



# IP Camera语音通话噪声和回声处理

Leo Zhang

2018. 11. 19

# 目录

- 1. 语音通话噪声
- 2. 语音通话回声
- 3. 回声消除，噪声抑制方案

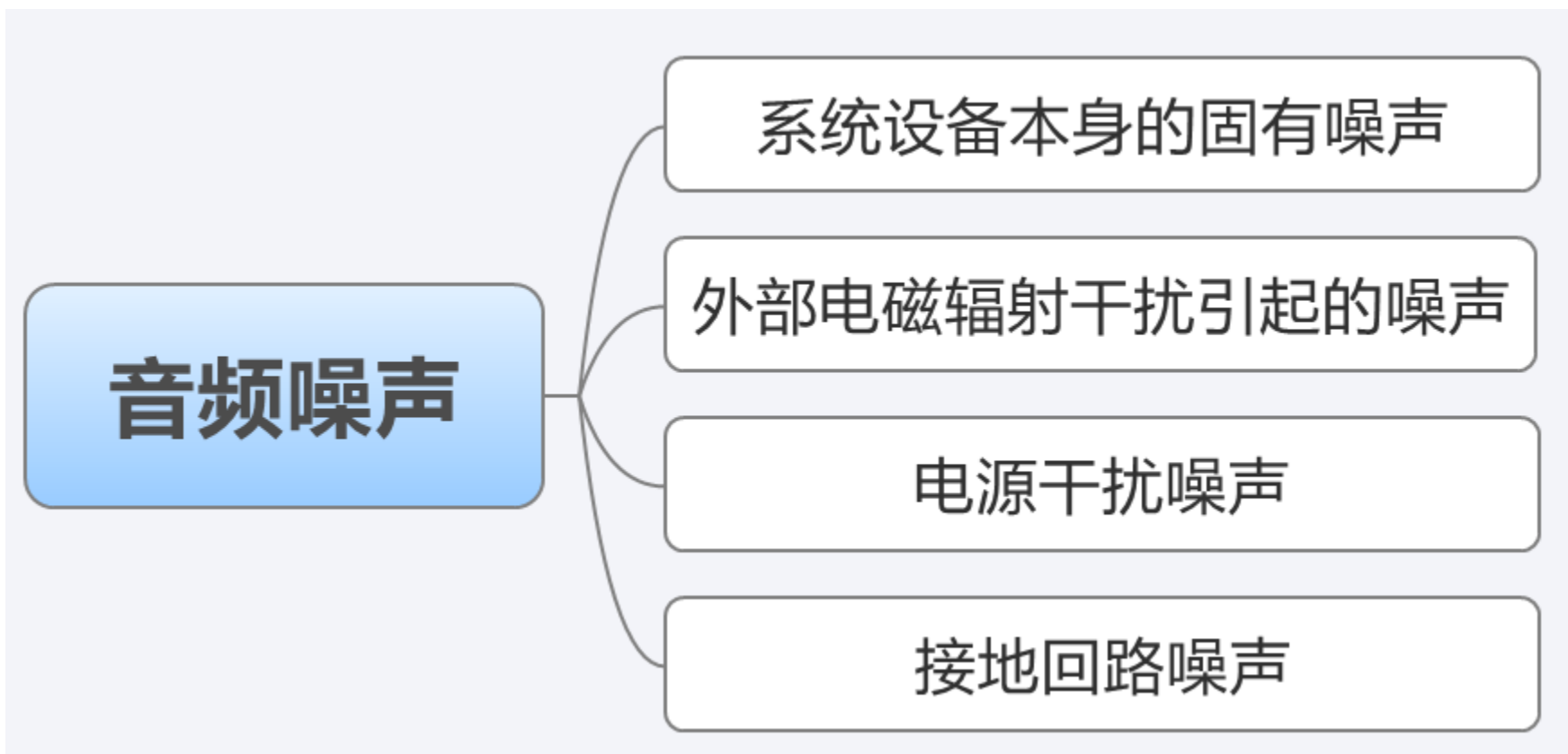
1.1 语音通话噪声来源

1.2 语音通话降噪方案

1.3 语音通话降噪算法

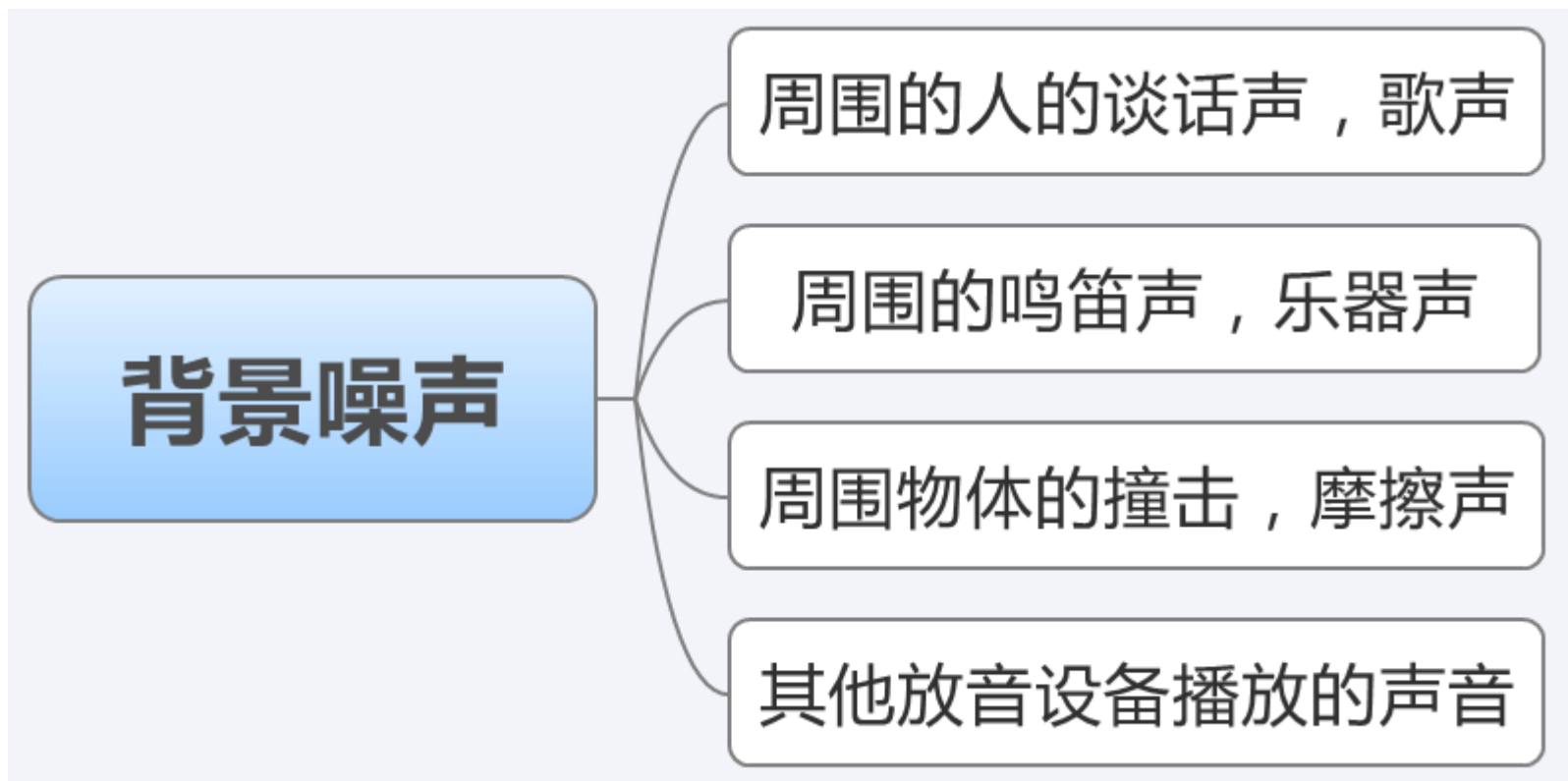
# 1.1 语音通话噪声来源

音频噪声：广义理解为不需要的声音讯号。



# 语音通话场景的噪声

背景噪声：与主体声音信号存在与否无关的一切干扰。多为不需要的讯号。



# 语音通话场景的噪声

背景噪声：与主体声音信号存在与否无关的一切干扰。多为不需要的讯号。

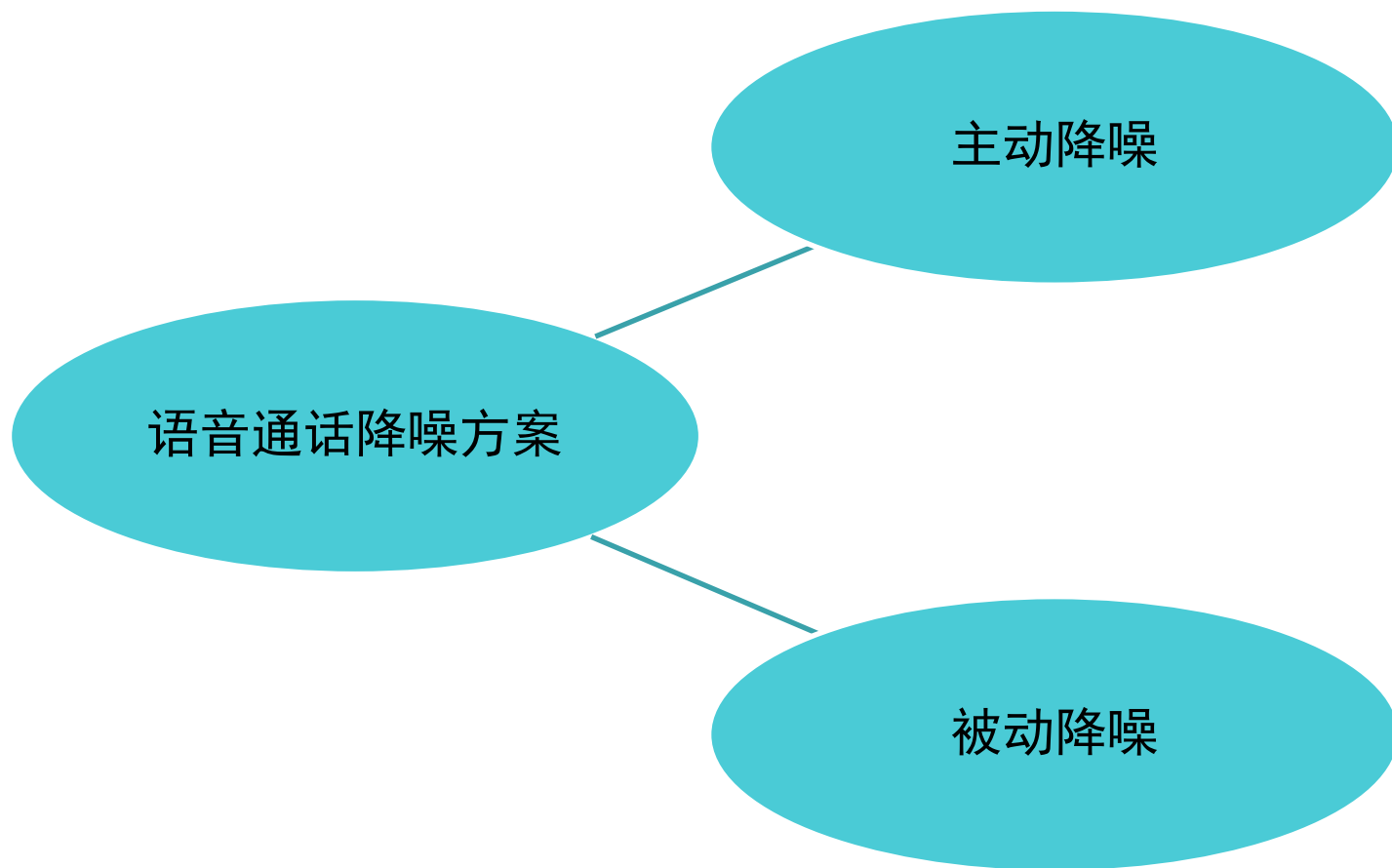


1.1 语音通话噪声来源

1.2 语音通话降噪方案

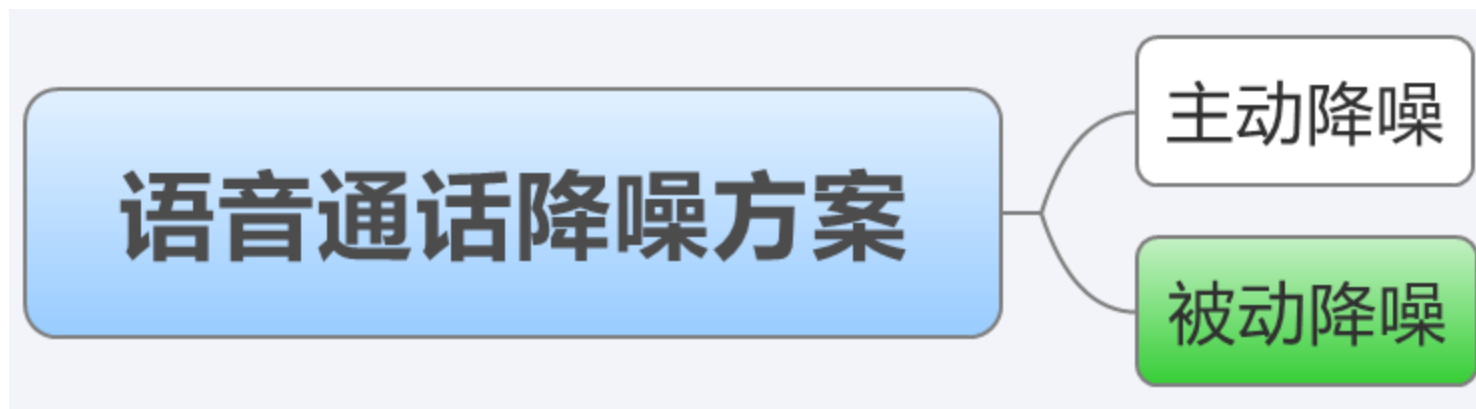
1.3 语音通话降噪算法

## 1.2 语音通话降噪方案



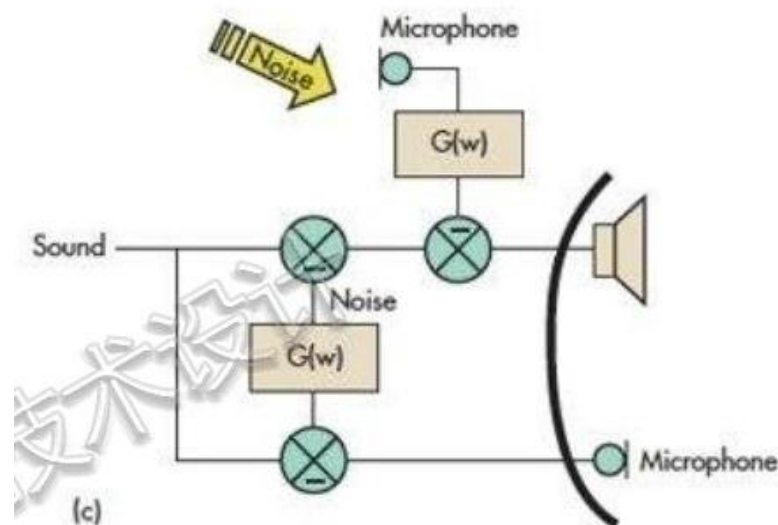


## 1.2 语音通话降噪方案



## 1.2.1 主动降噪

- 在有参考噪声信号的情况下，根据自适应算法抵消掉混在有用信号中的噪声。主动降噪会根据输入信号的变化调整自身的参数来去除噪声。



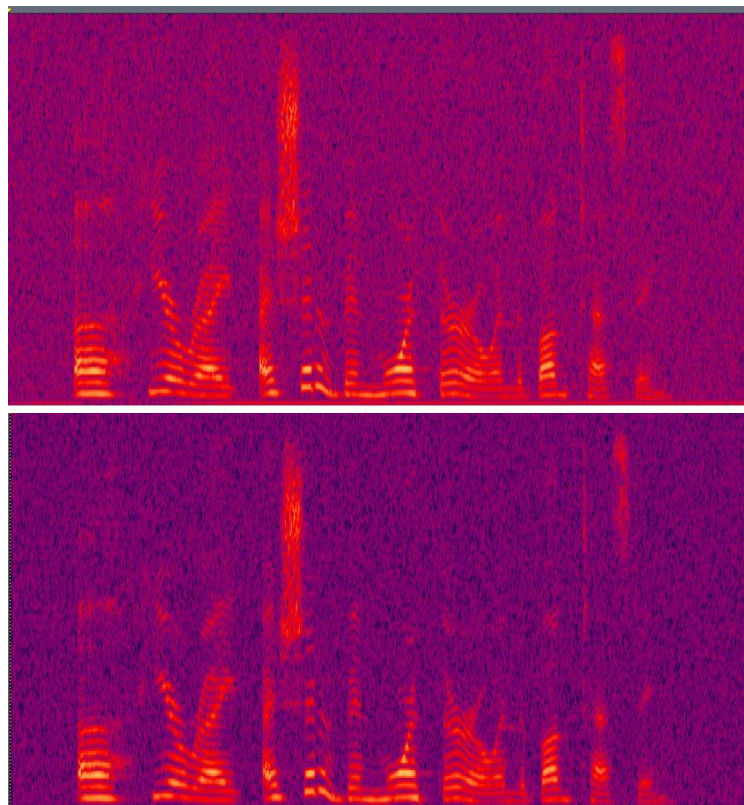
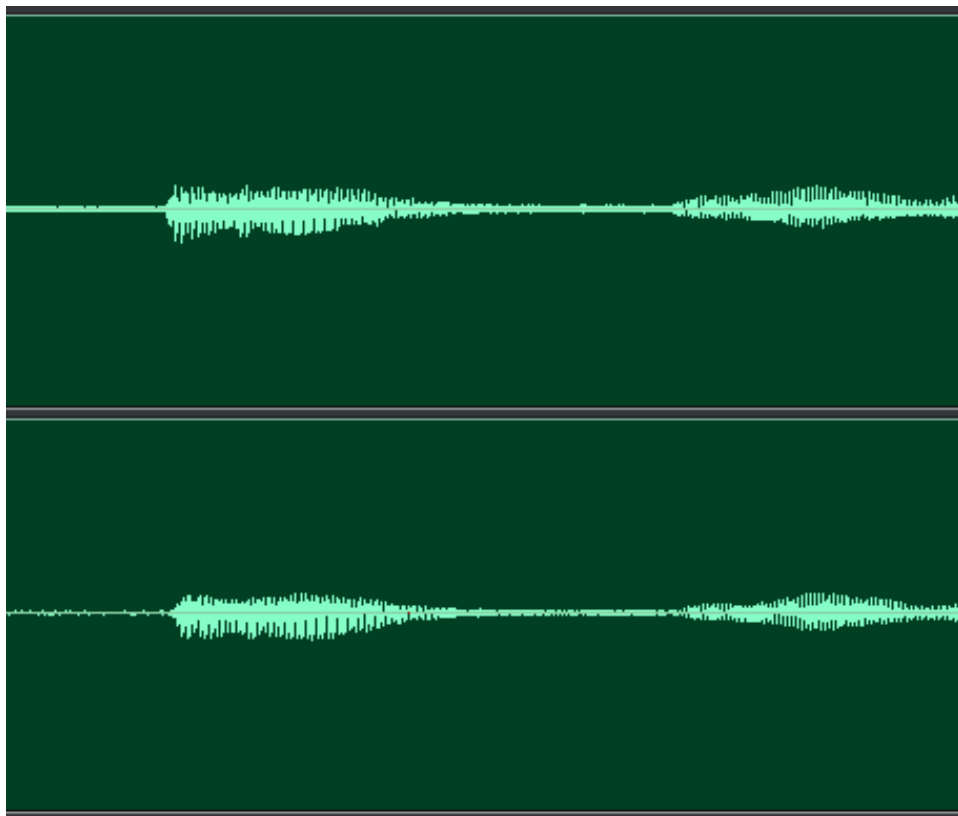
- 主动降噪案例：iphone 4S双麦克风降噪， 主动降噪耳机

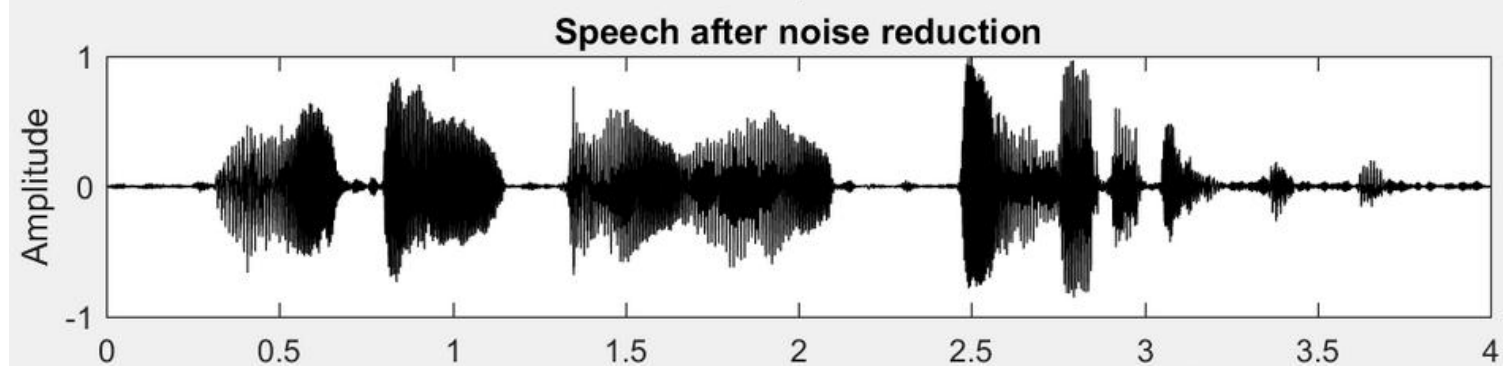
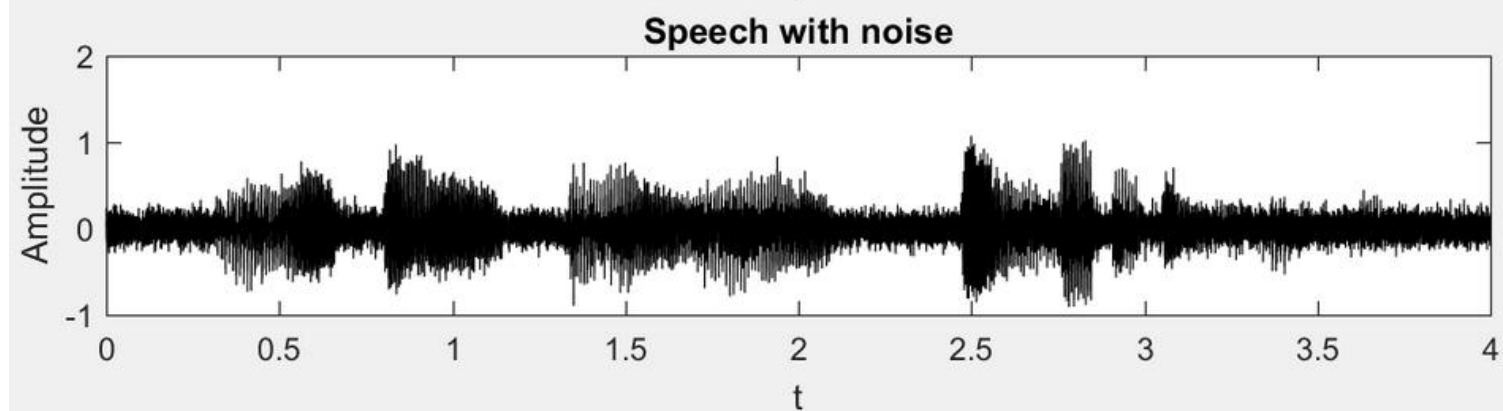
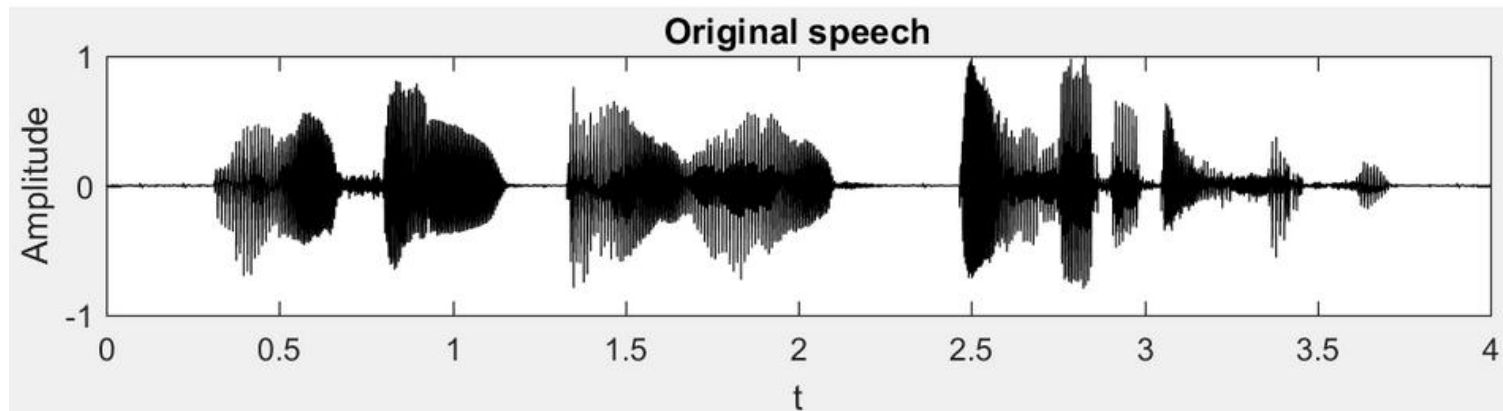
# iphone 4S双麦克风降噪



## 1.2.2 被动降噪

在没有参考噪声信号的情况，利用已有的消噪模型对输入信号进行消噪。  
例如：分析频谱，使用滤波器进行滤波降噪。





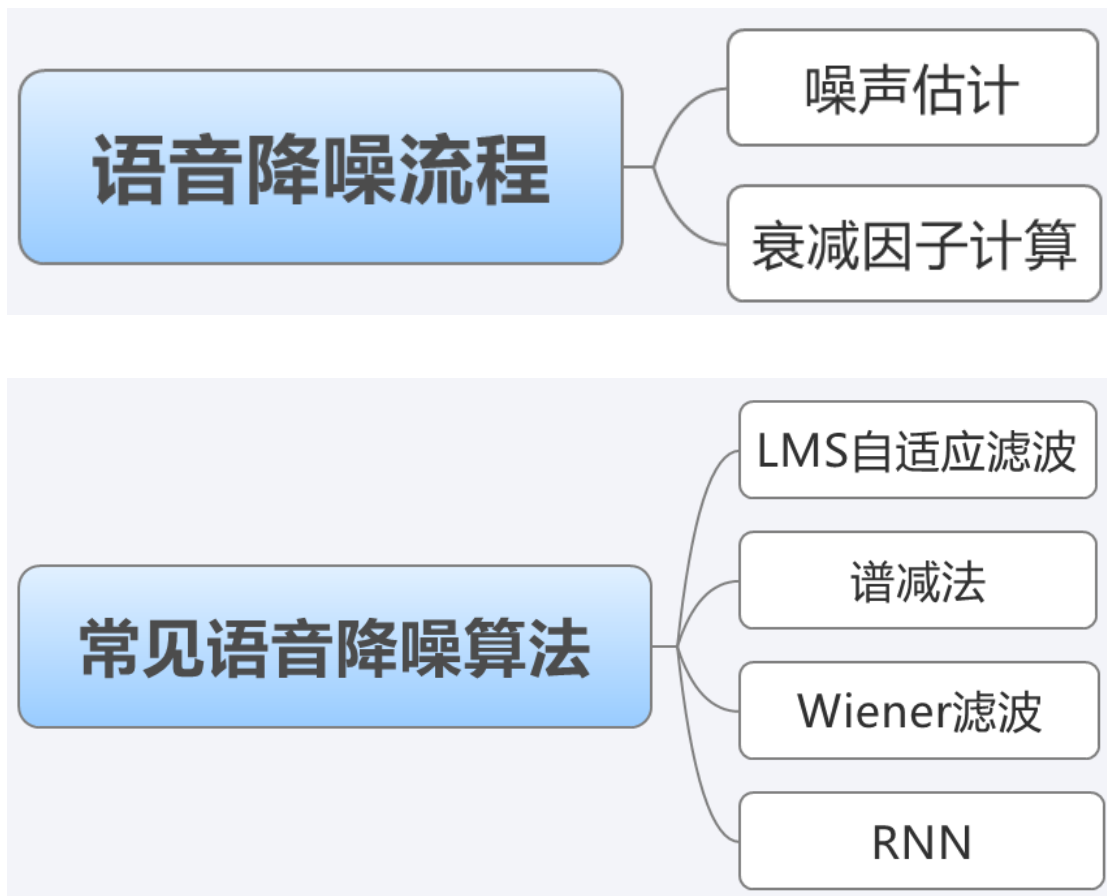
1.1 语音通话噪声来源

1.2 语音通话降噪方案

1.3 语音通话降噪算法

# 1.3 语音通话降噪算法

语音降噪，属于语音增强的范畴



# 目录

- 1. 语音通话噪声
- 2. 语音通话回声
- 3. 回声消除，噪声抑制方案



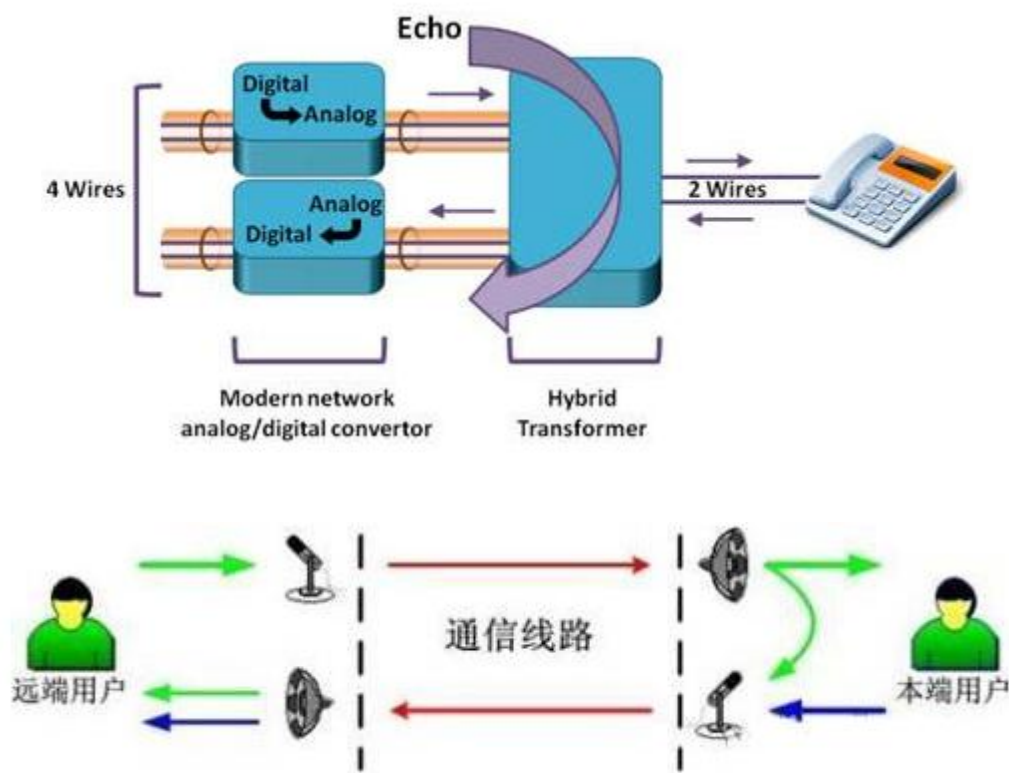
2.1 语音通话回声来源

2.2 语音回声处理方案

2.3 语音回声消除算法

## 2.1 语音通话回声来源

回声：是指说话者通过通信设备发送给其他人的语音又重新回到自己的听筒里的现象。包含“线路回声”和“声学回声”。

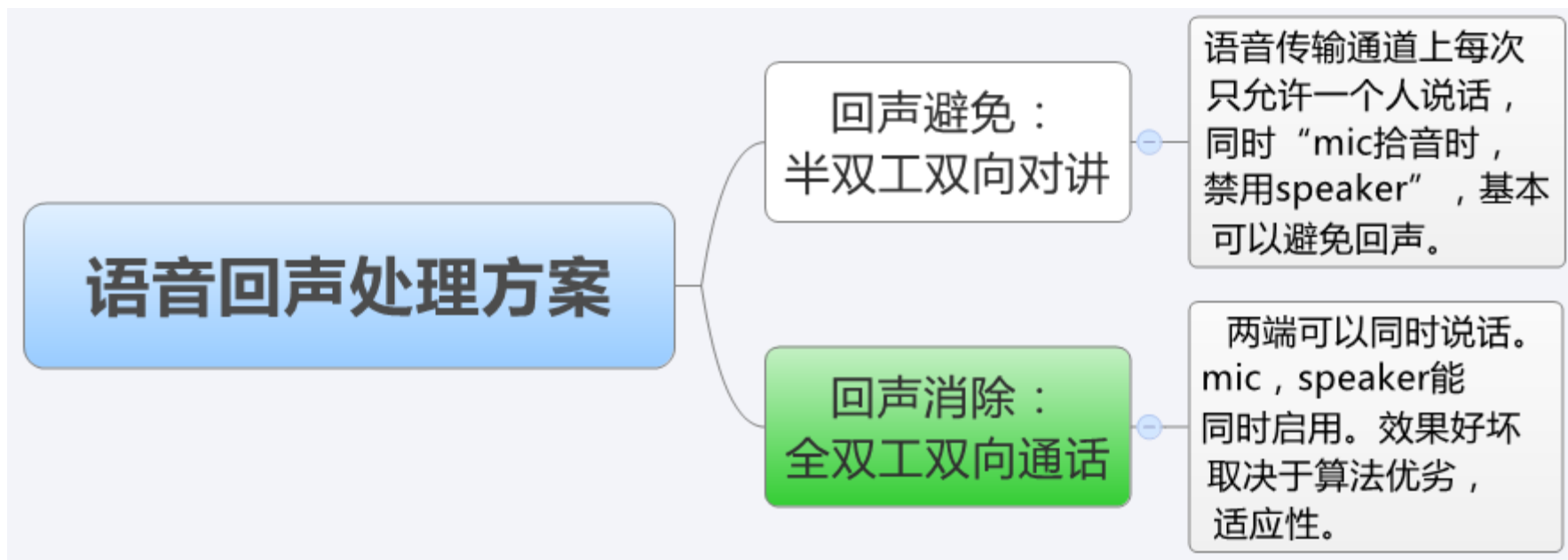


2.1 语音通话回声来源

2.2 语音回声处理方案

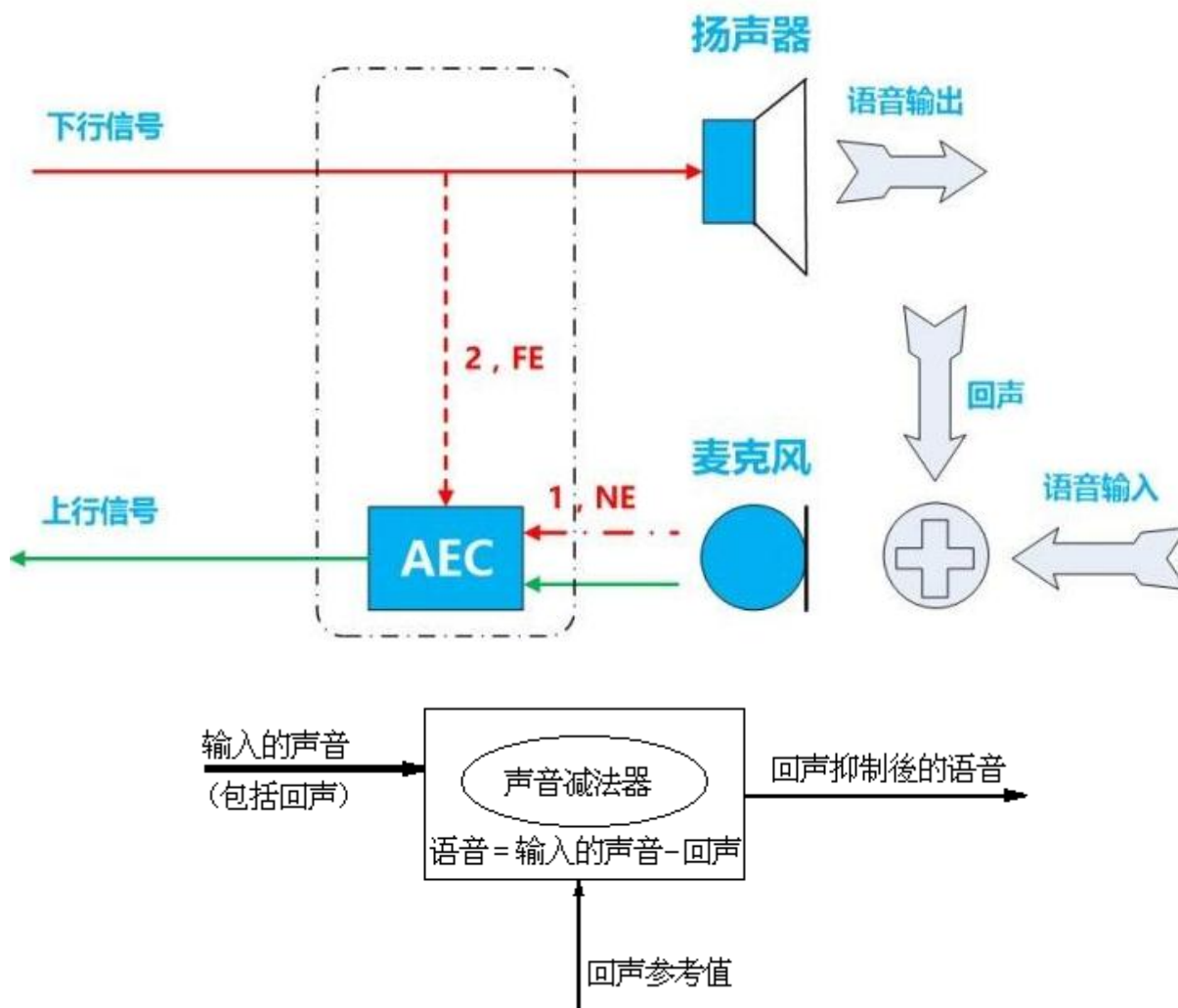
2.3 语音回声消除算法

## 2.2 语音回声处理方案



# 声学回声消除

原理：参考远端语音信号，估计回声。将回声估计值从近端拾音信号中减去。



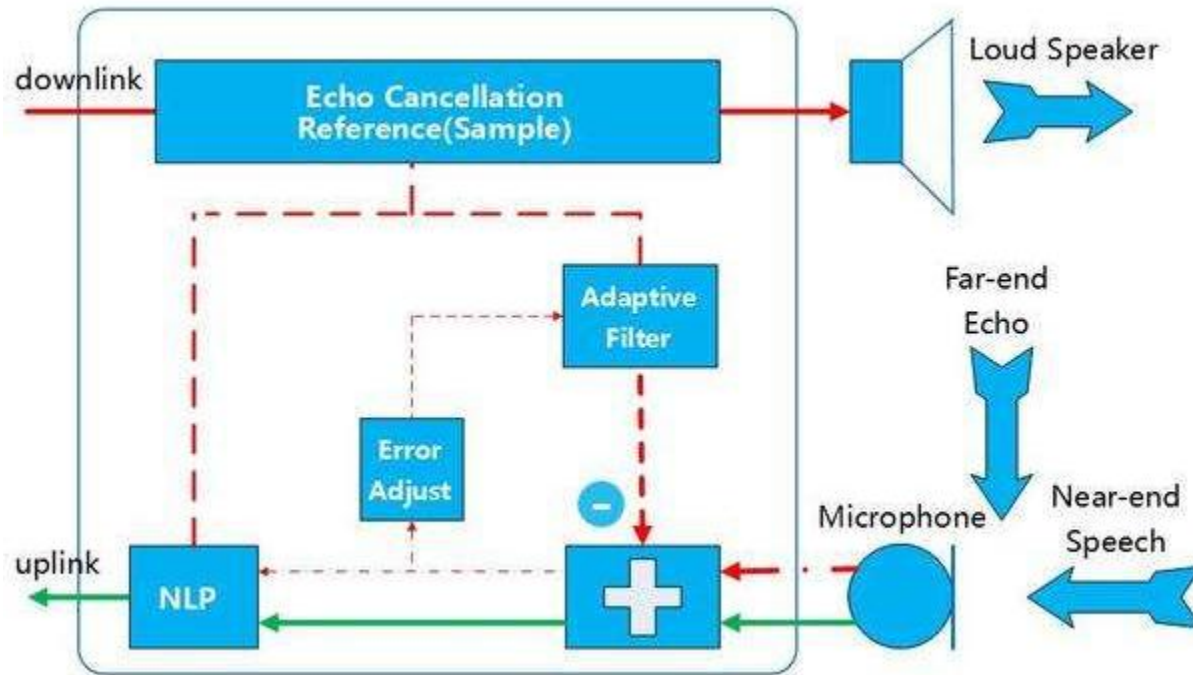
2.1 语音通话回声来源

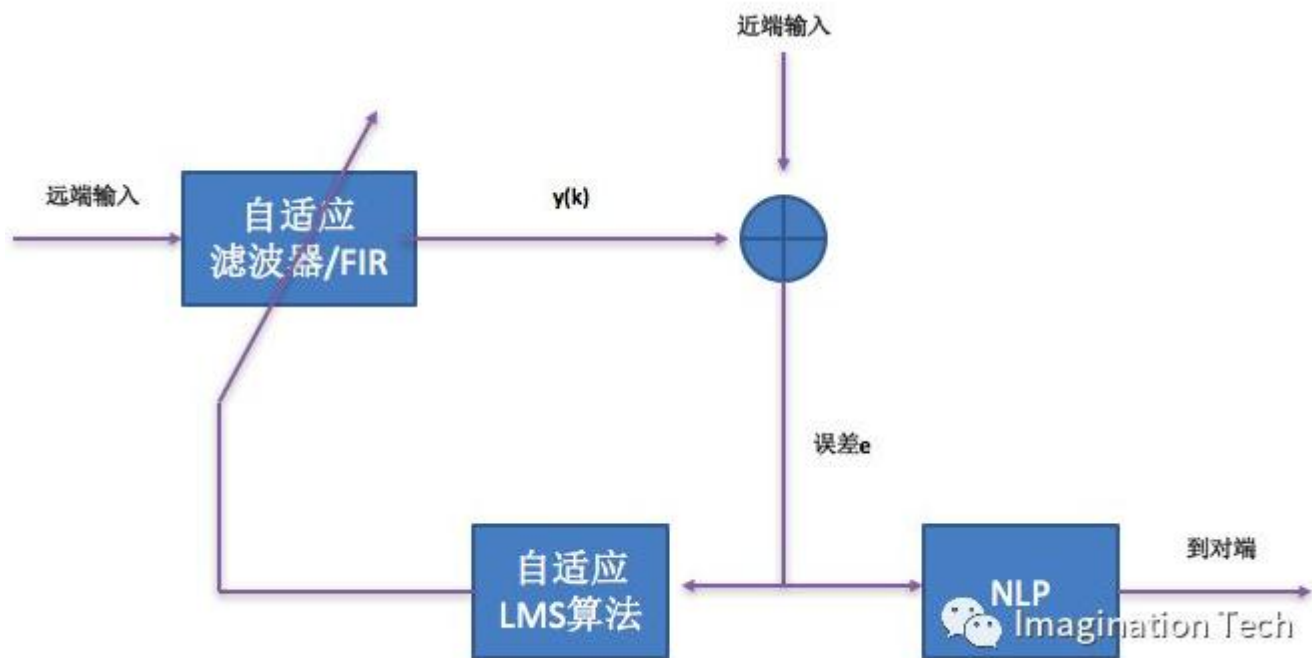
2.2 语音回声处理方案

2.3 语音回声消除算法

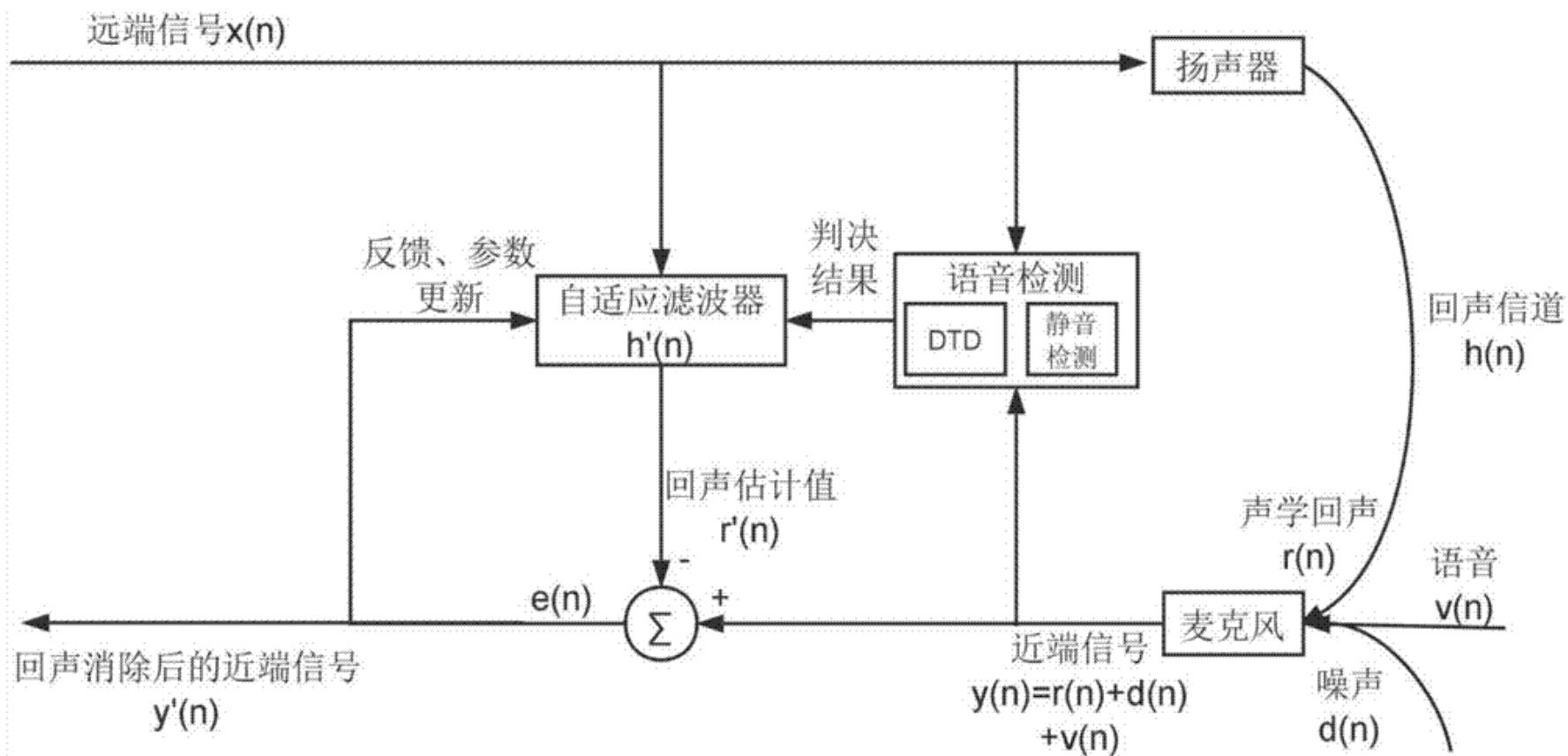
## 2.3 语音回声消除算法

AEC算法： 大部分都基于频域分块和自适应滤波算法。









# 目录

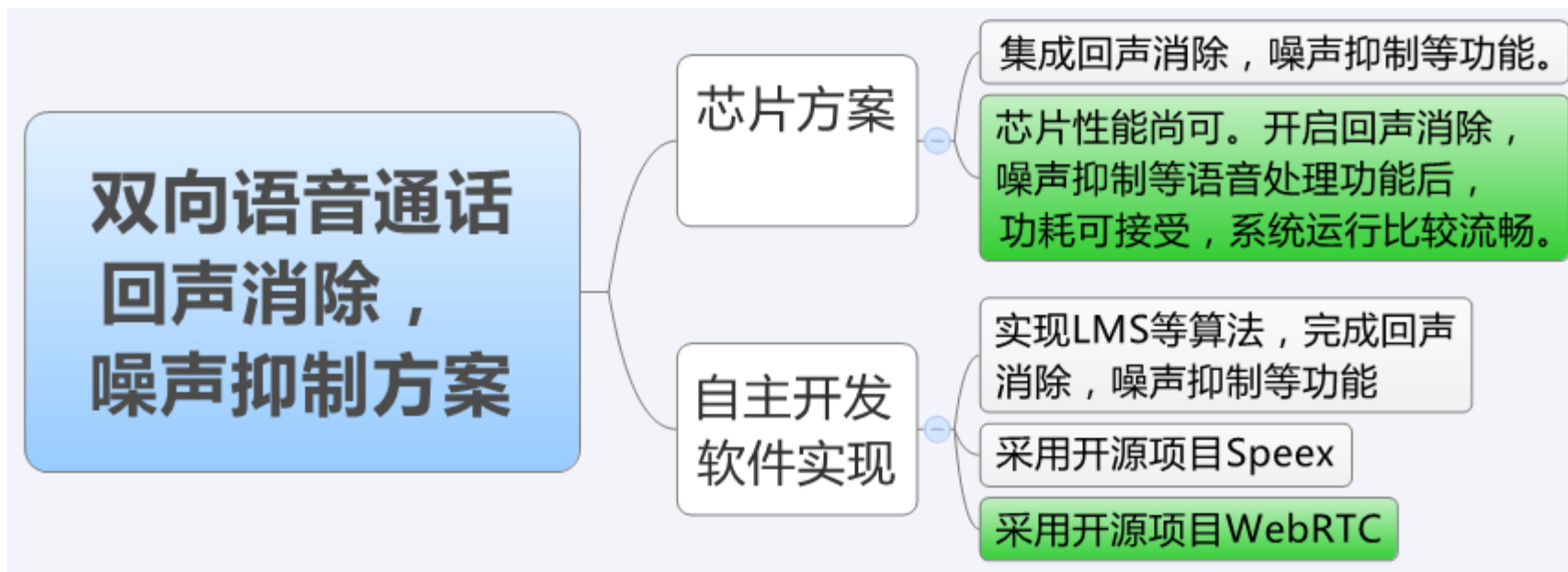
- 1. 语音通话噪声
- 2. 语音通话回声
- 3. WebRTC语音处理方案

## 3.1 语音处理方案选择

## 3.2 Webrtc语音处理移植

## 3.3 Webrtc语音处理测试

## 3.1 语音处理方案选择



# Speex

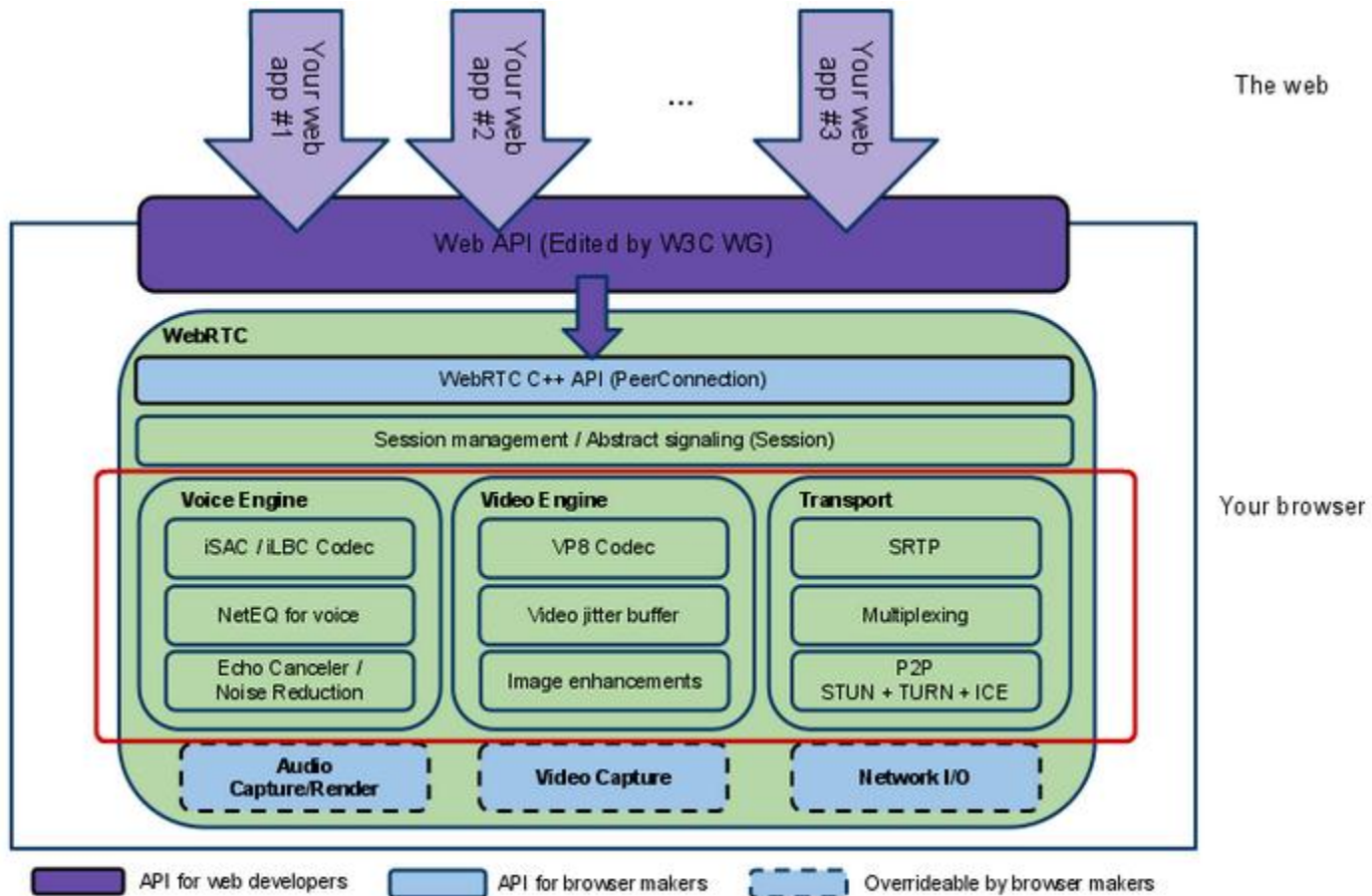
开源音频编码器，项目网址：<https://www.speex.org/>

主要特性如下：

- Narrowband (8 kHz), wideband (16 kHz), and ultra-wideband (32 kHz) compression in the same bitstream
- Intensity stereo encoding
- Packet loss concealment
- Variable bitrate operation (VBR)
- Voice Activity Detection (VAD)
- Discontinuous Transmission (DTX)
- Fixed-point port
- Acoustic echo canceller
- Noise suppression

# WebRTC

Google开源项目，最初的目的是提供Web浏览器之间实时多媒体通信功能，后来扩展到移动平台。项目网址：[webrtc.org](http://webrtc.org)，<http://webrtc.org.cn/>



WebRTC总体架构图

# Speex VS WebRTC

- 如果样本非线性不严重，两者的AEC效果都不错
- 对于非线性样本，`speex aec`效果较差，`webrtc aec`的效果更好
- 双向通话时，使用`webrtc aec`会有吃音现象， 需要加入双讲检测

选择WebRTC开源项目

3.1 语音处理方案选择

3.2 WebRTC语音处理移植

3.3 WebRTC语音处理测试



# 移植--方案选择

➤ 方案一：从WebRTC Native代码中裁剪出语音处理模块。

（1）优点：能确保下载代码是最新的，最新版本的代码可能解决了一些问题，增加了一些功能。

（2）缺点：从众多项目代码文件中提取所需文件，工作量较大，会遇到较多编译和链接错误。

➤ 方案二：寻求现成的已经裁剪语音处理模块代码，进行移植。

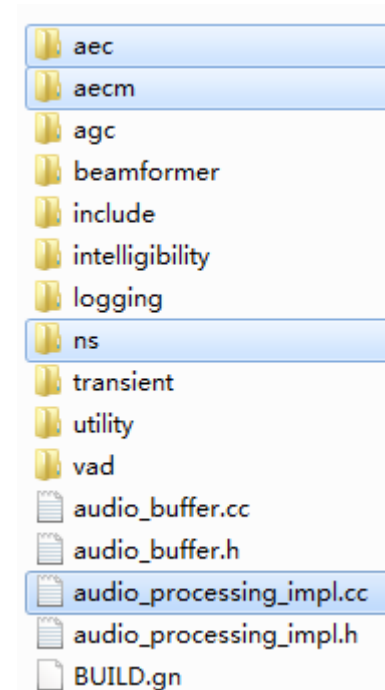
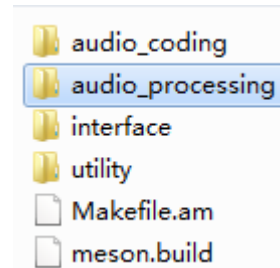
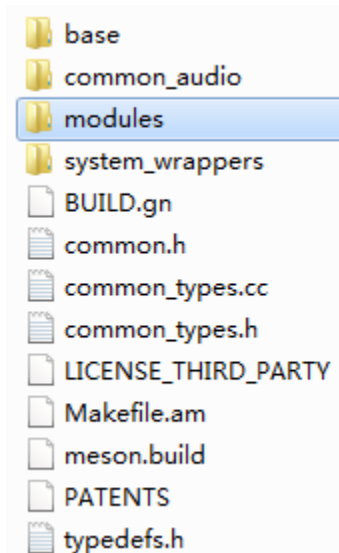
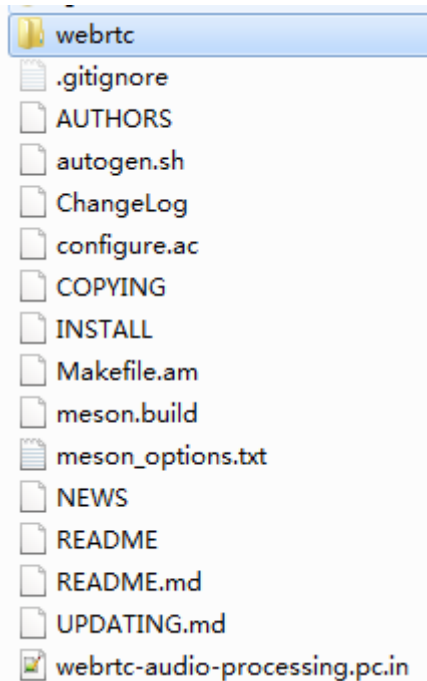
（1）优点：移植时所需改动较少，遇到的编译和链接问题相对较少。

（2）缺点：项目代码不是最新，可能会遇到在最新代码上已经被解决的问题。也无法使用新功能。

➤ 最终我选择了方案二。

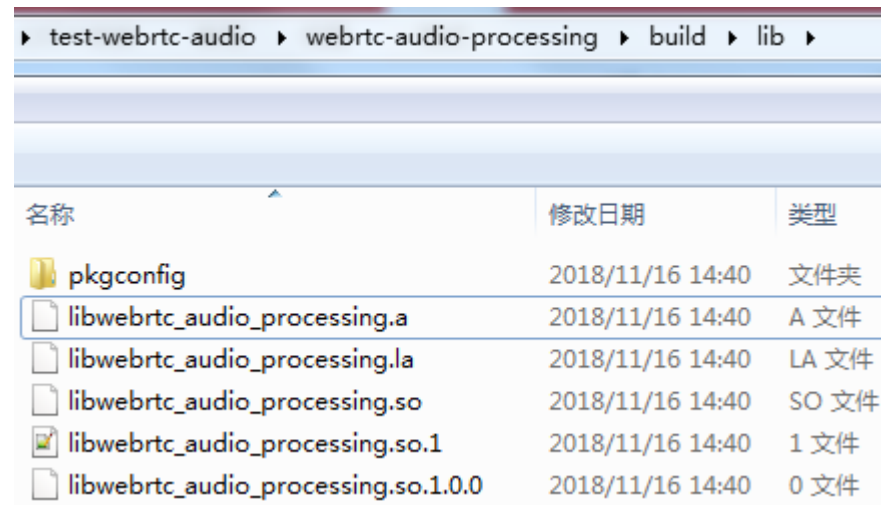
# 移植--代码下载

- 代码下载路径: <http://freedesktop.org/software/pulseaudio/webRTC-audio-processing/>
- 代码结构如下:



# 移植--代码编译

- 1. 执行./autogen.sh，通过automake工具生成configure文件， Makefile文件。如果系统中没有安装automake, libtool，需要通过sudo apt-get install 手动安装。
- 2. 执行./configure --prefix=/some/local/path做相关的配置，指定文件的生成目录。
- 3. 执行make。
- 4. 执行make install将生成libwebrtc\_audio\_processing.a静态链接库文件和libwebrtc\_audio\_processing.so动态链接库文件。



名称	修改日期	类型
pkgconfig	2018/11/16 14:40	文件夹
libwebrtc_audio_processing.a	2018/11/16 14:40	A 文件
libwebrtc_audio_processing.la	2018/11/16 14:40	LA 文件
libwebrtc_audio_processing.so	2018/11/16 14:40	SO 文件
libwebrtc_audio_processing.so.1	2018/11/16 14:40	1 文件
libwebrtc_audio_processing.so.1.0.0	2018/11/16 14:40	0 文件

# 移植--编写测试程序

```
AudioProcessing* apm = AudioProcessing::Create(0);  
apm->high_pass_filter()->Enable(true);
```

```
apm->echo_cancellation()->enable_drift_compensation(false);  
apm->echo_cancellation()->Enable(true);  
apm->noise_reduction()->set_level(kHighSuppression);  
apm->noise_reduction()->Enable(true);
```

```
// Speaker将要播放的远端语音，回声消除功能开启时才需要调用  
apm->ProcessReverseStream(render_frame);
```

```
// 这个延时非常重要，稍后讲述  
apm->set_stream_delay_ms(delay_ms);
```

```
// 处理MIC接收到的语音（包含有效语音和回声）  
apm->ProcessStream(capture_frame);
```

# 编写测试程序--构造AudioFrame数据

```
webrtc::AudioFrame *echo_frame = NULL;
if (is_echo_cancel) {
    echo_frame = new webrtc::AudioFrame();
    echo_frame->sample_rate_hz_ = 8000;//16000;
    echo_frame->samples_per_channel_ = echo_frame->sample_rate_hz_
* frame_step / 1000.0;
    echo_frame->num_channels_ = 1;
}

bool ReadFrame(FILE* file, webrtc::AudioFrame* frame) {
    // The files always contain stereo audio.
    size_t frame_size = frame->samples_per_channel_;
    size_t read_count = fread(frame->data_,
                                sizeof(int16_t),
                                frame_size,
                                file);
}
```

# 编写测试程序--修改Makefile.am

根据**automake**的规则，需要添加的内容如下：

```
bin_PROGRAMS = audio_test_main
```

```
audio_test_main_SOURCES = audio_test_main.cc
```

```
audio_test_main_CFLAGS = $(AM_CFLAGS) $(COMMON_CFLAGS)
```

```
audio_test_main_CXXFLAGS = $(AM_CXXFLAGS)
```

```
$(COMMON_CXXFLAGS)
```

```
audio_test_main_LDADD =
```

```
webrtc/modules/audio_processing/.libs/libwebrtc_audio_processing.la
```

# 编写测试程序--LD\_LIBRARY\_PATH

- 如果使用静态链接库libwebrtc\_audio\_processing.a，不需要指定。
- 如果使用动态链接库libwebrtc\_audio\_processing.so，则需要指定程序执行时，动态链接库的查找路径。
- 可通过“读取任意pcm文件，测试语音降噪功能”来测试程序是否可以正常执行。

# 关于stream\_delay的说明 ( 1 )

This must be called if and only if echo processing is enabled.

Sets the |delay| in ms between AnalyzeReverseStream() receiving a far-end frame and ProcessStream() receiving a near-end frame containing the corresponding echo. On the client-side this can be expressed as

$$\text{delay} = (t_{\text{render}} - t_{\text{analyze}}) + (t_{\text{process}} - t_{\text{capture}})$$

- $t_{\text{analyze}}$  is the time a frame is passed to AnalyzeReverseStream()
- $t_{\text{render}}$  is the time the first sample of the same frame is rendered by the audio hardware.
- $t_{\text{capture}}$  is the time the first sample of a frame is captured by the audio hardware.
- $t_{\text{process}}$  is the time the same frame is passed to ProcessStream().



# 关于stream\_delay的说明（2）

- 实现声学回声消除时，需要参考远端语音信号，估计近端回声。
- 需要从近端mic拾取的语音（语音信号+回声信号等）中减去回声信号，那么就需要对齐“近端mic的语音信号”与“远端发送给speaker播放的语音信号”，以便后续查找。
- **stream\_delay**就是 “时间差 = 接收到第一个近端mic语音样本的代码执行时刻 - 接收到第一个远端语音样本的代码执行时刻”

3.1 语音处理方案选择

3.2 WebRTC语音处理移植

3.3 WebRTC语音处理测试

# 测试程序使用说明

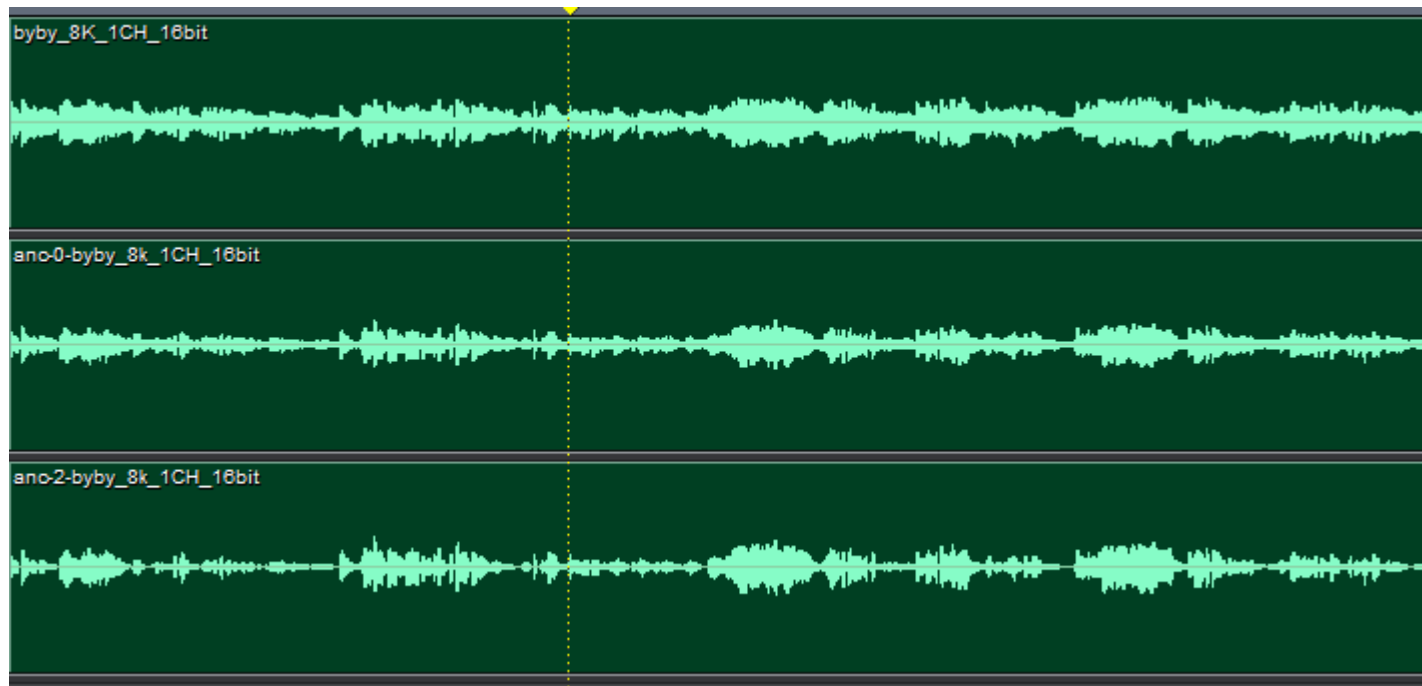
- 输入 `./audio_test_main` 或者 `./audio_test_main -h` 获得程序使用提示：

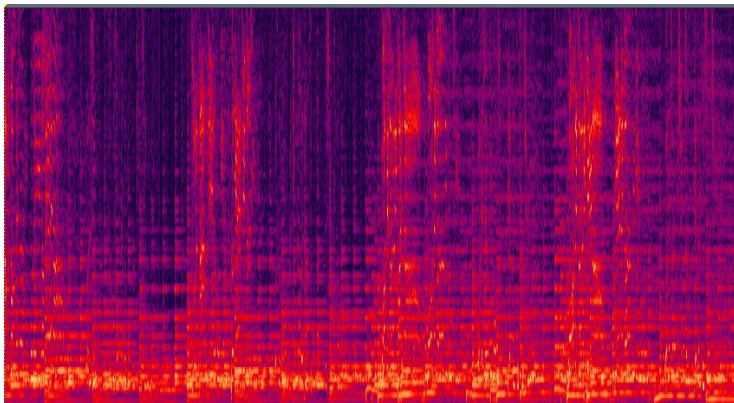
```
Usage: audio_test_main -anc|-agc|-aec value input.raw output.raw [delay_ms echo_in.raw]
```

- 说明如下：
- `-anc`: auto noise control, 对应着noise suppression功能。
- `-aec`: auto echo control, 对应着acoustic echo cancellation功能。
- `delay_ms`: stream\_delay时间, 以毫秒为单位, 当前用代码中的固定值。
- `echo_in.raw`: 使用回声消除功能时, 需要指定的远端参考语音文件。

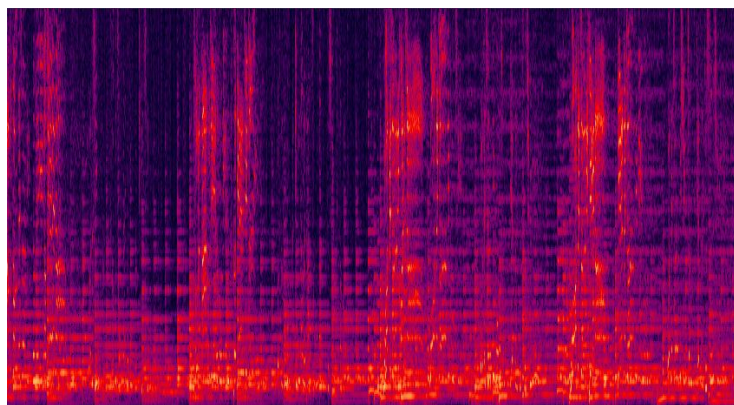
# noise suppression功能测试

```
-processing/build/bin$ ./audio_test_main -anc 2 ../../data/byby_8K_1CH_16bit.pcm ..  
../../data/out/anc-2-byby_8k_1CH_16bit.pcm
```

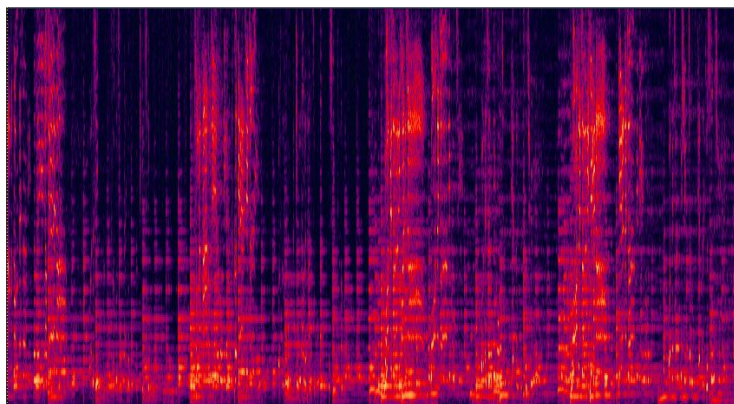




原音



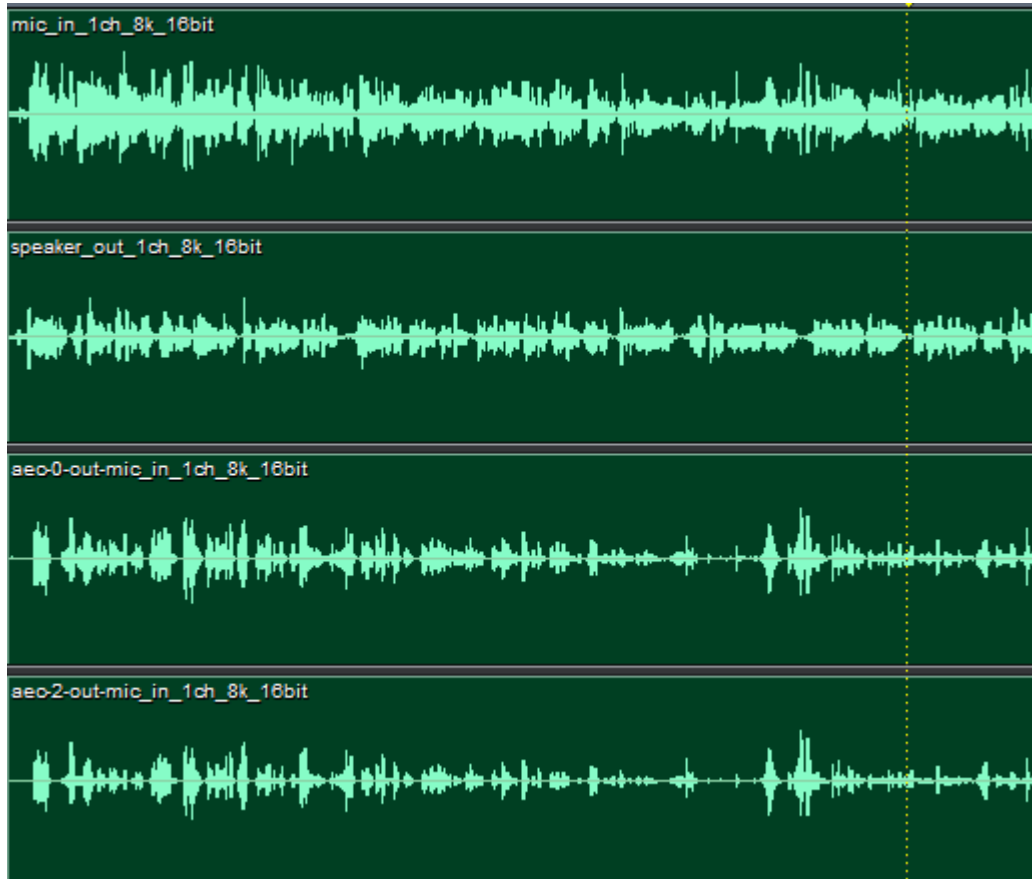
ANC-0



ANC-2

# Echo cancellation功能测试

```
server@ubuntu14:~/test-webrtc-audio/PulseAudio-webrtc-audio-processing/webrtc-audio-processing/build/bin$ ./audio_test_main -aec 0 ../../data/mic_in_1ch_8k_16bit.pcm  
../../data/out/aec-0-out-mic_in_1ch_8k_16bit.pcm 20 ../../data/speaker_out_1ch_8k_16bit.pcm
```



Near-Data-  
Mic-in



Far-Data-  
Speaker-out



aec-0-  
mic-data



aec-2-  
mic-data



THANK YOU

Q&A