International Journal of Multimedia Information Retrieval manuscript No.
(will be inserted by the editor)

Tonal Representations for Music Retrieval: From Version Identification to Query-by-Humming

Justin Salamon · Joan Serrà · Emilia Gómez

Received: 21 June 2012 / Revised: 25 September 2012 / Accepted: 13 November 2012

Abstract In this study we compare the use of different music representations for retrieving alternative performances of the same musical piece, a task commonly referred to as version identification. Given the audio signal of a song, we compute descriptors representing its melody, bass line and harmonic progression using state-of-the-art algorithms. These descriptors are then employed to retrieve different versions of the same musical piece using a dynamic programming algorithm based on nonlinear time series analysis. First, we evaluate the accuracy obtained using individual descriptors, and then we examine whether performance can be improved by combining these music representations (i.e. descriptor fusion). Our results show that whilst harmony is the most reliable music representation for version identification, the melody and bass line representations also carry useful information for this task. Furthermore, we show that by combining these tonal representations we can increase version detection accuracy. Finally, we demonstrate how the proposed version identification method can be adapted for the task of query-by-humming. We propose a melody-based retrieval approach, and demonstrate how melody representations extracted from recordings of a cappella singing can be successfully used to retrieve the original song from a collection of polyphonic audio. The current limitations of the proposed approach are discussed in the context of version identification and query-by-humming, and possible solutions and future research directions are proposed.

Keywords Music similarity · version identification · cover song detection · query by humming · melody extraction · bass line · harmony · music retrieval

This research was funded by: Programa de Formación del Profesorado Universitario (FPU) of the Ministerio de Educación de España, Consejo Superior de Investigaciones Científicas (JAEDOC069/2010), Generalitat de Catalunya (2009-SGR-1434) and the European Commission, FP7 (Seventh Framework Programme), ICT-2011.1.5 Networked Media and Search Systems, grant agreement No. 287711.

Justin Salamon
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
E-mail: justin.salamon@upf.edu

Joan Serrà
Artificial Intelligence Research Institute (IIIA-CSIC)
Spanish National Research Council, Bellaterra, Spain
E-mail: jserra@iiia.csic.es

Emilia Gómez
Music Technology Group
Universitat Pompeu Fabra, Barcelona, Spain
E-mail: emilia.gomez@upf.edu

1 Introduction

Music similarity is a key aspect in the development of music retrieval systems [25], and developing automatic methods for quantifying music similarity addresses part of a more general problem: making sense of digital information [28]. Music similarity is, however, an ambiguous term. Apart from involving different musical facets such as instrumentation, tonality or rhythm, it also depends on cultural (or contextual) and personal (or subjective) aspects [15, 21]. There are many factors involved in music similarity judgments, and some of them are difficult to measure [1].

To assess the similarity between music documents, some Music Information Retrieval (MIR) researchers have devoted their efforts to the related task of *version identification*. Remarkably, and in contrast to music similarity, the relation between versions is context-independent and can be objectively measured [35]. In addition, research on version identification can yield valuable clues on how music similarity can be modeled. This task, of automatically detecting versions of the same musical piece, has received much attention from the research community over recent years (see [36] for a survey). Potential applications range from the detection of copyright violations on websites such as YouTube, to the automation of computational analyses of musical influence networks [4]. Version identification on its own also represents an attractive retrieval task for end users.

Systems for the automatic detection of versions exploit musical facets which remain mostly unchanged across different renditions, primarily tonal information [36]. In this context, the term tonality refers, in a broad sense, to "the systematic arrangement of pitch phenomena and relations between them" [16]. Perhaps the most common tonal representation for version identification is chroma. Chroma features (also called pitch class profiles) represent, in polyphonic music signals, the relative intensity of each of the 12 semitones in an equal-tempered chromatic scale, discarding octave information. As such, chroma features are related to harmony, understood as "the combining of notes simultaneously, to produce chords, and successively, to produce chord progressions" [5]. Common techniques for comparing sequences of chroma features include dynamic time warping and simple cross-correlation [36]. Another tonal representation that has been considered for version identification is the predominant melody, either by attempting to fully transcribe it [42], or by using it as a mid-level representation for computing similarity [23]. Melodic representations have also been widely used for related tasks such as query-by-humming [6] or music retrieval using symbolic data [43].

Whilst good results have been achieved using single music representations (in particular harmony represented by chroma features [36]), some recent studies suggest that version detection could be improved through the combination of different musical cues [9, 19, 35]. However, not much research has been carried out in this direction. One of the first studies to automatically extract features derived from different music representations for version identification was conducted by Foucard et al. [9], in which a source separation algorithm was used to separate the melody from the accompaniment. The authors then compared the performance of a version identification system using the melody, the accompaniment, the original mix, and their combination, by employing different fusion schemes. The study showed that considering different information modalities (i.e. main melody and accompaniment) is a promising research direction, but also noted the intrinsic limitation of simple fusion schemes whose capabilities seemed to be limited to merging modalities that carry more or less the same type of information. In the work of Ravuri and Ellis [29], the task of detecting musical versions was posed as a classification problem, and different similarity measures were combined to train a classifier for determining whether two musical pieces were versions or not. However, only chroma features were used to derive these similarity measures. Therefore, they were all in fact accounting for the same musical facet: the harmony.

In this paper we expand the study of version identification using different music representations. In particular, we explore three related yet different tonal representations: melody, bass line and harmony. To extract the melody, we employ a state-of-the-art melody extraction algorithm [32] which returns a per-frame estimate of the fundamental frequency corresponding to the pitch of the predominant melody. The bass line is extracted using a modified version of the same algorithm. Harmony is represented by means of chroma features [11], which have already been used successfully in state-of-the-art version identification systems [36]. Beyond comparing identification performance for each of the three tonal representations separately, we also study their combination. For this we use the power of a standard classification approach, similar to Ravuri and Ellis [29]. In addition, we compare a number of classification algorithms and assess their ability to fuse the information coming from the three different representations.

As mentioned earlier, a task very much related to version identification is that of query-by-humming (QBH). A QBH system allows users to search for songs by singing or humming part of the melody. As such, QBH can be considered a special case of version identification in which the version used as a query is produced by the user (rather than an artist) and contains only melody information. The task has received much attention from the research community, both because of the challenges it presents in computational music similarity and because of its attractive potential application for end users (see [6] for a comparative evaluation and [18] for a more recent survey). One important problem in the creation of QBH systems is the generation of a melody database (song index) against which the sung queries are to be compared. Until recently, the lack of reliable melody extraction algorithms meant that most effort was focused on matching queries against symbolic databases (e.g. MIDI files) [6]. Whilst it is possible to find MIDI versions of many songs on the Internet, such an approach will always be limited since it is not feasible to generate (i.e. transcribe) MIDI files manually for very large music collections, not to mention all the new music that will be composed in the future. Another solution that has been proposed is to match queries against other queries, as performed by services such as SoundHound[1] and Tunebot [27]. Whilst this avoids the need for manual transcription, the approach still suffers from the same "cold start" problem – a song "does not exist" until someone records it, so the need for manual labour remains. In light of this, a fully automated solution in which the melody database can be generated automatically is highly attractive. Audio-to-audio QBH algorithms have been proposed in, e.g., [7, 30, 41]. However, results indicate there is still much work to be done before these systems perform as well as audio-to-symbolic QBH (e.g. the approach in [30] obtains an MRR of 0.57 for a database of 427 audio songs compared to 0.91 for a database of 2048 symbolic songs). In the final sections of this paper we describe how our proposed version identification approach can be adapted into a fully automated audio-to-audio QBH system with very few modifications.

The structure of the remainder of the paper is as follows: in Section 2 we describe the music representations compared in this study, how we compute descriptors to represent them, the computation of version similarity, and our approach for descriptor fusion. In Section 3 we start by describing our evaluation strategy for version identification, including the music collection and evaluation measures used to assess the retrieval accuracy. This is followed by the results of the evaluation for both individual descriptors and descriptor fusion, including a detailed discussion of the results. In Section 4 we explain how the proposed approach can be applied to the related task of query-by-humming. We describe the test collections, evaluation measures and queries used to evaluate the approach and discuss the retrieval results. The contributions of the paper are summarised in Section 5.

2 Methodology

In the following subsections we present our approach for (dis)similarity-based music retrieval. We start by describing the different tonal representations extracted from the audio signal, which are based on the melody, bass line and harmonic progression of a musical piece. For melody and bass line, we define a representation abstraction process for removing performance-specific information that may hinder the matching procedure. Next, we explain how song matching is performed using a local alignment algorithm based on nonlinear time series analysis. Finally, we describe our approach for integrating the different music representations for version identification using a classification technique.

2.1 Tonal Representations

2.1.1 Melody Representation

To extract a representation of the melody from the audio signal we use the state-of-the-art melody extraction algorithm presented in [32]. This algorithm, which is based on the characterisation of melodic pitch contours, obtained the highest mean overall accuracy in the most recent MIREX evaluation campaign [31]. A brief summary of the different stages of the algorithm is provided here.

In the first stage of the algorithm, the audio signal is analyzed and spectral peaks (sinusoids) are extracted [32, 33]. This process is comprised of three main steps: first, a time-domain equal loudness filter is applied [45], which has been shown to attenuate spectral components belonging primarily to non-melody sources [33]. Next, the short-time Fourier transform is computed and the local maxima (peaks) of the spectrum are detected at each frame. In the third step, the estimation of the spectral peaks' frequency and amplitude is refined by calculating each peak's instantaneous frequency (IF) using the phase vocoder method [8] and re-estimating its amplitude based on the IF. The detected spectral peaks are subsequently used to compute a representation of pitch salience over time: a *salience function*. The salience function is based on harmonic summation with magnitude weighting, and spans a range of almost five octaves from 55Hz to 1760Hz. Further details are provided in [33]. In the next stage, the peaks of the salience function are grouped over time using heuristics based on auditory streaming cues [3]. This results in a set of pitch contours, out of which the contours belonging to the melody need to be selected. The contours are automatically analyzed and a set of contour characteristics is computed. In the final stage of the system, the contour characteristics and their distributions are used to filter out non-melody contours. The melody F0 at each frame is selected out of the remaining pitch contours based on their salience. A full description of the melody extraction algorithm, including a thorough evaluation, is provided in [32].

2.1.2 Bass Line Representation

The bass line is extracted by adapting the melody extraction algorithm described above. Instead of applying an equal loudness filter (which attenuates low frequency

[1] `http://www.soundhound.com/`

content), we apply a low-pass filter with a cutoff frequency of 261.6Hz, as proposed in [12]. The window size is increased to 185ms, since for the bass we require more frequency resolution. The salience function is adjusted to cover a range of two octaves from 27.5Hz to 110Hz. As before, the salience peaks are grouped into pitch contours. However, since we do not expect other instruments to compete for predominance in the bass frequency range, the detailed contour characterisation and filtering used for melody extraction is less important in the case of bass line extraction. Therefore, the bass line is selected directly from the generated contours based on their salience.

2.1.3 Representation Abstraction

Once the melody and bass line sequences are extracted, we must choose an adequate representation for computing music similarity or, in the case of this study, a representation for retrieving versions of the same musical piece. Since the matching algorithm can handle transposition, a first guess might be to use the extracted representation as is, i.e. to compare the F0 sequences directly. However, initial experiments showed that this (somewhat naïve) approach is unsuccessful.

When considering the task of version identification, we must take into consideration what kind of musical information is maintained between versions, and what information is subject to change. In the case of the melody, we can expect the general melodic contour to be maintained. However, more detailed performance information is likely to change between versions [36]. Besides changing the key and tempo in which the melody is sung (or played), performers might change the octave in which some segments of the melody are sung to adjust them to their vocal range. More importantly, the use of expressive effects (such as ornaments, glissando and vibrato) will obviously vary across versions. Overall, this means we should aim for a representation which abstracts away specific performance information and details, whilst maintaining the basic melodic tonal progression. To this effect, we defined the following types of information abstraction:

– Semitone abstraction: quantise pitch information into semitones. This will help in removing some local expressive effects.
– Octave abstraction: map all pitch information onto a single octave. This will help in removing potential octave changes of the melody within the piece.
– Interval abstraction: replace absolute pitch information with the difference between consecutive pitch values (delta values). This may provide robustness against key changes.

Before applying any abstraction, all frequency values were converted into a cent scale, so that pitch is measured in a perceptually meaningful way. We then ran initial matching experiments comparing the different degrees of abstraction applied to melody sequences: none, semitone, interval, interval+semitone, and semitone+octave (by definition, the interval and octave abstractions are not compatible). For these experiments we used a collection of 76 songs, described in Section 3.1.1, and evaluated the results as detailed in Section 3.1.2. We found that results using the semitone+octave abstraction were considerably better than the other types of abstraction, obtaining a mean average precision of 0.73, compared to 0.26–0.28 for all other abstractions considered. Perhaps not surprisingly, we note that this abstraction process is quite similar to the one applied for computing chroma features (described in the following section). In particular, the observations above suggest that octave information can be quite detrimental for the task of version identification. For the remainder of the study we use the semitone+octave abstraction for both the melody and bass line descriptors.

The exact abstraction process is as follows: first, all frequency values are converted into cents. Then, pitch values are quantised into semitones, and mapped onto a single octave. Next, we reduce the length of the sequence (whose original hop size is 2.9ms), by summarizing every 150 frames as a pitch class histogram[2]. This produces a shortened sequence where each frame is a 12-bin vector representing the distribution of the pitch classes of the melody over roughly half a second. This window length has been reported to be suitable for version identification by several authors, see e.g. [19, 36]. The motivation for the summary step is two-fold: firstly, it reduces the sequence length and therefore reduces the computation time of the matching algorithm. Secondly, it reduces the influence of very short pitch changes which are more likely to be performance specific (e.g. ornamentations). Finally, the vector of each frame is normalised by the value of its highest bin. The steps of the representation abstraction are depicted in Figure 1 for a melody and in Figure 2 for a bass line.

2.1.4 Harmony Representation

To represent harmony, we compute the sequence of harmonic pitch class profiles (HPCP) [10,11], a specific chroma feature implementation. The HPCP is derived from the frequency-dependent energy in a given range (typically from 50 to 5000Hz) in short-time spectral

[2] The contribution of each frame to the histogram is weighted by the salience of the melody at that frame, determined by the melody extraction algorithm.

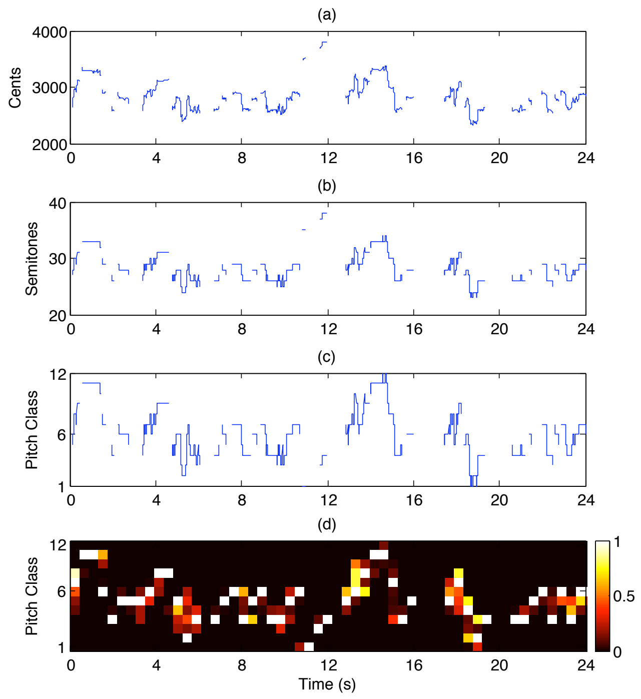

Fig. 1 Melody representation abstraction process: (a) melody pitch in cents, (b) quantised into semitones, (c) mapped onto a single octave, (d) summarised as a pitch histogram and normalised.

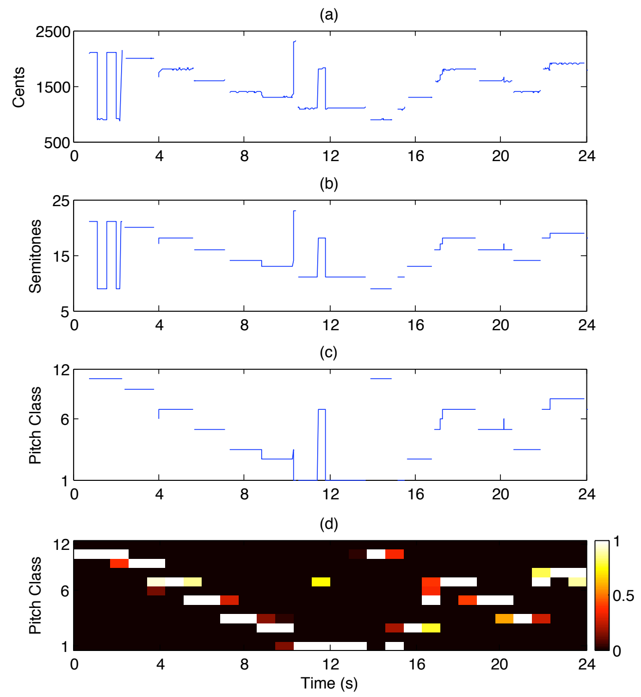

Fig. 2 Bass line representation abstraction process: (a) bass line pitch in cents, (b) quantised into semitones, (c) mapped onto a single octave, (d) summarised as a pitch histogram and normalised.

representations of the audio signal (e.g. 100ms; frame-by-frame extraction). The energy is mapped into an octave-independent histogram representing the relative intensity of each of the 12 semitones of the equal-tempered chromatic scale (12 pitch classes). To normalise with respect to loudness, the histogram is divided by its maximum value, leading to values between 0 and 1. Three important preprocessing steps are applied during the computation of the HPCP: tuning estimation, sinusoid extraction and spectral whitening [10]. This means the HPCP is tuning-frequency independent and robust to noise and changes in timbre, which makes it especially attractive for version identification.

Chroma features are a standard tool in music information research, and the HPCP in particular has been shown to be a robust and informative chroma feature implementation [10,24,37]. For more details we refer the interested reader to [10] and references therein. For the purpose of this study, and in order to facilitate the comparison with previous work on version identification, the HPCP is computed using the same settings and parameters as in [39].

2.2 Matching

For deriving a similarity measure of how well two versions match we employ the $Q_{\max}$ method [39]. This is a dynamic programming algorithm which computes a similarity measure based on the best subsequence partial match between two time series. Therefore, it can be framed under the category of local alignment algorithms. Dynamic programming approaches using local alignment are among the best-performing state-of-the-art systems for version identification [36], and have also been extensively used for melody-based retrieval [6].

The $Q_{\max}$ algorithm is based on general tools and concepts of nonlinear time series analysis [17]. Therefore, since the algorithm is not particularly tied to a specific time series, it can be easily used for the comparison of different (potentially multivariate) signals. Furthermore, the $Q_{\max}$ method has provided the highest MIREX accuracies in the version identification task, using only HPCPs [39]. Therefore, it is a very good candidate to test how melody and bass line compare to HPCPs, and to derive competitive version similarity measures to be used in our fusion scheme.

Given a music collection containing various sets of covers, we use the $Q_{\max}$ algorithm to compute the similarity, or in the case of our method, the dissimilarity, between every pair of songs in the collection. The resulting pairwise dissimilarities are stored in a dissimilarity matrix which can then be used either to evaluate the performance of a single descriptor (as explained in Section 3.1.2), or for descriptor fusion as described in the following section.

2.3 Fusing Descriptors

In addition to evaluating each representation separately for version identification, another goal of this study is to see whether there is any information overlap between these representations, and whether results for this task can be improved by combining them. To this end, we propose a classification approach similar to [29] – each descriptor is used to calculate a dissimilarity matrix between all query-target pairs as described in Section 2.2 (4,515,625 pairs in total for the collection used in this study). Every query-target pair is annotated to indicate whether the query and target are versions or not. We then use five different balanced subsets of 10,000 randomly selected query-target pairs to train a classifier for determining whether two songs are versions of the same piece. The feature vector for each query-target pair is three-dimensional, and contains the dissimilarities produced by the matching algorithm using each of the three representations: melody, bass line and harmony (feature columns are linearly normalised between 0 and 1 prior to classification). In this way we can study different combinations of these descriptors, and most importantly, rather than imposing a simple fusion scheme, the combination of different descriptors is determined in an optimal way by the classifier itself. The only potential limitation of the proposed approach is our employment of a late-fusion strategy (as opposed to early-fusion). Nonetheless, in addition to being straightforward, previous evidence suggests that late-fusion provides better results for version identification [9]. Since the modalities employed in this study are different from the ones in [9], the preferability of a late-fusion strategy over early-fusion should still be validated, and we intend to explore this in future work.

The classification is performed using the Weka data mining software [13]. We compare five different classification algorithms: random forest, support vector machines (SMO with polynomial kernel), simple logistic regression, k-star, and Bayesian network [47]. For all classifiers we use the default parameter values provided in Weka. By comparing different classifiers we are able to assess which classification approach is the most suitable for our task. Furthermore, by verifying that any increase (or decrease) in performance is consistent between classifiers, we ensure that the improvement is indeed due to the descriptor fusion and not merely an artefact of a specific classification technique.

3 Version Identification

Our main goal is to evaluate the proposed tonal representations for the task of version identification. We start by describing our evaluation methodology for this task, followed by the evaluation results and an in-depth discussion of the results. The complete matching process for version identification, using either a single tonal representation or descriptor fusion, is depicted in the block diagram of Figure 3.

3.1 Evaluation Strategy

3.1.1 Music Collection

To evaluate the performance of our method (using either a single music representation or the descriptor fusion strategy), we use a music collection of 2125 songs [38]. The collection includes 523 version sets (i.e. groups of versions of the same musical piece) with an average set cardinality of 4.06. The collection spans a variety of genres including pop, rock, electronic, jazz, blues, world, and classical music. We note that this collection is larger than the collection used in the MIREX version identification task[3], and as such contains a greater variety of artists and styles.

For training the parameters of the $Q_{\max}$ matching algorithm, a small subset of 76 songs from the full collection was used. This 76-song collection was also used for the preliminary experiments on information abstraction outlined in Section 2.1.3. Importantly, we made sure that all songs in this subset have a main melody (and all but 3 have a clear bass line). The full collection, on the other hand, includes versions where there is no main melody (e.g. minus one versions of jazz standards) or no bass line (e.g. singing voice with acoustic guitar accompaniment only). In a manually annotated random sample of 300 songs from the full collection, 88.7% had a melody, 89.7% a bass line and 95.3% included harmony (the confidence intervals for the statistics as representative of the full collection with 95% confidence are 3.3, 3.2 and 2.2 respectively). Whilst we can expect this difference to affect the relative performance of the melody and bass-line-based representations, the statistics are representative of a real-world music collection and, as such, the results for this collection will reflect those we would expect to obtain in a real-world scenario.

3.1.2 Evaluation Measures

The dissimilarity matrix produced by each descriptor can be used to generate an ordered list of results for each query. The relevance of the results (ideally versions of a query should all appear at the top of the list) can

[3] `http://www.music-ir.org/mirex/wiki/Audio_Cover_Song_Identification`

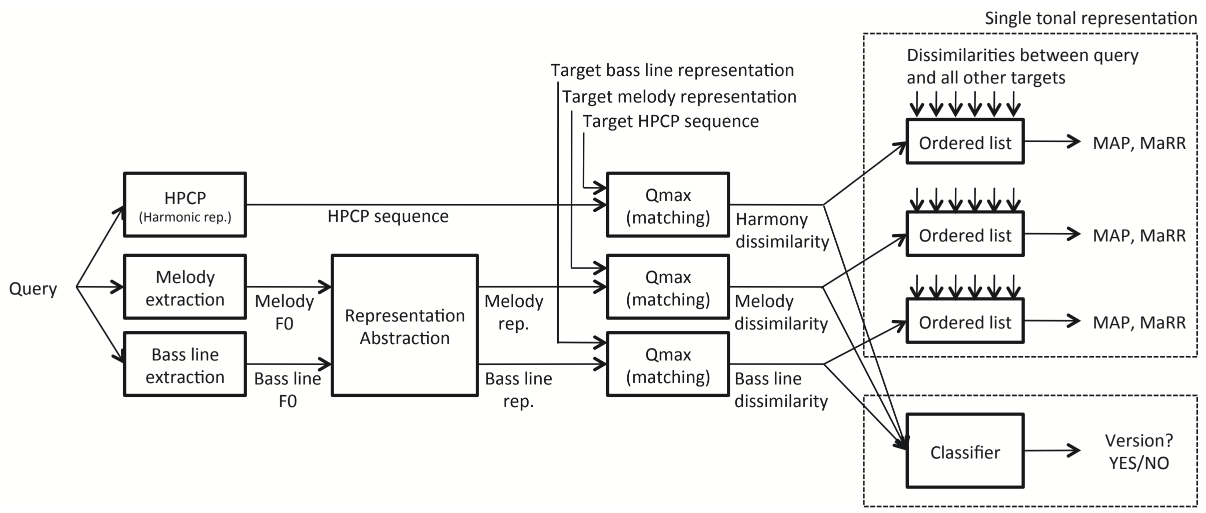

Fig. 3 Matching process for version identification using either a single tonal representation (top right corner) or descriptor fusion (bottom right corner).

then be evaluated using standard information retrieval metrics, namely the mean average precision (MAP) and the mean reciprocal rank (MRR) [22]. Note that since we want to retrieve all versions of the query in our collection we cannot compute the MRR (which assumes there is only one correct answer) directly. Hence, we define a measure we refer to as the *mean averaged reciprocal rank* (MaRR). For a given query, the averaged reciprocal rank (aRR) is given by computing the average of the reciprocal ranks of all the targets in the result list that are versions of the query. The MaRR is then the mean aRR over all queries. For assessing the results, recall that the the MAP ranges from 0 (worst case) to 1 (best case). The MaRR range depends on the number of versions a query has in the collection. For example, if a query has 3 target versions in the collection, the highest possible MaRR will be $(\frac{1}{1} + \frac{1}{2} + \frac{1}{3})/3 = 0.61$. Since the average version-set cardinality in our collection is approximately 4, we can consider this value (0.61) as a rough upper-bound for the MaRR. Both the MAP and MaRR measures are a common choice for assessing the accuracy of version identification systems based on a single information source [36].

Since we use classification to fuse different information sources (dissimilarities based on different descriptors), an alternative evaluation approach is required to evaluate the results obtained using descriptor fusion. Here, the results produced by each classifier are evaluated in terms of classification accuracy (%) using 10-fold cross validation, averaged over 10 runs per classifier. The classification is carried out using the balanced subsets of 10,000 randomly selected query-target pairs mentioned in Section 2.3. We repeat the evaluation process for 5 such subsets (non-overlapping), and average the results over all subsets. Note that we ensure each training subset contains an equal amount of pairs that are versions and pairs that are not. In this way we ensure the subsets are not biased and, therefore, the baseline accuracy (corresponding to making a random guess) is 50%. The statistical significance of the results is assessed using the paired t-test [46] with a significance threshold of $p < 0.001$.

Table 1 Results for single tonal representation (76 songs).

Feature	**MAP**	**MaRR**
Melody	0.732	0.422
Bass line	0.667	0.387
Harmony	0.829	0.458

Table 2 Results for single tonal representation (full collection, 2125 songs).

Feature	**MAP**	**MaRR**
Melody	0.483	0.332
Bass line	0.528	0.355
Harmony	0.698	0.444

3.2 Results

3.2.1 Single Tonal Representation

We start by comparing the results obtained when using a single descriptor, either the melody, the bass line or the harmony. In Table 1 we present the MAP and MaRR results for the 76-song subset which was used for training the parameters of the matching algorithm. At first glance we see that the harmonic representation yields better results compared to the melody and bass line descriptions. Nonetheless, the results also sug-

Table 3 Fusion results for the different classifiers considered.

Feature	Random Forest	SMO (PolyKernel)	Simple Logistic	KStar	Bayes Net
M	69.84	76.73	75.29	77.98	77.90
B	73.34	81.03	78.98	81.31	81.03
H	82.04	87.69	86.42	87.74	87.58
M+B	79.80	82.05	80.91	84.62	84.46
H+M	84.29	87.73	86.51	88.01	87.81
H+B	84.72	87.80	86.77	88.32	88.14
H+M+B	86.15	87.80	86.83	88.46	88.24

gest that the latter two representations do indeed carry useful information for version identification. Evidence for the suitability of melody as a descriptor for version identification has been reported elsewhere [23,36,42]. However, no evidence for the suitability of bass lines has been acknowledged prior to this study. Moreover, to the best of our knowledge, no direct comparison between these three music representations has been performed previously in the literature.

To properly assess the performance of each descriptor, however, a more realistic collection size is required. Thus, we now turn to the results obtained using the full 2125 song collection, presented in Table 2. As expected, there is a drop in performance for all three representations (cf. [36]). The harmonic representation still outperforms the melody and bass line descriptors, for which the drop in performance is more considerable. It is worth noting that the MAP results we obtain using melody or bass line, though lower than those obtained using harmony, are still considerably higher than those obtained by other version identification systems using similar (and different) types of descriptors [36].

As suggested earlier, one probable reason for the superiority of the harmonic representation is that some versions simply do not contain a main melody, and (though less often) some songs do not contain a bass line (e.g. a singer accompanied by a guitar only). Still, as seen in the results for the 76-song subset, even when the melody and bass line are present, the harmonic representation produces better matching results in most cases. This can be attributed to the different degree of modification applied to each tonal representation across versions: whilst some versions may apply reharmonisation, in most cases the harmony remains the least changed out of the three music representations. Differences in the melody and bass line may also be increased due to transcription errors, an additional step which is not necessary for computing the HPCP.

Since the HPCP is computed using the complete audio mix, we know that the melody and bass line may also be, to some degree, represented in the HPCP. Thus, even though the HPCP descriptor is considered to be related to harmony, it is interesting to ask to what degree is the information it encapsulates different from the melody and bass line descriptors. This aspect, albeit very simple, has not been formally assessed before. To answer this question we turn to the second part of the evaluation, in which we examine whether fusing the different representations results in improved matching or not.

3.2.2 Fusion of Music Representations

The classification results for individual descriptors and fusion approaches are presented in Table 3, where we use "M" for melody, "B" for bass line and "H" for harmony (HPCP). Several observations can be made from the results. Firstly, we note that for all descriptors and all classifiers the results are significantly above the baseline of 50%. We see that most classifiers perform relatively similarly, though there are some notable differences. In particular, the random forest classifier provides lower results, whilst k-star consistently provides the highest (the difference between the two is for all cases statistically significant). As before, we note that when using only a single representation, the harmony provides the best performance, followed by the bass line and, finally, the melody.

Perhaps the most interesting results are those obtained by descriptor fusion. For all classifiers, combining the melody and bass line provides increased classification accuracy compared to using either of the two descriptors separately (the increase is statistically significant). Not surprisingly, this confirms that the two music representations carry complementary information and hence their combination results in increased performance. Still, using melody and bass line together does not outperform using the harmony on its own. The remaining question is thus whether combining harmony with other descriptors improves classification accuracy.

The results are less straightforward this time. In the case of the random forest classifier, the improvement is clear and statistically significant. However, for the remainder of classifiers the increase is not as considerable. This suggests that the benefits of considering different music representations are particularly important when the classifier has (relatively) low performance. Nonetheless, if we consider the results of the best performing classifier (k-star), it turns out that the increase in accuracy when combining harmony, melody, and bass line compared to harmony alone is in fact statistically significant. Still, the small increase in accuracy (less than 1%) indicates that our harmonic representation (HPCP), to a great extent, carries overlapping information with the melody and bass line.

3.3 Discussion

To better understand how these different tonal representations can complement each other, we manually examined cases where the melody or bass line descriptors produced better matching results than the HPCP. In Figure 4 we present three dissimilarity matrices of 10 queries compared to 10 targets, where the same 10 songs are used both as the queries and the targets. The three dissimilarity matrices are computed using (a) HPCP, (b) melody, and (c) bass line. The dissimilarities in each matrix are normalised by the greatest value in each matrix so that they are visually comparable. Cells for which the query and target are versions of the same musical piece are marked with a black box.

An example where melody works better than the HPCP can be seen for the version group with IDs 3, 4, and 5. We see that when using the HPCP, song 4 is considered relatively different from songs 3 and 5 (light color), whilst the dissimilarity is much smaller (darker colour) when using the melody. The three songs are different versions of the song “Strangers in the Night” popularized by Frank Sinatra. Listening to the songs we found that whilst versions 3 and 5 have relatively similar orchestral arrangements, version 4 includes several reharmonisations and entire sections where the melody is played without any accompaniment. It is clear that in such a case using the melody on its own will produce smaller dissimilarities between the versions. The bass line descriptor on the other hand does not work well in this example, for the very same reasons.

Another interesting example is provided by the version group with IDs 8, 9 and 10. The three songs are different versions of the song “White Christmas” by Irving Berlin, made famous by Bing Crosby back in 1941. Here we see that whilst song 8 is poorly matched to songs 9 and 10 using either HPCP or melody, it is well matched to song 10 when we use the bass line. When listening to the songs we found that unlike versions 9 and 10, in version 8 there are sections where the melody is solely accompanied by the bass line. In other parts of the song the accompaniment, played by a string section, consists of melodic motifs which interleave with

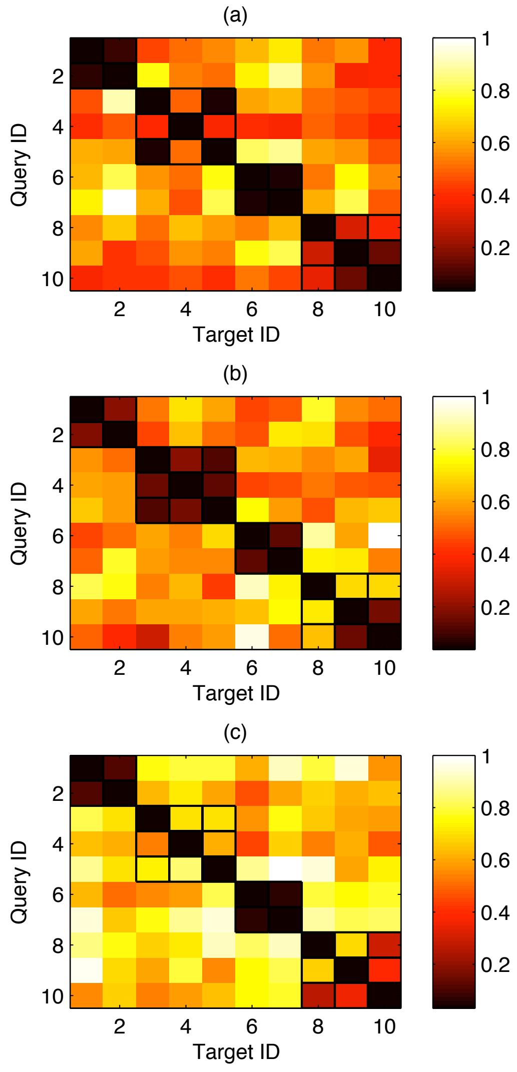

Fig. 4 Dissimilarity matrices for 10 query and 10 target pieces, produced using: (a) HPCP, (b) melody, (c) bass line.

the singing. Furthermore, unlike the more traditional vocal renditions in 9 and 10, the melody in 8 is sung in a more “talk-like” fashion, which combined with the predominant melodic motifs of the string section causes greater confusion in the melody extraction. The various aforementioned differences explain why in this case the bass line succeeds whilst the melody and HPCP do not perform as well. Curiously, whilst song pairs 8-10 and 9-10 are well matched using the bass line, the pair 8-9 is not. Though investigating the exact cause for this inequality is beyond the scope of this study, a possible explanation could be the greater degree of transcription errors in the extracted bass line of song 9. Since the dissimilarity computation is not metric, it is possible for transcription errors to have a greater effect on the matching of some songs compared to others.

The results above show that, while in most cases the HPCP (most closely related to the harmony) is the most reliable music representation for version matching, the melody and bass line can provide useful information in cases where the harmony undergoes considerable changes or is otherwise completely removed (e.g. a cappella singing in unison). Although this observation may seem somewhat obvious, approaches for version matching using descriptor fusion such as [9] and the one proposed in the current study do not take this into account since they always use all descriptors even when one of them may not be appropriate. Thus, a potential approach for improving matching accuracy would be, rather than always using all descriptors, to first attempt to determine which descriptors will provide the most reliable matching results and then use only those. For example, if we detect that one version has accompaniment and the other does not, we might decide to use just the melody rather than the melody, bass line and harmony. Another possibility would be to train a classifier for each representation using a classification algorithm that returns a confidence measure along with its prediction. Then, the prediction of the classifier with the highest confidence could be selected, unless there is no clear winner in which case we could use the prediction obtained by descriptor fusion as described in this study.

Whilst the generality of the matching algorithm employed in this study (Section 2.2) means it can be easily adapted to different types of time series, it is still relevant to ask whether it is the most appropriate matching approach for the melody and bass line sequences. Since the algorithm was originally designed to work with chroma features (HPCPs), it is possible that it introduces a slight bias towards this type of time series. Another conjecture is that the intrinsic lower dimensionality of the melody and bass line features may in part be the cause for the reduced performance of these features. One of our goals for future work will be to address these questions by evaluating and comparing different matching algorithms with the melody and bass line representations proposed in this study.

4 Query-by-Humming

As mentioned earlier, query-by-humming can be considered a special case of the more general task of version identification. We are still interested in matching different renditions of the same musical piece, only this time one of the renditions is monophonic and produced by the user themselves. The QBH method proposed here is an almost direct application of the version identification approach proposed in this study, with very little modification. One important difference is that for this task, we only use the melody-based representation. This is because unlike the version vs. version scenario, in this case queries will only contain melody information (users cannot/will rarely sing the harmony or bass line and will almost always focus on the main melody).

The melody descriptor database is created using the same process described earlier for representing a song for version identification: given a polyphonic recording, we first extract the melody using [32] as outlined in Section 2.1.1, and then perform the representation abstraction described in section 2.1.3. This step only needs to be performed once for every song in the database and, as noted earlier, is fully automatic.

To query for a song, the user records a (relatively) short segment of the melody by either singing or humming into a microphone. The query may contain any part of the melody (i.e. it does not necessarily start at the beginning of the song), in any key, with or without lyrics. The recording must then be converted into the same representation used for complete songs in the database. Conveniently, the polyphonic melody extraction algorithm used for full songs [32] also works very well for monophonic transcription. This is because a query can be viewed as a simplified song where only the melody is present without any accompaniment. The only significant change made to the algorithm is the removal of the voicing detection step, since all detected pitch contours will belong to the melody. Additionally, a simple energy threshold is applied to filter any microphone noise detected during silent segments of the query. Once the query pitch is extracted we apply the same representation abstraction applied to full songs.

Matching is performed as before using the $Q_{\max}$ algorithm [39]. The query representation is compared against every song in the database and songs are then returned in order of increasing dissimilarity (i.e. the song most similar to the query is returned first, then the second closest song, etc.). A block diagram of the proposed approach is presented in Figure 5.

It is important to note that the proposed approach is a proof-of-concept prototype. Most importantly, in a large-scale QBH system it might not be feasible to compare the query against every song in the database in a reasonable amount of time, and some type of indexing technique would be required to speed up the search. Although the dissimilarity measure returned by $Q_{\max}$ is not metric, indexing could be achieved using techniques based on hashing [34] or vantage indexing [40,44]. Alternatively, methods exist for converting non-metric distance measures into metric distances [20], which would allow the use of more standard indexing methods [2].

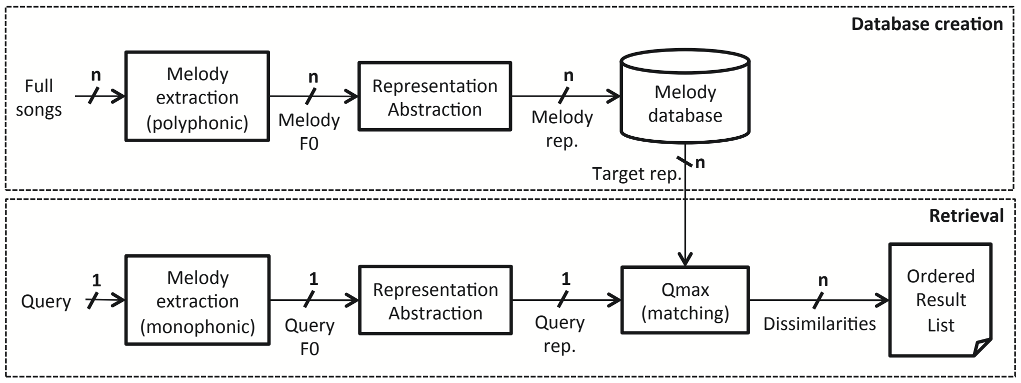

Fig. 5 Block diagram of the proposed query-by-humming system.

4.1 Evaluation Strategy

4.1.1 Music Collections

The proposed QBH approach is evaluated using two collections. The first collection contains only the canonical version of every song from the full 2125 song collection described in Section 3.1.1. Version sets which do not contain the original/canonical version were discarded due to potential confusion of the user when recording a query (33 out of the 523 version sets described in Section 3.1.1). This resulted in a total of 481 songs for this "canonical" collection. Note that for this collection there is only one correct answer for every query.

The second collection is the full set of 2125 songs [38]. Note that this collection presents both an advantage and a disadvantage compared to the canonical collection. On the one hand, it is more than four times larger, and an increase in database size often results in a decrease in retrieval performance [34,36]. On the other hand, the additional songs are (almost) all versions of the songs in the canonical collection, which might increase the probability of successful retrieval since a query that is poorly matched against the canonical version of a song might be well matched against one of its alternative versions. A list of the songs included in the canonical and full collections is provided online[4].

4.1.2 Queries

For the purpose of this evaluation, we recorded a set of sung queries corresponding to songs in the canonical collection described in the previous section. Subjects were presented with a list of all 481 songs in the collection out of which they were asked to select songs and record queries. A total of 118 queries were recorded by 17 subjects (9 female and 8 male). The smallest amount of queries recorded by a subject was 1 and the largest was 11, with a mean of 6.8 queries per subject. The musical experience of the subjects ranged from none at all to amateur musicians. To simulate a realistic scenario all queries were recorded using a basic laptop microphone and no post-processing was applied. Query duration ranged from 11 seconds to 98 seconds and the average query duration was 26.8 seconds. The complete set of 118 queries is available online[4].

One factor that we expected to have a notable influence on the performance of the system was the self-tuning of the sung queries. By this we refer not to the difference in key between the query and the target song, but to whether the singer maintains the same reference tuning throughout the query. If they do not, the same (theoretical) note might be represented by very different frequencies within a single query, thus dramatically changing the contour of the melody. Such detuning was observed for roughly one third of the subjects, where the reference tuning was abruptly changed several times during a single query. To observe the effect of this de-tuning on performance, we manually divided the subjects into two groups: "Good Tuning" and "Bad Tuning". It is important to note that this division was only based on the tuning of the singer with respect to themselves, not on the resemblance of the sung query to the original melody nor on any other singing quality criterion.

4.1.3 Evaluation Measures

Two evaluation measures are used to asses the performance of the proposed approach. The first is the mean reciprocal rank (MRR) mentioned previously, which is the standard measure for evaluating QBH algorithms [6]. Recall that this measure is different from the MaRR used to evaluate version identification in the previous sections – for version identification we were interested in recovering all versions of the query, and hence for every query we computed the reciprocal ranks of all

[4] `http://mtg.upf.edu/download/datasets/MTG-QBH`

Table 4 QBH results for the canonical collection (481 songs).

	MRR	Top-X hit rate (%)			
Singers		1	3	5	10
Good Tuning	0.56	50.00	60.00	61.25	63.75
Bad Tuning	0.23	21.05	21.05	23.68	26.32
All	0.45	40.68	47.46	49.15	51.69

Table 5 QBH results for the full collection (2125 songs).

	MRR	Top-X hit rate (%)			
Singers		1	3	5	10
Good Tuning	0.67	61.25	70.00	73.75	78.75
Bad Tuning	0.33	28.95	34.21	34.21	39.47
All	0.56	50.85	58.47	61.02	66.10

target versions and averaged them, and then averaged this value over all queries. For QBH, we are only interested in the rank of the highest correct match. Thus, the MRR (which ranges from 0 in the worst case to 1 in the best case) for N queries is defined as:

$$MRR = \frac{1}{N}\Sigma_{i=1}^{N}\frac{1}{r_i} \tag{1}$$

where r_i is the highest rank obtained by any version of the correct target song for query i.

The second measure is the top-X hit rate, which reports the proportion of queries for which $r_i \leq X$. If the system had an interface which returned X results for every query, the top-X hit rate would describe the percentage of queries for which at least one of the displayed results corresponds to the correct target song.

4.2 Results and Discussion

The retrieval results for the canonical collection are presented in Table 4. The first thing we note is that there is a significant difference in performance between subjects with good tuning and subjects with bad tuning. For the former group, the correct song is ranked first in the results list 50% of the time, whilst for the latter group only 20% of the time. Still, it is worth noting that for all singers results are well above the random baseline (returning a song at random from the database) which would obtain an MRR of 0.01, top-1 hit rate of 0.15% and top-10 hit rate of approximately 2%. It is also interesting to note that the top-1 hit rate for singers with good tuning is not too far below the top-1 hit rate reported for humans attempting to identify queries manually (66%), as reported by Pardo and Birmingham [26]. When comparing the two groups we note that whilst for good tuning increasing the size of the result list increases the chance of finding the correct answer (14% increase between the top-1 and top-10 hit rate), for subjects with poor tuning the increase is a lot smaller (5%).

Another interesting observation is that even for subjects with good tuning, the correct song appears in the top-10 results just 64% of the time, suggesting there is still much room for improvement. Nonetheless, the results are definitely encouraging, obtaining comparable performance to [30] (and outperforming [7]) on a collection of similar size even though only few changes were made to adapt the approach from version identification to QBH. For singers with poor tuning on the other hand it is clear that there is much work to be done. It is interesting to note that despite its significant influence on performance, the issue of query self-tuning has not been addressed in the previous studies mentioned here. In the future we intend to further investigate the influence of query de-tuning on performance, and research techniques for overcoming such de-tuning. Another possibility would be to try and avoid the de-tuning problem altogether by helping subjects maintain a fixed reference tuning (e.g. providing a fixed reference tone during query recording). Finally, it is probably valid to ask whether we should expect a singing-based retrieval system to work for subjects with poor tuning, who might be directly better off trying search methods which do not involve singing such as query-by-example [34] or query-by-tapping [14].

Next, we turn to the results obtained for the full 2125 song collection (Table 5). We see that there is a significant improvement in performance for both subject groups. As proposed earlier, the cause for this improvement (despite the increased database size) is the addition of cover versions to the collection, meaning we increase the possibility of finding a correct match for queries that do not match the canonical version of the song. Another potential cause for this improvement is that using several versions of each song increases the probability of extracting at least one version of the melody with high-accuracy, thus improving retrieval performance for songs where the melody extraction step did not work well for the canonical version.

The improved performance for the full collection is encouraging, with an MRR of 0.67 and top-10 hit rate of almost 80% for subjects with stable reference tuning. It also highlights an important fact – the more versions we have of the same song in the collection, the better the chances of retrieving it will be. This fact is exploited by approaches such as [27] where the database is directly composed of queries. By combining the two approaches, we can obtain a system that does not suffer from the cold start problem (the initial descriptor database is created using the melody extraction algorithm) and whose performance improves the more people use it (by adding successful queries into the database).

5 Conclusion

To date, the use of different music representations for computing version similarity has not received the attention it deserves. In this paper we have taken a necessary step in this research direction, which not only holds the promise of improving identification accuracy, but also improving our understanding of the relationship between different musical cues in the context of music similarity. Three types of descriptors were compared in this study, related to the melody, bass line and harmony. We studied different degrees of abstraction for representing the melody and bass line, and found that abstracting away octave information and quantising pitch information to a semitone level are both necessary steps for obtaining useful descriptors for version identification. The new melody and bass line descriptors were evaluated on a relatively large test collection, and shown to carry useful information for version identification. Combined with the proposed matching algorithm, our melody and bass line descriptors obtain MAP results comparable to (and in some cases higher than) other state-of-the-art version identification systems. Still, it was determined that in most cases the harmony based descriptor gives better matching accuracy. We have also shown that by using a classification approach for descriptor fusion we can improve accuracy, though the increase over using harmony alone is (albeit significant) small.

Finally, we have demonstrated how the proposed version identification method can be adapted for the related task of query-by-humming. A prototype system was presented and evaluated against two collections, one containing only the canonical version of each song and the other containing both the canonical and cover versions of each song. The approach was shown to obtain results comparable to those presented in previous studies, and current limitations were identified for future improvement. It was then shown how performance can be increased significantly by including more than one version of each song in the target database. In the future we intend to investigate the influence of different factors on retrieval performance such as query length and melody extraction accuracy. Also, as with the proposed version identification method, we intend to evaluate the proposed QBH approach using different distance measures and compare the results to those obtained using $Q_{\max}$. Whilst there is still much work to be done in this area, the results presented here serve as a proof-of-concept and will hopefully lead to the future development of fully automated high-performance QBH solutions.

Acknowledgements The authors would like to thank all the people who recorded queries for the query-by-humming experiments. They would also like to thank the reviewers for their valuable feedback and suggestions for improvement.

References

1. Berenzweig, A., Logan, B., Ellis, D.P.W., Whitman, B.: A large scale evaluation of acoustic and subjective music similarity measures. Computer Music Journal **28**(2), 63–76 (2004)
2. Bozkaya, T., Ozsoyoglu, M.: Indexing Large Metric Spaces for Similarity Search Queries. ACM Transactions on Database Systems **24**(3), 361–404 (1999)
3. Bregman, A.: Auditory scene analysis. MIT Press, Cambridge, Massachussetts (1990)
4. Bryan, N.J., Wang, G.: Musical influence network analysis and rank of sampled-based music. In: Proceedings of the 12th International Society for Music Information Retrieval Conference. Miami, Florida (2011)
5. Dahlhaus, C.: Harmony. Grove Music Online. Oxford Music Online (2012). URL `http://www.oxfordmusiconline.com/subscriber/article/grove/music/50818`
6. Dannenberg, R.B., Birmingham, W.P., Pardo, B., Hu, N., Meek, C., Tzanetakis, G.: A comparative evaluation of search techniques for query-by-humming usig the MUSART testbed. Journal of the American Society for Information Science and Technology (2007)
7. Duda, A., Nürnberger, A., Stober, S.: Towards query by singing/humming on audio databases. In: 8th Int. Conf. on Music Info. Retrieval. Vienna, Austria (2007)
8. Flanagan, J.L., Golden, R.M.: Phase vocoder. Bell Systems Technical Journal **45**, 1493–1509 (1966)
9. Foucard, R., Durrieu, J.L., Lagrange, M., Richard, G.: Multimodal similarity between musical streams for cover version detection. In: Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 5514–5517 (2010)
10. Gómez, E.: Tonal description of music audio signals. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona, Spain (2006). URL `http://mtg.upf.edu/node/472`
11. Gómez, E.: Tonal description of polyphonic audio for music content processing. INFORMS Journal on Computing, Special Cluster on Computation in Music **18**(3), 294–304 (2006)
12. Goto, M.: A real-time music-scene-description system: predominant-f0 estimation for detecting melody and bass lines in real-world audio signals. Speech Communication **43**, 311–329 (2004)
13. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explorations Newsletter **11**(1), 10–18 (2009)
14. Hanna, P., Robine, M.: Query by tapping system based on alignment algorithm. In: IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 1881–1884 (2009)
15. Harwood, D.L.: Universals in music: a perspective from cognitive psychology. Ethomusicology **20**(3), 521–533 (1976)
16. Hyer, B.: Tonality. Grove Music Online. Oxford Music Online (2012). URL `http://www.oxfordmusiconline.com/subscriber/article/grove/music/28102`

17. Kantz, H., Schreiber, T.: Nonlinear time series analysis, 2nd edn. Cambridge University Press, Cambridge, UK (2004)
18. Kotsifakos, A., Papapetrou, P., Hollmén, J., Gunopulos, D., Athitsos., V.: A survey of query-by-humming similarity methods. In: Conf. on Pervasive Technologies Related to Assistive Environments (PETRA) (2012)
19. Liem, C.C.S., Hanjalic, A.: Cover song retrieval: a comparative study of system component choices. In: Proc. of the Int. Soc. for Music Information Retrieval Conf. (ISMIR), pp. 573–578 (2009)
20. Liu, D., Hua, K.A.: Transfer non-metric measures into metric for similarity search. In: 17th ACM international conference on Multimedia, MM '09, pp. 693–696. ACM, New York, NY, USA (2009). URL `http://doi.acm.org/10.1145/1631272.1631390`
21. Lynch, M.P., Eilers, R.E., Oller, D.K., Urbano, R.C.: Innateness, experience and music perception. Psychological Science **1**(4), 272–276 (1990)
22. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge, UK (2008)
23. Marolt, M.: A mid-level representation for melody-based retrieval in audio collections. Multimedia, IEEE Transactions on **10**(8), 1617–1625 (2008)
24. Ong, B.S., Gómez, E., Streich, S.: Automatic extraction of musical structure using pitch class distribution features. In: Workshop on Learning the Semantics of Audio Signals (LSAS), pp. 53–65 (2006)
25. Pachet, F.: Knowledge management and musical metadata. In: D. Schwartz (ed.) Encyclopedia of Knowledge Management. Idea Group, Harpenden, UK (2005)
26. Pardo, B., Birmingham, W.: Query by humming: How good can it get? In: Workshop on Music Information Retrieval, pp. 107–109. Toronto, Canada (2003)
27. Pardo, B., Little, D., Jiang, R., Livni, H., Han, J.: The vocalsearch music search engine. In: ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL) (2008)
28. Ratzan, L.: Understanding information systems: what they do and why we need them. American Library Association (2004)
29. Ravuri, S., Ellis, D.P.W.: Cover song detection: From high scores to general classification. In: Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 65–68 (2010)
30. Ryynanen, M., Klapuri, A.: Query by humming of midi and audio using locality sensitive hashing. In: IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pp. 2249–2252 (2008)
31. Salamon, J., Gómez, E.: Melody extraction from polyphonic music: Mirex 2011. In: 5th Music Information Retrieval Evaluation eXchange (MIREX). extended abstract, Miami, USA (2011)
32. Salamon, J., Gómez, E.: Melody extraction from polyphonic music signals using pitch contour characteristics. IEEE Transactions on Audio, Speech, and Language Processing **20**(6), 1759–1770 (2012)
33. Salamon, J., Gómez, E., Bonada, J.: Sinusoid extraction and salience function design for predominant melody estimation. In: Proc. 14th Int. Conf. on Digital Audio Effects (DAFX-11). Paris, France (2011)
34. Salamon, J., Rohrmeier, M.: A quantitative evaluation of a two stage retrieval approach for a melodic query by example system. In: 10th Int. Soc. for Music Info. Retrieval Conf., pp. 255–260. Kobe, Japan (2009)
35. Serrà, J.: Identification of versions of the same musical composition by processing audio descriptions. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona, Spain (2011)
36. Serrà, J., Gómez, E., Herrera, P.: Audio cover song identification and similarity: background, approaches, evaluation, and beyond. In: Z.W. Raś, A.A. Wieczorkowska (eds.) Advances in Music Information Retrieval, *Studies in Computational Intelligence*, vol. 274, chap. 14, pp. 307–332. Springer, Berlin, Germany (2010)
37. Serrà, J., Gómez, E., Herrera, P., Serra, X.: Statistical analysis of chroma features in western music predicts human judgments of tonality. Journal of New Music Research **37**(4), 299–309 (2008)
38. Serrà, J., Kantz, H., Serra, X., Andrzejak, R.G.: Predictability of music descriptor time series and its application to cover song detection. IEEE Trans. on Audio, Speech and Language Processing **20**(2), 514–525 (2012)
39. Serrà, J., Serra, X., Andrzejak, R.G.: Cross recurrence quantification for cover song identification. New Journal of Physics **11**(9), 093,017 (2009)
40. Skalak, M., Han, J., Pardo, B.: Speeding melody search with vantage point trees. In: 9th Int. Soc. for Music Info. Retrieval Conf. Philadelphia, USA (2008)
41. Song, J., Bae, S.Y., Yoon, K.: Mid-level music melody representation of polyphonic audio for query-by-humming system. In: 3rd Int. Conf. on Music Info. Retrieval. Paris, France (2002)
42. Tsai, W.H., Yu, H.M., Wang, H.M.: Using the similarity of main melodies to identify cover versions of popular songs for music document retrieval. Journal of Information Science and Engineering **24**(6), 1669–1687 (2008)
43. Typke, R.: Music retrieval based on melodic similarity. Ph.D. thesis, Utrecht University, Netherlands (2007)
44. Typke, R., Walczak-Typke, A.: A tunneling-vantage indexing method for non-metrics. In: 9th Int. Conf. on Music Info. Retrieval, pp. 683–688. Philadelphia, USA (2008)
45. Vickers, E.: Automatic long-term loudness and dynamics matching. In: Proc. of the Conv. of the Audio Engineering Society (AES) (2001)
46. Wasserman, L.: All of statistics: a concise course in statistical inference. Springer, Berlin, Germany (2003)
47. Witten, I.H., Frank, E.: Data mining: practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, Waltham, USA (2005)