A Mid-Level Representation for Melody-based Retrieval in Audio Collections

Matija Marolt, *Member, IEEE*

***Abstract*— Searching audio collections using high-level musical descriptors is a difficult problem, due to the lack of reliable methods for extracting melody, harmony, rhythm, and other such descriptors from unstructured audio signals. In the paper, we present a novel approach to melody-based retrieval in audio collections. Our approach supports audio, as well as symbolic queries and ranks results according to melodic similarity to the query. We introduce a beat-synchronous melodic representation consisting of salient melodic lines, which are extracted from the analyzed audio signal. We propose the use of a two-dimensional shift-invariant transform to extract shift-invariant melodic fragments from the melodic representation and demonstrate how such fragments can be indexed and stored in a song database. An efficient search algorithm based on locality-sensitive hashing is used to perform retrieval according to similarity of melodic fragments. On the cover song detection task, good results are achieved for audio, as well as for symbolic queries, while fast retrieval performance makes the proposed system suitable for retrieval in large databases.**

***Index Terms*—audio collections, information retrieval, melody, music**

I. Introduction

DIGITAL libraries of music as well as personal song collections are growing at a fast pace and will continue to grow with an increasing number of online retailers selling music in digital formats, such as mp3, and the ongoing digitization of music libraries and archives. Digital home storage units and media players are being put on the market by major consumer electronics manufacturers and are taking centre stage in the home entertainment market. With all this digital content, the demand for efficient methods of search and retrieval in such collections is already here.

To search and organize a collection of items, a similarity measure needs to be defined that quantifies to what extent two items from the collection are alike by associating a numeric value with a pair of items. A higher value indicates greater similarity. Music is multifaceted and its many aspects are also reflected in researches related to music similarity. Timbral similarity or similarity according to musical genre has been extensively explored [1, 2]. Methods employing timbral similarity rely on low-level signal descriptors that are usually combined with temporal features to yield similarity measures that estimate music with similar timbre and tempo as similar. Such similarity measures frequently correspond well with genre labels, so these methods are used for searching and organizing music collections according to genre. Another application area relying on low-level signal descriptors is audio fingerprinting [3]. These systems are used for audio identification tasks, where the goal is not to establish similarities between pieces, but to identify the queried recording in a music collection. Techniques that do not rely on features of musical signals, but use other data sources, such as the web, are also being explored for music and artist similarity measures [4].

Melody is an important descriptor of music [5] and is therefore a very natural descriptor to be used for measuring music similarity. In the symbolic domain, melodic similarity has been extensively studied [6-8], but the work cannot be applied to audio signals directly, while methods for transcribing polyphonic audio signals to symbolic representations are not yet accurate enough [9, 10]. Shwartz et al. [11] presented a probabilistic model for query by melody that uses symbolic melodies to query audio collections. To bridge the semantic gap between low-level and symbolic representations, a number of works recently began to use mid-level representations to search audio collections. The main advantage of mid-level representations is that they reduce the semantic gap by incorporating some higher-level semantic features extracted from music signals, while still avoiding symbols. Most researchers use representations that emphasize harmonic aspects of music signals and therefore yield similarities based on harmonic structure. Ellis and Poliner [12] used cross correlations of beat-synchronous chromagrams to find similar songs. Müller et al. [13] also used chroma-based features, but they compared sequences of statistical features to calculate song similarity. Serrà [14] extracted sequences of harmonic pitch class profiles from audio signals and aligned them with dynamic programming, while Bello [15] performed HMM-based chord estimation on audio signals and used string alignment to compare sequences of chord labels.

In contrast to the above mentioned works, the approach presented in this paper uses melodic similarity to search audio collections. We propose a mid-level melodic representation that consists of salient melodic lines extracted from the

Manuscript received November 13, 2007. This work was supported in part by two Slovenian Government-Founded R&D projects: EthnoMuse: multimedia digital archive of Slovenian folk music and folk dance culture, and EthnoCatalogue: creating semantic descriptions of Slovene folk song and music based on melodic and metro-rhythmic analysis.

Matija Marolt is with University of Ljubljana, Faculty of Computer and Information Science, Trzaska 25, 1000 Ljubljana (phone: +386 1 4768483; fax: +386 1 4264647; e-mail: matija.marolt@ fri.uni-lj.si).

analyzed audio signal. We extend our previous work [16], where we compared melodic patterns obtained from self-similarity matrices, and base our retrieval algorithm on shift-invariant melodic fragments extracted from the mid-level representation. To make retrieval feasible in real-world scenarios, locality-sensitive hashing is employed to index melodic fragments stored in the database. We demonstrate that the mid-level melodic representation makes it possible to use the proposed system for retrieval with audio as well as symbolic queries, which makes it particularly valuable in music collections, where both kinds of query should be supported.

The organization of the paper is as follows. In Section II we introduce a mid-level melodic representation that forms the basis of our approach. In section III we propose the use of a shift-invariant transform for extraction of shift-invariant melodic fragments from the melodic representation and outline a retrieval system based on locality-sensitive hashing of fragments for efficient retrieval. In section IV we present an evaluation of the system's performance and discuss the results. Section V concludes the paper.

II. Mid-level Melodic Representation

Melody is an important descriptor of music, so melody-based searching (query by melody) is a very natural way of interacting with music collections. Melody can be defined as an auditory object that maintains its identity under certain transformations along the dimensions of pitch, tempo, timbre, loudness, spatial location, and reverberant environment; sometimes with changes in rhythm; but rarely with changes in contour [17]. As with other cognitive percepts, it is very difficult to extract melody from real-world signals; for polyphonic audio signals, current state of the art melody extraction algorithms yield accuracy of approx. 73% [18]. We therefore propose a novel mid-level representation that enables melody-based search and retrieval in audio collections by approximating the predominant melody of a piece of music. The representation is based on salient melodic lines extracted from a music signal – it consists of a single melodic line, where one line is predominant, but may contain several parallel melodic lines, where several lines compete for attention. No segmentation and separation of individual voices are performed, so all the found salient melodic lines are included in the representation. As we show in section IV, the proposed representation outperforms a harmonic representation for the cover song detection task. Its additional advantage lies in the fact that by approximating the predominant melody, it also enables the use of symbolic melodies (e.g. MIDI) for querying audio collections.

A. Estimation of Salient Melodic Lines

Estimation of salient melodic lines is based on Klapuri's joint multipitch estimator [19]. His algorithm was modified to extract only the dominant pitches and to form salient melodic lines by: (1) adding a psychoacoustic loudness model to improve the selection of dominant pitch candidates, (2) adding spectral modeling synthesis analysis to reduce effects of noise and track dominant pitches in time, and (3) adapting the algorithm to calculate dominant pitches in a period of several consecutive time frames. The algorithm is summarized in Fig. 1 and is described below with emphasis given to the proposed extensions.

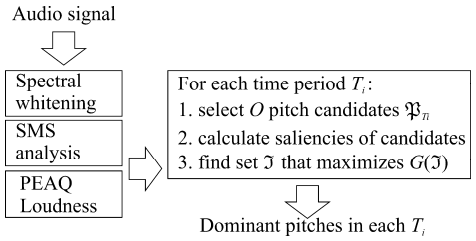

Fig. 1. Overview of the dominant pitch extraction algorithm.

The whitened [19] audio signal is passed through spectral modeling synthesis (SMS) analysis [20] to extract harmonic components from the signal. SMS analysis was added to reduce effects of noises (drums, transients etc.), to provide more accurate partial frequency estimates in each time frame, and to form salient melodic lines from the found dominant pitches. Let $p(t) = \{f(t), a(t)\}$ represent a harmonic component (partial) found by the SMS analysis. The partial is defined by its time-varying frequency $f(t)$ and amplitude $a(t)$. Estimation of dominant pitches is performed in successive time periods $T_i=[t_i,\ t_{i+1})$. Amongst the set of all partials observed within a time period T_i, we choose O loudest partials according to the psychoacoustic loudness model PEAQ [21], to obtain a set of pitch candidates $\mathfrak{P}_{Ti}$. The PEAQ loudness model is based on the well known Zwicker's loudness model. It weigths the power spectrum of each frame by the frequency response of the outer and middle ear and groups the resulting power spectral energies into critical bands, which are used to estimate loudness. Introduction of the loudness model lets us reliably pick a small number of dominant pitch candidates from the set of all partials, thus improving the percentage of relevant melodic lines and reducing the overall number of salient melodic lines found by the algorithm.

The resulting set of O pitch candidates $\mathfrak{P}_{Ti}$ is further enhanced by adding first subharmonics of these partials to $\mathfrak{P}_{Ti}$, as SMS analysis often finds incomplete and fragmented fundamentals at lower frequencies (up to 400 Hz), usually due to interference with partials of bass lines or noises such as drums. For each pitch candidate we compute its salience as:

$$s(p) = \sum_{t \in T_i} \sum_{m=1}^{NP} g(f_p(t), m) a_{pm}(t), \quad p \in \mathfrak{P}_{Ti}\ , \qquad (1)$$

where $f_p(t)$ are frequencies of partial p and $a_{pm}(t)$ magnitudes of the m-th harmonic of partial p. Function $g(f,m)$ defines the spectral shape of a tone with pitch f and is calculated as described by Klapuri [19]. We then evaluate combinations of up to L candidate pitches to find one that maximizes function $G(\mathfrak{I})$ [19]:

$$G(\mathfrak{I}) = |\mathfrak{I}|^{-\gamma}\left(\sum_{p\in\mathfrak{I}} s(p) - \sum_{p\in\mathfrak{I}}\sum_{r\in\mathfrak{I}\setminus p} Inh(p,r)\right),\ |\mathfrak{I}| \le L$$
$$Inh(p,r) = \frac{d}{2}\sum_{t\in T_i}\sum_{(m,n)\in\mathfrak{K}_{pr}} g(f_p(t),m)a_{pm}(t)g(f_r(t),n) \quad (2)$$

where $\mathfrak{I}$ is a set of up to L partials taken from $\mathfrak{P}_{Ti}$, $\mathfrak{K}_{pr}$ a set of tuplets of indices of overlapping harmonics of partials p and r, γ a factor which adjusts preferences towards finding larger or smaller sets of partials, and d a parameter controlling the strength of inhibition defined by function *Inh*. The set of partials $\mathfrak{I}$ that yields the highest value of G represents dominant pitches within the time period T_i. The procedure is repeated for all time periods resulting in a set of dominant pitches for each time period. SMS partial tracking information is used within and between time periods to link the found dominant pitches into salient melodic lines. Lines are defined as triplets $l(t) = \{f(t), a(t), s(t)\}$; next to frequency f and amplitude a, a melodic line is also characterized by the salience s of its pitch as calculated by eq. (1).

We evaluated the method on a set of 33 song excerpts with annotated melody taken from the public data set of MIREX 2004 and 2005 melody extraction tasks [18]. Results are given in Table 1. We estimated how well the found salient melodic lines correspond to the annotated melodic line (ignoring octave errors - column 2: % accurate pitch class) and counted the average number of simultaneous melodic lines found (column 3: average $|\mathfrak{I}|$ per time period). The desired result is to have high coverage of the annotated melody with a low average number of simultaneous melodic lines (polyphony). Results are given for four methods: Klapuri's joint multipitch estimator [19], a melodic line extraction method used in our previous work [16], the described approach using a single time frame for dominant pitch estimation (t_{i+1}-t_i=1), and the described approach, where time periods T_i were estimated by a beat tracker [22] (times t_i correspond to the found beats). The following values of model parameters were used: SMS analysis window frame size of 93 ms, step size of 10 ms, O=4, NP=11, L=3, γ=0.83 and d=0.5.

TABLE 1. COMPARISON OF DOMINANT PITCH ESTIMATION METHODS

method	% accurate pitch class	average $\|\mathfrak{I}\|$
Klapuri	67.8	2.6
previous work	80.6	2.3
single frame	81.4	1.3
beat synchronous	79.7	1.3

The presented approach significantly improves the accuracy of dominant pitch estimation in comparison to Klapuri's original method. We attribute this to two factors: instead of estimating harmonic components of pitch candidates from FFT frames we use partials as calculated by the SMS analysis. SMS analysis reduces effects of various noises (drums, transients etc.), provides more accurate partial frequency estimates, and consequently results in more accurate pitch salience calculation. The loudness model used for picking pitch candidates is the second factor, as reducing the large set of possible pitch candidates to only a few candidates before multipitch extraction not only speeds up the process, but also improves its accuracy. We do not claim, however, that improvements to Klapuri's approach lead to a better multipitch estimator, we only demonstrate that they lead to a better dominant pitch estimator, which is the topic of our interest. Surprisingly, using beat-synchronous time periods did not improve dominant pitch estimation over the frame-by-frame approach. We expected that longer time periods will result in higher saliencies of relevant melodic lines, which are usually louder and also longer in duration. As it turns out, especially in vocal music, vowels that vary a lot in frequency and amplitude are split into several partials, which within a particular time period compete for dominance with the more stable accompaniment partials. The latter are then often picked instead of their perceptually louder counterparts. In our further experiments, we chose to use frame based dominant pitch estimation as the first step in calculating the mid-level melodic representation.

B. Mid-level Melodic Representation

Salient melodic lines form the basis for calculation of the mid-level melodic representation. We first remove some irrelevant melodic lines in order to put more emphasis on the predominant melody: (1) lines with mean salience below a threshold T_1 are removed; (2) lines with mean relative amplitude below a threshold T_2 are removed. Relative amplitude is the ratio of melodic line's amplitude to a global amplitude envelope that is calculated by zero-phase filtering the amplitude envelope of the analyzed music signal using a first order low-pass filter with cutoff frequency of 0.3 Hz. The first step removes melodic lines, where pitch was not estimated reliably, which indicates that lines may have a very unstable pitch or may be masked by other lines. The second step removes a large number of accompaniment lines, which are quieter than the predominant melody, but are usually detected between melodic parts (see Fig. 2 top). Experimentally, threshold values T_1 and T_2 were set to 0.05 and 0.5, respectively.

Events in a piece of music are not perceived in direct relation to time, but in relation to their place within a metric hierarchy, whose basic elements are beats. These in turn relate to time according to tempo and its variation within a piece. Calculating intra- and inter-piece similarities is difficult when tempo varies; dynamic programming approaches can be used to solve this problem [14, 15]. We prefer to make our representation tempo-invariant by using a beat tracker [22] to detect beats and then aligning the representation to the beat grid, thus making it beat-synchronous (Ellis [12] used a similar approach). Before alignment, the beat grid is made denser by doubling the tempo, so that events that are shorter than one beat (typically a quarter note) may be represented. We also enforce octave invariance in the alignment process by discretizing pitches of salient melodic lines into a set of 12 pitch classes representing 12 pitches of the chromatic scale (of twelve-tone equal temperament):

$pc = \lfloor 12\log_2(f/27.5) + 0.5 \rfloor (\mathrm{mod}\, 12)$. The mid-level representation m of a set $\mathfrak{M}$ of salient melodic lines $l(t) = \{pc_l(t), a_l(t), s_l(t)\}$ is calculated as follows. For each melodic line within each time period, we calculate its pitch class according to amplitude and salience of the line:

$$pcm_l(i) = \arg\max_{pc} \sum_{t \in [b_i, b_{i+1})} a_l(t) s_l(t) \delta(pc_l(t) - pc) \quad , \qquad (3)$$

where b_i are beat positions and δ the Dirac delta function. The representation m is then calculated by summing contributions of all melodic lines:

$$m(pc, i) = \sum_{l \in \mathfrak{M}} \delta(pcm_l(i) - pc) \frac{1}{b_{i+1} - b_i} \sum_{t \in [b_i, b_{i+1})} a_l(t) s_l(t) \quad . \qquad (4)$$

The obtained representation is tempo and octave invariant. It contains salient melodic lines that represent the predominant melody and may also include parallel melodic lines, where a single line is not clearly predominant. Lines belonging to different instruments (e.g. voice, solo instrument, bass) are all included in the representation, if they are predominant in any time period, because no segmentation of lines into different voices is performed. This may be observed in Fig. 2, which shows an excerpt taken from the mid-level representation of "Almost like being in love" (the excerpt used is part of the MIREX 2005 melody extraction training set).

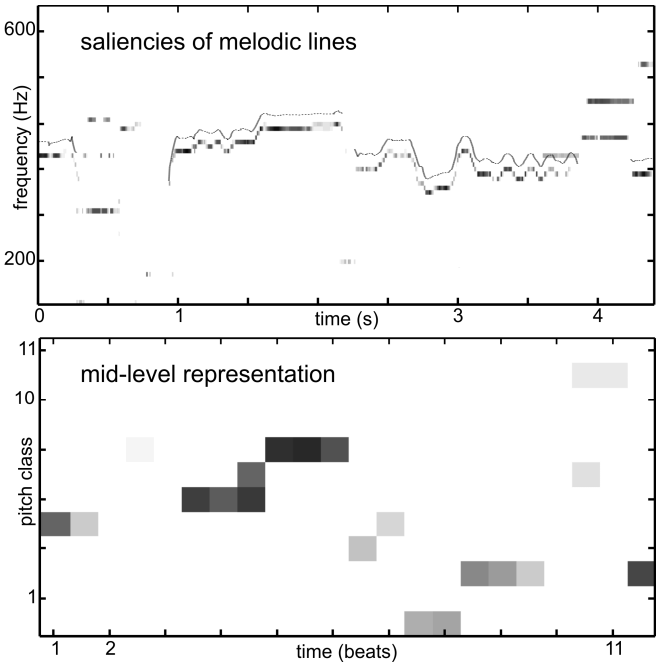

Fig. 2. Saliencies of melodic lines (top) and the mid-level representation (bottom) of an excerpt from "Almost like being in love".

The top part of the figure shows saliencies of the found melodic lines as calculated by the algorithm described in section *A*. The hand-annotated melody, drawn on top of the found lines, demonstrates that all parts of the vocal melody are correctly identified. The mid-level representation (bottom) is beat-synchronous and divided into 12 pitch classes. In addition to melody, it contains other salient melodic lines, which belong to accompaniment bass and piano. Note that some of these irrelevant lines have been removed in the filtering process, as previously described.

III. Retrieval in Audio Collections

With a growing number of music collections, one is often faced with a task of finding songs that are similar to song *X*. As melody is an important music descriptor, we find it natural to describe and search for songs based on similarities of their melodies. If symbolic notation of musical pieces is available, many reliable approaches exist for melody-based querying of such collections. In this section, we demonstrate how the mid-level representation proposed in section II can be applied to the problem of retrieval in collections of audio recordings. We first introduce the concept of melodic fragments that are extracted from the mid-level representation and used as basic units of search and retrieval. We propose to use a shift-invariant transform to transform the fragments into a shift-invariant space. We finally outline how locality-sensitive hashing of shift-invariant melodic fragments and term frequency-inverse document frequency weighting can be combined into an efficient retrieval system that supports audio, as well as symbolic querying of music collections.

A. Locality-Sensitive Hashing of Melodic Fragments

We define $\mathfrak{S}=\{S\}$ to be a set of songs in an audio database and m_s the mid-level representation of song *S*. We partition the mid-level representation of a song into overlapping regions of length *N*: $m_{si} = m_s(0..11, i..i+N-1)$ by shifting a rectangular window of length *N* over the entire representation. We call regions m_{si} melodic fragments. Fragments (Fig. 2 (bottom) shows an example) are stored in the database, indexed, queried and retrieved. They contain short melodies of length *N*, which makes our approach somewhat analogous to *n*-gram algorithms [6] commonly used for querying symbolic music collections, where all note sequences of length *n* are stored and indexed in the music database.

Searching through all melodic fragments stored in a database without proper indexing is not suitable for real-world applications. Like the work of Casey and Slaney [23], our solution uses locality-sensitive hashing to hash melodic fragments. A locality-sensitive hashing scheme is a distribution on a family $\mathfrak{F}$ of hash functions operating on a collection of objects, such that for two objects *x*, *y* $\Pr_{h \in \mathfrak{F}}[h(x) = h(y)] = \mathrm{sim}(x, y)$, where $\mathrm{sim}(x, y) \in [0,1]$ is some similarity function defined on the collection of objects [24]. Such hash functions enable the construction of efficient algorithms for approximate nearest neighbor search and clustering, because the similarity of objects can be estimated from their hash values. For hashing melodic fragments, we use random hyperplane-based hash functions, which are designed to approximate cosine similarity of hashed vectors. A random hyperplane is drawn from a multidimensional Gaussian distribution (each coordinate is drawn from a 1-dimensional Gaussian distribution) and used to hash input vectors. Given a melodic fragment m_{si} and a hyperplane defined by *p*, we

calculate a hash function h_p as:

$$h_p(m_{si}) = \mathrm{sgn}(m_{si} \cdot p) \ , \qquad (5)$$

where sgn is the sign function taking values 1, if its argument is larger or equal to 0, and 0 otherwise. Each possible choice of p defines a single function. It can be proved that, for two fragments x and y: $Pr\left[h(x) = h(y)\right] \approx \cos\left(\theta(x,y)\right)$, where $\theta(x,y)$ is the angle between x and y. We can thus estimate cosine similarity, which is a commonly used measure of music similarity [25], by comparing hash values of two melodic fragments. To calculate the similarity, we choose a set of B random hyperplanes to obtain a B-bit hash value representing each melodic fragment. Cosine similarity of two fragments can be estimated by calculating the normalized Hamming similarity between hash values of two vectors [24]:

$$\mathrm{sim}(m,n) = \cos(\theta(m,n)) \approx 1 - \frac{1}{B}\sum_{p=1}^{B}\left|h_p(m) - h_p(n)\right| \qquad (6)$$

B. Transposition Invariance

When calculating melodic similarity of two musical pieces, one must take into consideration that songs may be recorded in different musical keys. These transposed versions are melodically equivalent to most listeners, as pitches are perceived relative to each other rather than in absolute categories. In symbolic databases, transposition invariance is usually achieved by using melodic contours rather than notes to represent melodies. This is not easily achieved in our case. A possible solution is to store 12 (for 12 pitch classes) transpositions of each melodic fragment in the database or to query the database with 12 transpositions of the query. The former results in a large number of fragments in the database, while the latter (used by most researchers) increases query times. As an alternative, we propose the use of a family of shift invariant transforms widely used in image processing. Examples of such transforms include the power spectrum (Fourier transform of the autocorrelation), Fourier-Mellin transform, R-transform, and shift-invariant operators derived from higher-order spectra.

The 2D power spectrum of a melodic fragment is a (circular) shift invariant representation in pitch class, as well as time. Fig. 3 demonstrates four shifted melodic fragments, which all yield identical 2D power spectra. While shift invariance is desirable with key transpositions (Fig. 3B) and small time shifts (Fig. 3C), larger time shifts may lead to fragments, which are musically not similar (Fig. 3D). With this in mind, we have also explored shift-invariant operators derived from higher order spectra [26], which may be designed to be invariant to key transpositions only. These operators are derived from two-dimensional complex moment invariants based on the observation that there is a duality between rotation and shift invariance, and can be calculated efficiently with the discrete Fourier transform. As results were not improved, they are not discussed in this paper.

In section IV, we show that using the power spectrum as the shift-invariant (and therefore key-invariant) representation of melodic fragments, results in significant improvements in retrieval accuracy, as well as speed, when compared to the approach of querying in all transpositions.

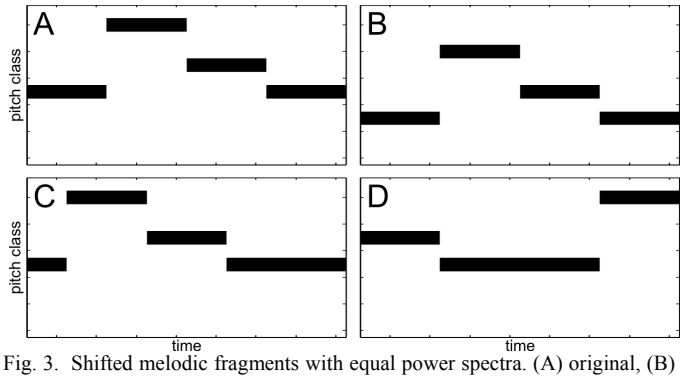

Fig. 3. Shifted melodic fragments with equal power spectra. (A) original, (B) transposed by two semitones, (C) time shifted by 1 beat, and (D) time shifted by 8 beats resulting in a musically different fragment.

C. Beat Tracking Errors and Static Melodic Fragments

The mid-level representation relies on a beat tracking algorithm to attain tempo invariance. Beat tracking is prone to errors, especially the so-called tempo octave errors, where tempo is estimated to be twice or half as fast as the real tempo. Tempo octave errors have a large influence on our system's performance, as they compress or expand melodic fragments, resulting in retrieval mismatches. To reduce the effect of these errors, we include two versions of each song in the database: one with tempo as calculated by the beat tracker and another with double or half this tempo, whichever has larger salience according to the pulse period salience curve, as defined by Parncutt [27]. This curve is derived from tapping data and represents preferred tempi that listeners in his experiments assigned to rhythms played at different tempi. By using the pulse period salience curve, we try to guess the most likely alternative tempo of a song in case the beat tracker made an octave error and add the mid-level representation of the song in this alternative tempo to the database.

Melodic fragments that contain a single dominant long note (often encountered in accompaniment lines and intros) contain too little information to be treated as musically similar to other fragments. They are a frequent cause of retrieval errors, as they match well across different songs. To reduce effects of such static melodic fragments on retrieval accuracy, we exclude them from the database. The criterion for exclusion is based on the kurtosis of each fragment's pitch profile pp:

$$pp(pc) = \frac{1}{N}\sum_{b=1}^{N} m(pc,b) \qquad (7)$$

Kurtosis is a measure of peakedness of a probability distribution and results in high values for fragments, which have a highly peak-like pitch profile; in our case these are patterns with one dominant pitch. Fragments with kurtosis exceeding a threshold T_k are excluded from the database. Experimentally, the threshold value was set to 5.

D. Retrieval

Given a query song or song excerpt q, we calculate its mid-level representation m_q and melodic fragments m_{qi}. For each

melodic fragment, its *B*-bit hash value is calculated according to eq. (5). Normalized Hamming similarities of this hash value to values stored in the database are measures of similarity of the queried fragment to fragments in the database. Finding fragments similar to a given query fragment is therefore reduced to finding nearest neighbors of its hash value in the database. To perform this nearest neighbor search efficiently, we use the algorithm proposed by Charikar [24], which is based on random permutations of hash values. Given a database of hash values consisting of *B* bits each, we choose *P* random permutations of the bits. We calculate a sorted order of permuted hash values for each random permutation,. Given a query hash value h_q, we find its approximate nearest neighbors by performing a binary search to locate hash values closest to h_q for each permutation. We now search in each of the sorted orders examining *M* elements above and below the position returned by the binary search to find all vectors with Hamming similarity to h_q larger than a threshold T_h.

For each queried melodic fragment m_{qi}, the approximate nearest neighbor algorithm described above returns up to *M* approximate nearest neighbors $m_{nn} \in NN\left(m_{qi}\right)$ from the database. We can estimate the similarity of the query *q* to a song *S* in the database in several ways. The most straightforward is the Sum Common approach, where we sum similarities of all fragments common to both the query and the song to obtain a cumulative similarity measure:

$$\mathrm{sim}(q,S) = \sum_{m_{qi} \in m_q} \sum_{m_{sj} \in NN(m_{qi}) \cap m_s} \mathrm{sim}(m_{qi}, m_{sj}) \ . \qquad (8)$$

Another approach is to use tf-idf (term frequency-inverse document frequency) weighting on cosine similarity estimates. The cumulative similarity measure of songs *q* and *s* can thus be written as:

$$\mathrm{sim}(q,S) = \sum_{m_{qi} \in m_q} \sum_{m_{sj} \in NN(m_{qi}) \cap m_s} \mathrm{sim}(m_{qi}, m_{sj})\, tfidf(m_{sj}) , \qquad (9)$$

where *tfidf* is the weighting function, which represents the importance of each melodic fragment in the database. The importance of a fragment increases as the frequency of the fragment increases in a song (i.e. refrain parts are more important than intros or solos), but the degree of importance diminishes depending on how common the fragment is in all songs in the database (fragments that appear often in different songs are given lower importance). A high tf–idf weight is reached by high fragment frequency (in the given song) and low overall frequency of the fragment in the whole song collection. We calculate the weight as:

$$idf(m_{sj}) = \log \frac{\left|\mathfrak{S}\right|}{\left|\left\{t \in \mathfrak{S} \mid \exists m_{ti} \in m_t, \mathrm{sim}(m_{ti}, m_{sj}) > T_h\right\}\right|} \qquad (10)$$

$$tfidf(m_{sj}) = \frac{\left|\left\{m_{si} \in m_s \mid \mathrm{sim}(m_{si}, m_{sj}) > T_h\right\}\right|}{\left|\left\{m_{si} \in m_s\right\}\right|} idf(m_{sj})$$

A query thus results in a set of up to *N*M* approximate nearest neighbors of the queried song, ranked according to cumulative similarity defined by eqs. or . *N* is the number of melodic fragments of the query, whereas *M* is the maximum number of nearest neighbors returned by each query.

IV. Experiment and Discussion

We evaluated the performance of our system on the task of identifying cover songs in an audio collection. Finding cover songs in audio collections can be considered a special case of the more general task of finding songs with similar melody. Cover songs are different interpretations of the same song by the same or by different artists. The common denominator of all song covers is lyrics (if present) and melody of at least part of the song, while song structure, instrumentation and rhythm may differ. To evaluate our retrieval system, we set up a database of 2424 songs, mostly of rock and pop genres, which included cover versions of 34 songs (two to sixteen covers per song, totaling 147 covers; a list of songs can be found at http://lgm.fri.uni-lj.si/matic/coversongs). We evaluated retrieval accuracy by querying the database with each of the 147 songs and calculating average precision, which is a commonly used evaluation measure in multimedia retrieval [28], as well as counting the number of relevant cover songs returned in the top 10 hit list. Both evaluation measures were also used in the 2007 MIREX algorithm evaluation for the audio cover song identification task. We calculate average precision for a query as:

$$AP = \frac{1}{R} \sum_{j=1}^{H} \frac{rel(j)}{j} \sum_{k=1}^{j} rel(k) \ , \qquad (11)$$

where *H* is the number of hits returned, *R* the number of relevant hits returned and *rel*(*k*) a binary function returning 1, if the *k*-th entry in the hit list is relevant. Relevant cover songs that are not retrieved in the returned list of hits are also included in the calculation of average precision; their rank is set to $|\mathfrak{S}|/2$. We evaluated the performance of our system when changing its key parameters, as well as when queried with symbolic representations of monophonic melodies of 8 cover songs.

A. Fragment Size and Hashing

Length of melodic fragments (*N*) and the number of bits used for fragment hashing (*B*) are two key parameters of our retrieval system. We evaluated how the system's performance changes under different values of the two parameters, and the Hamming threshold T_h. The values of other parameters were: the number of permutations used for nearest neighbor search *P*=8, and the maximum number of nearest neighbors returned *M*=15. Due to the randomness of the hashing process and nearest neighbor search, retrieval was repeated ten times for each set of parameters. Friedman's test for significant differences on mean average precision scores of all queries was used to estimate, which parameter sets were significantly different. Results are shown in Fig. 4.

The optimal length of melodic fragments is 32 beats. This corresponds to 8 bars in 4/4 time signature, which in turn corresponds to the typical length of a phrase in verse and refrain sections in popular music [29]. With longer or shorter fragment sizes, retrieval accuracy starts to decrease. We attribute such behavior to the fact that cover songs consist of

similar phrases, but may differ in overall structure. When structure differs, fragments become more and more dissimilar as the fragment size grows beyond the phrase size, which causes the loss in retrieval performance. Smaller fragment sizes, on the other hand, are too general and match too many songs, so performance suffers as well.

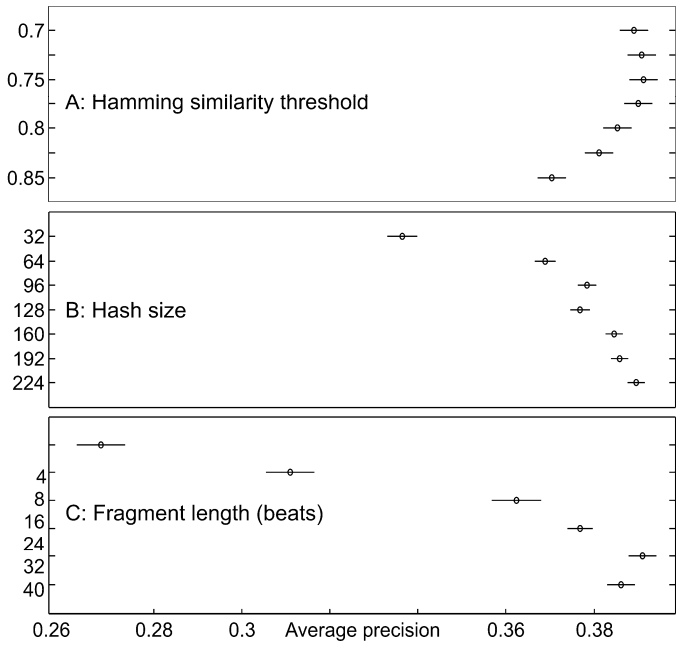

Fig. 4. Performance under varying parameter values: (A) Hamming threshold T_h, (B) number of bits used for fragment hashing B, (C) length of melodic fragments N.

As expected, larger hash sizes improve similarity estimates and thus the accuracy of nearest neighbor search and retrieval. We chose 160 as our optimal hash size; it provides reasonable retrieval speed and contained storage size, while results do not improve significantly with larger hash sizes. We have also varied the Hamming similarity threshold T_h, which turned out to have only a small influence on the overall performance. We chose 0.75 as the value for our experiments.

B. Choice of Representation and Transposition Invariance

To assess how the proposed mid-level representation and transposition invariance techniques affect the accuracy of retrieval, four variants of our system were tested. Results are given in Table 2.

In (a), beat-synchronous chromagram was used instead of the melodic representation in (b,c,d). In (a,b), querying in all 12 transpositions was used to achieve key-invariance instead of the power spectrum used in (c,d). In (a,b,c), the sum commons approach was used to calculate the cumulative similarity, while tf-idf weighting was applied in (d). We also provide results of Ellis' system [12] on the same database (e). Table 2 shows mean average precision scores and the number of songs found in the top 10 ranked hits for the compared systems on all 147 queries, as well as the average speed of a single query on a 2GHz Pentium M machine.

TABLE 2. COMPARISON OF VARIANTS OF THE PROPOSED ALGORITHM ON THE COVER SONG DETECTION TASK

System	mean avg. precision	top 10 hits	Retrieval speed (s)
(a) Chroma with all transpositions	0.32	92	9.3
(b) Melodic with all transpositions	0.36	108	9.3
(c) Melodic with power spectrum	0.39	132	1.8
(d) Melodic with pow. sp. and tf-idf	0.40	142	1.8
(e) Ellis	0.36	135	660

The use of the melodic representation (b,c,d) yields better performance when compared to the chromagram representation. The difference is significant, but not overwhelming. We attribute the difference to the fact that chromagrams of short song fragments are not as distinctive as short melodic fragments. Accuracy might increase, if we use longer fragments, as is shown by good results of Ellis' system that compares chromagrams of entire songs; but then accuracy is lost, when song structure differs. Using the power spectrum (c,d) instead of querying in all transpositions (a,b) results in better performance, as well as shorter execution time. Although the power spectrum is invariant in key as well as time (see Fig. 3), this does not seem to affect the performance, as fragment lengths are large enough, so that all time-shifted fragments are musically still similar. In fact, it is the invariance in time that increases retrieval accuracy over the "all transpositions" approach, as different transitions between song segments and timing deviations are tolerated to a higher degree (see Fig. 3C). Tf-idf weighting provides an additional (although small) gain in performance that can be attributed to reduced mismatches caused by patterns which are uncommon within songs (e.g. intros, solos), and patterns, which are very common throughout the collection.

Looking at the performance on individual songs (Table 3 shows mean average precision and number of top 10 hits for three songs), the comparison to Ellis' system shows some interesting variations.

TABLE 3. COMPARISON OF PER-SONG RESULTS

Song	our approach		Ellis	
	mean AP	top 10	mean AP	top 10
A Hard Day's Night	0.70	8	0.78	12
All by Myself	1	12	0.53	5
Over the Rainbow	0.15	5	0.13	2

When the predominant melodic line is not conspicuous enough to be estimated reliably, such as on a baroque performance of The Beatles' A Hard Day's Night, the mid-level melodic representation is not accurate enough to find relevant hits. In such cases, better results are achieved by using a harmonic representation and longer song segments. On the other hand, for songs with solos of different lengths or otherwise varying song structure, such as with All by Myself (AA'BA'BBCAB'BBB for Celine Dion's version vs. AA'BBAA''BCABB for Eric Carmen's version), our system performs better, as melodic fragments are short enough not to be corrupted by the changing structure. For songs where interpretations vary a lot in melody, as well as in rhythm (e.g.

interpretations of Over the Rainbow), both systems struggled, either due to poor beat tracking or because of the high variance of underlying representations. In general, it is hard to estimate how errors in beat tracking affect our results, as we are lacking ground truth for all the data and cannot make a proper evaluation.

The use of locality-sensitive hashing and approximate nearest neighbor search reduces computational complexity of retrieval to matrix multiplication and binary search. Retrieval of approximate nearest neighbors of a melodic fragment is in the order of $O\left(NB + P\log\left(|m||\mathfrak{S}|\right)\right)$, where N is the fragment size, B the hash size, P the number of permutations in nearest neighbor search, $|m|$ the average number of fragments in a song, and $|\mathfrak{S}|$ the number of songs in the database. N, B and P are parameters of the system and can be considered constant, so retrieval time increases only logarithmically with the size of the database: $O\left(\log\left(|\mathfrak{S}|\right)\right)$, assuming that the average number of fragments per song (song length) remains constant. This makes the approach suitable for real-world applications on large databases. Results also show that the overhead of calculating the power spectrum is small when compared to repeating the query for all transpositions, so the key invariant representation not only improves accuracy, but also speeds up searching.

A direct comparison of our results to results of different algorithms at MIREX 2007 algorithm evaluation [18] is difficult, because neither implementations of the algorithms nor the MIREX evaluation dataset are available. Overall, mean average precision of our system is comparable to the best MIREX systems with the notable exception of Serra and Gomez' approach [14] , which outperformed others by a clear margin (its mean average precision was 0.521). Although not as accurate, the proposed approach has two important advantages: (1) the melodic representation makes it possible to perform retrieval with audio and symbolic queries (as shown in sections IV B and C), and (2) locality-sensitive hashing makes the approach applicable to large real-world databases.

C. Symbolic Queries

An advantage of the proposed mid-level melodic representation is that it also supports symbolic queries (e.g. scores, MIDI files or melodies extracted from query-by-humming front ends). If we convert the symbolic representation of a song's melody into pitch class piano roll representation, we can use the retrieval algorithm described in section III to query a database of audio recordings. We define the symbolic representation of a song as a set of note triplets $n=(nn,b_s,b_e)$, where nn is the MIDI note number, b_s the starting and b_e the ending beat of a note. All MIDI notes are considered to be equally loud. The decay of loudness is emulated by a simple exponential note decay model (the decay factor in eq. (12) was estimated experimentally). We calculate the piano-roll representation m_n that can be used for querying an audio database (as described in section III.D), as:

$$m_n(pc,i) = \begin{cases} e^{-0.4(i-b_s)}, \exists(nn,b_s,b_e) \mid pc = nn \,(\mathrm{mod}\, 12), b_s \le i \le b_e \\ 0, \text{otherwise} \end{cases}, \quad (12)$$

where pc represents pitch class and i represents beat number.

We collected symbolic representations of eight melodies of cover songs in our database and used them to estimate retrieval accuracy with symbolic queries (MIDI files of these queries can be found at the aforementioned URL).

TABLE 4. MEAN AVERAGE PRECISION OF SYMBOLIC VS. AUDIO QUERIES

Song	mean AP audio q.	AP symbolic q.
A Hard Day's Night	0.70	1
All by Myself	1	0,26
Love is in the Air	0.94	0.80

Table 4 shows average precisions for three songs. On average, performance with symbolic queries is comparable to audio queries. For some cases, such as A Hard Day's Night, performance is even improved. In this case, for a baroque song version, the predominant melodic line is very difficult to approximate, because the piece contains several parallel melodic lines. A symbolic query with a clearly defined melody therefore results in better retrieval performance. When performance suffers, it is mostly due to quantization of symbolic events and lack of dynamics (all notes are equally loud). Overall, songs with faster tempi and more rigid pop/dance interpretations are detected with high accuracy (e.g. A Hard Day's Night or Love is in the Air), while looser interpretations that may be successfully retrieved with audio queries because of similar phrasing (e.g. All by Myself), may not be detected with symbolic queries due to their rigidity.

V. CONCLUSION

We presented a novel approach to melody-based retrieval in audio collections. We first introduced a novel melodic representation of an audio signal that approximates the predominant melody by salient melodic lines extracted from the audio signal. Using the power spectrum, we introduced shift-invariance to melodic patterns obtained from this representation. Locality-sensitive hashing is used for indexing and retrieval of shift-invariant melodic fragments, while the tf-idf weighting is employed to find more relevant hits. An important contribution of the presented approach is that it supports retrieval with audio as well as symbolic queries, which makes it particularly valuable in music collections, where both kinds of query should be supported. Retrieval algorithm is fast and scales well to large databases due to its logarithmic computational complexity, while its accuracy is comparable to other state of the art systems. Further work will be dedicated to improvements in melody detection, especially reduction of irrelevant melodic lines. We plan to add harmonic information to our representation to emphasize the context, in which the melody is placed, as this should improve retrieval of fragments, where melody is too static or too difficult to

extract. Other distance measures such as the earth mover distance, which may be embedded within a locality-sensitive hashing framework, will also be explored.

REFERENCES

[1] A. Berenzweig, B. Logan, D. P. W. Ellis *et al.*, "A large-scale evaluation of acoustic and subjective music-similarity measures," *Computer Music Journal,* vol. 28, no. 2, pp. 63-76, Sum, 2004.

[2] E. Pampalk, S. Dixon, and G. Widmer, "Exploring music collections by browsing different views," *Computer Music Journal,* vol. 28, no. 2, pp. 49-62, Sum, 2004.

[3] J. Haitsma, and T. Kalker, "A highly robust audio fingerprinting system with an efficient search strategy," *Journal of New Music Research,* vol. 32, no. 2, pp. 211-221, Jun, 2003.

[4] W. W. Cohen, and W. Fan, "Web-collaborative filtering: recommending music by crawling the Web," *Computer Networks-the International Journal of Computer and Telecommunications Networking,* vol. 33, no. 1-6, Jun, 2000.

[5] E. Selfridge-Field, "Conceptual and Representational Issues in Melodic Comparison," *Melodic Similarity: Concepts, Procedures, and Applications*, MA: MIT Press, 1998.

[6] J. S. Downie, "Evaluating a simple approach to music information retrieval: Conceiving melodic n-grams as text," Faculty of Graduate Studies, University of Western Ontario, London, Ontario, Canada, 1999.

[7] A. L. Uitdenbogerd, and J. Zobel, "An architecture for effective music information retrieval," *Journal of the American Society for Information Science and Technology,* vol. 55, no. 12, pp. 1053-1057, Oct, 2004.

[8] R. Typke, "Music Retrieval based on Melodic Similarity," Utrecht University, Utrecht, 2007.

[9] A. Klapuri and M. Davy (eds.), "Signal Processing Methods for Music Transcription," New York: Springer, 2006.

[10] G. E. Poliner, D. P. W. Ellis, A. F. Ehmann *et al.*, "Melody transcription from music audio: Approaches and evaluation," *IEEE Transactions on Audio Speech and Language Processing,* vol. 15, no. 4, pp. 1247-1256, May, 2007.

[11] S. Shalev-Shwartz, S. Dubnov, N. Friedman *et al.*, "Robust temporal and spectral modeling for query by melody," in 25th annual international ACM SIGIR conference on Research and development in information retrieval, Tampere, Finland, 2002, pp. 331 - 338.

[12] D. P. W. Ellis, and G. E. Poliner, "Identifying Cover Songs With Chroma Features and Dynamic Programming Beat Tracking," in IEEE International Conference on Acoustics, Speech and Signal Processing, Hawai'i, 2007, pp. IV-1429-IV-1432.

[13] M. Müller, F. Kurth, and M. Clausen, "Audio Matching via Chroma-based Statistical Features," in ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, 2005, pp. 288-295.

[14] J. Serrà, "Music similarity based on sequences of descriptors: tonal features applied to audio cover song identification," Department of Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain, 2007.

[15] J. P. Bello, "Audio-Based Cover Song Retrieval Using Approximate Chord Sequences: Testing Shifts, Gaps, Swaps and Beats," in ISMIR 2007, 8th International Conference on Music Information Retrieval, Vienna, Austria, 2007, pp. 239-244.

[16] M. Marolt, "A Mid-level Melody-based Representation for Calculating Audio Similarity," in ISMIR 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 2006, pp. 280-285.

[17] D. J. Levitin, "Memory for Musical Attributes," *Music, Cognition and Computerized Sound : An Introduction to Psychoacoustics*, P. R. Cook, ed.: The MIT Press, 1999.

[18] Wiki. "Music Information Retrieval Evaluation eXchange (MIREX) 2004-2007," http://www.music-ir.org/mirexwiki/index.php/Main_Page.

[19] A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," in ISMIR, 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 2006.

[20] X. Serra, and J. Smith, "Spectral modeling synthesis - a sound analysis system based on a deterministic plus stochastic decomposition," *Computer Music Journal,* vol. 14, no. 4, pp. 12-24, Win, 1990.

[21] J. Timoney, T. Lysaght, M. Schoenwiesner *et al.*, "Implementing Loudness Models in Matlab," in DAFx'04, Naples, Italy, 2004, pp. 177-180.

[22] S. Dixon, "Automatic extraction of tempo and beat from expressive performances," *Journal of New Music Research,* vol. 30, no. 1, pp. 39-58, Mar, 2001.

[23] M. Casey, and M. Slaney, "Song Intersection by Approximate Nearest Neighbor Search," in ISMIR, 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 2006.

[24] M. S. Charikar, "Similarity Estimation Techniques from Rounding Algorithms," in ACM Symposium on Theory of Computing, 2002.

[25] M. Cooper, and J. Foote, "Automatic Music Summarization via Similarity Analysis," in ISMIR, 2002, 3th International Conference on Music Information Retrieval, Paris, France, 2002.

[26] J. Heikkila, "A new class of shift-invariant operators," *Ieee Signal Processing Letters,* vol. 11, no. 6, pp. 545-548, Jun, 2004.

[27] R. Parncutt, "A perceptual model of pulse salience and metrical accents in musical rhythms," *Music Perception,* vol. 11, pp. 55, 1994.

[28] J. Serrà, "A qualitative assessment of measures for the evaluation of a cover song identification system," in ISMIR 2007, 8th International Conference on Music Information Retrieval, Vienna, Austria, 2007.

[29] R. Middleton, *Studying Popular Music*, 1. edition ed.: Open University Press, 1990.

Matija Marolt (M'96) received his B.S. and Ph.D. degrees, both in computer science, from University of Ljubljana, Faculty of Computer and Information Science, Slovenia in 1995 and 2002 respectively.

He is an assistant professor at Faculty of Computer and Information Science, University of Ljubljana, where he has been working since 1995, first as young researcher and later as assistant. He is a member of Laboratory of Computer Graphics and Multimedia at the Faculty, where his research interests include music information retrieval, specifically extraction of semantic descriptions from audio signals and retrieval in audio collections, as well as organization of ethnomusicological archives and audio-visual interaction.