手机超声波通信

这其实是两年前的一个产品，使用场景类似于基于苹果推出的iBeacon实现室内定位，利用一个小巧的超声波硬件设备，周期性的广播信标信号，手机直接使用麦克风接收这个音频信号并且解码，得到信标信号中的有效数据，最后再根据这个数据进行室内定位的算法逻辑处理。声波通信部分，是技术基础，开发难度比较大，在当时的情况下，对于产品来说，是一个技术壁垒。现在已经过去两年时间了，而且其实出于商业层面的原因，团队也早已放弃这个产品转战其他方向了，所以我还是准备把其中的一些技术细节记录下来。

**基于使用场景而提出的一些技术参数指标**

1. 只能使用18kHz~21kHz这个范围内的音频信号(人耳要尽量听不见这个声音，手机要能够接收到这个声音，所以只能限制在这个频率范围内)
2. 通信距离最远要达到10米
3. 还有一些是硬件设备的性能要求，和通信协议无关，不是这次的讨论重点，可以忽略

**关键技术点**

1. 为了减小数据处理过程中的延迟现象，采取的是实时的进行音频数据采集和数据处理，而不是像某些类似SDK中使用的技术那样，先录音得到一小段音频文件，然后再进行数据处理。具体到iOS平台上，就是使用audio unit框架来搭建PCM音频数据的采集管道，在管道的最后一个节点上，对得到的PCM数据再进行进一步的处理。
2. 仍然是为了降低延迟，使用手机的DSP硬件来进行快速傅里叶变换，具体到iOS上，就是用了Accelerate.framework框架中的相关函数。
3. 为了提高数据传输和解码的成功率，在2FSK的基础上，做了一些调整(magic trick)。

前两点没有太多可说的，对应的开发文档中有很详尽的描述，只不过稍微偏底层一些，只要静下心来老老实实的啃啃文档，还是可以搞定的。第3点中用到的技巧，可能不太常见，我会详细解释一下。

基于标准的2FSK，假如约定用18kHz的音频信号表示二进制的0，用19kHz的音频信号表示二进制的1，同时约定每一个bit持续的发送时间为50ms，假设要发送一个8bit的二进制数据0b11001010(忽略同步和校验部分的bit)，对于发送端来说，代码逻辑其实比较简单，只需要让特定频率的引号信号发送特定的时间就行了。但是对于接收端来说，代码就很困难了，虽然用的是2FSK，但是并没有专用的硬件来完成调制解调过程，所以要完全用代码来模拟整个过程，这个里面就涉及到了傅里叶变换、滤波等大量的数字信号处理里面的内容，这些处理完后，才会真正的进入到通信协议栈里面处理二进制的0和1。

如果按照标准的2FSK方式，接收端的代码必须用定时器记录0或1(18kHz或19kHz)持续的时间，然后用这个时间值和50ms做比较，才能判断出这一部分音频片段对应了多少个连续的0或1。而且这仅仅是理论上对解码算法的描述，实际情况中，发送端维持的每个bit位的持续时间是50ms，进入空气中后，会和其他的各种各样的音频信号混杂在一起，然后才进入接收端进行变换和滤波等操作，这个时候，是很难保证每个bit位仍然能够维持在50ms的(即便有50ms，代码仍然会很难编写)，正式因为这些原因，成功解码数据的概率并不高。为了改善这种情况，对2FSK做了一些调整，这里借鉴了数字电路里面的一些概念和技术。在数电的串行接口电路中，使用高低电平来表示二进制的1和0，根据传输比特率的约定，每个电平会持续特定的时间，这类似于我们的音频系统中约定的每个bit持续发送50ms，这通常称为电平检测(根据电平值持续的时间进行检测)，还有另外一种称为边缘检测的技术，它不依赖于每个电平值持续的时间，而是依赖于电平值的变化事件，比如电平从高变为低(从1变为0)。这里就是使用了边缘检测这种方式来处理音频信号，接收端需要关注的，是音频信号频率值的变化，而不是每个频率值持续的时间。为了实现这种方式，还需要对之前的约定做一些调整，调整为18kHz和19kHz的音频信号都可以表示二进制的0，20kHz和21kHz的音频信号都可以表示二进制的1，如果是为了表示两个连续的0，那么就应该是18kHz的音频信号持续50ms，然后变成19kHz的音频信号持续50ms(或者先发送19kHz的，再发送18kHz的)，对于连续的1，也采用类似的策略。举个例子，对于二进制数据0b11101000，转换成频率值后，可能就会是这样的一组值 [20kHz，21kHz，20kHz，18kHz，21kHz，18kHz，19kHz，18kHz]，因为每一个bit对应的频率值都会发生变化，那么接收端就可以忽略每个bit持续的时间，只需要检测出每一次频率值发生变化就行了，每一次变化后得到的数值，就可以对应到当前的bit位的二进制值。用了这种调制解调的思路后，接收端的代码，写起来就很容易了:]

# [声波通信原理及源代码](http://blog.csdn.net/jcgu/article/details/12834483" \t "_blank)

分类： [声波通信](http://blog.csdn.net/jcgu/article/category/1691611) [声音传输](http://blog.csdn.net/jcgu/article/category/1691613) [android](http://blog.csdn.net/jcgu/article/category/1346001) [源代码](http://blog.csdn.net/jcgu/article/category/1390565)2013-10-17 11:28 2965人阅读 [评论](http://blog.csdn.net/jcgu/article/details/12834483#comments)(15) 收藏 [举报](http://blog.csdn.net/jcgu/article/details/12834483#report)

[声波通信](http://www.csdn.net/tag/%E5%A3%B0%E6%B3%A2%E9%80%9A%E4%BF%A1)[声音传输](http://www.csdn.net/tag/%E5%A3%B0%E9%9F%B3%E4%BC%A0%E8%BE%93)[android](http://www.csdn.net/tag/android)[源代码](http://www.csdn.net/tag/%E6%BA%90%E4%BB%A3%E7%A0%81)

        目前声波通信已经在iphone和android中广泛的应用起来了，涉及到数据和文件传输，以及支付等众多领域。比如iphone中的chirp，android中的茄子快传，支付宝的声波支付，小米快传等。这些传输技术大多都是使用声波作为握手信号，然后使用wifi或其他信道传输数据。比如茄子快传可能的实现为，接收方先建立wifi热点，然后将热点名称通过声波发送出去，发送方在收到声波后解码出wifi热点名称，然后自动链接热点并传输文件，整个过程不需要人工干预。这些程序的核心技术和难点在于声波通信，下面讲解声波通信原理。

        声波通信的原理其实比较简单，主要是用单频率声音信号对数据进行编码，然后播放这些单频率声音，接收方在收到声音后，识别出频率，然后根据频率解码出数据。比如：我们可以将1500HZ的正弦波对应数字1，1600HZ的正弦波对应数字2，1700HZ的正弦波对应数字3。那么数字串3123就对应4段正弦波，规定每段正弦波持续100ms，则3123对应400毫秒的声音段。接收方录制声音，对收到的声音进行解析，识别出1700HZ，1500HZ，1600HZ，1700HZ四段正弦波频率，然后查找码本，解码出的数字就是3123。

        说了这么多，大家可能还会觉得比较抽象，没关系，大家可以下载声波通信的源代码自己理解，csdn下载地址：[点击打开链接](http://download.csdn.net/detail/jcgu/7503955)。

http://download.csdn.net/detail/jcgu/7503955