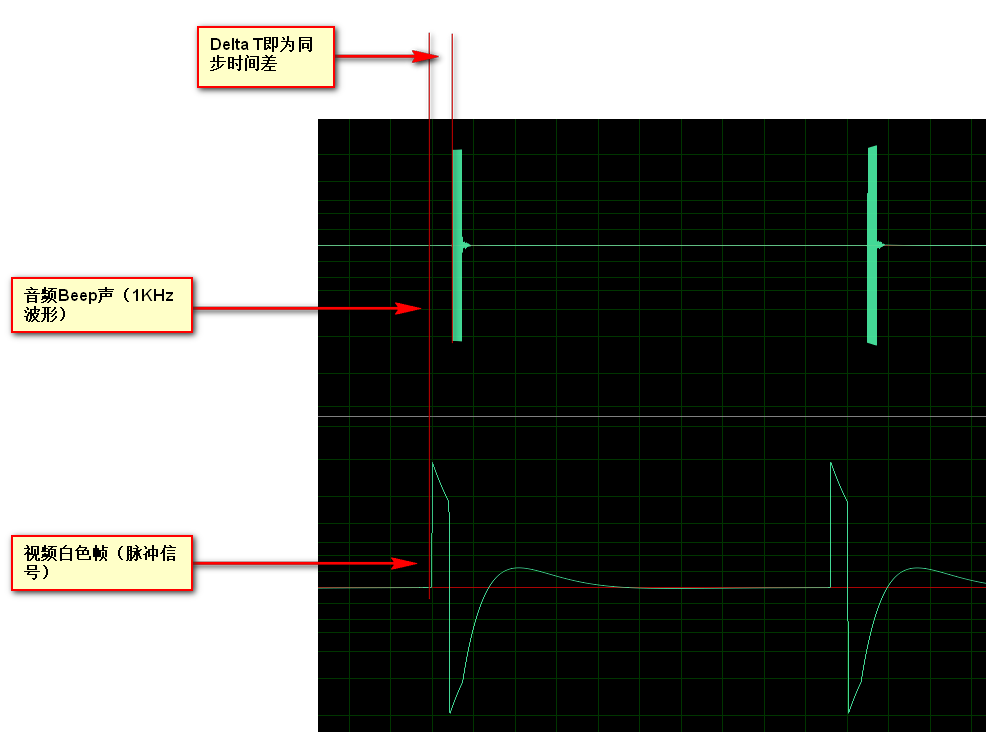


图2.1 音视频播放电信号图

实验结果与国际音视频同步标准作比较，如下图所示：

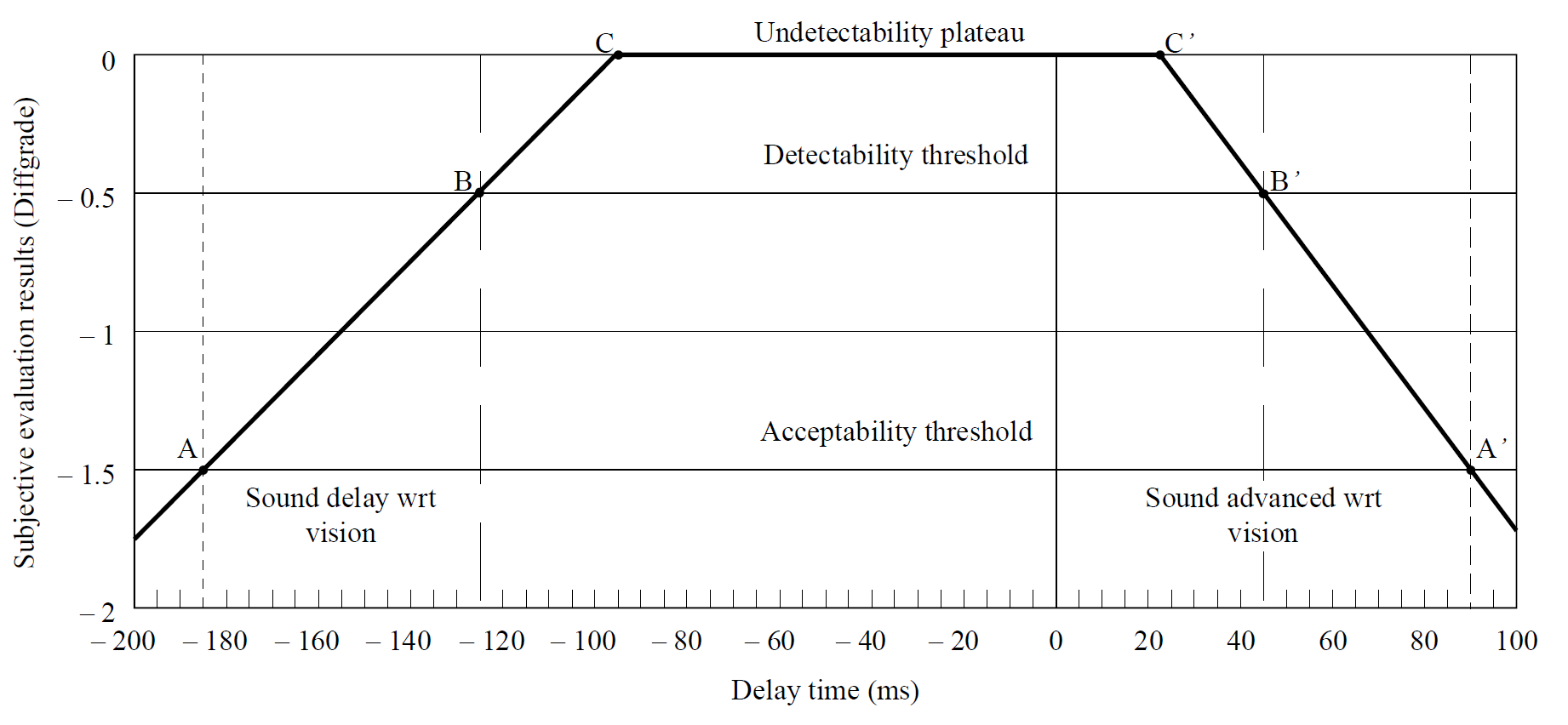


图2.2 国际标准图

### 有新意的方法

1. 想过在播放端使用双缓冲技术，类似显卡硬件的播放原理。

2. 由于Jitter Buffer需要参照网络环境，所以根据网络的哪些参数作哪些调整是最关键的问题。

3. 其他的暂时不知道了，打算先分别把音频和视频的本地播放端实现出来，再研究怎么进行本地的音画同步和网络传输的音画同步。