Audio Engineering Society
Convention Paper 6137
Presented at the 116th Convention
2004 May 8–11 Berlin, Germany

A Query by Humming system using MPEG-7 Descriptors

Jan-Mark Batke, Gunnar Eisenberg, Philipp Weishaupt, and Thomas Sikora

Communication Systems Group, Technical University of Berlin

Correspondence should be addressed to Jan-Mark Batke (batke@nue.tu-berlin.de)

ABSTRACT

Query by Humming (QBH) is a method for searching in a multimedia database system containing meta data descriptions of songs. The database can be searched by hummed queries, this means that a user can hum a melody into a microphone which is connected to the computer hosting the system. The QBH system searches the database for songs which are similar to the input query and presents the result to the user as a list of matching songs. This paper presents a modular QBH system using MPEG-7 descriptors in all processing stages. Due to the modular design all components can easily be substituted. The system is evaluated by changing parameters defined by the MPEG-7 descriptors.

1. INTRODUCTION

A Query by Humming (QBH) system enables a user to hum a melody into a microphone connected to a computer in order to retrieve a list of possible song titles that match the query melody. The system analyzes the melodic and rhythmic information of the input signal. The extracted data set is used as a database query. The result is presented as a list of e.g. ten best matching results. A QBH system is a

typical application of the MPEG-7 standard.

Generally, a QBH system is a Music Information Retrieval (MIR) system. A MIR systems provides several means for music retrieval, which can be hummed audio signal, but also music genre classification or text information about the artist or title. An overview on existing MIR systems is presented in [16].

1.1. Basic requirements

To compare two melodies, the representation of the melody is important. The user's query signal has to be transformed into a representation that is appropriate for melody similarity measurement. For QBH systems the melody contour is used often. To turn a hummed query into such a representation, the task of automatic transcription has to be done. Various approaches for the transcription of the singing voice are suggested [2, 4, 10].

The melody contour representation turned out to be sufficient for many cases. The simplest form is to use three contour values describing the intervals from note to note, up (U), down (D) and repeat (R). Coding a melody using U, D, and R is also known as PARSON-Code [13]. MPEG-7 defines a more detailed representation using five steps [7] which was introduced in [8]. However, in some cases this representation is not sufficient as reported in [6].

In Parson-Code no rhythmical features are taken into account. However, rhythm ca be an important feature of a melody. The MPEG-7 melody contour also hold information on rhythm. It is even possible to use pure rhythmical information for a query [5].

For the comparison of the query which the melodies that reside in the database of the QBH system, a distance measure for the melody representations is required [8, 17, 18]. More information on distance measures is given in [5].

1.2. Existing QBH systems

Different QBH systems are already available on the World Wide Web (WWW). *Musicline* is a commercial QBH system developed by FRAUNHOFER IDMT which can be found at [12]. The database contains about 3500 melodies of mainly pop music. A Java interface allows to submit a hummed query.

Melodyhound [11] by RAINER TYPKE provides a database with tunes of about 17000 folksongs, 11000 classic tunes, 1500 rock/pop tunes and 100 national anthems. One or more of these categories can be choosen to narrow down the database and increase chances for correct answers. Melodyhound uses a three step melody contour representation. The query input can be submitted via keyboard input as Parson code or as whistled input, using a Java application. Furthermore, there is a forum to ask other users.

1.3. MPEG-7 Interfaces

The aim of this paper is to present the use of MPEG-7 descriptors in different stages of QBH systems. The MPEG-7 Audio descriptors are defined in [7]. An overview on MPEG-7 Audio is given in [9, 14].

MPEG-7 provides Descriptors (Ds) for low level signal description and Description Schemes (DS) that describe the signal in a more abstract level. Our system *Queryhammer* uses MPEG-7 descriptors in all processing stages. This allows user queries to be preprocessed by other applications and inserted into the processing path of Queryhammer at any stage.

1.4. Paper overview

This paper is organized as follows: first, the architecture of Queryhammer and a detailed description of all processing steps is given. This is followed by a brief description of the MPEG-7 descriptors used in the system. Then, an evaluation of Queryhammer and practical use of the MPEG-7 interfaces is presented.

2. SYSTEM ARCHITECTURE

The architecture of the system is depicted in figure 1. A microphone takes the hummed input and sends this as a PCM signal to extraction block. The extracted information results here in a MPEG-7 `AudioFundamentalFrequency D` which is given to the transcription part. The transcription block forms a MPEG-7 `MelodyContour` to be compared with all contours residing in the database. A result list is finally presented to the user.

2.1. Extraction

The extraction block is also referred as the *acoustic front end* [4]. In existing systems, this part is often implemented as a Java applet [11, 12].

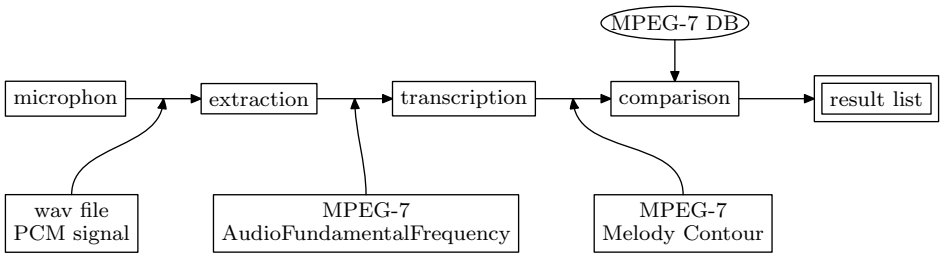

Fig. 1: The basic architecture of a QBH system. The Queryhammer system is connected to a MPEG-7 database.

Processing steps After recording the signal with a computer sound card the signal is band pass filtered to reduce environmental noise and distortion. In this system a sampling rate of 8000 Hz is used. The signal is band limited to 80 to 800 Hz, which is sufficient for sung input [10]. This frequency range corresponds to a musical note range of $D\sharp_2$–G_5. Figure 2 shows the score of a possible user query. This query results in a wave form as depicted in figure 3.

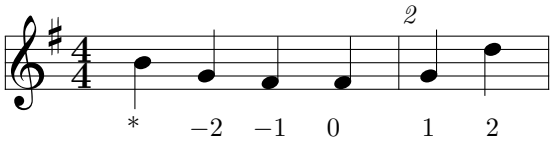

Fig. 2: Some notes a user might query.

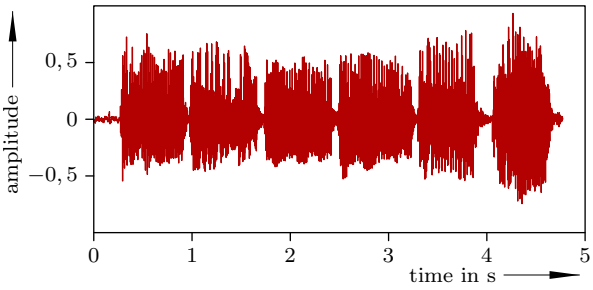

Fig. 3: The PCM signal of the user query.

Following preprocessing, the signal is analysed by a pitch detection algorithm. Queryhammer uses the autocorrelation method as used in the well known speech processing tool Praat by Paul Boersma [3]. This algorithm weights the autocorrelation function using a HANNING window, followed by a parabolic interpolation in the lag domain for higher precision. The result of the pitch detection done using the signal in figure 3 is shown in figure 4.

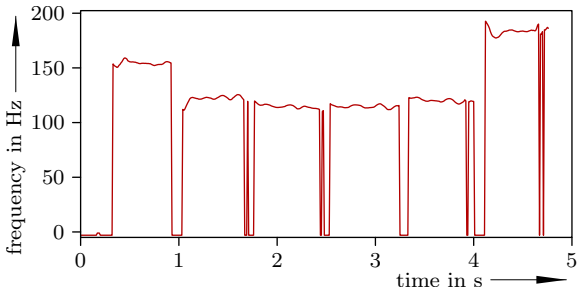

Fig. 4: The fundamental frequency of the singing input.

Tempo as an information on rhythmic features of the hummed query is an important feature. The extraction stage uses the beat detection algorithm for the estimation of beats per minute (BPM) as described in [15]. Note that extracted tempo information from sung input is usually estimated with a certain level of uncertainty.

Interfaces The extraction block reads a PCM signal and outputs the fundamental frequency. Therefore, the interface for this stage provides an input

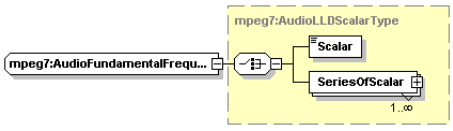

Fig. 5: The MPEG-7 `AudioFundamentalFrequencyType` used for the `AudioFundamentalFrequency D` [9].

interface for the PCM signal and an output interface using the MPEG-7 `AudioFundamentalFrequency D` [7]. The structure of this D is shown in figure 5. The `AudioFundamentalFrequency D` holds the frequency information in a `AudioLLDScalarType`, which can be a `Scalar` or a `SeriesOfScalar` type. For the `AudioFundamentalFrequency`, the `Scalar` holds information about the fundamental frequency of one block and a confidence measure between 0 and 1. The attributes `loLimit` and `hiLimit` of the `AudioFundamentalFrequency D` specify the range of possible frequencies. The `hopSize` defaults to 10 ms.

In current MPEG-7 Audio standard, no tempo information for music is described (it is expected to be part of the next version). Therefore, only an internal interface for passing the BPM information to the transcription stage is used.

2.2. Transcription

The transcription block transcribes the extracted information into the representation that is needed for comparison. The main task is to segment the input stream into single notes. This can be done using amplitude or pitch information [10].

Processing steps First, we segment the hummed query into single events using pitch information. Fundamental frequency is assigned to a note name of the well tempered scale. The frequencies of all notes of the chromatic scale can be calculated according to

$$f(n) = f_0 \cdot 2^{\frac{n}{12}} \tag{1}$$

where n is the number of the note in the scale. If f_0 is choosen standard pitch 440 Hz, $n = 0 \ldots 11$ result in frequencies of the chromatic scale from A_4 to $G\sharp_5$.

If f_0 is chosen to 8.1757989156 Hz, n corresponds to the MIDI note number (A at 440 Hz has number $n = 69$) [1]. Deviations from a tuned note with frequency f_1 can be measured in *cent* using

$$c(f) = 1200 \cdot \log_2 \left(\frac{f}{f_1} \right) \tag{2}$$

In the transcription block, a new event is detected if $|c(f)| > 50$ cent. All blocks with a smaller frequency deviation are assigned to one event. Figure 6 shows the segmented query.

Events shorter than 80 ms are discarded. Since no exact transcription of the singing signal is required, this is sufficient for building a melody contour (figure 7).

The melodic and rhythmic information is now transcribed into a more general representation, the melody contour. The contour used in this system is specified by the MPEG-7 standard and uses five contour values (figure 8).

Interfaces Input is the fundamental frequency descriptor. The output format, the MPEG-7 `MelodyContourType` is shown in figure 9. It contains a field `Contour` with the 5-level pitch contour representation of the melody [7] using the values shown in table 1. The field `Beat` contains the beat numbers where the contour changes take place, truncated to whole beats. The beat information is stored as a series of integers.

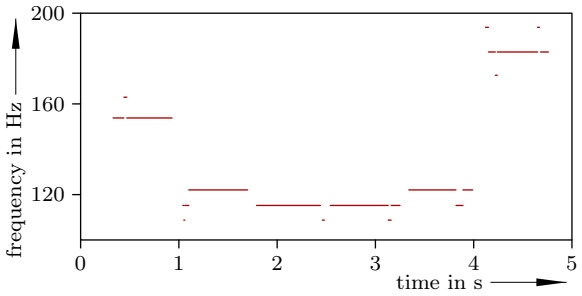

Fig. 6: The events extracted form the frequency signal.

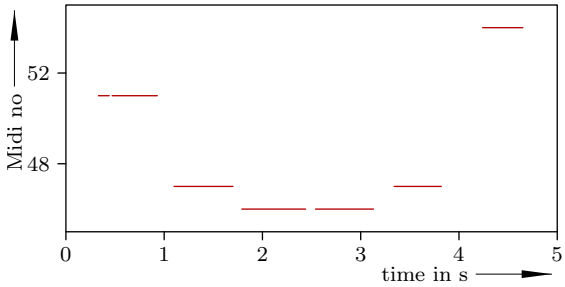

Fig. 7: The note events extracted from the eventlist.

Table 1: Melodic contour intervals defined for 5 step representation.

Contour value	Change of c(f) in cents
-2	$c \leq -250$
-1	$-50 \leq c < -250$
0	$-50 < c < 50$
1	$50 \leq c < 250$
2	$c \geq 250$

2.3. Comparison

The transcription result is used as database query. Several distance measures can be used to find a similar piece of music. The database contains a collection of already transcribed melodies formated according to the `MelodyContourType`. Both, MelodyData and BeatData can be taken into account for the distance measure. Queryhammer uses the algorithm proposed by Youngmoo Kim [8].

Processing steps The algorithm compares the contour values of each beat of query and investigated song. The query is aligned with the song beginning at beat B. A similarity score is then calculated for each beat. Subsequently, the scores of all beats are summed up and normalized by the amount of beats. This calculation is done for all beats B of the investigated song and the highest score chosen as the overall score.

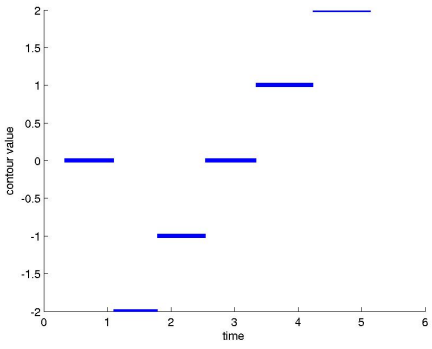

Fig. 8: The transcribed Melody contour shows all five steps from -2 to 2. The first value is chosen to 0.

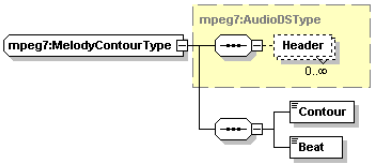

Fig. 9: MPEG-7 MelodyContourType [9].

Interfaces Both input interfaces of the comparison block use the MPEG-7 `MelodyContourType`.

3. TEST SYSTEM AND EVALUATION

The Queryhammer system is implemented in Matlab. The graphical user interface (GUI) is depicted in figure 10.

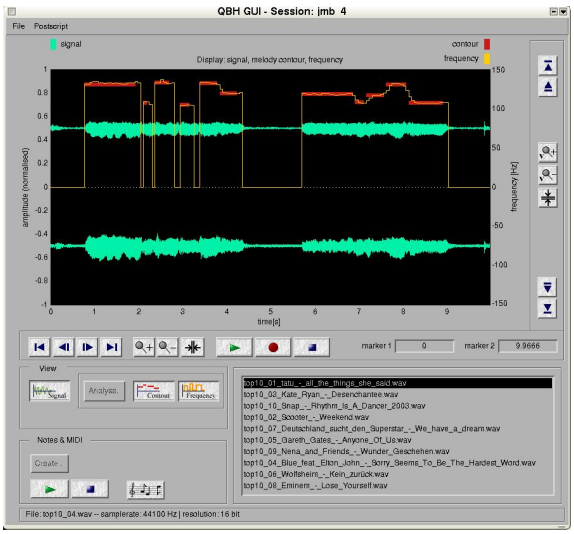

Fig. 10: The GUI of the Queryhammer system.

To query by humming, the user input can be recorded using a microphone. Alternatively, an existing signal can be loaded using the File dialog. Once the processing of the query is finished, a list of the ten best matching results is presented.

To use more sophisticated transcription tools, the MPEG-7 interface of Queryhammer can be used. A recorded hummed signal can thus be processed using any transcription tool that outputs the `AudioFundamentalFrequency D`. This description stored in an XML MPEG-7 file is loaded into the Queryhammer system, and a query is extracted.

Furthermore, a melody contour can be fed directly into the comparison stage using the `MelodyContourType`. A melody contour could be generated via keyboard input as it is done in [13].

The extraction of beat information from hummed input as done in the extraction stage is a challenging task. To evaluate our system, we extract two melody contour sets from our query database. Queryhammer generates contour and beat vector automatically. The second set is generated using a simpler transcription tool which is only capable of extracting the contour vector. The beat vector is manually transcribed.

4. RESULTS

The test setup consists of 59 data base songs and forty queries. The data base includes the German Top Ten Hits of March 2003. Midi files of all data base songs were retrieved from the WWW. Forty queries were generated by four test singers. They were asked to hum an *arbitrary* part of the melody of each of the Top Ten Hits from the data base (see table 2). Note, that this experimental setup is most challenging for the query system. Not two users used the same melodies, nor a metronome click was provided while humming.

A couple of tools are used to prepare the data for the evaluation. There is Wei Chai's tool *MelodyExtract* which transcribes the midi Top 10 songs into a text representation (`.mel`). A tool was written for transcribing those files into XML MPEG-7 melody contour files.

Thus, we created forty MPEG-7 melody contours for the queries and 59 MPEG-7 melody contours for the data base songs. To evaluate Queryhammer we use the matching algorithm described in [8]. This matching algorithm takes into account MPEG-7 contour and beat values. Comparison of each query and each data base song yields the following results (figure 11, figure 12).

The abscissa shows ten groups of four bars. The ten groups refer to the Top 10 songs the singers one to four have hummed parts of, the four bars of each group refer to the singers. The ordinate depicts distance, i.e. how far is the hummed song from being most similar to the query. If the distance is 0 the title hummed by the user is on position one in the

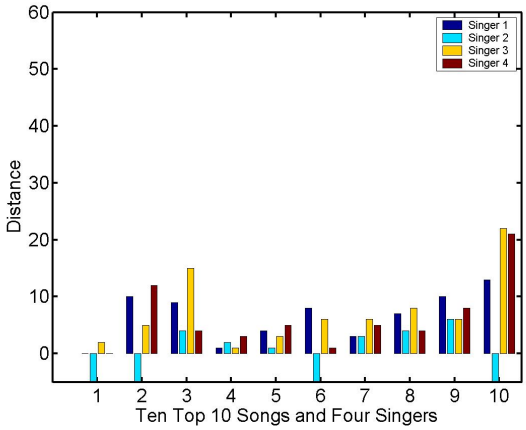

Fig. 11: Results for query set with manually transcribed beat vector.

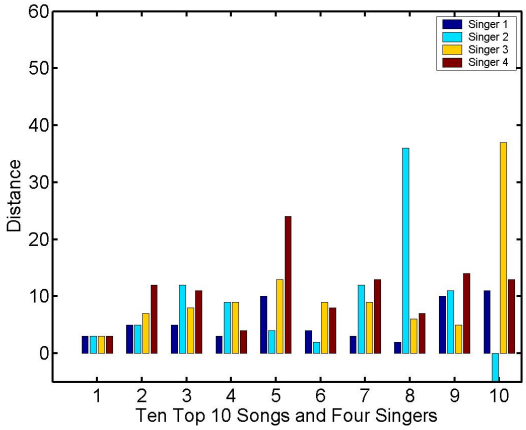

Fig. 12: Results for query set generated by Queryhammer.

Table 2: Artists and titles of the German Top Ten single charts from March 2003.

#	Artist	Title
1	TATU	All the Things She Said
2	Scooter	Weekend
3	Kate Ryan	Desenchantee
4	Blue feat. Elton John	Sorry Seems to Be the Hardest Word
5	Gareth Gates	Anyone of Us
6	Wolfsheim	Kein zurück
7	Deutschland sucht den Superstar	We Have a Dream
8	Eminem	Lose Yourself
9	Nena and Friends	Wunder geschehen
10	Snap	Rhythm Is a Dancer 2003

result list, distance 1 means position two and so on. If the results table contains two songs that have received equal score, the highest distance utilizing a data base of ten songs could then only be eight. A distance of -5 is assigned to invalid queries. Figure 11 depicts results for the query set with manually transcribed beat vectors, figure 12 shows the results for the query set extracted by Queryhammer.

Queries are marked *invalid* if the melody extraction tool has extracted less than three notes. This happened four times for singer two, who repeatedly sang the queries with fast tempo. In this case the extraction algorithm is not able to track pitch correctly. Thus, no reasonable melody contour is extracted.

As a result, the comparison in figure 11 and figure 12 shows better query matches for the hand transcribed query set. This is due to the unconfident tempo information obtained by the automatic extraction. On the other side, less invalid queries occur using the data set extracted by Queryhammer.

5. CONCLUSIONS AND FUTURE WORK

We tested our system using a real world szenario where users had free choice.

Our results differ for manually and automatically generated query sets. According to a real world scenario users had free choice of melody part, tempo and query length. Depending on the quality of queries, different extraction and transcription tools are useful. Therefore, flexible input interfaces for MIR systems are highly desirable. On the input side the existing system could be extended to use additional MPEG-7 low level descriptors, e.g. the `AudioPowerType` or the `SilenceType` might be useful for segmentation issues [7].

In the Queryhammer system the automatic beat detection of the hummed input turned out to be most difficult. To gain better results for the automatic transcription of queries more work has to be done in this field of research.

To use melody contours without beat information different distance measures can be used. In future work the comparison block is going to be enhanced by different distances measures.

6. REFERENCES

[1] MIDI Manufacturers Association. Website. `http://www.midi.org`.

[2] Juan Pablo Bello, Giuliano Monti, and Mark Sandler. Techniques for automatic music transcription. In *International Symposium on Music Information Retrieval*, 2000.

[3] Paul Boersma. Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound. In *IFA Proceedings 17*, 1993.

[4] L. P. Clarisse, J. P. Martens, M. Lesaffre, B. De Baets, H. De Meyer, and M. Leman. An auditory model based transcriber of singing sequences. In *Proceedings of the ISMIR*, pages 116–123, 2002.

[5] Gunnar Eisenberg, Jan-Mark Batke, and Thomas Sikora. Beatbank - an mpeg-7 com-

pliant query by tapping system. In *Proc. of the 116th AES Convention*, Berlin, May 2004.

[6] Emilia Gómez, Fabien Bouyon, Perfecto Herrera, and Xavier Amatriain. Using and enhencing the current mpeg-7 standard for a music content processing tool. In *Proc. of the 114th AES Convention*, Amsterdam, March 2003.

[7] ISO/IEC. *Information Technology — Multimedia Content Description Interface — Part 4: Audio*, 15938-4:2001 edition, June 2001.

[8] Youngmoo E. Kim, Wei Chai, Ricardo Garcia, and Barry Vercoe. Analysis of a contour-based representation for melody. In *Proc. International Symposium on Music Information Retrieval*, October 2000.

[9] B. S. Manjunath, Philippe Salembier, and Thomas Sikora, editors. *Introduction to MPEG-7*. Wiley, 1st edition, 2002.

[10] Rodger J. McNab, Lloyd A. Smith, and Ian H. Witten. Signal processing for melody transcription. In *Proceedings of the 19th Australasian Computer Science Conference*, 1996.

[11] Melodyhound. Melody recognition and search. `http://name-this-tune.com`. developed by Rainer Typke.

[12] Musicline. Die ganze Musik im Internet. `http://www.musicline.de/de/melodiesuche/input`. phononet QBH system powered by Fraunhofer IDMT.

[13] Lutz Prechelt and Rainer Typke. An interface for melody input. *ACM Transactions on Computer-Human Interaction*, 8(2):133–149, 2001.

[14] Schuyler Quackenbush and Adam Lindsay. Overview of mpeg-7 audio. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(6):727–729, June 2001.

[15] Eric D. Scheirer. Tempo and beat analysis of acoustic musical signals. *J. Acoust. Soc. Am.*, 103(1):588–601, January 1998.

[16] Rainer Typke. Mir systems: A survey of music information retrival systems. `http://mirsystems.info`.

[17] A. Uitdenbogerd and J. Zobel. Music ranking techniques evaluated. In M. Oudshoorn, editor, *Proceedings of the Australasian Computer Science Conference*, pages 275–283, Melbourne, Australia, January 2002.

[18] A. L. Uitdenbogerd and J. Zobel. Matching techniques for large music databases. In D. Bulterman, K. Jeffay, and H. J. Zhang, editors, *Proceedings of the ACM Multimedia Conference*, pages 57–66, Orlando, Florida, November 1999.