A SINGING TRANSCRIPTION SYSTEM USING MELODY TRACKING ALGORITHM BASED ON ADAPTIVE ROUND SEMITONE (ARS) PLUS MUSIC GRAMMAR CONSTRAINTS

[1]Chong-kai Wang, [2]Ren-yuan Lyu, [3]Yuang-chin Chiang

[1,2]Department of Electrical Engineering, Chang Gung University, Taoyuan 333, Taiwan

[3]Institute of Statistics, National Tsing Hua University, Hsinchu, 300, Taiwan

[1]ckwang.ee86@nctu.edu.tw [2]rylyu@mail.cgu.edu.tw
[3]chiang@stat.nthu.edu.tw

ABSTRACT

In this paper, an approach for melody tracking is proposed and applied to applications of automatic singing transcription. The melody tracker is based on Adaptive Round Semitones (ARS) algorithm, which converts a pitch contour of singing voice to a sequence of music notes. The pitch of singing voice is usually much more unstable than that of musical instruments. A poor-skilled singer may generate voice with even worse pitch correctness. ARS deals with these issues by using a statistic model, which predicts singers' tune scale of the current note dynamically. Compared with the other approaches, ARS achieves the lowest error rate for poor singers and seems much more insensitive to the diversity of singers' singing skills. Furthermore, by adding on the transcription process a heuristic music grammar constraints based on music theory, the error rate can be reduced 20.5%, which beats all the other approaches mentioned in the other literatures.

1. INTRODUCTION

Automatic musical transcription has been an interesting issue in multimedia applications. A system that is capable of transcribing orchestra input to musical notes is called a polyphonic transcription system. In contrast, a monophonic system takes as input single instrument sound or human singing. Generally speaking, a transcription system for musical instrument performs quite well satisfactorily because the input instrument sounds have contained accurate pitch information and thus are easily decoded. In the case of human singing, the task becomes more difficult due to the fact that a human singer can sing inconsistently and easily drifted away from standard musical scale. Unlike a musical professional, a common person might find it difficult to write down the melody in his mind or in his ears. He can choose to sing to a musical transcription system, but quite rarely he can sing as accurate as an instrument. This will deteriorate a transcription system expecting standard pitch scale input.

A singing transcription system accepts human singing voice, and transcribes it to musical notes. The fact that human singing rarely follows standard musical scale makes the implementation a challenge. There are two basic problems in a human singing transcription system: note tracking and melody tracking. Note tracking is to segment singing waveform into voiced/unvoiced region, for voiced region we mean a waveform segment where pitch period can be estimated. Each pitched region is often regarded as a musical note, and in music score, a syllable of a lyric word often mates one or many music notes. Melody tracking is to map singing pitches into musical notation correctly, even if a poor singer generates a bad melody.

Authors have explored a method for Note tracking using spectral standard deviation (see [1]). This paper reports our singing transcription system, in addition to detail techniques in Melody tracking task. In Melody Tracking module, a statistic model called Adaptive Round Semitones (ARS) and a music grammar constraint called Tune Map is applied to improve the robustness of the system.

This paper is organized as follows. Section 2 is a brief description for old Melody tracking algorithms. Section 3 is for the newly proposed ARS algorithm. Section 4 explores the mathematical calculations of Constraint table and Tune Map model. Section 5 is conclusion.

2. MELODY TRACKING

In this paper, we describe a singing transcription system, which could be divided into two modules. One is for the front-end voicing processing, including voice acquisition, end-point detection, and pitch tracking. The other is for the melody tracking, which maps the relatively variation pitch level of human singing into accurate music notes, represented as MIDI note number. The overall system block diagram can be shown as Figure 1.

2.1. Front-end voice processing

Singing voice utterance is first divided into voiced and unvoiced section. After segmenting the voiced region of waveform, extracting the pitch or fundamental frequency (F_0) of human singing signal follows. Each segmented section will correspond to a musical note. Here, a temporal pitch detection algorithm plus rule-based filtering is used to get the pitch contour.

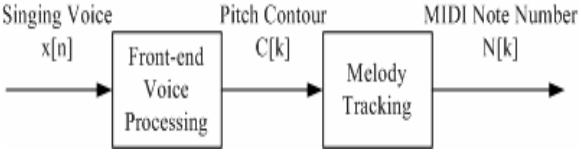

Figure 1: *System block diagram*

2.2. Converting pitch to MIDI number

The Melody Tracking module of the system deals with the average pitch value *f* (frequency in Hertz) for each note and transcribes them into music note representation, which have been defined as MIDI numbers for all possible semitones in the computer music processing domain. The pitch value and the corresponding MIDI number *N* can be related as the following formula:

$$N = 69 + 12 \times \log_2 \frac{f}{440} \tag{1}$$

For example, the note middle A with pitch being 440 *Hz* will be assigned a MIDI number 69, and the note A#, which is one semitone higher than middle A, is then assigned 70 as its MIDI number. On the equal tempered scale, the pitch interval between two consecutive MIDI numbers could be divided into 100 *cents*, e.g., the note middle A is 6900 cents. Thus, 100*N* is called pitch-in-cents.

2.3. Round MIDI and modification

A straightforward approach is to convert pitch-in-cents to MIDI number by rounding the real value of pitch-in-cents/100 to the nearest integer. This is termed as *Round MIDI*.

However, human's pitch generation process is much more unstable than that of musical instruments. An untrained singer rarely generates voice with correct pitch value. It was said that only about 1 in 10000 people claim to have tone-absolute pitch perception/generation [2]. Furthermore, the pitch level can drift upward or downward due to the singer's mood, and thus the pitch interval may not keep constant during singing. The experiments done in the other literatures show that the straightforward Round MIDI algorithm did not work well for melody tracking, thus several improvements have been proposed as follows.

- McNab's *Moving Tuning* [3] and *"Round Intervals"* rounds the difference of pitch in cents between adjacent pitched regions and treats them as the difference of MIDI number between adjacent notes.
- *Haus'* Modified *Round MIDI* [4] estimates an offset from all notes, adjusts all notes with this offset, and then does the same rounding as Round MIDI.

Here, we design a new method called Adaptive Round Semitones (ARS) which was shown to be the most effective among those in the literatures. [5]

3. ADAPTIVE ROUND SEMITONES (ARS)

The ARS was designed on the following three assumptions:

1. Most singers have only tone-relative pitch.
2. The tune scale of human singing voice is not necessarily 100 cents per semitone.
3. The tune scale will probably change with time while singing and be dependent on previous notes.

For example, while a singer singing a rising melody sequence, he will inadvertently sing higher and higher. For a poor singer, the melody sung outward does not always match what he wishes to sing, although the listeners usually can understand what he sang. This is because he has a changeable and unusual tune scale. One important motive of designing ARS is to deal with these issues.

3.1. Statistic model

The block diagram of ARS is shown in Figure 2, where the input is the sequence of the pitch in *cents***,** C[k], and the output is the sequence of the MIDI numbers, or equivalently the music notes, N[k].

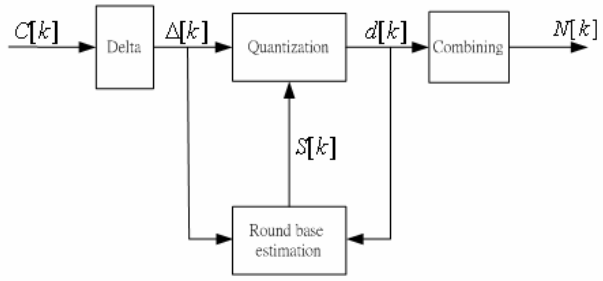

Figure 2: *Adaptive Round Semitone (ARS)*

All the state variables shown in Figure 2 will be described as follows:

$$\Delta[k] = C[k] - C[k-1] \tag{2}$$

$$d[k] = \begin{cases} R\left(\dfrac{\Delta[k]}{100}\right) & 1 \le k \le T \\ R\left(\dfrac{\Delta[k]}{S[k]}\right) & T+1 \le k \le K \end{cases} \tag{3}$$

$$S[k] = \begin{cases} \dfrac{\sum_{m=1}^{T} a[m] \cdot \Delta[k-m]}{\sum_{n=1}^{T} b[n] \cdot d[k-n]} & T+1 \le k \le K \\ 100 & 1 \le k \le T \end{cases} \tag{4}$$

$$N[k] = N[k-1] + d[k] \tag{5}$$

where Δ[k] is the input pitch difference (in cents), d[k] is the difference of the output MIDI numbers, N[k] is the final output MIDI number sequences, and $R(x)$ is a rounding function which rounds off the real number x to the nearest integer, and S[k] is the estimated tuning scale, which is initially set to 100 according to the assumption of equally tempered scale and is dynamically adjusted according to the information provided by the previous music notes.

Here we assume an autoregressive moving average (ARMA) model with parameter set {T; a[1], a[2],…a[T]; b[1], b[2],…b[T]} to estimate S[k]. The parameter set was determined empirically in this paper.

3.2. Rule-based post-processing

Furthermore, a lower and upper bounds for the tune scale S[k] is set to avoid over-tuning. A center-clipping and side-clipping process will be done to compensate for the overly tuning scale. We clip the value if tune scale which is below ±5% and above ±15%.

Besides, some local rules are added to deal with some exceptional cases:

- *Initial effect of tone*: if there are continuous notes identified to the same MIDI number, the tune scale S[k] will be set back to 100 as the initial value.
- *Memory effect of tone*: if pitch of the present note is near the second previous note, they will be identified to the same MIDI number.

4. TUNE MAP MODEL

4.1. Musical grammar constraint

According to well tempered tuning system, an octave is equally divided into twelve notes; a scale is a series of notes selected among these twelve notes. Each of these notes is called a degree, and each degree has its own name designated by a Roman numeral. Most songs are composed conforming to an underline music grammar. Tone distribution of a song melody is not fully randomized. A well-composed song could be clustered in music theory by its tunes, such as C-Major, E-minor…etc. A Major scale is a structure of tones, namely "Do", "Re", "Mi", "Fa", "So", "La", "Si", "Do" (Degree name: Ⅰ, Ⅱ, Ⅲ, Ⅳ, Ⅴ, Ⅵ, Ⅶ, Ⅰ), where the differences in semitones of each continuing tones are 2,2,1,2,2,2,1. Such a tone structure could be looked upon as a grammar to put a constraint on the possibility of notes, which can be followed by another note.

For example, if the pitch differences in semitones of three notes N_1, N_2, N_3, were identified as 1 and 2, then these three notes will be one of the sequences "Mi-Fa-So" and "Si-Do-Re". If there comes a confusing note N_4 which is 1 or 2 semitones higher than N_3. We can easily determine that N_4 is 2 semitones higher according to the Major scale structure distribution.

Form music theory of Bach's well tempered scale, a semitone-level shifting of a sequence melody will not change its inner relation. The singer's singing tune is not unique results from his/her singing key. Equal temperament made possible modern keyboard music with full modulation between all keys. The music grammar constraint is a method that adjusts singer's tune and song's tune in semitone-level.

The idea that adds constraint of music grammar to help track the melody comes from the language modeling technique widely used in speech recognition research, which improve the overall recognition performance significantly.

4.2. The Constraint Table

The Constraint Table lists the rules of melodies based on musical theory. We desired to know the degree name of the first note of a singing utterance. And, according to that, all candidate tones will be constrained. As shown in Table 1, if the degree name of the first note is " Ⅰ " in Major scale. All possible tones (between 2 octaves range) are {0, +2, +4, +5, +7, +9, +11} and {-1, -3, -5, -7, -8, -10} semitones relating to this tone. The tone offset is cyclic repeat every 12 semitones, so the Constraint Table lists 13 Tone offset simply.

Because that Major and Minor scale are cyclic shift in a similar manner, we can handle both Major and Minor scale songs using the same constraint table. Note that the probably degree name is not referring to the tune name, and it is only a symbol presenting the position of the tone in a tune structure. It is like a moving ruler with 7 candidates parting from 2, 2, 1, 2, 2, 2, 1 semitones, moving around the melody line. Aligning the ruler and the singing tones, seeking for fully match.

Tone offset	Probably Degree Name						
0	Ⅰ	Ⅱ	Ⅲ	Ⅳ	Ⅴ	Ⅵ	Ⅶ
+1			Ⅲ				Ⅶ
+2	Ⅰ	Ⅱ		Ⅳ	Ⅴ	Ⅵ	
+3		Ⅱ	Ⅲ			Ⅵ	Ⅶ
+4	Ⅰ			Ⅳ	Ⅴ		
+5	Ⅰ	Ⅱ	Ⅲ		Ⅴ	Ⅵ	Ⅶ
+6				Ⅳ			Ⅶ
+7	Ⅰ	Ⅱ	Ⅲ	Ⅳ	Ⅴ	Ⅵ	
+8			Ⅲ			Ⅵ	Ⅶ
+9	Ⅰ	Ⅱ		Ⅳ	Ⅴ		
+10		Ⅱ	Ⅲ		Ⅴ	Ⅵ	Ⅶ
+11	Ⅰ			Ⅳ			
+12	Ⅰ	Ⅱ	Ⅲ	Ⅳ	Ⅴ	Ⅵ	Ⅶ

Table 1: *The Constraint Table*

4.3. Tune Map

Music grammar constraints described above can be collected to a constraint table (Table 1). It shows the rules of music grammar according to music theory. In tone-relative pitch notion, we are interested in pitch difference of neighboring two notes. We construct a binary tree-like path map called Tune Map (Figure 4) which shows the moving path of notes. There are two kinds of nodes in Tune Map, one is "one-branch-node", and the other is "two-branches-node". Each node refers to a tone; the value on each note refers to the semitone offset between this tone and the first tone of the whole utterance, and the set besides the note refers to the probably degree name of this note.

We make a one-branch-node if the pitch interval rounding error of two neighbor notes is not exceeding ±20%. In the other case, we make it as a two-branches-node, and two of the branches will be connected to the node having tone upper and lower boundary of the rounding scale. The root node refers to the first note of the utterance, and then we add new branches and nodes for next notes. While adding a new node, we should look up the Constraint Table to certificate the node. Certification process doing logic AND for current probably degree name and the

corresponding probably degree name of current tone offset. In the last stage, we can connect a tone path form the survival node to the root node. Finally, the melody of the singing utterance can be tracked.

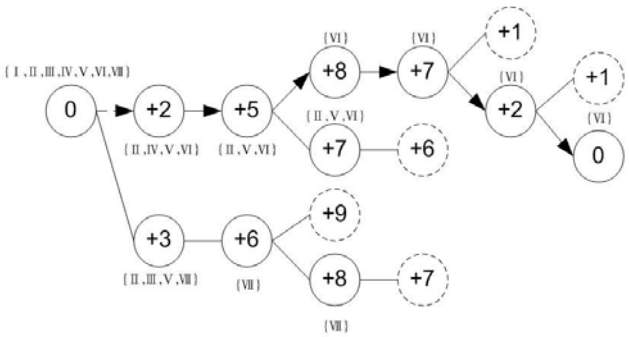

Figure 4: *Tune Map*

4.4. Multi-path of Tune Map

The result of the Tune Map may not be single. That is say, there are more than one path exist in Tune Map. This is because the number of notes of a singing utterance is not many enough to make a fully constraint. Because that, while establishing a two-branches-node in the Tune Map, we choose the nearer one from upper and lower bound of the rounding scale, and put it in the upper node. In the last step of the Tune Map, the uppermost and alive node is the starting node of trace back. Note that if the uppermost node is always alive in each node step, the result is the same as pure Adaptive Round Semitones.

5. PERFORMANCE

Experiments have been carried out with different approaches mentioned in the previous sections for comparison with the proposed one. Testing samples were collected from 13 people. Each person sings 9 pieces of melodies. While singing, the singing key and tempo is free. These testing pieces include melodies from several regions and multiple variations. Total number of testing notes is 2885. We consider them as poor singers if the error rate of Round MIDI algorithm exceeds 30%. There are 4 people (869 notes) in the set of poor singers, and 9 people (2016 notes) in the set of normal singers.

The error rates for each implemented algorithms were listed as in Table 2. , where error rates for all singers, normal singers, and poor singers are listed for all 6 melody tracking algorithms implemented or newly proposed in this paper. One can see that ARS plus music grammar not only achieves the lowest error rates 20.2% for all singers but also achieves the lowest performance degradation from the set of normal singers to the set of poor singers, i.e., least insensitive to the variety of multiple singers. With exception to the ARS plus Tune Map, even though prime ARS does not beat McNab's Moving Tuning algorithm in error rate for all singers, it indeed achieves less performance degradation (5.7%) than the others.

6. CONCLUSION

Music transcription is an interesting and useful technique. In early systems, they are short of any information help from music theory. Authors not only simulating human's performs on singing by an ARMA model but also add constraints from music theory to improve the totally performance. ARS plus Tune Map is a robust analysis algorithm in human singing signal transcription, and it makes "adding musical knowledge" no longer a future work in this kind of system.

7. REFERENCES

[1] Chong-kai Wang, Ren-yuan Lyu, and Yuang-chin Chiang. "An Automatic Singing Transcription System with Multilingual Singing Lyric Recognizer and Robust Melody Tracker," (Submitted to EuroSpeech 2003.)

[2] Profita, J. and Bidder, T.G.,"Perfect Pitch," *American Journal of Medical Genetics*, 29, 763-771, 1988

[3] R.J. McNab, L.A. Smith, and I. Witten, "Signal Processing for Melody Transcription," *Working Paper 95/22, Dept. of Computer Science*, University of Waikato, New Zealand, August 1995.

[4] G. Haus, and E. Pollastri, "An audio front end for query-by-humming systems," *In Proc. of International Symposium on Music Information Retrieval (ISMIR)* ,2001.

[5] Chong-kai Wang, Ren-yuan Lyu, and Yuang-chin Chiang, "A Robust Singing Tracker Using Adaptive Round Semitones (ARS)" *In Proc. of 3rd International Symposium on Image and Signal Processing and Analysis (ISPA03)*, Italy, 2003.

	Round MIDI	McNab's Moving Tuning	Haus' Modified Round MIDI	Round Intervals	**(Prime) Adaptive Round Semitone**	**Adaptive Round Semitone plus Tune Map**
Error rate for normal singers	28.1%	20.7%	29.6%	20.7%	**22.8%**	**19.8%**
Error rate for poor singers	33.6%	25.3%	33.1%	25.3%	**24.1%**	**20.5%**
Error rate for all singers	29.8%	22.0%	31.1%	22.0%	**23.1%**	**20.2%**
Relative Performance degradation from normal to poor singers	19.6%	22.2%	11.8%	22.2%	**5.7%**	**3.5%**

Table 2: *The experimental result for melody tracking algorithms implemented and newly proposed in this paper.*