Melody Recognition System

Katarzyna Adamska
Institute of Electronics
Technical University of Łódź
Łódź, Poland
kate_adams@windowslive.com

Paweł Pełczyński
Institute of Electronics
Technical University of Łódź
Łódź, Poland
pawel.pelczynski@p.lodz.pl

***Abstract* — The purpose of the presented research was the development of a melody search system that would allow to find a song in a music database, based on humming the tune of a known song fragment. The melody recognition in the developed system is based on comparing vectors of voice pitch values. The best match of the humming pitch vector against all the recordings in the database is searched. An original frequency and time scaling approach was implemented to improve recognition accuracy. The system performance was tested with the use of live humming recordings of four volunteers with a small database.**

Keywords* — *melody search, note frequency, query by humming, melodic contour.

I. Introduction

The task of recognizing the melody is one of the issues involved in the processing and storage of data related to music. The process of computerization of music helps not only people professionally involved in music, such as composers, instrumentalists or vocalists, but also provides new opportunities for the recipient of a musical work - the listener.

Currently, the most popular method of search for songs in musical databases is the so called "search by text" [1]. The searched text may be the name of the song, composer, artist or any fragment of the lyrics. It turns out, that it is not a satisfactory solution, when one remembers only a fragment of a song's melody. A melody recognition system allows the user to simply hum a song fragment, then treat it as a query to the database. As a result a list of tracks, that most fit to the hummed fragment, is returned. This is a special case of query by example, called the *Query by Humming* (QbH) [2]. For different recordings of the same musical work the only relatively constant feature of the song is the melody.

The melody is a fundamental part of any musical work. A song may consist of several tunes, and usually one of them is the main melody while the rest form the musical accompaniment. A melody consists of several notes occurring in a particular order. For each note one can specify the frequency and duration. Note representation of a song can be easily implemented in a musical database by direct note writing or can be retrieved from MIDI files. A properly hummed fragment of a song allows to recognize its melody for a human listener. Many QbH systems rely on note recognition. In *Note Interval Matching* [3] melodies are treated as strings. They are aligned to obtain the best similarity of two melodies with the use of dynamic programming. Frequency variation between consecutive notes serves as the measure of similarity. Another, well known approach, *N-Gram Matching* [4], is also based on relative note frequency, but it is quantized and an exact match is searched. A combination of the above two techniques is often used in a *Two-stage Search*: N-Gram Matching allows for rejection of the worst guesses and Note Interval Matching refines the result in a smaller database. Unfortunately, automatic note recognition is not an easy task [5]. Lack of clear boundaries between notes in a real humming signal, both in amplitude and frequency, makes automatic note recognition prone to errors due to inaccurate singing and background noise. To avoid the above problems the authors have decided to develop and investigate a melody search system without note recognition, as in *Melodic Contour Matching* techniques [4]. Instead of note symbols a vector of fundamental frequencies estimated in short time intervals, called "melodic contour", is produced. This vector is matched to similar data structures, stored in the database. The best match should occur for the proper piece of music, allowing correct recognition. Typically, *Dynamic Time Warping* is used to find the best match of two melodic contours. This algorithm is time consuming, which motivated the authors to replace it with fine scaling of tempo and pitch of the hummed melody.

II. The Concept of the Melody Regognition System

The presented project consists of three parts. The first is the analysis of the humming recording to identify the fundamental frequency for small time intervals and to find its variability in time. The second part relates to the extraction of the melodic contour of the songs in a database. The third part is the matching algorithm, which allows to search hummed tone sequence in the database of songs. The block diagram of the developed system is shown in Fig. 1. Signal processing and analysis procedures are represented with ovals, database records are enclosed in cylinders, and the produced data vectors are in rectangles. The research work is focused on the development and testing of these algorithms, it does not cover optimization problems of the music database search. To simplify the tests a MIDI representation of songs was chosen. It required the extraction of notes and its conversion to melodic contour of each record.

III. Extraction a Melodic Contour of Humming Sound Signal

Melody matching in the proposed approach is based on the only one signal feature - the frequency of the first signal

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on September 22,2021 at 17:02:50 UTC from IEEE Xplore. Restrictions apply.

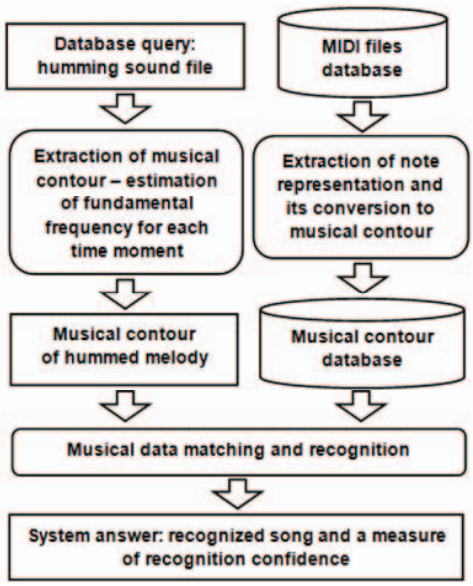

Figure 1. A Schematic diagram of the developed melody recognition system

harmonic, known also as the pitch. In general pitch is the subjectively perceived frequency, not always representing a real one [7]. Fortunately a humming is a periodic signal with the first harmonic of considerable amplitude. Thus, pitch estimation algorithms can be applied in the stage of feature extraction. A lot of methods for pitch estimation exist. The most known are based on an Autocorrelation Function (AF), Cepstrum, Average Magnitude Differential Function (AMDF) or Comb Transformation, [8],[9].

The authors chose the algorithm based on the accurate autocorrelation method [10]. This method is more accurate and robust, than methods based on cepstrum or filtering. It was implemented in Praat sound processing and analysis environment [11], that can be invoked as a command line application from other programs. The main parameters of the utilized algorithm (function: Sound To Pitch) were set as follows:

- Time step – 10ms,
- Pitch floor – 75Hz,
- Max number of candidates – 15.

Other parameters were set to their default values. Sound analysis performed by "Sound To Pitch" function produces a vector of frequency samples, that approximates the melodic contour of a given musical work (Fig. 2.). The obtained vector was then compared with musical database. The developed matching algorithm is described in section V.

IV. EXTRACTION OF THE MELODIC CONTOUR FROM MIDI FILES

Musical Instrument Digital Interface (MIDI) is a standard which allows collaboration between musical instruments, or between a computer and musical instrument. Communication between devices is realized using the MIDI protocol. This protocol is a fixed set of messages sent between devices. These

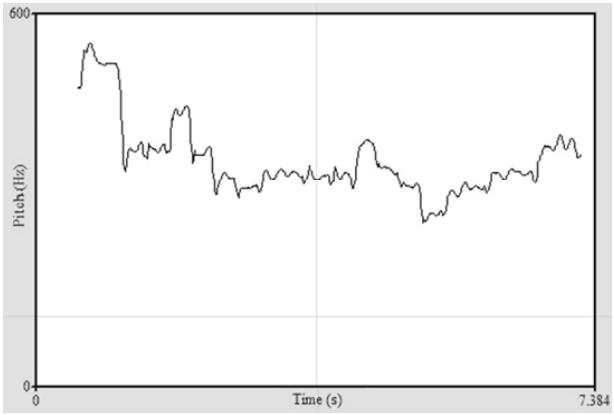

Figure 2. An example of the pitch vector extracted from a sample humming recording

messages equipped with time stamp form commands, that are stored in files, called MIDI files. MIDI commands can control an electronic, musical instrument or a synthesizer, that is often integrated in a sound card. A single MIDI file can be used to control up to 16 synthesizers – they are multipath. Each path carries the information about the played note numbers and their durations, which gives a synthetic representation of a melodic contour.

A specialized converter from a MIDI file to sampled musical contour was written in Java. It is capable of extracting a sequence of desired commands and converting them into a series of frequency values. The "javax.sound.midi" packet was used to read and perform an analysis of MIDI files. A conversion from note MIDI numbers to frequency required a knowledge of the frequency mapping. It is described by the following formula:

$$f_{note}=2^{x/12}\cdot f_{ref} \quad (1)$$

where:

$$x=n_{MIDI}-n_{ref\ MIDI} \quad (2)$$

Reference note is A4, its frequency f_{ref}=440Hz, and MIDI number $n_{ref\ MIDI}$=69. The coefficient $2^{1/12}$ is a halftone frequency interval.

Producing samples of melodic contour required both a decision of sampling period and measuring a time in MIDI. A choice of period was straightforward: it was the same as the time step in the Praat "Sound To Pitch" procedure. An estimation of time in MIDI depends on two parameters stored in the file: the tempo expressed in "beats per minute", and the resolution expressed in "tics per beat". Each event, such as the beginning and the end of a note, is expressed as a tick number, which is converted to discrete time based on the sampling period. Building of the melodic contour vector starts from determining its length and filling all the elements with zeros. Then, for each note, a proper range of vector elements is filled with note frequency. An example of extracted melodic contour can be seen in Fig. 3.a).

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on September 22,2021 at 17:02:50 UTC from IEEE Xplore. Restrictions apply.

V. MELODIC CONTOUR MATCHING AND SONG RECOGNITION

A recognition of a melody cannot be simply a result of comparing the unknown melodic contour with the contours stored in the database due to their different lengths and lack of time synchronization. It should be obvious that there is no need to hum the whole song and that the hummed fragment does not have to start from the beginning of the song. In these conditions a recognition is a result of finding the best match of the unknown melodic contour to the fragment of one of the contours from the musical database. Assuming, that the length of the unknown contour vector is not higher than the lengths of all the vectors in the database, an error of vector match to pattern p for a given time shift i can be defined as a Root-Mean-Square Error (*RMSE*):

$$e_p(i)=\{ [\Sigma_{j=0}^{N-1} (x(j)-p(j+i)^2] / N \}^{1/2}, \quad 0 \le i \le M-N \tag{3}$$

where:
x – unknown melodic contour vector of length N,
p – p-th melodic contour vector from database of length M.

Both vectors and a plot of error e_p are shown in Fig. 3.

Using *RMSE* as the measure of the matching accuracy makes the result independent of vector length and allows to express it in frequency units – [Hz]. The best match error me_p to the pattern p over all time shifts is the minimum of $e_p(i)$:

$$me_p = min\{ e_p(i) \}, \quad 0 \le i \le M-N \tag{4}$$

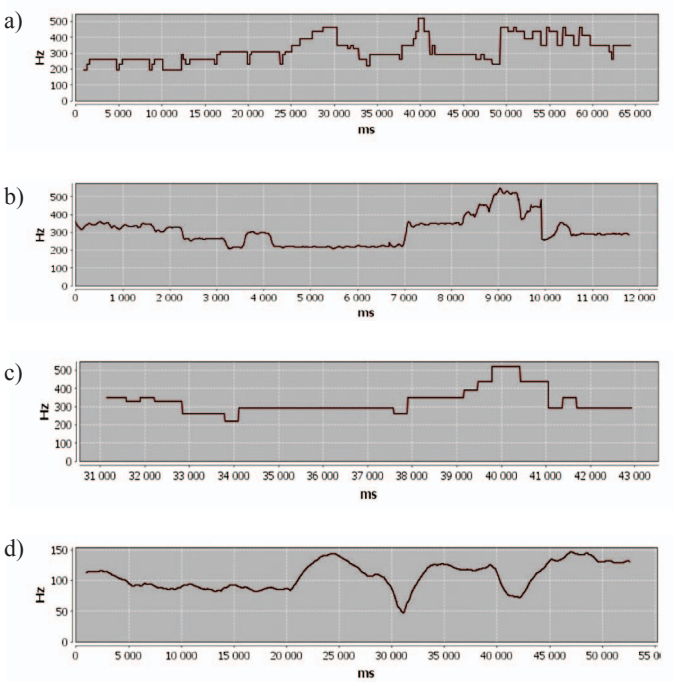

Figure 3. Examples of plots of melody contours: a) extracted from MIDI file, b) hummed by QbH system user, c) found fragment of a contour, and d) plot of match error as a function of time shift

Finally, the recognition of a song is defined as its classification k to one of the database examples p:

$$k = arg\{ min(me_p) \}, \quad 1 \le p \le P \tag{5}$$

where P is the number of songs in the database.

The above approach works very well in ideal conditions of perfect note frequency and tempo reproduction in the hummed fragment. Linear scaling of the unknown melodic contour was introduced to make the algorithm more robust to possible imperfections in melody humming. Scaling in time was obtained by signal resampling with the resulting sampling period Δt_{sl} defined as:

$$\Delta t_{sl} = tc \cdot \Delta t_s, \quad tc \in \{0.8, 0.85, \ldots, 1.2\} \tag{6}$$

where:
Δt_s – original sampling period,
tc – time scaling coefficient.

The scaling coefficient takes one of nine values, from 0.8 up to 1.2. Scaling in frequency was obtained by multiplying the signal by a power of 2:

$$f_m = 2^{m/12} \cdot f, \quad m \in \{-0.5 , -0.4 , \ldots, 0.5\} \tag{7}$$

where:
f – original frequency,
f_m – multiplied frequency.

The m coefficient takes one of nine values, from -0.5 up to 0.5. The range is set to compensate pitch errors in the range of a quarter of a musical tone. After an introduction a both scaling in time and frequency a number of necessary computations increases 9x9=81 times, making the algorithm much slower, but the probability of correct melody recognition increases.

VI. SYSTEM PERFORMANCE ESTIMATION

A series experiments were carried out on real data to evaluate the developed system's performance. The test database consisted of eight songs of popular music and four persons of different ages and genders participated in the experiment. Only one of the participants possessed musical experience. In all the cases recognition accuracy was 100%, so a distribution of the match error *me* over all the examples in the database was investigated. A matrix of match errors obtained for all the humming examples against all the examples in the database without scaling of the unknown melodic contour is shown in Table I. The lowest values are obtained for the proper examples, but other error values are not much higher. The certainty of matching is not very high, and a recognition error can occur in the case of lower recording quality. Introducing scaling of unknown melodic contour vector, both in time alone and in frequency, gave a reduction of the error level. The results are listed in Table III. The highest error reduction was observed after combining both scaling in time and frequency. In the same time the recognition certainty was increased (Table II.).

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on September 22,2021 at 17:02:50 UTC from IEEE Xplore. Restrictions apply.

TABLE I. MATCH ERROR WITHOUT SCALING OF MELODIC CONTOUR

Match error [Hz]	Hummed melody							
	1.	*2.*	*3.*	*4.*	*5.*	*6.*	*7.*	*8.*
Song 1.	47.0	63.3	55.2	53.4	32.3	51.0	24.1	48.5
Song 2.	63.0	55.6	61.5	56.4	38.4	58.1	38.9	57.6
Song 3.	64.2	76.8	47.9	65.5	50.6	59.3	41.3	49.1
Song 4.	76.4	66.7	74.1	32.2	58.9	95.5	44.7	56.5
Song 5.	61.1	70.5	63.8	73.3	12.8	60.0	34.8	57.0
Song 6.	90.6	105.7	85.2	114	42.1	19.1	49.8	61.5
Song 7.	68.7	65.8	59.0	71.6	33.0	66.1	22.0	48.7
Song 8.	71.3	71.6	65.9	74.0	45.2	71.9	35.0	18.3

TABLE II. MATCH ERROR FOR SCALING OF UNKNOWN MELODIC CONTOUR BOTH IN TIME AND FREQUENCY

Match error [Hz]	Hummed melody							
	1.	*2.*	*3.*	*4.*	*5.*	*6.*	*7.*	*8.*
Song 1.	39.5	57.7	51.6	46.8	23.7	34.2	23.3	43.8
Song 2.	56.7	31.3	54.9	47.9	35.7	53.4	34.3	51.8
Song 3.	60.5	70.3	29.0	50.9	45.4	54.4	36.8	43.1
Song 4.	59	55.7	60.6	25.6	46.7	84.8	32.9	49.1
Song 5.	54.6	59.6	61.5	65.6	12.3	53.0	28.9	52.3
Song 6.	85.5	90.9	78.3	87.9	40.9	16.8	43.1	55.0
Song 7.	66.7	54.3	56.8	60.8	27.5	56.1	15.5	45.2
Song 8.	63.4	55.7	55.4	66.4	42.8	58.7	33.8	18.1

TABLE III. MATCH ERROR FOR DIFFERENT COMBINATIONS OF SCALING OF UNKNOWN MELODIC CONTOUR

Match error [Hz]	Hummed melody							
	1.	*2.*	*3.*	*4.*	*5.*	*6.*	*7.*	*8.*
No scaling	47.0	55.6	47.9	32.2	12.8	19.1	22.0	18.3
Scaling in freq.	45.0	55.1	47.7	32.1	12.3	16.8	21.3	18.1
Scaling in time	42.0	32.0	29.3	25.8	12.8	18.1	16.3	18.3
Scaling in freq. & time	39.5	31.3	29.0	25.6	12.3	16.8	15.5	18.1

Another experiment was concentrated on finding the individual capabilities of each test participant to cooperate with the proposed QbH system. The results are listed in Table IV.

TABLE IV. MATCH ERROR FOR DIFFERENT PARTICIPANTS OF THE EXPERIMENT

Match error [Hz]	Hummed melody							
	1.	*2.*	*3.*	*4.*	*5.*	*6.*	*7.*	*8.*
Person 1	42.1	31.3	29.0	25.6	12.3	16.9	15.5	18.1
Person 2	44.7	36.0	33.7	53.3	19.2	15.5	16.6	33.6
Person 3	40.0	35.9	34.6	24.5	22.6	24.6	16.5	19.9
Person 4	33.9	33.1	34.3	22.3	8.99	19.2	19.9	18.1

Some variation can be observed, probably due to different style of melody humming and musical ability. The differences are still not crucial to the overall system performance.

The match error for correct melody guesses in Table I. and II. seem to be very high, compared to false guesses. One of the reasons for such a situation is the lack of the human ability to rapidly change the pitch between notes in a hummed song. It produces an additive error to all matches, which does not deteriorate the algorithm's performance. This was verified on synthetic melodic contours, for which all the errors were lower, but their variability was similar, as in the real examples.

VII. CONCLUSIONS

The proposed QbH system allows for accurate melody recognition in small databases. Its performance for the small group of participants was not significantly affected by musical ability. The created application allowed to test the distribution of RMSE over the introduced examples and made it possible to investigate the effect of time and frequency scaling on the outcome of the match.

The developed system is robust to slight errors in melody humming, such as singing a few false notes, singing the melody too fast or too slow, or shifting the tune's pitch. For the testing examples the application obtained the recognition accuracy of 100%.

REFERENCES

[1] M. Casey, R.Veltkamp, M. Goto, M. Leman, C. Rhodes & M. Slaney, "Content-based music information retrieval: Current directions and future challenges," Proceedings of the IEEE, 96(4), 2008, pp. 668-696.

[2] M. Raju, B. Sundaram, P. Tansen, "A query-by-humming based music retrieval system," National Conference on Communications, NCC 2003, Madras, 2003.

[3] R. B. Dannenberg, N. Hu, "Understanding Search Performance in Query-By-Humming Systems," Fifth International Conference on Music Information Retrieval, Barcelona, 2004.

[4] S. Doraisamy, S. Ruger, "Robust Polyphonic Music Retrieval with N-grams," Journal of Intelligent Information Systems, 21(1), 2003, pp. 53-70.

[5] B. Pardo, "Finding Structure in Audio for Music Information Retrieval," IEEE Signal Processing Magazine, Vol. 23, Issue 3, May 2006, pp. 126-132.

[6] R. B. Dannenberg, W. P. Birmingham, B. Pardo, N. Hu, C. Meek, G. Tzanetakis, "A comparative evaluation of search techniques for query-by-humming using the MUSART testbed," Journal of the American Society for Information Science and Technology, Volume 58, Issue 5, March 2007, pp. 611–762.

[7] J. Benesty, M. Sondhi, Y. Huang, "Handbook of Speech Processing," Springer, Berlin, 2008, pp. 185 – 188.

[8] P. De La Cuadra, A. Master, C. Sapp, "Efficient Pitch Detection Techniques for Interactive Music," Proceedings of ICMC 2001, International Computer Music Conference, La Habana, Cuba, September 2001.

[9] M. Dziubinski, B. Kostek, "High Accuracy and Octave Error Immune Pitch Detection Algorithms," Archives of Acoustics, No. 1, vol. 29, pp. 1 - 21, 1. 2004.

[10] P. Boersma, "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound," Proceedings of the Institute of Phonetic Sciences **17**: 97–110. University of Amsterdam, 1993.

[11] P. Boersma, D. Weenink, "Praat: A system for doing phonetics by computer," www.praat.org, 1992–2001.

Authorized licensed use limited to: Georgia Institute of Technology. Downloaded on September 22,2021 at 17:02:50 UTC from IEEE Xplore. Restrictions apply.