利用MATLAB实现的恶搞变声器

刘正浩 2019270103005

李仁轩 2019270103011

唐晨烨 2019270103003

**摘要：**本文通过不同方式对声音信号进行处理，得到了不同效果的变声器。

**关键词：**MATLAB 变声器 共振峰 滤波器

**1. 背景**

在某些场景下，为了避免暴露声音特征，我们需要变声器来对语音进行处理。最常用的变声器有变女声、变男声、变童声等，还有一些变声器具有特效效果，比如变为机器人声。

变声器的基本原理是通过改变输入声音的频率分布特征来改变音色、音调，从而使语音信号听上去与原信号不同。

要实现改变音色的功能，首先要知道音色是什么。语音理论可以告诉我们，语音生成的过程是在激励信号的激励下（即肺部呼气），声音经过谐振腔（如声道），由嘴或鼻辐射声波。习惯上，我们把声道传输频率响应上的极点称之为共振峰，而共振峰的分布特性决定着该语音的音色。也就是说，声音的频率范围高低决定着语音听起来是男声还是女声，而频率的分布特征是每个人独有的声音特征，即不同的频率分布特征决定了每个人语音的音色都是不同的。

所以，要进行变声器的设计，实际上就是通过改变共振峰的位置来改变音色，从而使声音听上去与原来不同。

**2. 方法**

（1）插值

这种方法的原理就是通过插值来改变信号的基频，从而达到变声的效果。

源码如下：

*%利用插值的方法改变基频变声——女变男*

*%读取女声文件*

[sound0,fs] = audioread('woman.wav');

sound1=sound0(:,1);

l = length(sound1);

l2 = l\*2-1;

sound2 = zeros(l2,1);

*%插值*

for  i=1:l2

    if(mod(i,2)<1)

        sound2(i) = (sound1(fix(i/2))+sound1(fix(i/2)+1))/2;

    else

        sound2(i) = sound1(fix((i+1)/2));

    end

end

sound(sound2,fs);

filename=('test\_interpolation.wav');

audiowrite(filename,sound2,fs);

经过我们的实验，使用这种方法确实能够将女声变为有些失真的男声。不过使用这种方法产生的声音信号长短和快慢都会发生变化，所以插值并不是一种较好的变声方法。

（2）频谱搬移

我们假设男声的共振峰比女声的共振峰更靠近原点，那么通过将整个频谱向频率升高的方向搬移的手段，是否就能将男声变为女声？

源码如下：

*%读取声音文件*

[sound0,fs] = audioread('woman.wav');

sound1=sound0(:,1);

ln=length(sound1);

sound2=zeros(ln,1);

*%与正弦信号相乘*

for i=1:ln

    sound2(i)=1.5\*sound1(i)\*sin(2\*pi\*i\*100/44100);

end

sound(sound2,fs);

filename=('test\_spectrumShift.wav');

audiowrite(filename,sound2,fs);

经过我们的实践，这种方法并不能成功将男声变为女声。我们分别输入男声、女声音频文件，得到的语音听起来仍然是男声和女声，只不过像是机器人发出的声音，而非人说话的声音。

（3）通过带通滤波器提取低频成分

这种方法类似使用均衡器对声音信号进行部分频谱成分进行放大，通过这种方式来突出频谱中的低频成分或高频成分，是否能达到改变共振峰位置的目的，从而实现变声？

在试验中，使用MATLAB的Filter Designer构建了一个低通滤波器，参数如下图

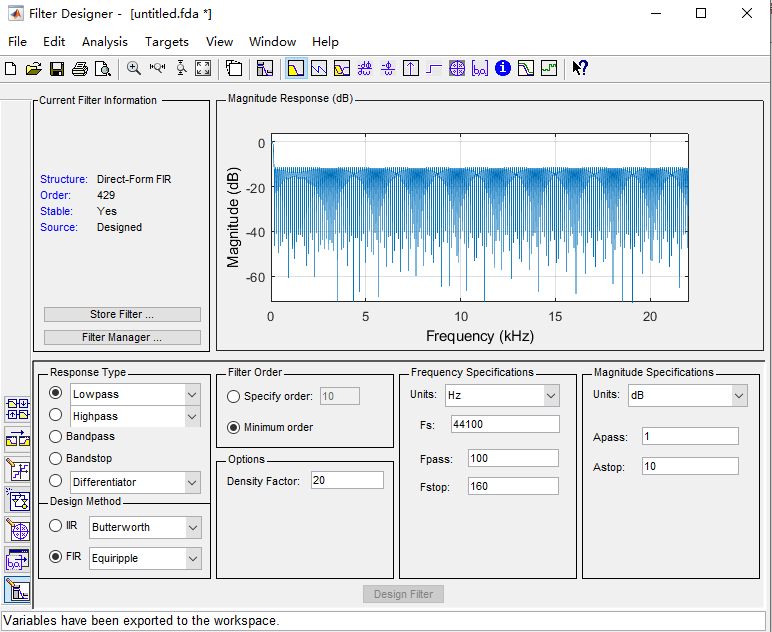


图 1 自行设计的低通滤波器

在得到滤波器参数Num后，可以使用filter()函数进行滤波。源码如下：

*%读取声音文件*

[sound0,fs] = audioread('woman.wav');

sound1=sound0(:,1);

ln=length(sound1);

*%带通滤波*

sound2=filter(Num,1,sound1);

sound(sound2,fs);

filename=('test\_bandpassFilter.wav');

audiowrite(filename,sound2,fs);

经过试验，这种方法同样不能男声变女声或女声变男声，并且得到的声音效果与之前频谱搬移得到的语音听起来基本相同。

可见，对频谱进行简单的搬移或者滤波无法实现我们的目标，而插值的方法虽然能得到效果不错的语音，但速度变化又成为了新的问题。那么能否通过改变速度而不改变频率的方法来使插值得到的语音速度与原来相同呢？

（4）通过TSM（Time-scale modification，时域压扩）处理插值得到的语音

时域压扩（Time-scale modification简称TSM，又叫变速不变调）是一个能够改变音频的语速而不改变其音调的算法，是一个在目前音频信号处理中必不可少的一种重要算法。然而音频信号是多种多样的，包含各种谐波（基音，泛音）冲激和瞬态分量。因为这种广泛的声学和音频特性，目前没有一种能够通用的TSM算法，在实际中需要依据音频特性来选择合适的TSM方法。

（i）重叠叠加（OLA，overlap-add）

重叠叠加是最简单的一种TSM方式，它的做法是，把音频信号分帧（每帧1024个数据点）后，取每帧的前半段进行拼接，这样得到的总时长与原语音的时长是相同的，同时没有改变语音的频率。代码如下：

*%用TSM缩短时长——变速不变调*

*%使用重叠叠加OLA*

*%读取女声文件*

[sound0,fs]=audioread('woman.wav');

sound1=sound0(:,1);

len=length(sound1);

len2=len\*2-1;

sound2=zeros(len2,1);

*%插值*

for i=1:len2

    if(mod(i,2)<1)

        sound2(i)=(sound1(fix(i/2))+sound1(fix(i/2)+1))/2;

    else

        sound2(i)=sound1(fix((i+1)/2));

    end

end

*%TSM*

sound3=zeros(len,1);

m=0;

for i=1:len-50

    sound3(i)=sound2(i+m)+sound2(i+m-512);

    if(mod(i,512)==0)

        m=m+512;

    end

end

sound(sound3,fs)

filename=('test\_TSM\_linear\_connect.wav');

audiowrite(filename,sound2,fs);

用这种方法得到的音频听起来是原速的男声，但由于波形不连续，在频率改变的部分会出现刺耳的噪音。

为了消除这种噪音，我们想到一种方法：将每帧的前五十个数据点改为上一帧的最后一个数据点与本帧的第50个数据点的线性插值，以此来减小两帧之间的不连续性。程序如下：

*%用TSM缩短时长——变速不变调*

*%使用重叠叠加OLA*

*%读取女声文件*

[sound0,fs]=audioread('woman.wav');

sound1=sound0(:,1);

len=length(sound1);

len2=len\*2-1;

sound2=zeros(len2,1);

*%插值*

for i=1:len2

    if(mod(i,2)<1)

        sound2(i)=(sound1(fix(i/2))+sound1(fix(i/2)+1))/2;

    else

        sound2(i)=sound1(fix((i+1)/2));

    end

end

*%TSM*

sound3=zeros(len,1);

m=0;

for i=1:len

    sound3(i)=sound2(i+m)+sound2(i+m-512);

    if(mod(i,512)==0)

        m=m+512;

    end

end

*%平滑*

for i=1:len-50

    if(mod(i,512)==0)

        p=(sound3(i)-sound3(i+50))/50;

        for j=1:50

            sound3(j+i)=sound3(i)-j\*p;

        end

    end

end

sound(sound3,fs)

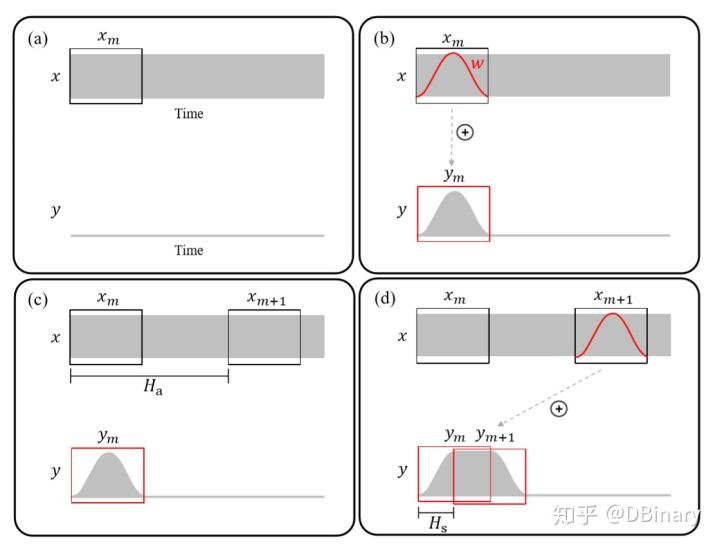
filename=('test\_TSM\_linear\_connect.wav');

audiowrite(filename,sound2,fs);

可以从输出的语音听出，插值后的语音比原来得到的语音噪音略小一些，但效果仍不理想。

（2）加窗后叠加

通过加窗后进行叠加（也就是用一个升余弦信号对帧进行调制），也可以减小这种不连续性。由于升余弦函数的性质，加窗后的帧叠加的幅值保持不变。



程序如下：

*%读取女声文件*

[sound0,fs]=audioread('woman.wav');

sound1=sound0(:,1);

len=length(sound1);

len2=len\*2-1;

sound2=zeros(len2,1);

*%插值*

for i=1:len2

    if(mod(i,2)<1)

        sound2(i)=(sound1(fix(i/2))+sound1(fix(i/2)+1))/2;

    else

        sound2(i)=sound1(fix((i+1)/2));

    end

end

*%汉宁窗*

win=hanning(1024,'periodic');

for i=1:len2

    sound2(i)=sound2(i)\*win(mod(i-1,1024)+1);

end

*%TSM*

sound3=zeros(len,1);

for i=1:512

    sound3(i)=sound2(i);

end

m=512;

for i=513:len

    sound3(i)=sound2(i+m)+sound2(i+m-512);

    if(mod(i,512)==0)

        m=m+512;

    end

end

*%发声并输出*

sound(sound3,fs)

filename=('test\_hanningWindow.wav');

audiowrite(filename,sound3,fs);

*%显示频谱*

sound1\_fft=fft(sound1);

abs\_sound1\_fft=abs(sound1\_fft);

sound3\_fft=fft(sound3);

abs\_sound3\_fft=abs(sound3\_fft);

s=linspace(0,fs,len+1);

x=s(1:len);

subplot (2,1,1);

semilogx(x,abs\_sound1\_fft);

xlim([20,20000]);

title('原音频频谱');

xlabel('Frequency/Hz');

ylabel('Amplitude/dB');

subplot (2,1,2);

semilogx(x,abs\_sound3\_fft);

xlim([20,20000]);

title('变声后频谱');

xlabel('Frequency/Hz');

ylabel('Amplitude/dB');

输出的音频噪声听起来基本上与男声吻合。我们通过傅里叶变换后画出频谱，观察重叠前后的频谱变化。频谱图像见附录图1。

从叠加前后的频谱可以看出，大部分频谱形状并没有发生改变，只是频率变为原来的二分之一。这也符合我们时域的操作.

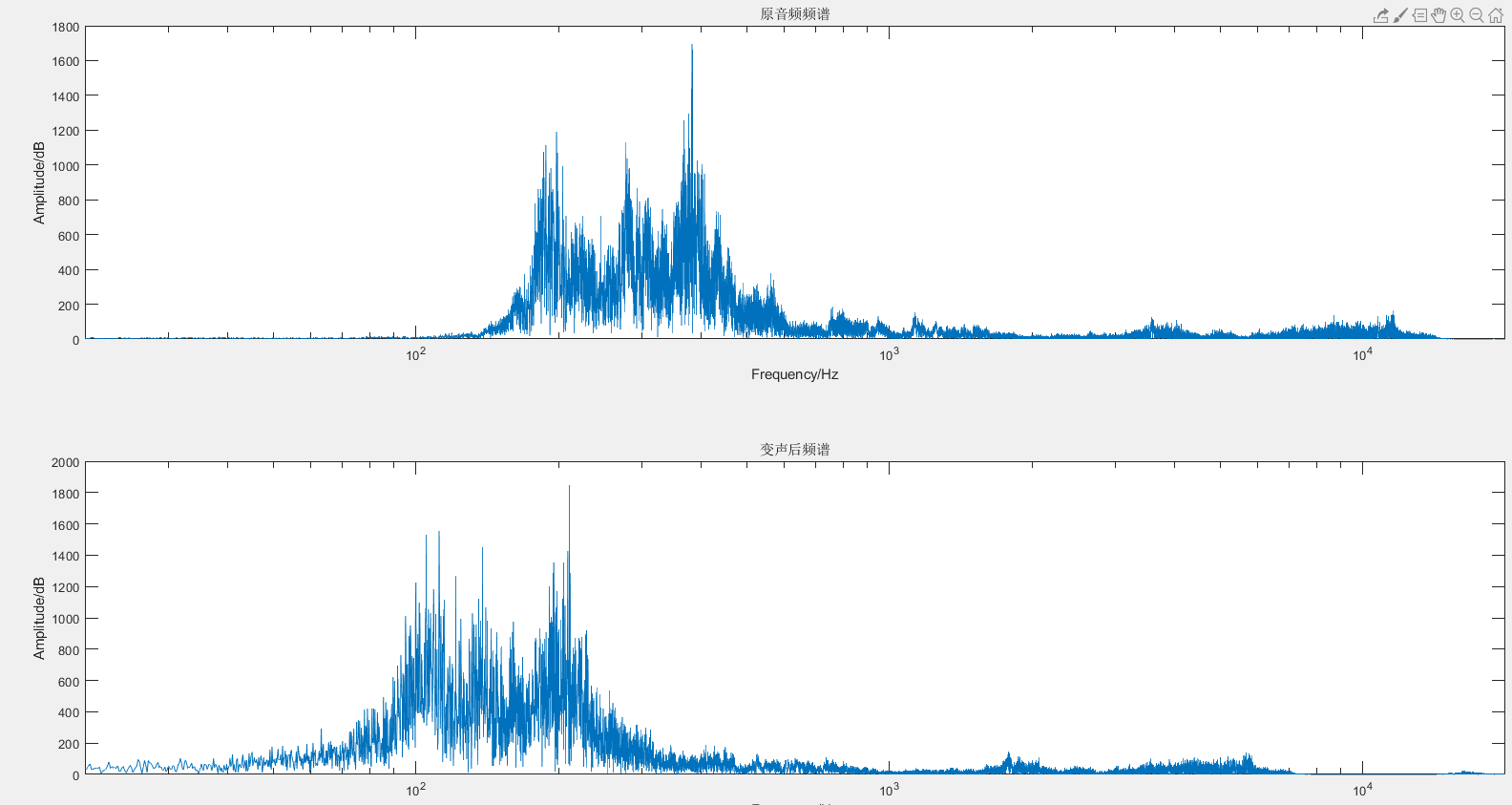
**参考文献**

[1]刘志城,陈超.基于matlab变声器的设计[J].电子世界,2013(12):132-133.

[2]李宝山,戴仁辉,刘客,周鹏,钱婷,孙俊海.Matlab在音频、图像噪声及数据拟合中的应用[J].电脑与信息技术,2019,27(03):10-14.

[3] TSM时域压扩(变速不变调)算法总结 <https://zhuanlan.zhihu.com/p/101198499>

[4] Driedger, J.; Müller, M. A Review of Time-Scale Modification of Music Signals. Appl. Sci. 2016, 6, 57.



附录图 1 叠加前后频谱