Multimed Tools Appl (2011) 53:253–269
DOI 10.1007/s11042-010-0510-6

An interactive Whistle-to-Music composing system based on transcription, variation and chords generation

Hung-Che Shen · Chung-Nan Lee

Published online: 6 April 2010

Abstract Most people can whistle, sing or hum a song that they are familiar with. However, it is rather difficult for common people without formal music skills or training to compose a song. In this paper, we have constructed an interactive Whistle-to-Music composing system with which a user can compose MIDI format music by whistling into a microphone. The user can experiment with computer-aided composition such as melodic variation and chords generation. The transcription speed is so fast that MIDI notes can be echoed as feedback immediately while the user is still whistling. For computer-aided composition, the given melodic fragments are developed and accompanied with stylish chords generation. We then study users' experiences and present the results of an experiment which tests how accurately people whistle along a target melody. This preliminary prototype of the proposed system is proved to be a handy tool for computer-aided MIDI music creation.

Keywords Whistle-to-MIDI · Transcription templates · Lead sheet notation · Melodic variation · Music templates

1 Introduction

In contrast to new digital musical instruments designed for professional and trained musicians, transcribing a monophonic source (single voice) for music composition has received little attention. This negligence is unfortunate, since one of the most popular musical instruments belongs to the human voice (singing, humming or whistling). Technically, it is much easier to analyze whistles than it is to analyze singing. Because of this, whistling sound has better

H.-C. Shen (✉)
Institute of Computer and Information Engineering, I-Shou University, 1, 1, Section 1, Hsueh-Cheng Rd., Ta-Hsu Hsiang, Kaohsiung, Taiwan 840, People's Republic of China
e-mail: shungch@isu.edu.tw

C.-N. Lee
Department of Computer Science and Engineering, National Sun Yat-Sen University, 70, Lien-hai Rd., Kaohsiung, Taiwan 804, People's Republic of China
e-mail: cnlee@mail.cse.nsysu.edu.tw

accuracy of the segmentation of notes and a higher pitch range in the spectrogram. This situation motivates our research on a Whistle-to-Music composition system. The Whistle-to-MIDI note transcription works as a convenient melodic MIDI input device via whistling. Most researches about monophonic melody transcription are primarily focused on content-based music retrieval [1, 2]. This paper reports a Whistle-to-Music composition system that extends melody transcription to music composition. Many music lovers who can whistle do not have the necessary score-writing technique, a Whistle-to-Music system aims at helping users to perform or make up a piece of music easily. Designing an intelligent, intuitive system that enables novices to compose music by whistling is a difficult task. We can view the problem as an integration of many musical subtasks. Three main musical subtasks are melody transcription, melody development and melodic accompaniment. In this paper, melody development and melodic accompaniment are restricted to melodic variation and chords generation. In the beginning, Whistle-to-MIDI translates a whistled tune into MIDI sequence data. The transcribed MIDI notes are displayed in the sequencer with a lead sheet view. Computer-aided composition can incorporate with musical knowledge such as key transposition, melody development and chords generation. The goal of Whistle-to-Music system is to enable a creative but musically-untrained individual to get a taste of songwriting and music creation.

1.1 Related work

To our knowledge, no system exists for generating a piece of MIDI music directly from a musical whistling. However, there has been a variety of prior work in tools and algorithms to help people make music, ranging from new forms of instruments to other systems that generate chords. Some of the most important researches related to our "Whistle- to-Music" composition system are included here. The related systems are classified based on Chong's report [3] in computer generated music composition.

1. *The instrument model systems*—rely on constant musical input from a user, similar to normal instrument. In addition to MIDI keyboards, some alternative music controllers are gestures such as gloves, dance, eyes [4] and facial expressions [5]. As for human voice as melody input, most systems include "automatic harmonization" that generates harmonies for a melody. Tsuruta et al. [6] was among one of the first to use vocal input for the instrument model system. MySong [7] is a system that automatically chooses chords to accompany a vocal melody.
2. *The radio model systems*—operate mostly independently from a user. The most important interaction is predetermined on a number of discrete alternatives. The music generation behaves much like a radio. Interaction is measured in terms of minutes and hours. Essl's Lexikon-Sonate [8] is one of the examples.
3. *High-level interactive music composition systems*—are somewhere in between the instrument and the radio models. They require constant, although much less active, interaction and offer high-level musical interaction primitives. For example, Jacob's *variations* [9] system works at the level of motives, which simplifies the organization of the music. Impro-Visor [10] (short for "Improvisation Advisor") is a music notation program designed to help jazz musicians compose and hear solos similar to ones that might be improvised.

Recently, numerous prototypes of user interfaces have been presented that are based on interpretation of non-verbal sounds produced by the users, such as humming, whistling, or hissing. The non-verbal vocal input (NVVI) that have been developed so far include

systems able to control mouse cursor pointer [11], operate computer games [12], create artwork [13], or emulate keyboard [14]. From the systems mentioned above, few focus on the integration of the three models. Our "Whistle-to-Music" system attempts to cover parts of the above three models. Firstly, Whistle-to-Music works as an instrument model to create some melodic fragments. For the radio model, we provide high-level interactions in computer-aided composition. They include various transcription templates, melody development methods and automatic accompaniment. In summary, the main contributions of this paper contain two aspects.

- **Fast Whistle-to-MIDI note transcription.** The Whistle-to-MIDI note transcription works as a direct MIDI input device that accepts a whistling tune at very fast tempi (up to 120 BPM). For any up-to-date commercial melody transcription systems such as Opcode's Studio Vision Pro [15] or Wildcat Canyon's Autoscore [16], a recording stage is required before an analysis stage can proceed. The pitch templates and rhythm templates are used to fit the transcribed notes to common musical notes.
- **Intuitive music interface design.** Some elements contribute to an intuitive music interface design for a live performance sequencer. They are lead sheet notation, music operators and high-level music templates. A lead sheet notation reduces a piece of music into a melody with chord symbols. In Whistle-to-MIDI, melodic operators are to vary a selected pattern in a wide variety of ways (i.e., transposed, orchestrated in innumerable ways, played backwards, notes deleted, added, changed, increased or decreased in volume, etc.). A transcription template in pitch can provide different whistlers to adjust the pitch range or pitch resolution for a wider or narrower pitch range.

1.2 Overview of Whistle-to-Music system

The overview of a "Whistle-to-Music" system is illustrated in Fig. 1. Three major components of this system are a note transcription, a MIDI sequencer, and a computer-aided composition.

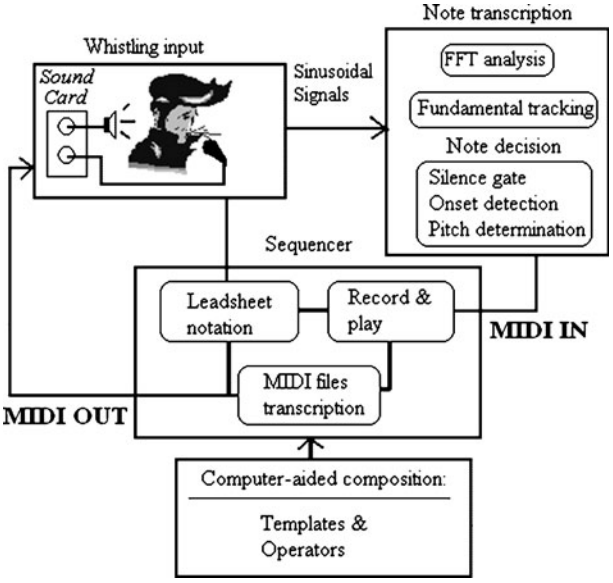

Fig. 1 Overview of the Whistle-to-Music

The note transcription works as a MIDI input device for the sequencer. It allows a user to input MIDI note messages to a sequencer via whistling. The whistle-to-MIDI input is accepted by a MIDI sequencer called "MaxSeq [17]". MaxSeq is a windows sequencer written in C++ using MFC. We modify this existing open source MIDI sequencer to extend its capabilities. In addition to allow a user to perform the musical tasks such as recording, playing and saving for a MIDI file, we provide a lead sheet notation for the composed MIDI music. A lead sheet is a musical composition represented by a chord progression and a melody line, usually on a single musical staff. The melody is written in modern Western music notation, the lyric is written as text below the staff and the harmony is specified with chord symbols above the staff. Various musical templates and operators are used for computer-aided composition such as melodic variation and automatic accompaniment.

The rest part of this paper is organized as follows. Whistle-to-MIDI note transcription is discussed in Section 2. Section 3 describes the implementation of Whistle-to-MIDI. Chords generation are given in Section 4. Section 5 is experimental results. Conclusion is given in Section 6.

2 Whistle-to-MIDI note transcription

Note transcription is the act of converting monophonic sound (e.g. whistling) into its music notation. The basic transcribed notation includes onset, duration, velocity and pitch of notes. One major advantage in using musical whistling as melodic input is that the sound spectrum of the human whistle (in the most common method of whistling) contains negligible harmonic frequencies and for practical purposes is a pure sinusoidal signal. This is easily verified by a modern spectrogram (or known as a frequency analyzer) that displays the time variation of the spectral content of a signal. Since our note transcription is configured as a MIDI In device, it can work as MIDI input to any sequencer program. The process of note transcription consists of three stages: FFT analysis, fundamental tracking and note decision.

Stage 1: **FFT Analysis**. At the first stage, FFT analysis is used to transform the sampling sound in time-domain into a spectrogram in frequency-domain representation. This signal conversion displays the pitch and duration tracks on the spectrogram. Real-time FFT:

$$A_k = \sum_{n=0}^{N-1} X_n e^{-2\pi ikn/N} \tag{1}$$

where, X_n is each sampled value, and N is the sampling number. A_k is the amplitude of specific frequency k.

Stage 2: **Fundamental Tracking**. It is to identify the pitch-contour each whistled note that are above the threshold loudness (intensity). Since FFT calculates the "spectrogram," or the intensities of various sine waves. One can use colors in spectrogram to indicate the intensities of various components of the sound. The traces that have colors above threshold intensities are determined as fundamental frequencies. By our experiments, the whistling sounds have a very sharp peak at fundamental frequency, that a thin line, probably red or yellow, depending on the volume, at the spectrogram.

Stage 3: **Note Decision**. This stage contains three subtasks that will be described in the following order: the silence gate, the onset detection and pitch determination.

A). *Silence Gate*. A silence gate first ensures the suppression of spurious candidates in background noise. Silence gate is set by two threshold values. One is band pass filter since the total pitch range of whistling is observed to be 700 Hz – 2,800 Hz. The other is by note velocity. When the signal drops under a certain level, the onsets are discarded. Because the noise level can dramatically change between different auditory scenes, this level can be adjusted to minimize the onsets detected during pauses and silences. We can use colors in spectrogram to indicate the intensities of various frequency components of the sound. As shown in Fig. 2, colors blue, green, yellow, orange, purple and red are used to indicate the intensity of a whistled note.

B). *Onset Detection*. The onset time is defined as the beginning time of a beat or note played (whistled). Onset detection is a combination of two methods. One is to detect power increasing, and the other is to divide two onsets at the pitch changing point. At first, the silent parts are determined by silence gate as rests. Then, the powers increasing points are detected as note starting points. If there is a pitch changing point between two rests, another onset is derived at that point. Figure 3 shows the spectrogram of whistling notes. The spectrogram represents the power of different frequencies at different time indices. It is a three dimensional plot of the energy (power) of the frequency content of a signal as it changes over time.

C). *Pitch Determination*. According to the MIDI Tuning Standard [18], the assignment of whistled note frequency to the MIDI semitone (from 0 to 127) is determined by the following formula:

$$\text{MIDI value} = 69 + \left[12 \times Log_2\left(\frac{freq}{440}\right)\right] \tag{2}$$

where freq is the frequency of whistled note and the operator of [ ] calculates the nearest integer value. 12 leads to the classic dodecaphonic musical scale, and 69 is the MIDI pitch that corresponds to central A (440 Hz). The tone middle C (261.6 Hz) corresponds to the MIDI note number 60. A tone n semitones above or below corresponds to the MIDI note number 60±n.

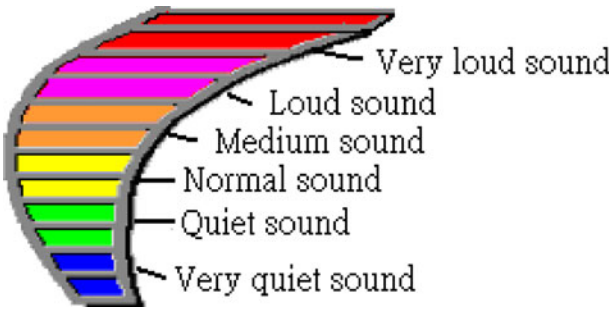

Fig. 2 The spectrum of a whistled note

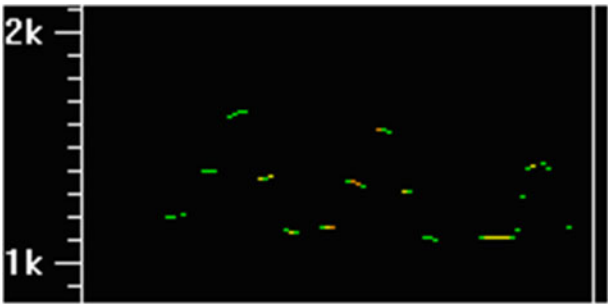

Fig. 3 The spectrogram of whistling notes

2.1 Spectrogram

A spectrogram is a two-dimensional function of time and frequency that displays the time variation of the spectral content of a signal. In MATLAB, the function *specgram* will compute the spectrogram. Figure 4 shows the results of applying *specgram* to the stepped-frequency sinusoids that make up the C-major scale.

We modify a real-time spectrogram program [19] to implement a Whistle-to-MIDI for a sequencer. Via a sound card, the microphone converts input sounds into voltage. It measures the voltage as often as 11,025 up to 44,100 times per second. In this program, we can set this number-of-measurement-per-second parameter using a drop-down combo box "Sampling Frequency" (here it shows 11 kHz). Each measurement is converted into an 8- or 16-bit number called a sample. 16-bit numbers allow for more accuracy in measuring subtle effects. We can select this parameter by clicking on one of the "Bits per sample" radio buttons. The result of sampling is a series of numbers. This series is displayed in the upper

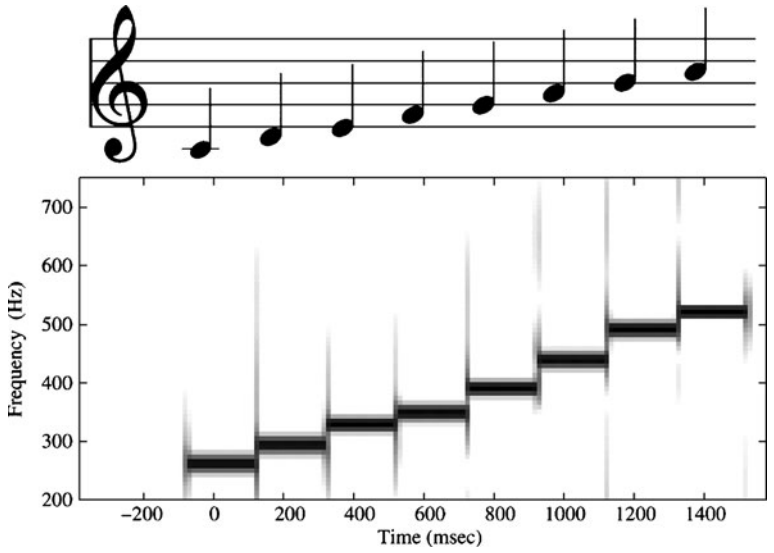

Fig. 4 Musical notation for the C-major scale and the corresponding spectrogram computed using MATLAB's *specgram* to the stepped-frequency sinusoids that make up the C-major scale

right pane of the Frequency Analyzer. That's where we can see the sine wave produced by our whistling. The bottom pane shows the spectrum of the sound.

Here's how we interpret the main pane of the Frequency Analyzer: There is a white vertical line moving from left to right. It leaves behind a colorful trace. The colors correspond to the intensities of various components of the sound. For instance, if we whistle into the microphone a pure sine wave of the frequency 1 kHz, the Fourier transform will have a sharp peak at 1 kHz. We'll see the program draw a thin line, probably red or yellow, depending on the volume, at the height corresponding to 1 kHz. If we lower the pitch of your whistle, the line will move downwards. If we increase the pitch, it will turn upwards. Figure 5 is the display of a modified spectrogram program. Notice that the traces of fundamental frequencies are very clear for a whistled tune.

3 Implementation of Whistle-to-MIDI

Figure 6 shows the user interface for Whistle-to-MIDI. Creating a melody with Whistle-to-MIDI begins with recording a whistled melody; the user presses a "record" button and whistles along with a computer-generated beat at a user-specified tempo. The user can hear and see the red colored whistled notes while he/she is whistling.

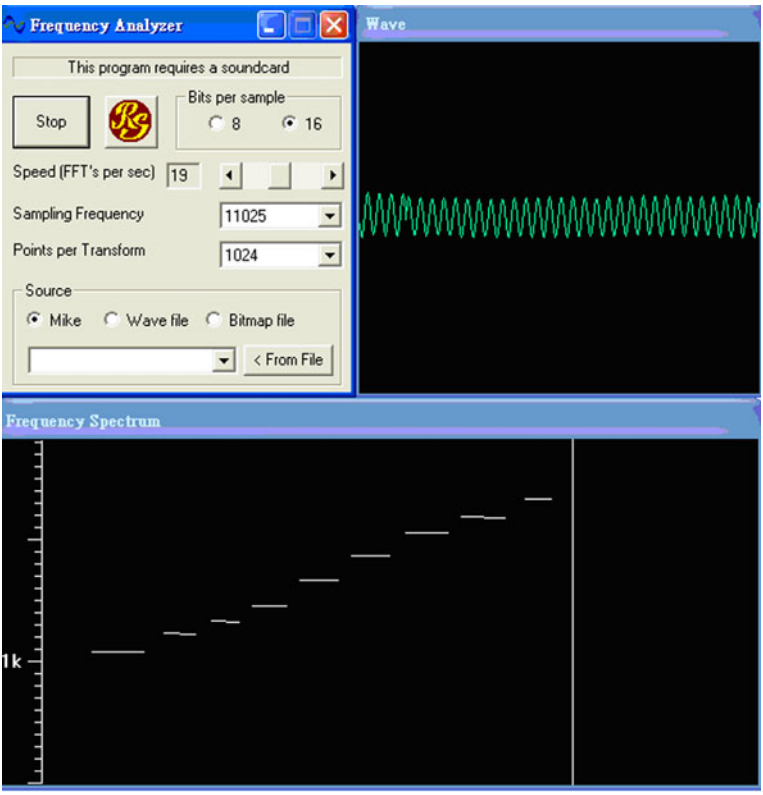

Fig. 5 Frequency analyzer display

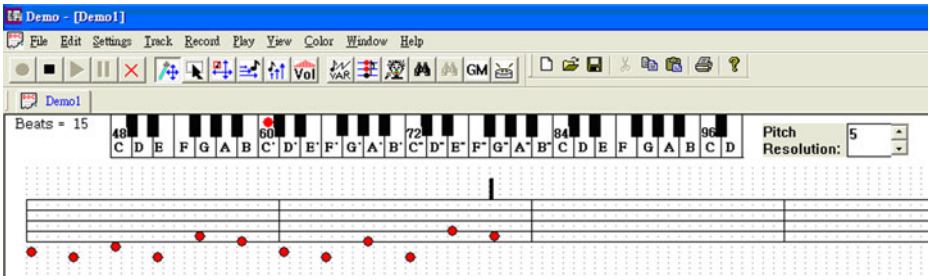

Fig. 6 Real-time Whistle-to-MIDI processing in a lead sheet interface display

There are two important physical human factors in the implementation of whistle-to-MIDI:

- The whistling range, i.e. how high?, how low?, or the interval of pitch that the user is capable to produce.
- The precision of intonation, especially the ability of the user to finish a tone within given interval.

Obviously, the larger the user's vocal range is and the more precise the user's intonation is, the wider set of values can be presented to the user to choose from. Therefore, the user can set the parameters in "tonic" and "pitch resolution" before performing Whistle-to-MIDI. The "tonic" is determined as the first note value for a whistled tune. The value of pitch resolution is to determine the size of a semi-tone between two pitches.

When the user stops whistling, the transcribed notes are immediately quantized into standard note durations and displayed as shown in Fig. 7.

3.1 Whistle-to-Music Interface

The interface uses direct manipulation principles, corresponding to the physical world. The standard lead sheet music interface was designed with two contradictory requirements in mind: (1) sheet music should be as large as possible and at least the size of whole screen, and (2) the menus should be legible, effective and efficient for users. This means the total space necessary for music and menus should be reduced. This can be done by reserving a small white area at the top, bottom, left and right side. The saved space can then be effectively used for showing or hiding different menus, buttons or musical controls at run-time as shown in Fig. 8. For examples, when the user want to perform transposition to any key desired, it is best to put the group of transposition controls into the above lead sheet so the user could have them nearby when needed. Those controls can be shown or hidden by pushing the same buttons.

The three main modes Whistle-to-Music interactions are 'whistle', 'sequence' and 'accompany'. Whistle-to-MIDI is to capture musical ideas quickly. The 'sequence' mode is

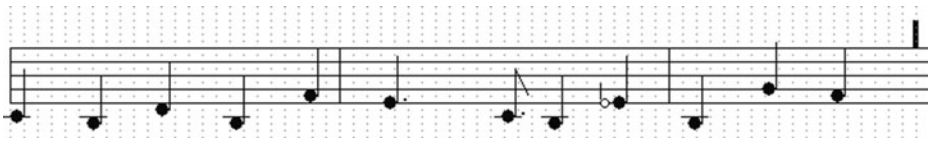

Fig. 7 A quantized melodic phrase from Whistle-to-MIDI

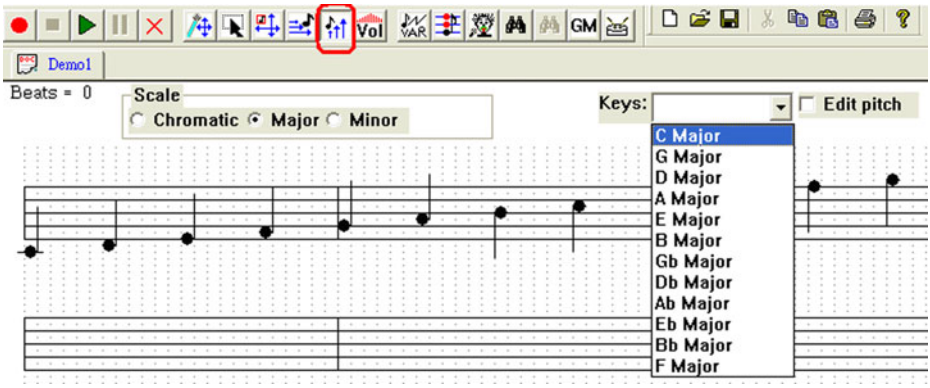

Fig. 8 A small white space above lead sheet for showing and hiding various musical controls in the Whistle-to-Music Interface

to organize those ideas in a useful manner. The 'accompany' can manipulate them based on musical styles. We focus on melodic development such as variation and chords generation as a computer-aided composition. Melodic variation is achieved through transform functions. Automatic accompaniment is to generate chords based on a given melody. It is discussed in Section 4.

3.2 Melodic composition in a 'sequence' mode

Figure 9 shows melodic composition in a 'sequence' mode. There are two ways to develop a melody in Whistle-to-Music. The first method is to concatenate the whistled phrases when giving Whistle-to-MIDI. The second is to select a fragment of melodic motive or phrase first and then transform it as shown in Fig. 9a. To rearrange the melodic motives, the user can select (cut) a certain melodic pattern in a score, and then paste this pattern to the desired measure based on a moved caret position as shown in Fig. 9b.

In the 'sequence' mode, transform functions are used to derive motive or phrase variants. A transform function is a mapping of members of one set, called the domain set, to another set, called the range set. The domain set usually contains the pitch elements or rhythmic elements. The mapping is described by an expression or, in some case by a table. Let S be an order set of a finite attribute $(x_1,\ldots,x_n)$ such as pitch elements and rhythmic elements. A general unary function is formulated as follows. $F_u(kS \pm n) = (kx_1 \pm n, \ldots, k\,x_n \pm n)$. There are some binary functions which take two input variables. Let T and L be order sets of a finite attributes $(x_1,\ldots,x_n)$ and $(y_1,\ldots,y_n)$. A general binary function is formulated as follows. $F_b(jT \pm n, kL \pm m) = [(jx_1 \pm n, ky_1 \pm m), \ldots, (jx_n \pm n, ky_n \pm m)]$. The meaning of the binary function is to take two order sets (T, L) and map them into a series of two-dimensional pairs (T_1, L_1), (T_2, L_2),…, (T_n, L_n). The first number in each pair indicates the temporal order T (horizontal axis), and the second number represents the pitch level L (vertical axis). Table 1 lists some binary transform functions. The first three of these correspond with the traditional mirror form groups of inversion, retrograde, and retrograde-inversion. The last four are new and are identified by the prefix quadrate. The quadrate function is derived from the prime motive by reflection about the ascending diagonal. The other quadrate forms, QI, QR, and QRI, may then be derived from quadrate function by the same operations performed on prime motive to get inversion, retrograde, and retrograde-inversion.

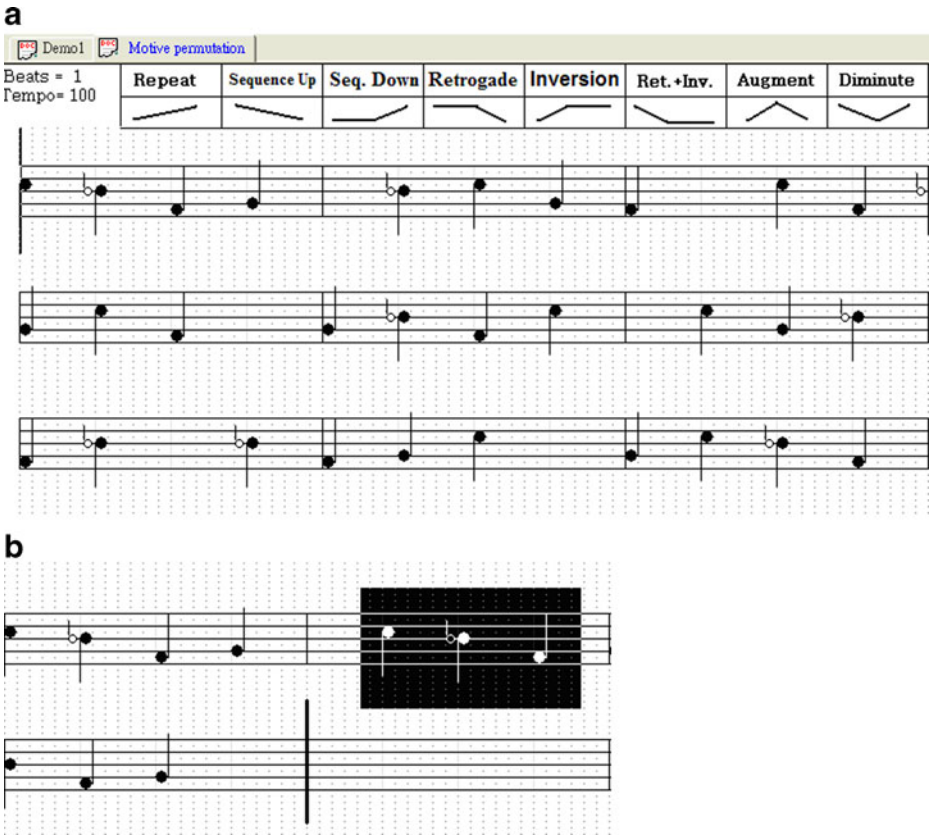

Fig. 9 Melodic composition in 'sequence' mode: **a** Variation. **b** Cut and paste operation

Figure 10a shows the geometry of binary functions. The binary functions are the results of reflective transformations of (T, L) according to their types of axes and operations. Axes ordinarily occur in time, pitch or both. Figure 10b shows the examples and the results of those functions. The temporal order T of original motive is (1, 2, 3, 4) and the pitch level L is (1, 3, 4, 2). The paired attributes of the original motive (T, L) are [(1,1), (2,3), (3,4),

Table 1 Functions that perform topological transformations

Functions	Descriptions
Inversion	$F_I(T, n-L)$
Retrograde	$F_R(n-T, L)$
Retrograde-Inversion	$F_{RI}(n-T, n-L)$
Quadrate	$F_Q(L, T)$
Quadrate -Inversion	$F_{QI}(L, n-T)$
Quadrate- Retrograde	$F_{QR}(n-L, T)$
Quadrate- Retrograde- Inversion	$F_{QRI}(n-L, n-T)$

* n is note count + 1 here

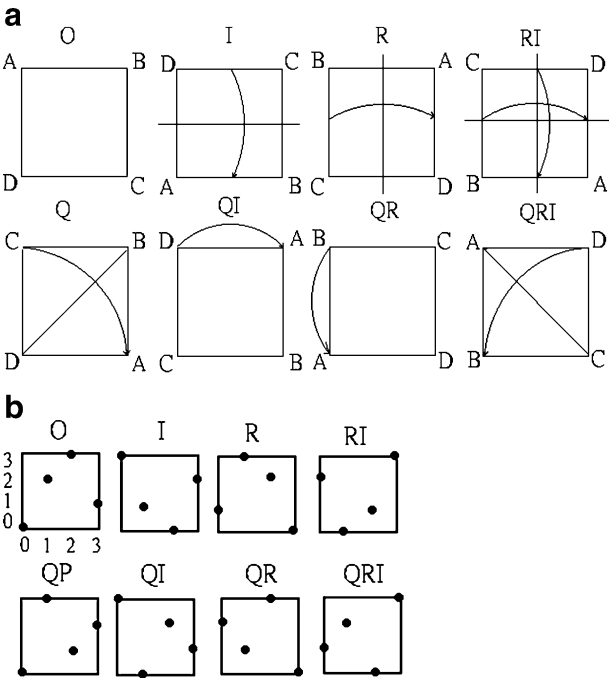

Fig. 10 **a** Topological transformation functions. **b** An example at the left top and the results after a series of topological transformations

(4,2)]. Since n is 5 (notes count +1) here, after those transformations, the sequence of pitch level attributes become $F_{\mathbf{I}}$(T, 5-L) = [(1,4),(2,2),(3,1),(4,3)], $F_{\mathbf{R}}$(5-T, L) = [(1,2),(2,4),(3,3),(4,1)], $F_{\mathbf{RI}}$(5-T, 5-L) = [(1,3),(2,1),(3,2),(4,4)], $F_{\mathbf{Q}}$(L, T) = [(1,1),(2,4),(3,2),(4,3)], $F_{\mathbf{QI}}$(L, 5-T) = [(1,4),(2,1),(3,3),(4,2)], $F_{\mathbf{QR}}$(5-L, T) = [(1,3),(2,2),(3,4),(4,1)] and $F_{\mathbf{QRI}}$(5-L, 5-T) = [(1,2),(2,3),(3,1),(4,4)].

Figure 10b shows more contour shapes that can be derived based on the original motive shape "$\wedge_{-h}$".

The unary function only takes one parameter. Table 2 lists some unary transform functions that accept a single attribute of pitch or rhythm as an input variable.

In order to vary the pitch and rhythm slightly, any of above unary functions may be applied to a subset of the attribute set upon which they operate. The subset is notated as a set of positive integers that denote the members of the primary set affected by the transformational function. For example:

$$\mathrm{Tc}(1,2,4,8,9)+3$$

It indicates that the first, second, fourth, eighth, and ninth members of the pitch set have all been transposed upwards by three semi-tones.

After the completion of melodic composition, the user can assign a general MIDI instrument for the melody as shown in Fig. 11. The interface is designed with bitmap buttons with context menu.

Table 2 Unary transform functions

Functions	Descriptions
Chromatic transposition	$T_C(p) = p + i$, where p is the pitch set and i is an integer from -11 to 11, and p is the pitch value.
Tonal transposition	$T_t(s) = s + i$, where s is the scale degree and i is an integer from -7 to 7, and s is the scale degree.
Interval Expansion	$I_e(i) = i + k$, where i the duration set and k is a positive real number larger than 1.
Interval Contraction	$I_c(i) = i - k$.
Durational augmentation	$R_a(d) = kd$, where d is the duration set and k is a positive real number larger than 1.
Durational diminution	$R_a(d) = kd$, where k is a fraction less than 1.
Unify rhythm	$R_a(d_i) = 1$,where d_i is each element of in the duration set.
Contrasting rhythm	$R_c(d_i) = kd_i$, if d_i > = a quarter note then k=2 else k=1/2.

4 Chords generation

It is important to note that there may have many accompaniments for a particular melody. Chord selection will vary among musicians and genres. Therefore our goal in designing Whistle-to-Music's chords generation is not to predict the "correct" chords for a given melody, but to produce subjectively appropriate chords according to a small set of parameters that are intuitive to a non-musically-trained user. Thus, chords generation can be thought of as a tool which produces a variety of suitable accompaniments that the user can choose from. Figure 12a shows the interface of manual chord generation. Based on a chord input dialog, a user can place a chord label in a caret position and then create a chord progression. A gray-scale color scheme is applied to the three-note triad so that the most important note is colored with black, the third note with light gray and the fifth note with dark grey. This color scheme is useful when the user perform chord inversion as shown in Fig. 12b.

The steps of generating a chord progression automatically for a given melodic phrase are described in the following.

Step-1. **Determine the key and harmonic pulse.** The harmonic pulse is how often the chords change. In Fig. 12a, the harmonic pulse is two chord per measure.

Step-2. **Analyze scale degree.** Each melodic note is represented as a scale degree based on a key. For example, a C major scale has seven scale degree range from 1 to 7.

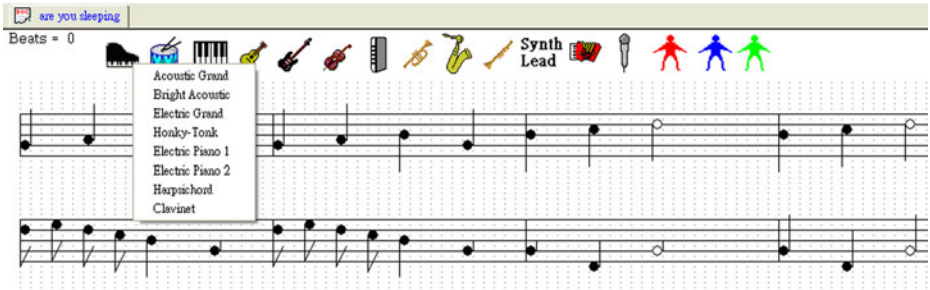

Fig. 11 General MIDI instrument assignment with bitmap buttons and context menus

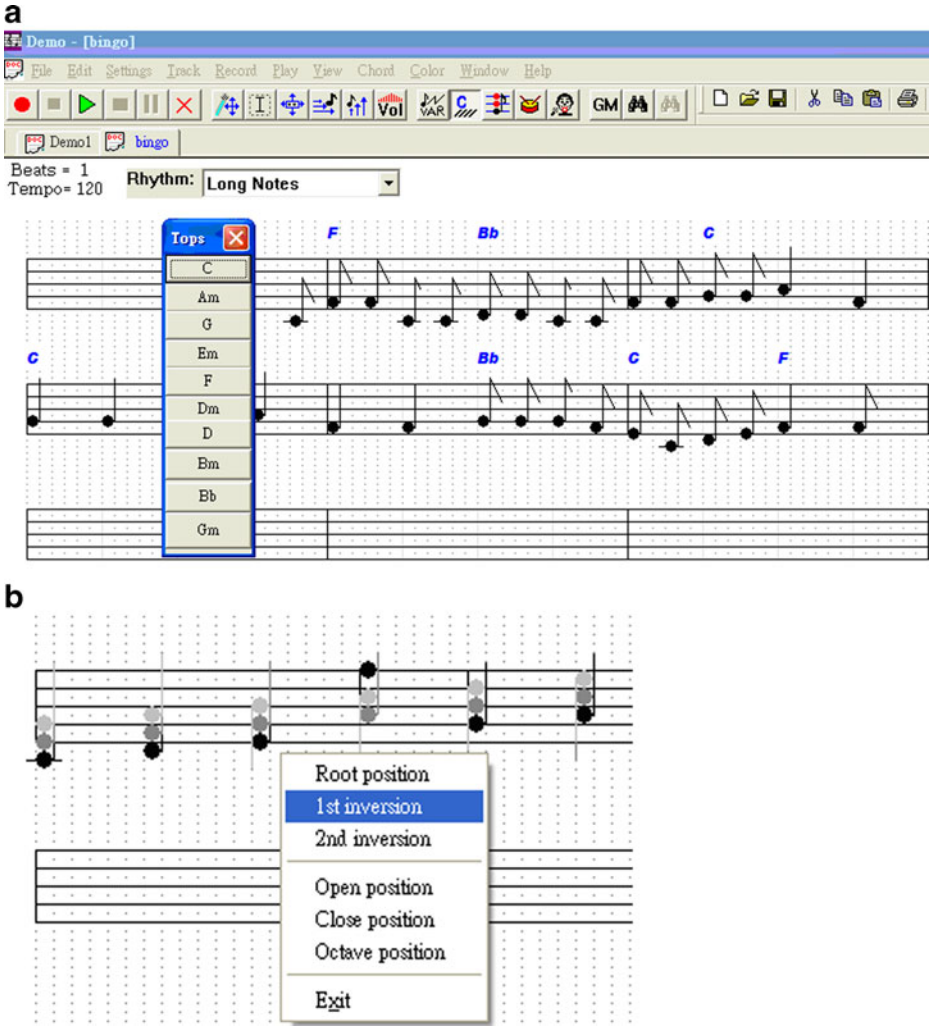

Fig. 12 **a** The interface of chord generation. **b** A gray-scale color scheme for a triad

Table 3 A harmonic table

Current chord	Priority of the next chord					
	I	II	III	IV	V	VI
I	1	6	4	2	3	5
II	3	6	4	5	1	2
III	6	4	5	1	2	3
IV	1	4	5	2	3	6
V	1	5	4	6	2	3
VI	6	1	3	2	4	5

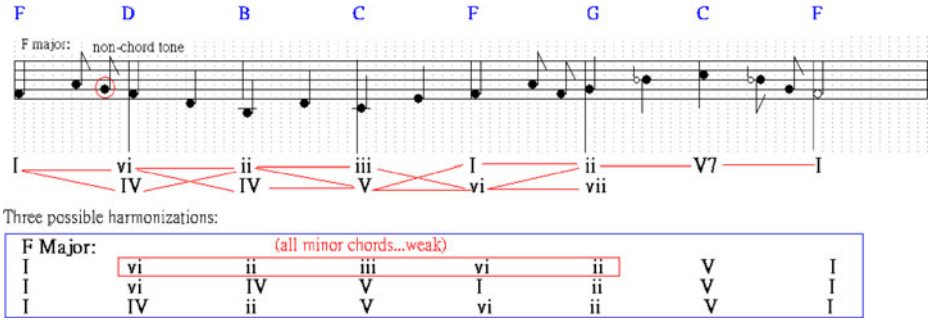

Fig. 13 The possible harmonization for a given melodic phrase

Step-3. **List all possible chords that cover most melodic notes if not all of the notes are within that beat.** For a single note, this will be three possible triads (the note will either be the root, third or fifth of a chord. For beats with more than one note, there may be less, or even only one possible chord for that beat. List all of the possible chords, since you can not know for certain which we may need to use yet.

Step-4. **Select the next chord progression that meet a given harmonic table**. Since certain chords are used far more often than others, a harmonic table as shown in Table 3 is used for a possible chord progression.

Step-5. **List all possible chords without a dead end. For a melodic phrase,** the last two or three beats are usually a harmonic cadence. So when you hit a "dead end" simply back up two or three chords and choose another possible chord. Figure 13 shows three possible chord progressions available for selection.

5 Experimental results

We are interested in the convenience of using oral whistling as a melodic input. However, no one can expect that the transcribed whistled notes are exactly the same notes with those appear in the musical staff. Therefore, a melodic subtasks and hit-rates table is designed to study user's experiences or satisfactions as shown in the Table 4. Firstly, a test user is asked to whistle along a target melodic motive. This table shows the hit-rates of each user attempting some melodic subtasks.

Subtasks from T1 to T3 denote the exact matching of transcribed notes with a target motive in note counts, pitch contours and duration contours. The 3-level pitch interval contour composes of three types of movement. The lower case letters "u", "d" or "r" are

Table 4 Melodic subtasks and hit-rates

	Users										
Melodic subtasks	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	Hit-rate
T1 Note counts	hit	fail	hit	hit	hit	hit	fail	hit	hit	hit	8
T2 pitch contours	fail	hit	fail	hit	fail	hit	hit	hit	fail	hit	6
T3 Duration contours	fail	hit	hit	hit	fail	fail	hit	hit	hit	hit	7

used to indicate whether a note is higher (u), lower (d) or same (r) in pitch compared to the previous note. For a 3-level rhythmic contour, it is a representation by the letter "l", "s" or "e". Those letters indicate whether a note is longer (l), shorter (s) or equal to (e) in inter onset interval compared to the previous note.

The results in Table 4 show the outcome of each attempt in three melodic subtasks. In the success rate, we found that it is easier to derive a correct note count in whistling transcription. As for pitch contours, there is a trend that with finer pitch interval contours (>3 directions) of the target melody the success rate falls. Actually, the incorrect pitch in MIDI note number can be solved based on an interactive "play and modify" interface design. We can let user select the incorrect pitch notes and use keyboard or mouse to adjust the pitch.

6 Conclusion

We have constructed an interactive Whistle-to-Music composing system to support computer-aided composition. The given melodic phrases are developed into a whole song via melodic variation and chords generation. The proposed system integrates a MIDI sequencer with Whistle-to-MIDI note transcription and computer-aided composition. This design can allow an ordinary user easily to explore his potential in musical creativity. One important aspect of this paper is the cost-effective MIDI controller by whistling. The transcription speed is improved and note segmentation is made easier when the melodic input is restricted to oral whistling. Another important aspect of this paper is melodic composition and chords generation based on an interactive approach.

New music instruments or controllers convey also new possibilities and paradigms to the act of making music. Currently we have only scratched a few, like melodic development and chords generation to bring the joy of real-time music creation to non-trained musicians. In the future, the proposed "Whistle to Music" will not only provide an effective framework for musical composition by whistling, but also provide the framework for other melodic inputs like humming and singing.

References

1. Asif, G., Jonathan, L., David, C., Brian, C.S. (1995). Query by humming: musical information retrieval in an audio database. In Proc. of ACM Multimedia (pp. 231–236)
2. Hung-che, S., Chong-nan, L. (2007). Whistle for music: using melody transcription and approximate string matching for content-based query over a MIDI database," Multimedia Tools and Applications, Vol 2, 35(3): 259–283
3. Chong, (John) Y. (1996). Computer generated music composition, MSc Thesis, M.I.T.
4. Anthony, H., Linda, S. (2004). EyeMusic: making music with the eyes. In Proceedings of the 2004 Conference on New Interfaces for Musical Expression (NIME04), Hamamatsu, Japan, June 3–5, 185–188
5. Mathias, F., Kazuhiro, K., Michael, J.L. (2005). Sonification of facial actions for musical expression. Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05), Vancouver, BC, Canada. Pp 127–131
6. Tsuruta S, Fujimoto M, Mizuno M, Takashima Y (1988) Personal computer-music system-song transcription and its application. IEEE Transactions on Consumer Electronics 34(3):819–823
7. Ian, S., Dan, M., Sumit, B. (2008). MySong: automatic accompaniment generation for vocal melodies. To appear in Proceedings of Computer-Human Interaction
8. Karlheinz, E. (1995). Lexikon-Sonate: an interactive real-time composition for computer-controlled piano. In proceedings of the II Brazilian Symposium on Computer Music
9. Bruce, L.J., (1996). Algorithmic composition as a model of creativity, Organised Sound 1(3): 157–165. Internet: http://www.ee.umd.edu/~blj/ algorithmic_composition/ algorithmicmodel.html

10. Robert MK, David RM (2007) A grammatical approach to automatic improvisation, Paper to appear in the Fourth Sound and Music Conference, SMC '07. Lefkada, Greece
11. Bilmes, J.A., Li, X., Malkin, J., Kilanski, K., Wright, R., Kirchhoff, K., Subramanya A., Harada, S., Landay, J.A., Dowden, P., Chizeck, H. (2005). The vocal joystick: a voice-based human-computer interface for individuals with motor impairments. In Human Language Technology Conf./Conf. on Empirical Methods in Natural Language Processing
12. Adam, J.S., Sri, H.K., Murni, M., Pavel, S. (2006). Non-speech input and speech recognition for real-time control of computer games. In The Proceedings of The Eighth International ACM SIGACCESS Conference on Computers & Accessibility, ASSETS 2006, Portland, Oregon
13. Sama'a, A.H. (2006). Blowtter. A voice-controlled plotter. In Proceedings of HCI 2006 Engage, The 20th BCS HCI Group conference in co-operation with ACM, vol. 2, London, England
14. Adam, J.S. (2009). Pitch in non-verbal vocal input. ACM SIGACCESS Accessibility and Computing. Issue 94. June, ACM
15. Opcode Studio Vision Pro (2006). http://www.dg.co.il/Partners/opcode.html, Last Visited May 30, 2006.
16. AutoScore by Wildcat Canyon Software (2006). http://www.wildcat.com/Web/Wildcat/Html/Site/Homepage.html, Last Visited May 30, 2006
17. Paul M (1998) Maximum MIDI. Manning, CT, pp 105–204
18. MIDI Manufacturers Association (1996). Complete MIDI 1.0 Detailed Specification
19. Frequency Analyzer. http://www.relisoft.com/freeware/index.htm

Hung-Che Shen completed his M.Sc. in computer science from Oklahoma State University, U.S. in 1990. He received the Ph.D. degree in 2007 from National Sun Yat-Sen University, Taiwan. He has been working as a Assistant Professor at the I-Shou University, Taiwan. His research interests include computer music, digital signal processing and consumer electronics

Chung-nan Lee received the B.S. and M.S. degree in electrical engineering from National Cheng Kung University, Tainan, Taiwan, in 1980 and 1982, respectively, the Ph.D. degree in electrical engineering from the University of Washington, Seattle, WA, in 1992. Since 1992, he has been with National Sun Yat-Sen University, Kaohsiung, Taiwan, where he was an Associate Professor in the Department of Computer Science and Engineering from 1992 to 1999; He was Chairman of the department of Computer Science and Engineering from August, 1999 to July, 2001 and is currently a Professor. His current research interests include wireless networking, information appliances, computer graphics, Web and Java computing, and parallel computing.