# DSP Final Project Team 10

## High Quality Voice Changers

Members: 蕭郁澄 105061245, 吳柏澔 105061132

# I. 問題定義/應用場景

聲音，幾乎可說是無處不在，只要有介質存在的地方，聲音就可以傳播。而所有聲音中，人聲的多變化性讓我們能透過不同的音域、音色辨認不同的人的聲音。因此，我們想做的是一個高品質變聲器，將輸入的人聲，依據想要調整的頻率進行升頻或降頻，最終輸出一段雜訊干擾少、音色接近真人的高品質聲音。

並且，我們不限制輸入和輸出音訊的頻率，也就是說，只要是人聲，無論輸入的聲音是男性 (低頻) 或女性 (高頻) ，變聲器都能將它輸出成一段被改變過頻率的人聲。在此，我們定義無論輸入的音頻是男性或女性的聲音，只要是做提高頻率的處理都稱之為男聲轉女聲 ( Male to Female )，只要是做降低頻率的處理都稱之為女聲轉男聲 (Female to Male ) 。

# II. 問題分析

要實現高品質的變聲器，我們認為大致上會需要處理以下幾個問題:

1. 輸入人聲的信噪比，會影響輸出結果的信噪比。

要確認輸入的音訊為高品質人聲，高品質意味著低雜訊，若是要改變一段本身雜訊比很高的聲音，輸出的聲音的雜訊比也會很高。除非我們先對訊號做降噪音的處理，但是降噪處理多少還是會對聲音訊號造成一些破壞，因此如果能確保輸入的聲音中有越少雜訊越好。

2. 合成方法的選擇，影響音色的保留。

人聲和其他聲音最大的差異來在於音色，每個人聲音中的獨特波形組合使得我們聽到人聲時會認為這段是「人聲」，但是變聲時若只是簡單的在頻譜上移動的話很容易就會破壞波型組合，使得最後合成出來的聲音聽起來不像人聲，因此，要讓合成出來的聲音聽起來像是人聲，使用的合成方法是音色保留的關鍵。

3. 輸出結果是聲音訊號，結果難以量化分析。

嚴格來說，聽覺是一種感官感受，受限於生理結構的不同，每個人對同樣的聲音的感受很可能會不同。並且，聲音訊號直觀上來看只是一維的時域信號，很難對應到人耳所聽到的感受，即使我們能透過傅立葉轉換之類的手段將訊號轉至頻譜分析，但依然很難透過頻譜的圖形對結果量化分析。

# III. 定義方法

我們最初想實現的是 Real-time 的變聲器，蒐集資料後發現有三種不同的方法能做出變聲器。分別是 Phase vocoder [1]、Sinusoidal modeling [2]、跟 Straight vocoder [3]，前兩者都是用到 Short-time Fast Fourier transform (ST-FFT) 來做分析與合成。

經過 MATLAB 實作後，我們發現 Phase vocoder 的處理速度是三者中最快的，但問題則是合成出來的音色會變成不自然的人聲，聽起來像是唐老鴨的聲音。而另外兩種做法能夠改善這個問題，Straight vocoder 將輸入的音訊經過 picth-shifting 後，聲音相對第一種更自然，合成的聲音品質為三者之冠。而 Sinusoidal modeling 的特色則是介於一跟三之間，處理速度夠快，合成聲音品質也不差，因此最終我們決定選擇 Sinusoidal modeling 作為我們 Final project 的主要方法，並嘗試加以改良。

在讀檔時，若音檔為雙聲道時則對兩個聲道作平均來轉成單聲道並做 resample 來統一 sampleing rates，另外因為音訊為長串訊號，因此將音訊切成多段相同 sampling points 的 frame 來做處理並且利用 50% overlap-add 方式來合成。

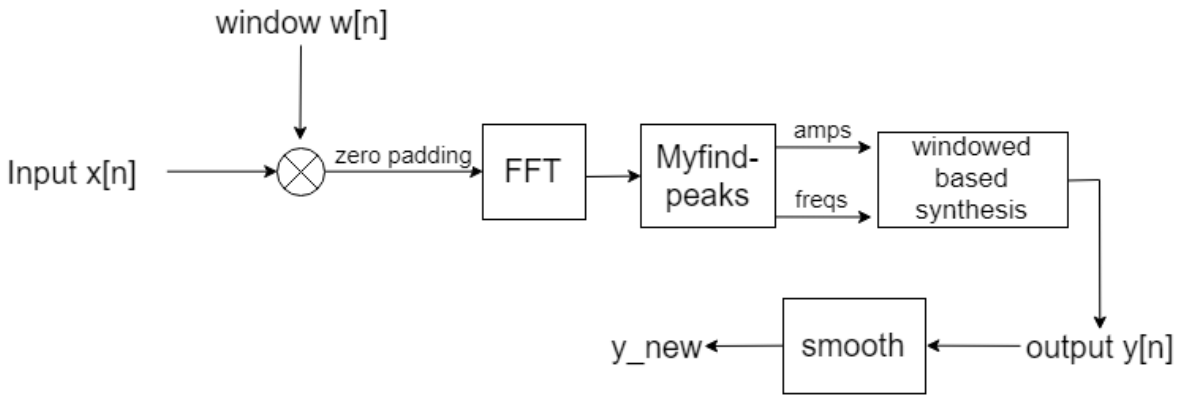



Figure.1 Sinusoidal modeling 的 Flow Diagram

Flow Diagram 說明:

Input 除了輸入音檔之外，還有一個參數"Maxpeaks"可以調整。Maxpeaks 的數值表示每個 frame 中選取 peaks 的數量 (如 Figure.2)。一段 frame 中的 peak 代表著這段區間中的主頻率，Maxpeaks 越大，表示由大到小選取的 peaks 越多。

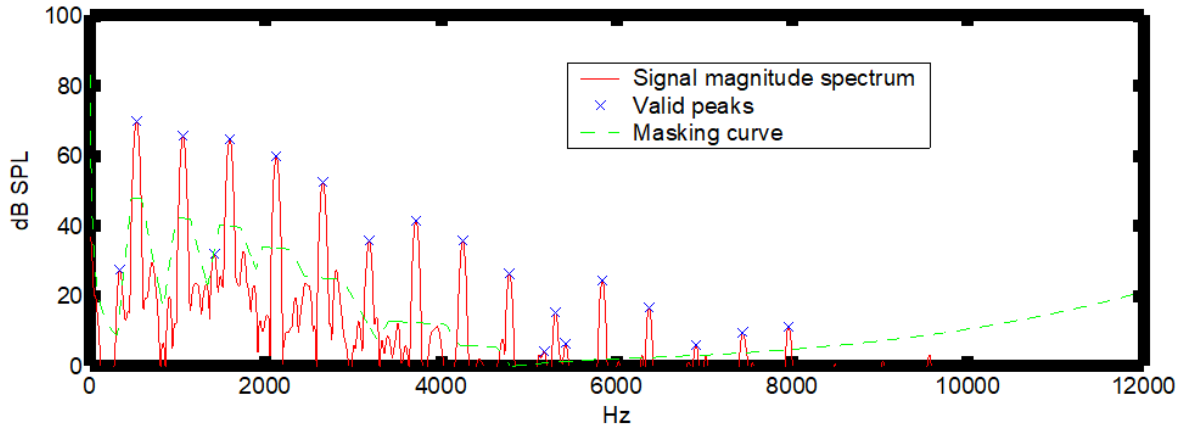
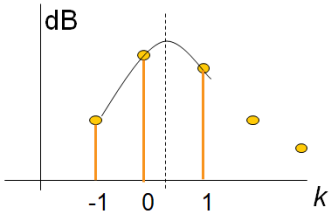



Figure.2 The blue cross is valid peaks

Myfindpeaks function 裡用 quadratic interpolation ( Figure 3. ) 從選出來的 peaks 中得到 amplitude 和 frequency envelop(amps 和 freqs)，

$$L(x) = Ax^2 + Bx + C = A(x-q)^2 + \hat{L}$$
$$L(-1) = L^-, L(0) = L^0, L(1) = L^+$$




$$A = \frac{L^+ + L^- - 2L^0}{2}$$
$$B = (L^+ - L^-)/2$$
$$C = L^0$$

$$q = -\frac{B}{2A}$$
$$\hat{L} = L^0 - B^2/4A$$

Figure.3 Quadratic interpolation

合成方式則是用 reference [2] 中 window based 的公式如 Figure.4，用 amps 和 freqs 以 window 和 sinusoidal 的方式來 sum up all trajectories。

$\{A_m\}_{m=0}^{M-1}$ .....an amplitude envelope

$\{f_m\}_{m=0}^{M-1}$ .....a frequency envelope

Continuous phase updates:

$$\phi_m = \phi_{m-1} + 2\pi\left(\frac{f_{m-1} + f_m}{2}\right)hT$$

$$s[n] = \sum_{m=0}^{M-1} A_m w[n - mh]\cos\left(2\pi f_m nT + \phi_m\right)$$

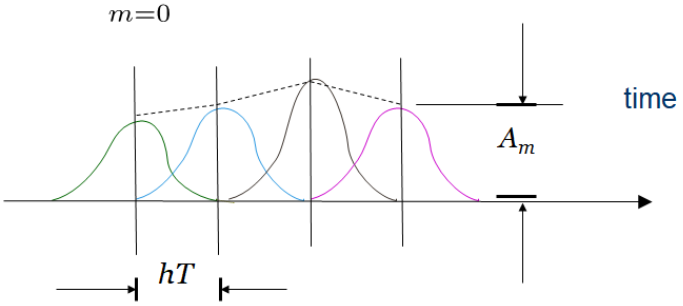



Figure.4 Window-based Synthesis

# IV. 結果分析

## 分析方法:

首先我們想要解決 **II-1** 提到的問題，提高輸入訊號的信噪比，我們採取的方法是使用好的收音設備(ex:錄音筆)，讓聲音本身的信噪比很高。再來要解決 **II-2** 提到的音色保留問題，因此我們採用了三種不同的合成方法，觀察他們合成出來的結果後，我們選擇使用 Sinusoidal modeling 作為我們主要分析的方法。最後，如 **II-3** 所述，此 project 遇到的一大困難乃是結果難以量化分析，而我們採取的方法是分析每個合成結果的「spectrogram」。此種圖形是一種 2D 的三維圖片，結合時間、頻率和能量強度三個維度來呈現音訊資料，如 Supplyment.pdf 中的圖所示，橫軸是時間軸，縱軸是頻率軸，圖形上的顏色表示訊號能量強度。

因此我們一方面透過聆聽的方式比較不同方法不同參數合成出來的結果好壞，另一方面分析 Spectrogram，找到那些我們聽完覺得是好聲音的 Spectrogram 特徵，最後將這些特徵歸納出幾個**觀察**後，回頭調整方法及參數設定，以達到比較好的成果，接下來我們將分成三部分來分析我們的結果。

## 第一部分: 三種方法之比較

這一部份我們著重於 **III.定義方法** 提到的三種方法之比較，並且如同我們在 **I 問題定義/應用場景**所提到的，我們將變聲定義成兩種，一種是將聲音頻率調高 ( Male to Female )，另一種是將聲音頻率調低 (Female to male )。從 Sup-Figure. 1/2/3/4 ，我們可以看到依序是未處理過的聲音、經過 Phase vocoder 升高頻率、經過 Sinusoidal modeling 升高頻率跟經過 Straight vocoder 升高頻率。

我們先聽合成出來的結果，聽起來較似人聲 : Straight vocoder > Sinusoidal modeling > Phase vocoder，再觀察人聲主要分布區段 0~ 2K Hz，可以發現 Sup-Figure.2 的圖形較為鬆散不清楚，而 Sup-Figure 3.和 Sup-Figure.4 都可清楚看到塊狀結構，更仔細比較後發現，Sub-Fugure.4 的塊狀結構清楚又最大，Sub-Fugure.3 的塊狀結構雖然清楚卻零碎，因此我們歸納出**觀察一：「合成出來的聲音，塊狀結構越大越清晰者，聽起來越像人聲。」**

為何選擇升高頻率而非降低頻率，乃是因為降低頻率等於是把 spectrogram 上的資訊更加擠壓在一起，會讓特徵更不容易觀察，從 Sup-Figure. 5/6/7/8，我們能看到如上所述的情形，肉眼並不容易看出差異，這也正是 **II-3** 所提到的難以量化分析之困難。

## 第二部分: Sinusoidal modeling 分析

首先，從 **III.定義方法**中的 amps 跟 freqs 兩種 envelope 來看，主要的功用是去保留人聲的音色，而這些音色的保留程度取決於取 maxpeaks 的數量。然而，maxpeaks 的數量既不能太小也不能太大，例如當 maxpeaks 約小於 5 時，因為 peak 數量取的太少，音色的保留程度太低，在做 interpolation 時會無法形成較接近原始的聲音，部分聲音會缺少甚至於消失，反之，當 peak 取超過 20、30 時，機械聲會更加明顯，原因可以從 Figure.2 來看，peak 取太多有可能會取到噪音的部分，使得在合成過程中產生非人聲的聲音。

可以從 Sup-Figure.9/10/11 來觀察改變 maxpeaks 對人聲降頻的差異，而 maxpeaks 分別取的是 3、10、30，與 Sup-Figure.5 (原始音檔) 比較可以發現 Sup-Figure.9 的圖形非常的零碎，在 frequecy domain 的範圍顯得十分狹窄，而當 maxpeaks 太大時(如 Sup-Figure.11)，可以看到在 time domain 和 frequency domain 上，多出了許多原始音檔未有的資訊，尤其是 frequency domain 特別明顯，另外在升高頻率的部分也可觀察到類似的情形( 如 Sub-Figure.12/13/14)。因此，**觀察二:「maxpeaks 需考慮原始音檔的頻譜特徵以及信噪比例，不能隨意選取。」**

## 第三部分: Smooth

Smooth function 實際上可以分成兩個部分: Wavelet denoising + Low-pass filtering 。從 Sup-Figure. 15/16/17 ，我們可以看到依序是經過 Sinusoidal modeling 降低頻率但尚未 denoise&filter、經過 denoise 但未 filter 過、經過 denoise 和 filter 處理過的聲音。如同前面的分析所述，Straight vocoder 的合成聲音品質最好，因此 Smooth 這個部分我們嘗試讓 Sinusoidal modeling 的 spectrogram 特徵越接近 Straight vocoder 的 spectrogram 特徵。再比較 Sup-Figure. 15 跟 Sup-Figure.16 ，Sup-Figure. 15 的 spectrogram 有淺綠色的塊狀結構集中在 0 ~ 2K，而 Sup-Figure.16 的圖形則可以看出淺綠色的區塊均勻分布在各個頻率，類似於白噪音，使得人耳聽起來的結果不會覺得不自然。

於是我們試圖讓 Sup-Figure. 15 的 spectrogram 看起來更像 Sup-Figure. 16，最後發現 MATLAB Wavelet toolbox 的 wdenoise function 能實現我們的想法，讓輸出的結果 y[n]_smooth 聽起來沒有不自然機械音 。此外，因為人聲的頻率主要分布於 20Hz ~ 2KHz，且人耳對於高頻率的細微變化感受不太到差異，因此我們再將 y[n]_smooth 經過 MATLAB 內建的 Lowpass function 進行濾波 (pass frequency : 4K )，減少高頻的訊號，讓合成出來的 y[n]_new 的音訊能量集中在 0~4K Hz，讓聲音品質更好。

# V. 結論

大致上可以將 Sinusoidal modeling 分成兩個階段。第一階段是音訊分析，讀取輸入音訊在頻域上的 Peaks，得到訊號主要的頻率後再使用 quadratic interpolation 的方式重建頻域訊號，第二部分則是合成音訊，利用 Figure 4.的 cosine function 還原出音色。這樣的方法有好有壞，好處是處理流程簡單，因此處理速度比起 Straight vocoder 來得快上許多，缺點就是還原人聲的部分依然不夠清晰，並且會受到 Peaks 的選取數量影響。於是我們透過分析不同的 spectrogram，得到了三個觀察，分別是:

**1.合成出來的聲音，塊狀結構越大越清晰者，聽起來越像人聲。**

**2.maxpeaks 需考慮原始音檔的頻譜特徵以及信噪比例，不能隨意選取。**

**3.頻譜上若有集中的噪音會影響人聲的判定，因此我們將其轉換成頻譜上均勻分布的白噪音，雖然聲音品質依舊比不上 Straight vocoder，但比起單純的 Sinusoidal 的成果來得好了。**

# VI. 貢獻

## A. 方法貢獻

Phase vocoder 跟 Straight vocoder 分別是參考[1]跟[3]，Phase vocoder 只有更動參數來改變 pitch，而 Straight vocoder 是用他的 function 來做 pitch shifting，因為 2^(1/12)接近音階上的一個半音，我們改動的則是用這個倍數來對基頻做乘除，再合成回去。Sinusoidal modeling 則是用 EE664100-ASAS 所出的 Lab 來修改，參考[2]講義中的公式來合成，最後為了想要達到跟 straight vocoder 的 high qulity，我們多加 filter 和 denoise 來讓聲音聽起來更接近與清楚。

## B. 實作貢獻

Phase vocoder 是已經寫好的 source code，可以直接執行; Straight vocoder 是利用他的 library 中的部分 function 來實作; Sinusoidal modeling 部分是有 strart code ，要實作的部分是 interpolation 的部分，最後 filter 和 denoise 是另外想要額外嘗試的部分。

# VII.組內分工

程式實作 : 蕭郁澄 (30%) 、吳柏澔 (70%)
結果分析 : 蕭郁澄 (70%) 、吳柏澔 (30%)

# VIII. 參考

[1] Pitch Shifting and Time Dilation Using a Phase Vocoder in MATLAB
https://ww2.mathworks.cn/help/audio/examples/pitch-shifting-and-time-dilation-using-a-phase-vocoder-in-matlab.html
[2] EE664100 Analysis and synthesis of digital audio signals-Lec6 SinMod
[3] Straight vocoder https://github.com/HidekiKawahara/legacy_STRAIGHT