Multi-pitch Estimation

Nguyen Tien Dung, Luong Chi Vu, Feng Shuyu
HT055497N, U027353N, HT050618J
{g0505497, u0201065, g0500618}@nus.edu.sg

Abstract

Pitch estimation is an important subproblem of music transcription which transforms an acoustic musical signal into a MIDI-like symbolic representation. The aim of pitch estimation is to find fundamental frequencies of a given music sound. For monophonic music, pitch estimation algorithms have achieved high precision. However, for polyphonic music, where several music tones are played concurrently, the problem has not fully be solved. In this project, we implement two state-of-the-art algorithms and evaluate them by comprehensive experiments.

Index Terms

pitch estimation, multi-F0, predominant-F0, auditory model, harmonic partials

I. Introduction

Pitch estimation is an important subproblem of music transcription which transforms an acoustic musical signal into a MIDI-like symbolic representation. The aim of pitch estimation is to find fundamental frequencies of a given music sound. For monophonic music, pitch estimation algorithms, like YIN [1], have achieved high precision. However, for polyphonic music, where several music tones are played concurrently, the problem has not fully be solved. Existing methods cannot achieve high accuracy when concurrent tones number increases, because the probability of coinciding frequency partials (refer to Chapter 6.4 of [2]) increases as the number of concurrent tones increases. In this project, we implement two state-of-the-art algorithms presented in [3], [4]. The first one is based on auditory model and the second one sums harmonic partials of a tone based on a weighting function learned from sample database. Because there are a lot of commons between these two methods, we combine them in a a single framework. This framework becomes a general framework that can support various harmonic based multiple F0 estimation methods by dividing salience calculating strategies and F0 estimation strategies into two independent parts. One salience calculating strategy can use any F0 estimation strategy and vice versa. The framework save us a lot of time to implement a new method and comparing results of those methods becomes easily. In the following of this report, we will present related work to multiple F0 estimation in section 2. The algorithms used in our implementation are described in section 3. Section 4 describes evaluation database and database building process. Section 5 shows the results of our implementation compared with results in original papers. Finally, Conclusion and future works will be discussed in section 6 and 7.

II. Related Work

According to [5], existing methods for multi-pitch estimation can be classified into five categories. Refer to literature review for comprehensive discussion about related work.

- Auditory-scene-analysis based approach

  Methods in this category generally follow Bregman's theory [6] which primarily concerns with the **psychology** of auditory perception. Auditory scene analysis (ASA) is used here. ASA is a cognitive function used by human auditory system, which is very effective in perceiving and recognizing individual sound sources in mixture signals. Computational ASA (CASA) is usually viewed as a two-stage process: it first decomposes an incoming signal into its elementary time-frequency components, i.e. *sinusoidal tracks* which are time-varying sinusoidal components, and then organized them to their respective sound source. Some acoustic "cues" can promote the grouping of time-frequency components to the same sound source in human listeners. For example, proximity in time-frequency, harmonic frequency relationships, synchronous changes in the frequency or amplitude of the components, and spatial proximity (i.e., the same direction of arrival). [7], [8] introduce methods to extract sinusoidal tracks from music signals. [9], [10], [11], [12] all use auditory-scene-analysis approach based on sinusoid tracks.

- Auditory periphery modeling based approach

  Meddis et al. proposes "unitary" pitch model [13], [14], which addresses the more peripheral (largely **physiological**) parts of hearing. Simulation experiments show that this model is capable of reproducing a wide range of phenomena in human pitch perception.

  [15] extends the unitary model to multipitch estimation in mixture signals. They propose a system where pitch estimation is followed by cancelation of the detected sound, and the estimation is then repeated for the residual signal iteratively. The cancelation is performed either by subband selection or by performing within-band cancelation filtering. Evaluation results are reported only for synthetic and perfectly periodic signals. Later, [16], [17], [2], [3] are all based on auditory periphery modeling.

  [2], [3] propose certain modifications to the unitary pitch model in order to obtain a reliable multiple-F0 estimation tool for use in music signals. The first two steps of the unitary model (bandpass filtering, compression, and rectification) are retained but the ACF calculations are replaced by a technique called harmonic selection and a more complex subband-weighting is applied when combining the results across bands. Computational efficiency is achieved by approximating the HWR operation in the frequency domain according to Equation 1. The method is evaluated by calculating error rates for random mixtures of recorded musical instrument samples. Good accuracy is achieved using analysis frame sizes of 46 ms or longer.

$$\hat{Y}(k) = \frac{\sigma_x}{\sqrt{8\pi}}\delta(k) + \frac{1}{2}X(k) + \frac{1}{\sigma_x\sqrt{8\pi}} \sum_{j=-K/2+k}^{K/2-k} X(j)X(k-j) \tag{1}$$

  where $\delta(k)$ is the unit impulse function and σ_x is the standard deviation of $x(n)$. On the right-hand side of this equation, the first term is a dc-component, the second term represents the spectrum of the input signal,

and the last term represents the beating components of the amplitude-envelope spectrum.

- Signal-model based probabilistic inference

  As musical signals are highly structured, it is possible to state the whole multiple-F0 estimation problem in terms of a signal model, where parameters should be estimated. [18], [19], [4]

  In [4], the salience $s(\tau)$ of a period candidate τ is modeled as

$$s(\tau) = \sum_{m=1}^{M} g(\tau, m)|Y(f_{\tau,m})| \tag{2}$$

  where $f_{\tau,m} = mf_s/\tau$ is the frequency of the mth harmonic partial of a F0 candidate f_s/τ, f_s is the sampling rate, and function $g(\tau, m)$ defines the weight of partial m of period τ in sum. The weight function $g(\tau, m)$ is got through training the sample sets. It is reported that this method outperforms two previous methods of the same author [20], [3].

- Data-adaptive approach

  The source signals are estimated from the data. Typically, it is not even assumed that the source (which here refer to individual notes) have harmonic spectra. However, for real-world signal, the performance of these methods are poor. By placing certain restrictions on the sources, these methods become applicable in realistic cases. Such restrictions are, e.g., **independence of the sources and *sparseness* which means that the sources are assumed to be inactive most of the time**.

  [21] adds temporal continuity constraint to the sparse coding paradigm. It uses the signal model (equation 3), where the power spectrogram of the input, $S(t, f)$ is represented as a linear sum of N static source spectra $S_n(f)$ with time-varying gains at n. The term $S_e(t, f)$ represents the error spectrogram. It also proposes an iterative optimization algorithm which estimates non-negative at n and $S_n(f)$ based on the minimization of a cost function which takes into account reconstruction error, sparseness, and temporal continuity. The algorithm is used to separate pitched and drum instruments in real-world music signals.

$$S(t, f) = \sum_{n=1}^{N} a_{t,n} S_n(f) + S_e(t, f) \tag{3}$$

  [22], [23] are unsupervised learning techniques, and [24] is a classification approach.

- Other approach

  [20] proposes a frequency-domain separation methods for multi-F0 estimation. It is a progmatic approach which decomposes the problem into smaller subproblems, which are noise suppression, predominant-F0 estimation, spectral smoothing and detected sound removing, and solve them one-by one.

  There are batch of other approaches, which can refer to [5].

III. Methods and Implementation

A. Framework

[3] and [4] are two recent methods which have achieved relatively high accuracy. The reasons we choose these two methods are: 1. they are quite new papers, having improvement compared with some old papers; 2. they

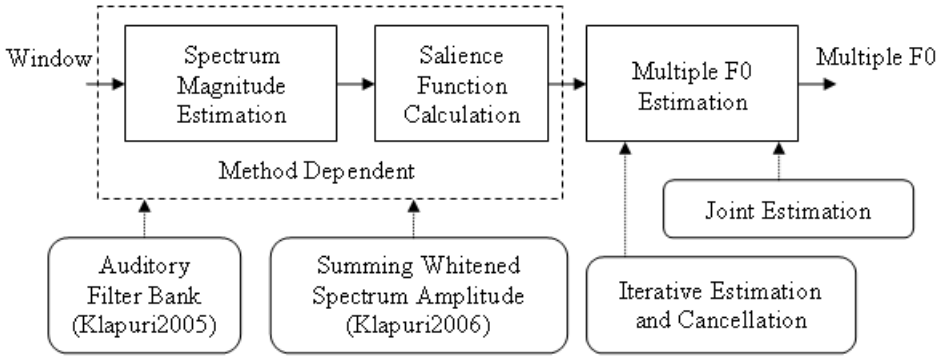


Fig. 1. Framework for spectrum-magnitude based multiple F0 estimation

achieve high accuracy, outperforming other state-of-the-art reference methods; 3. they have well-formed theory, not like [20] which is a engineering work. 4. They are easier to implement. In the following, we will briefly introduce implementation steps of these two methods.

Because there are a lot of commons between Klapuri2005 and Klapuri2006 methods, we decided to combine both methods in a single framework which is showed in Figure 1. There are three main blocks in the framework: Spectrum Magnitude Estimation, Salience Function Calculation, and Multiple F0 Estimation.

Spectrum Magnitude Estimation, and Salience Function Calculation blocks are method dependent. Based on methods used to process the input signal window, auditory filter bank in Klapuri2005 or summing whitened spectrum in Klapuri2006, the output of two blocks is the salience function of fundamental period τ. Based on salience function, the Multiple F0 Estimation output multiple F0 using difference strategy (Iterative Estimation and Cancelation or Joint Estimation). Currently, only Iterative Estimation and Cancelation is implemented because we did not have enough time so we chose one that is the most practical and efficient.

By using this framework, we will save a lot of time when implementing a new method, and make testing and comparing results between methods more easily.

B. Auditory Based Method

[3] uses a computational model of human auditory periphery, followed by autocorrelation function (ACF) estimation to analyze periodicity. Estimation of multiple fundamental frequencies is achieved by canceling each detected sound from the mixture and repeating the estimation for the residual. Computational load of the method remains reasonable since the peripheral hearing model needs to be computed only once. Implementation follows steps:

1) Auditory filterbank and neural transduction

   For the auditory filterbank, “gammatone” filters proposed by Slaney [25] is used. A total of 72 filters between 65Hz and 2.1kHz are employed. The signal at the output of the auditory filter at channel c is denoted by $x_c(n)$.

Hair cell transduction is modeled by a cascade of compression, half-wave rectification and lowpass filtering for each subband signal $x_c(n)$. Compression is implemented by

$$FWC(x) = \{ \begin{array}{ll} x^v & x \geq 0 \\ -(-x)^v & x < 0 \end{array} \tag{4}$$

where $v = 0.33$. Half-wave rectification is defined as $HWR(x) = \max(x, 0)$. Lowpass-filter is to reject the distortion spectrum at twice the center frequency.

2) Periodicity analysis

The object is to find fundamental period candidate τ which maximize $\tilde{\lambda}(\tau)$: $\tau = \arg\max(\tilde{\lambda}(\tau))$. $\tilde{\lambda}(\tau) = (1 + b\ln(f_s/\tau))\lambda(\tau)$, where $b = -0.04$, and $\lambda(\tau)$ is the relative strength, or *salience*, of a fundamental period candidate τ:

$$\lambda(\tau) = \frac{f_s}{\tau} \sum_{j=1}^{\tau/2} (\max_{k \in K_{j,\tau}} (H_{LP}(k)U(k))) \tag{5}$$

where f_s denotes the sampling rate, $H_{LP}(k) = \frac{1}{0.108 f_s k/K} + 24.7$, $U(k) = \sum_c |Z_{c,n}(k)|$ is a summary magnitude spectrum (SMS) across the subbands. $K_{j,\tau}$ defines a range of frequency bins in the vicinity of the j:th overtone partial of the F0 candidate f_s/τ.

3) Iterative estimation and cancelation

An iterative technique is used, where the F0 estimation is followed by the cancelation of the detected sound from the mixture and the estimation is then repeated for the residual signal. The residual SMS recalculated as $U_R(k) \leftarrow \max(0, U(k) - dU_D(k))$, where $d = 0.5$ controls the amount of the subtraction and is a free parameter of the algorithm. $U_D(k)$ is a spectrum of detected sounds.

C. Harmonic Summing Method

[4] is a conceptually simple and computationally efficient method. The salience of a F0 candidate is modeled as a weighted sum of the amplitudes of its harmonics partials: $s(\tau) = \sum_{m=1}^{M} g(\tau, m)|Y(f_{\tau,m})|$. The mapping from Fourier spectrum to a "F0 salience spectrum" is found by optimization using generated training parameters. Three different estimators are proposed. We use iterative estimation and cancelation method in our reimplementation. Implementation follows steps:

1) Spectral whitening

The discrete Fourier transform $X(k)$ of the input signal $x(n)$ passes through a bandpass filterbank. Center frequencies c_b of subbands are distributed uniformaly on the critical-band scale $c_b = 229 \times (10^{(}b+1)/21.4 - 1)$, and each subband $b = 1, ..., 30$ has a triangular power response $H_b(k)$ that extends from c_{b-1} to c_{b+1} and is zero elsewhere. Standard deviations σ_b within the subband b are calculated: $\sigma_b = (\frac{1}{K} \sum_k H_b(k)|X(k)|^2)^{1/2}$, where K is the length of the Fourier transform.

Band-wise compression coefficients $\gamma_b = \sigma_b^{v-1}$ are linearly interpolated between the center frequencies c_b to obtain compression coefficients $\gamma(k)$ for all frequency bins k, where $v = 0.33$ is a parameter determining the amount of spectral whitening.

The whitened spectrum $Y(k)$ is obtained by weighting the spectrum of the input signal by the compression coefficients, $Y(k) = \gamma(k)X(k)$.

2) Calculation of the salience function $s(\tau) = \sum_{m=1}^{M} g(\tau, m)|Y(f_{\tau,m})|$, where the weight function is calculated by $g(\tau, ,) = \frac{f_s/\tau+\alpha}{mf_s/\tau+\beta}$. $\alpha = 27Hz$, $\beta = 320Hz$ are used.

3) Iterative estimation and cancelation

This step is the same as the implementation of [3].

IV. Evaluation and Results

A. Database

To evaluate our implementation of [3], [4], we try our best to follow the author's evaluation method. As there are no standard database for multi-pitch estimation now, we carefully prepare the testing database. We also conduct experiment on note-mixing of different instruments. Details of the experiments are in the following sections.

1) Prepare Data: In the author's paper, he used samples from three different data source.

- University of Iowa website [26] (Free)
- IRCAM Studio Online (Not Free)
- McGill University Master Samples collection (Not Free)

There are together 2842 samples from 32 musical instruments. **Note**: marimba and vibraphone are excluded since they're quite inharmonic.

Advantages:

- Three data sources: provide several different sound production mechanisms.
- Many samples and instruments: provide a variety of spectra.

Our work:

As only the first source is free, we collect samples from The University of Iowa, Electronic Music Studio website [26]. It contains samples of 20 instruments, with a large range (i.e. for piano, from Bb0 to C8). For some wind instruments, there are vibrato and non-vibrato samples. For violin, there are acro and pizz samples.

Although we have only one data source now, which means we cannot provide several different sound production mechanisms comparing with Klapuri's experiment, we can still provide a variety of spectra because there are large range of samples and many instruments.

File Format: The samples are categorized based on the instruments. For example, FLUTE with one sample like ' **flute.vib.pp.B3B4.aiff**' specifies that this sample contains wave for notes range from B3 to B4 (see figure 2). The target is to separate this sample into several single notes. For the above example, we should cut into 12 notes, which got pitch from B3 to B4.

Furthermore, since there are 3 levels of strength in the playing, we choose **mf** as it is the medium level of strength.

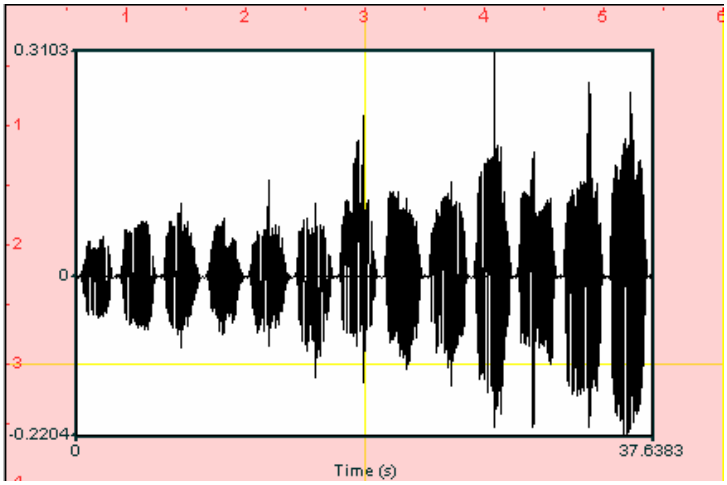


Fig. 2. Waveform of flute.vib.pp.B3B4.aiff

2) Build DSV Database: In the author's paper, data got from the above data sources are randomly mixed to generated test cases. There are two different schemes, *random mixture* and *musical mixture*.

- Random mixture: generated by first allotting an instrument and then a random note from its whole playing range, restricting however, the pitch over five octaves between 65 Hz and 2100 Hz, ranging from $C2$ to $C7$. This was repeated to get the desired number of sounds which were mixed with equal mean-square level (see "Note" below).
- Musical mixtures: generated in a similar manner, but favoring different pitch relationships according to a statistical profile in classical Western music ([27] p.68). In brief, octave relationships are the most frequent, followed by consonant musical intervals, and the smallest probability of occurrence is given to dissonant intervals. Musical mixture is harder to resolve because of coinciding partials.

Note: In [3], simultaneous notes number is 1, 2, 4, and 6 (see Figure 3 in the paper). In [20], the number is from 1 to 6.

Our work:

Samples from Iowa data source have larger range than the author's actave range. We generate Random Mixtures across instruments. There are 632 single note samples, and 1000 samples of 2, 4 and 6 notes, respectively. This is not like the author's methods. We didn't generate Musical mixture set.

The process of building DSV database is

(1) **Build Single Note Database**

The algorithm and concrete steps are as follows:

For every instrument:

For every sample:

Using Praat to roughly estimate region for every wave note.

Manually marking the region for each wave note.

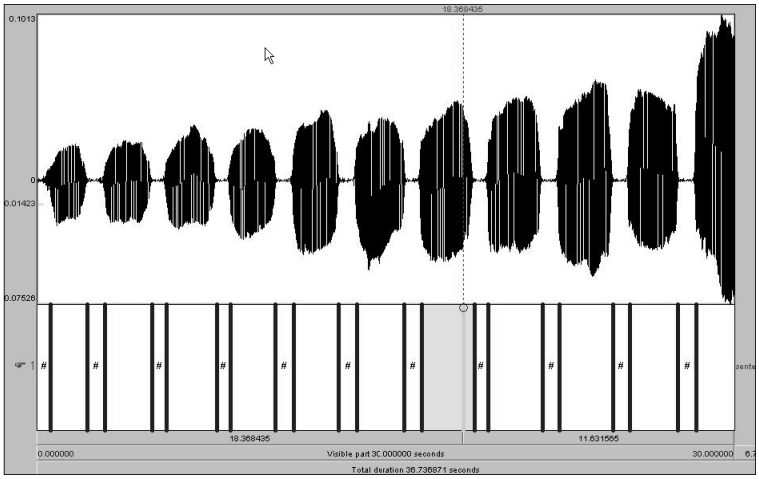


Fig. 3. Labelling in Praat

Run Praat Script to cut into single wave note

For every cut wave note:

Calculate single pitch and detect onset.

Compare with standard pitch.

Then discard bad sample wave note.

a) **PRAAT marking and separating process**

For every sample, we use Praat to pre-estimate the region of the inside notes. The Praat Script will produce a grid file; we call it TextGrid, for the purpose of region labelling.

Suppose the generated file for '$flute_novib_mf_B3B4.aiff$' is '$flute_novib_mf_B3B4.TextGrid$'. We open it in PRAAT to discover further (see figure 3):

The region marked with "♯" will be used as a marker to cut the big sample into small samples. Therefore, in this manual step, we try to correctly mark it as best as possible.

b) **Onset Detection**

At first, we intend to use some open-source Onset detection toolkit. However, after some consideration, based on the definition given in Klapuri paper, 2004, we define the Onset Function ourselves.

Definition:

The onset of the sound is defined to be at the time where waveform reaches 1/3 of its maximum value over the beginning 200ms.

Therefore, the work is quite simple at this stage. We find the onset of the single wave file, and then put it into the single pitch estimation process to find the dominant F0.

c) **MATLAB YIN toolbox for Single pitch estimation**

After getting single wave file, we will then calculate the fundamental frequency of this wave.

The motivation for using YIN is that this toolbox gets a very high accuracy in detecting single pitch, at about 97%.

The Matlab files for this process is in figure 4:

F0 will be calculated as figure 5:

d) **Checking frequency validity**

SinglePitchEstimation

- **CenterClipping.m**: separate a signal into 2 region based on a clip level.
- **DisplayPitchAndWave.m**: display information.
- **PitchDetection.m** : find the pitch for each frame
- **PitchEstimation.m** : do a median of pitches found on a set of frames
- **r.m**: main process entry
- **readaif.m**: Read in an aiff file

Fig. 4. Single pitch estimation files

```
function F0 = YINTest(x, onset, fs)
endpoint = min(round(0.100*fs) + onset, length(x));
s = signal(x(onset:endpoint), fs);
F0 = PitchEstimation(s);
```

Fig. 5. F0 estimation algorithm

Definition: The correct F0 estimate is defined to deviate less than 3% from the reference F0.

Table I is our brief explanation about this definition.

Therefore, with a table of reference F0s in F0LookUpTable.txt, this process is simply to choose note that has frequency matching its name. The naming is done during separation step since you know exactly how many notes in the samples and they are orderly played.

(2) **Build Multiple Notes Database For one sample**

The algorithm is:

For one sample:

For other sample:

Mixing together to produce mixed database.

After the process of building single note is carefully done, this step is just simple. A Matlab file will handle an ADDITION of different instrument single notes and produce a file with correct name. This file is a multinote single sample.

We generate 2, 4 and 6-multinote samples as a evaluation database since Klapuri2005 uses this structure. So, for example of 2-multinote sample, $aflu@A5_aflu@Db6.wav$, contains AtoFlute A5 and AltoFlute Db6. Another

TABLE I

DEFINITION EXPLANATION

Starting at any note, the frequency to other notes maybe calculated from its frequency by:

Freq (N^{th}Note) = Freq (currentNote) x $2^{N/12}$

Therefore, the next node will be differrent from the previous note by 1 semitone, and this difference will have the value of:

$(2^{(1/12)} - 1)$ * Freq (currentNote) = 0.594* Freq (currentNote).

According to hearing theory, our ear can distinguish note with frequency F0 in the range of half a semitone. So the value of half the semitone is: 0.0297*note. That's why the value 3% exists.

TABLE II

NAMING CONVENTION TABLE

Instrument full name	Short name
AltoFlute	aflu
AltoSaxophone	asax
BassClarinet	bcla
BassFlute	bflu
Bassoon	bsoo
BbClar	bcla
Cello	cell
EbClar	ecla
Flute	flut
Horn	horn
Oboe	obeo
Piano	pian
SopSaxophone	ssax
TenorTrombone	trom
Trumpet	trum
Viola	vila
Violin	vili

example of 6-multinote sample, $aflu@Ab3_bflu@Gb5_ssax@Gb4_bcla@Ab2_trom@Gb2_bsoo@F4.wav$, contains AltoFlute Ab3, flute Gb5, SopSaxophone Gb4, BassClarinet Ab2, TenorTrombone Gb2 and Bassoon F4.

Table II is the naming convention table:

B. Evaluation Method

1) Evaluation Criteria:

- predominant-F0 (the first detected F0)

  error rate=(No. of predominant-F0 errors)/(No. of random sound mixture)

  Note: it is No. of random sound mixture, not No. of notes.

- multi-F0

  There are substitution error, deletion error and insertion error.

  error rate= percentage of all F0s that are not correctly detected in the input signals

2) Evaluation Settings: In the experiments, we compare our reimlementation of [3] and [4] with the author's results in the papers. Parameters used in the experiment are the same as the ones in the papers.

- Frame length: 46 ms and 93 ms.
- Number of concurrent notes: 1, 2, 4, 6.

3) Estimate Predominant-F0: We compare Klapuri05, Klapuri06. Error rate refers to "Evaluation Criteria".

4) Estimate Multi-F0: We compare Klapuri05, Klapuri06. Error rate refers to "Evaluation Criteria".

C. Results and Analysis

There are four types of test cases in evaluation database. They are 1, 2, 4, and 6 polytones. There are 632 monotone test cases (due to lack of monotone samples in Iowa MIS), and 1000 test cases for 2, 4 and 6 polytone

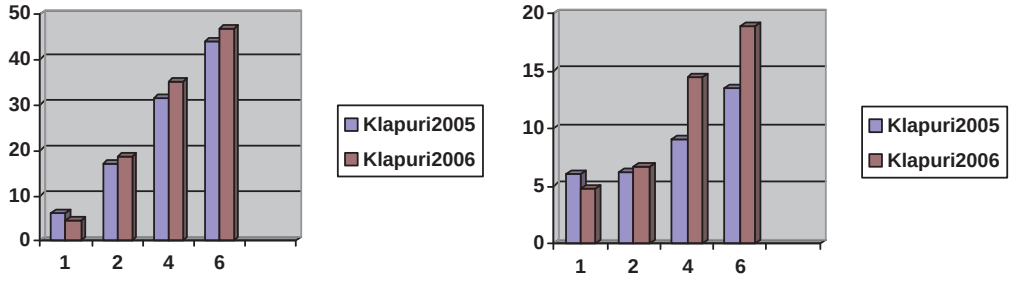
test cases.

Figure 6 and Figure 7 show the final results for 46ms and 93ms frames, respectively. Generally, Klapuri2005 method is better than Klapuri2006 in both multiple-F0 estimation and predominant-F0 estimation but the running time of Klapuri2005 is much slower than that of Klapuri2006 because it used auditory filter bank to produce spectrum magnitude for salience function calculation. The results suggest that auditory-based method is more stable and data independent than summing spectrum amplitude method which multiple harmonic amplitudes with a coefficient function learned from data (we used the same function described in author's paper).

The results are somewhat lower than results presented in author papers. For Klapuri2005, the different is minor but for Klapuri2006 method, the different is clearly distinguished. This different could be caused by the cancellation part, which is not presented clearly in author's papers. We have implemented the cancellation part base on our own understanding.

Another observation is that the results for 93ms analysis frames are better than that for 46ms analysis frames, which is in accordance with the author's results. An important factor is the onset selection method. The analysis frames were positioned immediately at the onsets of the sound. Since the analysis frames might contain noisy beginning transients, longer frames got better results.

There are some other observations. The error rates increase when the number of concurrent polyphonic tones increase. One important reason is that the cancellation part tends to corrupt the remaining signal. In 93ms frame predominant-F0 estimation, the error rate for 1 note is higher than that for 2 notes, which may be cause by the number of samples. There are 632 samples for 1 note while 1000 samples for 2 notes.

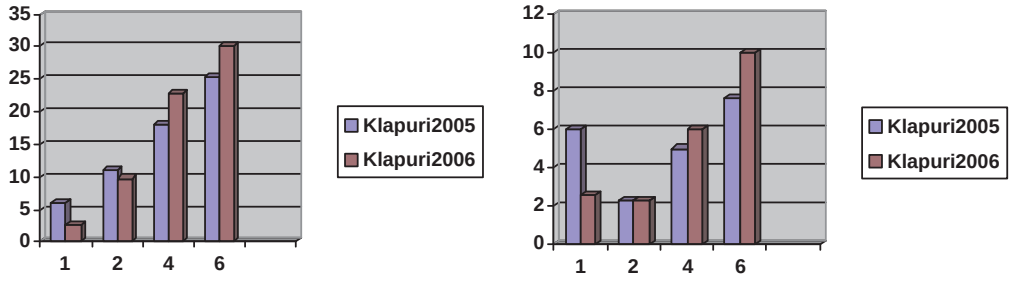


(a) Multiple-F0 Estimation (46ms) (b) Predominant-F0 Estimation (46ms)

Fig. 6. Evaluation Results

V. Conclusion and Future Work

Two state-of-the-arts multiple F0 estimation methods were implemented in a single framework. This framework is general enough to support new spectrum-based multiple F0 estimation methods. The auditory-based method is more accurate than summing spectrum amplitude method but its run time is much slower.



(a) Multiple-F0 Estimation (93ms) (b) Predominant-F0 Estimation (93ms)

Fig. 7. Evaluation Results

For future work, the cancellation part of iterative estimation and cancellation strategy should be carefully study and re-implemented to get closer results to those presented in original papers. Speed improvement is also an important task if we want to apply above methods to a real music transcription application. Finally, joint F0 estimation strategy could be consider to be implemented if we need better results because this strategy gives more accurate F0 estimation but the trade-off is runtime.

ACKNOWLEDGMENT

We'd like to thank Dr Wang for his guidance in this project. We also thank Adrian for sharing his codes and report.

REFERENCES

[1] A. de Cheveigne and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *J. Acoust. Soc. Am.*, 2001.

[2] A. Klapuri, "Signal processing methods for the automatic transcription of music," Ph.D. dissertation, Tampere University of Technology, Finland, 2004.

[3] A. Klapuri, "A perceptually motivated multiple-f0 estimation method," in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, New York, Oct. 2005.

[4] A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," in *7th International Conference on Music Information Retrieval (ICMIR)*, Victoria, Canada, Oct. 2006.

[5] A. Klapuri, "Automatic music transcription as we know it today," in *Journal of New Music Research*, 2004, pp. 269–282.

[6] A. Bregman, *Auditory Scene Analysis*. Cambridge, MA: MIT Press., 1990.

[7] X. Serra, "Musical sound modeling with sinusoids plus noise," in *Roads, C., Pope, S., Picialli, A., De Poli, G. (eds.), Musical signal processing*, 1997.

[8] S. Levine, "Audio representation for data compression and compressed domain processing," Ph.D. dissertation, University of Stanford, 1998.

[9] K. Kashino and H. Tanaka, "A sound source separation system with the ability of automatic tone modeling," in *Proc. International Computer Music Conference*, Tokyo, 1993, pp. 248–255.

[10] K. Kashino and N. Hagita, "A music scene analysis system with the mrf-based information integration scheme," in *Proc. of International Conference on Pattern Recognition (ICPR)*, vol. 2, 1996, pp. 725–729.

[11] A. Sterian, "Model-based segmentation of time-frequency images for musical transcription," Ph.D. dissertation, University of Michigan, MI, USA, 1999.

[12] D. Godsmark and G. Brown, "A blackboard architecture for computational auditory scene analysis," *Speech Communication*, vol. 27, pp. 351–366, 1999.

[13] R. Meddis and M. J. Hewitt, "Virtual pitch and phase sensitivity of a computer model of the auditory periphery. i: Pitch identification," in *Journal of the Acoustical Society of America*, 1991, pp. 2866–2882.

[14] R. Meddis and L. O'Mard, "A unitary model of pitch perception," in *Journal of the Acoustical Society of America*, 1997, pp. 1811–1820.
[15] A. Cheveigné and H. Kawahara, "Multiple period estimation and pitch perception model," *Speech Communication*, vol. 27, pp. 175–185, 1999.
[16] T. Tolonen and M. Karjalainen, "A computationally efficient multipitch analysis model," *IEEE Trans. Speech Audio Processing*, vol. 8, pp. 708–716, 2000.
[17] K. D. Martin, "Sound-source recognition: A theory and computational model," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, MIT, June.
[18] M. Goto, "A predominant-f0 estimation method for realworld musical audio signals: Map estimation for incorporating prior knowledge about f0s and tone models," in *Proc. Workshop on Consistent and reliable acoustic cues for sound analysis*, Aalborg, Denmark, 2001.
[19] M. Davy and S. Godsill, "Bayesian harmonic models for musical signal analysis," *J.M. Bernardo, J.O. Berger, A.P. Dawid, and A.F.M. Smith (Eds.), Bayesian Statistics VII*, 2003.
[20] A. Klapuri, "Multiple fundamental frequency estimation by harmonicity and spectral smoothness," *IEEE Trans. Speech and Audio Processing*, vol. 11, pp. 804–816, 2003.
[21] T. Virtanen, "Sound source separation using sparse coding with temporal continuity objective," in *Proc. International Computer Music Conference*, Singapore, 2003.
[22] P. Smaragdis and J. C. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 2003.
[23] S. A. Abdallah and M. D. Plumbley, "Unsupervised analysis of polyphonic music by sparse coding," *IEEE Transactions on Neural Networks*, vol. 17, pp. 179–196, Jan. 2006.
[24] G. E. Poliner and D. P. W. Ellis, "A classification approach to melody transcription," in *6th International Conference on Music Information Retrieval*, London, UK, 2005.
[25] M. Slaney, "An efficient implementation of the patterson holdsworth auditory filter bank," *Perception Group, Apple Computer Tech. Rep. 35*, 1993.
[26] T. U. of Iowa Musical Instrument, "http://theremin.music.uiowa.edu."
[27] C. L. Krumhansl, *Cognitive Foundations of Musical Pitch*. New York: Oxford Univ. Press, 1990.