Pattern Recognition Letters 36 (2014) 272–280

Contents lists available at SciVerse ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Query by humming: Automatically building the database from music recordings

Martín Rocamora [a,*], Pablo Cancela [a], Alvaro Pardo [b]

[a] Institute of Electrical Engineering, School of Engineering, Universidad de la República, Uruguay
[b] Department of Electrical Engineering, School of Engineering and Technologies, Universidad Católica del Uruguay, Uruguay

ARTICLE INFO

Article history:
Available online 20 April 2013

Communicated by Luis Gomez Deniz

Keywords:
Voice based multimodal interfaces
Music information retrieval
Query by humming
Singing voice separation
Melody extraction

ABSTRACT

Singing or humming to a music search engine is an appealing multimodal interaction paradigm, particularly for small sized portable devices that are ubiquitous nowadays. The aim of this work is to overcome the main shortcoming of the existing query-by-humming (QBH) systems: their lack of scalability in terms of the difficulty of automatically extending the database of melodies from audio recordings. A method is proposed to extract the singing voice melody from polyphonic music providing the necessary information to index it as an element in the database. The search of a query pattern in the database is carried out combining note sequence matching and pitch time series alignment. A prototype system was developed and experiments are carried out pursuing a fair comparison between manual and automatic expansion of the database. In the light of the obtained performance (85% in the top-10), which is encouraging given the results reported to date, this can be considered a proof of concept that validates the approach.

1. Introduction

The constant increase in computer storage and processing capabilities has made possible to collect vast amounts of information, most of which is available online. Today, people interact with this information using various devices, such as desktop computers, mobile phones or PDAs, posing new challenges at the interface between human and machine. Yet, the most common case of information access still involves typing a query to a search engine. There is a need for new human–machine interaction modalities that exploit multiple communication channels to make our systems more usable.

Among the information available there are huge music collections, containing not only audio recordings, but also video clips and other music-related data such as text (e.g. tags, scores, lyrics) and images (e.g. album covers, photos, scanned sheet music). A query for music search is usually formulated in textual form, by including information on composer, performer, music genre, song title or lyrics. However, other modalities to access music collections can also be considered that allow more intuitive queries. For instance, to provide a musical excerpt as an example and obtain all the pieces that are similar in some sense, namely query-by-example,[1] or to retrieve a musical piece by singing or humming a few notes of its melody, which is called query-by-humming (QBH). This offers an interesting interaction possibility, in particular for small size devices such as portable audio players, and requires no music theoretical knowledge from the user. Additionally, it can be combined with traditional metadata-based search and visual user interfaces to offer multimodal input and output, in the form of visual and auditory information.

Dealing with multimodal music information requires the development of methods for automatically establishing semantic relationships between different music representations and formats, for example, sheet music to audio synchronization or lyrics to audio alignment (Müller et al., 2012). Much research in audio signal processing over the last years has been devoted to music information retrieval (Müller, 2007; Lerch, 2012), i.e. the extraction of musically meaningful content information from the automatic analysis of an audio recording. This involves diverse music related problems and applications, from computer aided musicology (Leech-Wilkinson, 2009), to automatic music transcription (Klapuri and Davy, 2006) and recommendation (Celma, 2010). Many research efforts have been devoted to dealing with the singing voice, tackling problems such as singing voice separation (Li and Wang, 2006) and melody transcription (Ryynänen and Klapuri, 2006). The incorporation of these techniques into multimodal interaction systems can lead to novel and more engaging music learning, searching and gaming applications.

Even though the problem of building a QBH system has received a lot of attention from the research community for more than a decade (Pardo et al., 2003), the automatic generation of the melody

* Corresponding author. Tel.: +598 27110974; fax: +598 27117435.
E-mail addresses: rocamora@fing.edu.uy (M. Rocamora), cancela@fing.edu.uy (P. Cancela), apardo@ucu.edu.uy (A. Pardo).

[1] Audio fingerprinting techniques are used in this case, being Shazam (http://www.shazam.com/) probably one of the best known commercial services of this kind.

0167-8655/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.patrec.2013.04.006

database against which the queries are matched remains an open issue. In all the proposed systems – with very few exceptions – the database consists of music in symbolic notation, e.g. MIDI files. This is due to the lack of sufficiently robust automatic methods to extract the melody directly from a music recording. Although there is a great amount of MIDI files online, music is mainly recorded and distributed as audio files. Hence, the scope of this approach is limited because of the need of manually transcribing (i.e. audio to MIDI) every new song of the database. A way to circumvent this problem is to build a database of queries provided by the users themselves and to match new queries against the previously recorded ones (Pardo et al., 2008). This approach drastically simplifies the problem and is applied in music search services such as SoundHound.[2] However, the process is not automatic but relies on user contributions. Besides, a new song can not be found until some user records it for the first time. In order to extend QBH systems to large scale it is necessary to develop a full automatic process to build the database. There are only a few proposals of a system of this kind (Song et al., 2002; Duda et al., 2007; Ryynnen and Klapuri, 2008; Salamon et al., 2013) and results indicate there is still a lot of room for improvement to reach the performance of the traditional systems based on symbolic databases.

In this paper a method for automatically building the database of a QBH system is described, in which the singing voice melody is extracted from a polyphonic music recording. In our previous work (Rocamora and Pardo, 2012) a technique for singing voice detection and separation was presented. The contribution of the present work is the application of this technique to a music retrieval problem involving a voice-based multimodal interface. A prototype is built as a proof of concept of the proposed method and a study is conducted that compares the performance of a previously developed QBH system (López and Rocamora, 2006) when using a database of MIDI files and when using melodies extracted automatically from the original recorded songs. The rest of this document is organized as follows. Next section briefly describes the QBH system used in the experiments. The method for extracting the singing voice melody from polyphonic music recordings is presented in Section 3. In Section 4 the experiments carried out for assessing the performance of the QBH on the automatically obtained database are described and results are reported. The paper ends with some critical discussion on the present work and conclusions.

2. Query-by-humming system

The existing QBH systems can be divided, from its representation and matching technique, basically into two approaches. The most typical solution is based on a note by note comparison (Ghias et al., 1995; McNab et al., 1996). The query voice signal is transcribed into a sequence of notes and the best occurrences of this pattern are identified in a database of tunes (typically MIDI files). The melody matching problem poses some challenges to be considered. A melody can be identified in spite of being performed at different pitch and at different tempo. Additionally, sporadic pitch and duration errors or expressive features modify the melodic line but still allow the melody to be recognized. In the matching step, pitch and tempo invariance are typically taken into account by coding the melodies into pitch and duration contours. By means of flexible similarity rules it is possibly to achieve some tolerance to singing mistakes and automatic transcription errors. Automatic transcription of the query inevitably introduce errors that tend to deteriorate matching performance. For this reason, another usual approach avoids the automatic transcription, comparing melodies as fundamental frequency (F0) time series (Hu and Dannenberg, 2002; Zhu and Shasha, 2003). Unfortunately, this involves working with long sequences, very long compared to note sequences, and therefore it implies high computational burden. Moreover, in many proposals the user is required to sing a previously defined melody fragment (Hu and Dannenberg, 2002; Zhu and Shasha, 2003) in order that the query exactly matches an element of the database. This is because of the difficulty of searching subsequences into sequences providing pitch and tempo invariance.

In our previous work (López and Rocamora, 2006), a way of combining both approaches was introduced that exploits the advantages of each of them. Firstly, the system selects a reduced group of candidates from the database using note by note matching. Then, the selection is refined using fundamental frequency time series comparison. Finally, a list of musical pieces is retrieved in a similarity order. The system architecture is divided in two main stages, as depicted in Fig. 1. The first one is the transcription of the query into a sequence of notes. To do that, the F0 contour is computed using a very well know technique based on the difference function (de Cheveigné and Kawahara, 2002). Then, the audio signal is segmented into notes by computing energy envelopes from different frequency bands and detecting salient events (Klapuri, 1999). Besides, evident pitch changes that do not exhibit an energy increment are identified (e.g. legato notes) and considered in the segmentation. Each note is described by a pitch value, an onset time and a duration. To assign a pitch value to each note the median of its fundamental frequency contour is taken. Then the tuning of the whole sequence is adjusted by computing the most frequent deviation from the equal tempered scale, subtracting this value for every note and rounding to the nearest MIDI number (Pollastri, 2003).

In the second stage, the notes of the query are matched to the melodies of the database. The pitches sequence $A = (a_1, a_2, \ldots, a_n)$ is encoded as a sequence of intervals $\overline{A} = (a_2 - a_1, a_3 - a_2, \ldots, a_n - a_{n-1})$, so that a melody $\widehat{A}$ transposition of A has the same interval representation. In a similar way, given the duration sequence, $B = (b_1, b_2, \ldots, b_n)$, a tempo invariant representation is computed as the relative duration sequence $\overline{B} = (b_2/b_1, b_3/b_2, \ldots, b_n/b_{n-1})$ (Pardo and Brimingham, 2002). When singing carelessly gross approximations in duration take place, so the inter-onset interval is used as a more consistent representation of duration and relative durations are smoothed and quantized through $q_i = \text{round}(10\log_{10}(b_{i+1}/b_i))$, obtaining the sequence $B_q = (q_1, q_2, \ldots, q_{n-1})$ (Pollastri, 2003).

Finding good occurrences of the codified query in the database is basically an approximate string matching problem. For this task, Dynamic Programming is used to compute an edit distance that combines duration and pitch information (Lemström, 2000). In this combination, pitch values are considered more important because duration information is less discriminative and not so reliable. The edit distance, d_{ij}, is computed recursively as the minimum of the set of values shown in Eq. (1).

$$d_{ij} = \min \begin{cases} d_{i-1,j} + 1, & & \text{(insertion)} \\ d_{i,j-1} + 1, & & \text{(deletion)} \\ d_{i-1,j-1} + 1, & & \text{(note substitution)} \\ d_{i-1,j-1} - 1, & |\overline{a}_i - \overline{a}'_j| < 2 \text{ and} & \text{(coincidence)} \\ & |q_i - q'_j| < 2 & \\ d_{i-1,j-1} & |\overline{a}_i - \overline{a}'_j| < 2 & \text{(duration substitution)} \end{cases} \tag{1}$$

The last two values of the set are only considered if the corresponding conditions are met, where $\overline{a}$ and $\overline{a}'$ refer to the pitch interval of the query and the database element respectively, whereas q and q' correspond to their quantized relative duration. Finally, a similarity score is computed normalizing the edit distance to take values between 0 and 100,

[2] http://www.soundhound.com/.