AN EFFICIENT SIGNAL-MATCHING APPROACH TO MELODY INDEXING AND SEARCH USING CONTINUOUS PITCH CONTOURS AND WAVELETS

Woojay Jeon, Changxue Ma, and Yan Ming Cheng
Applied Research and Technology Center
Motorola, Inc.
Schaumburg, IL, U.S.A.
{woojay, Changxue.Ma, fyc002}@motorola.com

ABSTRACT

We describe a method of indexing and efficiently searching music melodies based on their continuous dominant fundamental frequency (f0) contours without obtaining note-level transcriptions. Each f0 contour is encoded by a redundant set of wavelet coefficients that represent its shape in level-normalized form at various locations and time scales. This allows a query melody to be exhaustively compared with variable-length portions of a target melody at arbitrary locations while accounting for differences in key and tempo. The method is applied in a Query-by-Humming (QBH) system where users may search a database of recorded pop songs by humming or singing an arbitrary part of the melody of an intended song. The system has fast retrieval times because the wavelet coefficients can be effectively indexed in a binary tree and a vector distance measure instead of dynamic programming is used for comparisons. Using automatic pitch extraction to obtain all f0 contours from acoustic data, the method demonstrates practical performance in an experiment with an existing monophonic data set and in a preliminary experiment with real-world polyphonic music.

1. INTRODUCTION

It has been suggested in the past that using "continuous" (or frame-based) pitch contours may result in more robust matches of music melodies [1] compared to using symbolic string representations (usually note transcriptions). Both methods require reliable extraction of the dominant pitch contour from both query and target for matches to be successful, but the latter approach requires an extra transcription stage of converting the continuous contours to symbolic strings, which can exacerbate the effect of pitch tracking errors because it makes hard decisions on note boundaries and quantization levels. However, the former approach also has the major drawback of high computational complexity, especially when applying string matching techniques to handle differences in tempo and key as well as the well-known insertion, deletion, and substitution errors. Piecewise approximations of the contours have been used for greater efficiency [2], but this still requires query and target melodies to have roughly similar tempi.

Another problem in melody search is the length and location of queries within their target songs. Query-by-Humming(QBH) applications often limit queries to specific music phrases or hooks, hence simplifying the search space, but in other melody search scenarios, the query may be a completely random portion of a song, e.g. a briefly audible segment of a tune in a TV commercial that the viewer wishes to identify.

In this study, we present a method that tries to address both issues – the computational complexity when using continuous pitch contours *and* allowing the search of partial melodies at arbitrary locations – by using redundant wavelet transformations to index and match pitch contours. The method avoids edit-distance comparisons and instead uses distance measures between fixed-dimension vectors while explicitly resolving tempo and key differences from the very beginning of the search process. This is done by dividing target melodies into overlapping, level-normalized segments over a range of lengths and using wavelets to efficiently represent the segments and match them with queries. The wavelet coefficients are stored in vectors that are in turn indexed in a binary K-D Tree [3] for fast search. Although rhythmic inconsistencies within queries are ignored for computational efficiency, the results show that in practice we can achieve reasonable performance. Searching continuous pitch contours at arbitrary locations was tried in the past [4], but computation-intensive dynamic programming was used for the matching.

While we agree that symbolic melody descriptions are the future for robust melody-matching, with reliable music modeling and transcription methods pending we believe it worthwhile to explore the use of continuous pitch contours in a somewhat traditional, signal-matching framework that is fast enough for practical use.

In addition, it is hard to tell from QBH experiments using MIDI target data how well the same system would

perform on arbitrary polyphonic music for which the transcriptions are unavailable and must be extracted automatically and crudely. Assuming perfect note transcriptions could lead to QBH methods that are overly sensitive to the integrity of the transcription and turn out to have little value in such real-world scenarios. Therefore, in this study we also conduct a preliminary QBH experiment on "real-world" data, i.e., commercial recordings of polyphonic music from which dominant pitch contours are obtained using an automatic f_0 tracking method.

Wavelets [5] have a rich history of diverse applications in the areas of signal coding and matching. In particular, they have been used in the past to match whole image contours [6] with robustness to affine transformations, and also to encode f_0 contours for speaker identification [7]. In the former case, the wavelet coefficients were used to match whole contours, while in the latter, to encode the f_0 contour using compact dyadic wavelet coefficients. In our study, to match f_0 contours for the purpose of melody matching, we employ "redundant" sets of wavelets defined on non-integer scale and time indices to encode segments of varying locations and time scales.

Note that throughout this paper, we conveniently assume that "main melody" and "dominant pitch contour" both mean "dominant f_0 contour," although strictly speaking, all three concepts have subtle differences.

2. INDEXING VIA REDUNDANT WAVELETS

2.1 Brief Overview of Wavelets and Notation

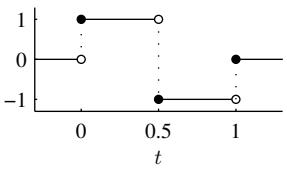


Figure 1. The Haar wavelet, $\psi(t)$

It is well known that a real, continuous-time signal $x(t)$ may be decomposed into a linear combination of a set of wavelets that form an orthonormal basis of a Hilbert Space [5]. First, we define a wavelet as

$$\psi_{m,n}(t) = 2^{-m/2}\psi\left(2^{-m}t - n\right) \tag{1}$$

for $m, n \in \mathcal{R}$ (real numbers) where m is a dilation factor, n is a displacement factor, and $\psi(t)$ is some mother wavelet function. In this paper, we use the Haar Wavelet in Fig.1. It is easy to see that the support of (1), then, is

$$t \in [n2^m, (n+1)\,2^m) \tag{2}$$

The corresponding wavelet coefficient of a signal $x(t)$ is

$$\langle x(t), \psi_{m,n}(t)\rangle = \int_{-\infty}^{+\infty} x(t)\psi_{m,n}(t)\,dt \tag{3}$$

It is well known that when m, n are integers $j, k \in \mathcal{Z}$ (integers), $\{\psi_{j,k}\}$ form an orthonormal basis and $x(t)$ is a linear combination of the resulting "dyadic" wavelet coefficients:

$$x(t) = \sum_{j,k\in\mathcal{Z}} \langle x(t), \psi_{j,k}(t)\rangle\, \psi_{j,k}(t) \tag{4}$$

Since signals are often represented by a compact set of coefficients, we can efficiently compare real signals using

$$\int_{-\infty}^{+\infty} \{x(t) - y(t)\}^2\,dt = \sum_{j,k\in\mathcal{Z}} \left(\langle x, \psi_{j,k}\rangle - \langle y, \psi_{j,k}\rangle\right)^2 \tag{5}$$

Throughout this paper, we always assume $m, n \in \mathcal{R}$ and $j, k \in \mathcal{Z}$.

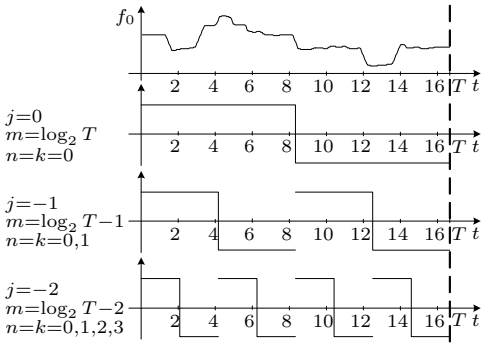


Figure 2. Example query pitch contour $q(t)$ with support $[0, T)$ and "dyadic-equivalent" wavelets $\psi_{m,n}$ that correspond to some of the dyadic wavelets $\psi_{j,k}$ of $q(Tt)$. The vertical dotted line indicates T. The wavelet amplitudes in the figure are not plotted to scale.

2.2 Application of Wavelets to Pitch Contour Matching

Assume some query f_0 contour $q(t)$, shown in Fig. 2. Also assume a pitch contour $p(t)$ of a target song, shown in Fig. 3 representing the "dominant" f_0 in a piece of polyphonic music. The query contour closely resembles a portion of the target contour, and our goal is to locate this segment. Given two contour segments representing identical melody, there are two different types of scaling that must be considered before attempting to directly compare them. The first one is in frequency, resulting from difference in musical key, which will cause one contour to be a scaled version of the other in the linear frequency domain. In the log-frequency domain, it will be a linear translation. The second scaling is in the time domain, resulting from difference in tempo. Notice that the two example melodies are sung at different speeds. The query is about 17 seconds long, while the matching segment in the target is about 12 seconds long. Both of these issues prevent us from directly comparing $p(t)$ and $q(t)$, and they will now be addressed.

2.2.1 Key Normalization

First, assume some signal $x(t)$ defined arbitrarily on $[0, 1)$ and 0 elsewhere. Since $\psi_{j,0} = 2^{-j/2}$ in $[0, 1)$ when $j > 0$, we have

$$\langle x, \psi_{j,0}\rangle = 2^{-j/2} S_x \;(j > 0)\,,\; S_x \triangleq \int_0^1 x(t)\,dt \tag{6}$$

Also note that $\langle x, \psi_{j,k} \rangle = 0$ in $[1, 0)$ for $j > 0$ and $k \neq 0$. From these relations it follows that the wavelet expansion of $x(t)$ can be decomposed as follows:

$$\begin{aligned} x(t) &= \sum_{j \leq 0, k \in \mathcal{Z}} \langle x, \psi_{j,k} \rangle \psi_{j,k} + \sum_{j>0, k=0} \langle x, \psi_{j,k} \rangle \psi_{j,k} \\ &\quad + \sum_{j>0, k \neq 0} \langle x, \psi_{j,k} \rangle \psi_{j,k} \\ &= x_N(t) + S_x \sum_{j>0, k=0} 2^{-j} + 0 = x_N(t) + S_x \end{aligned} \quad (7)$$

where we have defined

$$x_N(t) \triangleq \sum_{j \leq 0, k \in \mathcal{Z}} \langle x, \psi_{j,k} \rangle \psi_{j,k} \quad (8)$$

From the orthogonality property of the wavelets, and the fact that $x(t)$ is 0 outside of $[0, 1)$, note that

$$\langle x_N, \psi_{j,k} \rangle = \begin{cases} \langle x, \psi_{j,k} \rangle & (j,k) \in \mathcal{W} \\ 0 & \text{all other } j, k \end{cases} \quad (9)$$

where we define the set $\mathcal{W}$ of tuplets (j, k) that correspond to the dyadic wavelets in $[0, 1)$:

$$\mathcal{W} = \left\{ (j,k) : j \leq 0, 0 \leq k \leq 2^{-j} - 1, j \in \mathcal{Z}, k \in \mathcal{Z} \right\} \quad (10)$$

Now, assume another signal $y(t) = x(t) + c$ in $[1, 0)$ and 0 elsewhere. Since $S_y = S_x + c$, we can see from (7) that $y_N(t) = x_N(t)$. Hence, for any arbitrary $x(t)$ and $y(t)$ on $[0, 1)$, we can obtain "level-normalized" signals $x_N(t)$ and $y_N(t)$ that are independent of constant bias. In our case, "level" is in fact "key" when $x(t)$ and $y(t)$ are log-frequency pitch contours, since key shifts will result in constant biases. To compute their mean squared distance in a "level(key)-normalized" way, we use, instead of (5),

$$\int_{-\infty}^{+\infty} \{x_N(t) - y_N(t)\}^2 dt = \sum_{j,k \in \mathcal{W}} (\langle x, \psi \rangle - \langle y, \psi \rangle)^2 \quad (11)$$

2.2.2 Time and Key Normalization of the Query

Assume that the query signal $q(t)$ is defined arbitrarily in $[0, T)$ and 0 elsewhere. The first step is to time-scale it into a "time-normalized" signal $q'(t)$ defined on $[0, 1)$ and 0 elsewhere:

$$q'(t) \triangleq q(Tt) \quad (12)$$

Using (3) and (1), it is easy to see that

$$\begin{aligned} \langle q'(t), \psi_{j,k}(t) \rangle &= T^{-1/2} \langle q(t), \psi_{m,n}(t) \rangle \\ m = j + \log_2 T, \; n &= k \quad (j,k \in \mathcal{Z}) \end{aligned} \quad (13)$$

Fig. 2 shows $\psi_{m,n}$ for $(j, k) \in \mathcal{W}$ when $j = 0, -1$, and -2, which corresponds to $m = \log_2 T$, $-1 + \log_2 T$, and $-2 + \log_2 T$, respectively. The wavelets $\{\psi_{m,n}\}$ could be regarded as the "dyadic-equivalent" wavelets of $q(t)$ – the wavelets applied to $q(t)$ that are equivalent to the dyadic wavelets applied to its time-normalized version $q'(t)$.

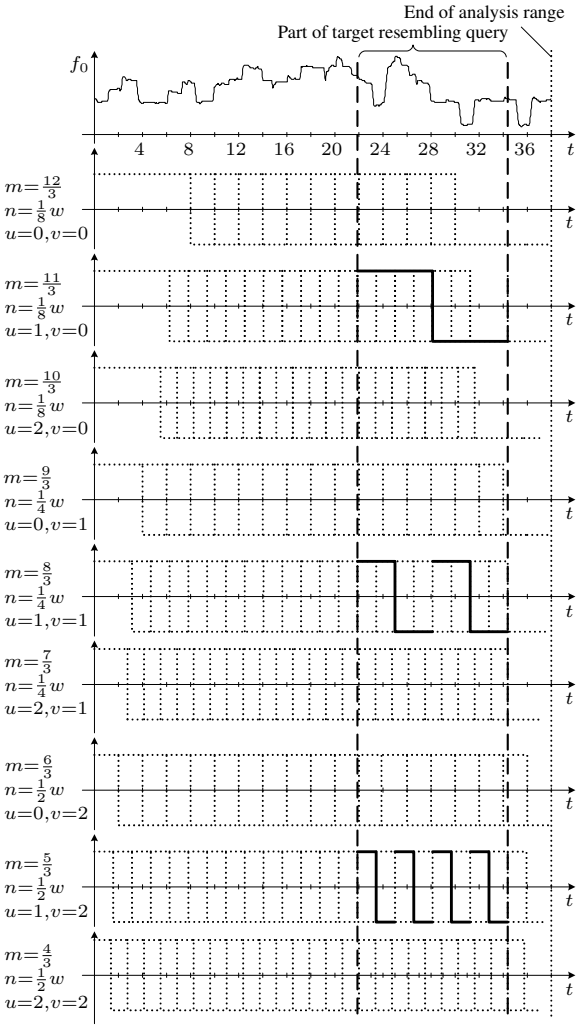


Figure 3. Example target pitch contour $p(t)$ and a redundant set of wavelets with design parameters $D = 3$, $M = 12$, $V = 2$, and $E = 3$ encoding the contour at different locations over a range of time scales. The bold broken line shows the segment resembling the query in Fig.2, and the bold lines show the "dyadic-equivalent" wavelets that encode this segment

Now, if we only compute those wavelet coefficients for $(j, k) \in \mathcal{W}$, we can obtained the *key-normalized*, *time-normalized* signal $q'_N(t)$. From (9) and (13), we have

$$\langle q'_N, \psi_{j,k} \rangle = \begin{cases} T^{-1/2} \langle q, \psi_{m,n} \rangle & \begin{matrix} (j,k) \in \mathcal{W} \\ m = j + \log_2 T, n = k \end{matrix} \\ 0 & \text{all other } j, k \end{cases} \quad (14)$$

2.2.3 Normalization and Redundant Encoding of Targets

For the target pitch contour $p(t)$, we do a *redundant* wavelet analysis so that we can search multiple, overlapping sections of varying time scales in $p(t)$. Some sort of regularity must be imposed on the scale factors and analysis intervals so that the coefficients can be used efficiently. Note that there can be many ways to do this, and here we are proposing one such method. While we present a general

formulation of our design, the easiest way to understand this section is by studying the specific example in Fig.3.

We compute a "redundant" set of wavelet coefficients $\{\langle p, \psi_{m,n}\rangle : u, v, w\}$, where we set

$$m = \frac{M-u}{D} - v,\ u = 0, 1, \cdots, D-1,\ v = 0, 1, \cdots, V \tag{15}$$

The constant D represents the amount of resolution in the time scales over which the redundant analysis is done. $M > D$ represents some upper limit in m, u is a time scale factor, v is a nonzero integer, and $V < \frac{M}{D}$ represents some lower limit in m. For each m, the possible values of n are

$$n = \frac{1}{2^{E-v}} w,\ w = 0, 1, \cdots \tag{16}$$

$E > V$ represents the amount of time resolution. Fig. 3 shows the wavelets with $D = 3$, $M = 12$, $V = 2$ and $E = 3$.

Now, consider the part of $p(t)$ in

$$t \in [n_0 2^{m_0}, (n_0 + 1)\, 2^{m_0}) \tag{17}$$

which is exactly the support of ψ_{m_0,n_0} by (2). We also constrain m_0 and n_0 to conform to (15) and (16):

$$\begin{cases} m_0 = \dfrac{M-u_0}{D} - v_0 & 0 \le u_0 \le D-1, 0 \le v_0 \le V, \\ & u_0, v_0 \in \mathcal{Z} \\ n_0 = \dfrac{1}{2^E} w_0 & w_0 \ge 0, w_0 \in \mathcal{Z} \end{cases} \tag{18}$$

The time-normalized version of this portion of $p(t)$, assuming zero elsewhere, is

$$p'(t) = \begin{cases} p\left(2^{m_0}(t + n_0)\right) & t \in [0, 1) \\ 0 & \text{elsewhere} \end{cases} \tag{19}$$

It is easy to see that the corresponding key-normalized, time-normalized signal $p'_N(t)$ will have wavelet coefficients

$$\langle p'_N, \psi_{j,k}\rangle = \begin{cases} 2^{-m_0/2} \langle p, \psi_{m',n'}\rangle & \begin{matrix}(j,k)\in\mathcal{W} \\ m'=m_0+j \\ n'=k+2^{-j}n_0\end{matrix} \\ 0 & \text{all other } j, k \end{cases} \tag{20}$$

Now, from (15), (16), and (18) one can see that all coefficients $\langle p, \psi_{m',n'}\rangle$ required above can always be found in the set of wavelets $\{\langle p, \psi_{m,n}\rangle : u, v, w\}$ up to scale level $j = v_0 - V$. One can also notice that many wavelet coefficients can be "reused" in the sense that they contribute to more than one contour segment. In the example in Fig.3, we see $\{\psi_{m',n'}\}$ for $j = 0, -1, -2$ with $m_0 = \frac{11}{3}$ and $n_0 = \frac{14}{8}$, which encode the section of $p(t)$ that pertains to the query $q(t)$ in Fig.2.

Using the wavelet coefficients in (14) and (20), we can compute the distance between the key- and time-normalized query $q'_N(t)$ and target segment $p'_N(t)$ using (11). The distance will be an approximation, since we cannot take the coefficients over the entire set $\mathcal{W}$ but over a finite number of scale levels that provides sufficient accuracy (e.g., $j = 0$ to $j = -4$ for a 7s pitch contour sampled at 10ms).

To account for variations in tempo, we compare segments over a range of values of m_0. Note first that if the query and target had the same tempo, we should have $m_0 = \log_2 T$, which would produce portions of $p(t)$ with length T in (17), to obtain the most accurate match. Now, if we allow the query's tempo to be as slow as half the target's tempo and as fast as twice the target's tempo, we can let m_0 vary within the range

$$-1 + \log_2 T < m_0 < 1 + \log_2 T \tag{21}$$

which results in around $2D$ different values of m_0 according to the system design.

2.2.4 Two-Stage Search of Arbitrary Target Locations

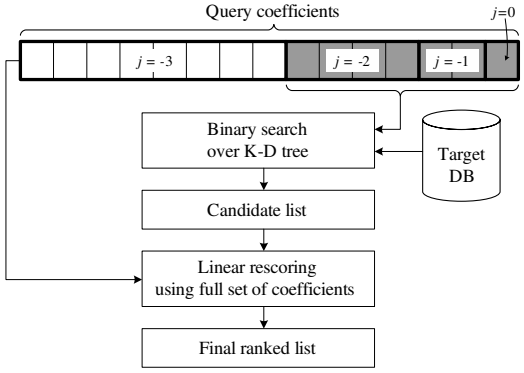


Figure 4. Schematic overview of two-stage search. In this example, 7 wavelet coefficients are indexed by a K-D Tree, and 15 coefficients are used for the linear rescoring.

The variable n_0 in (19) controls the location of the target segment compared with the query. The resolution of the wavelet locations can be controlled to find a good compromise between speed and accuracy. For efficient comparison of a query with a large number of targets, the possible dyadic-equivalent coefficients embedded in every target (i.e., the coefficients in (20)) can be indexed as coordinates in a binary K-D Tree [3] with a fixed number of dimensions. Each leaf in the tree coarsely represents a melodic fragment in the target database. At the first stage of the search, the query coefficients are appropriately scaled to form a search sphere that is used to find tree leaves that are spatially close to the query, resulting in a list of candidate melody fragments. At the second stage, a linear search is conducted over the candidates using a larger number of coefficients to more accurately compute (11), which is then used to rank the results as shown in Fig.4.

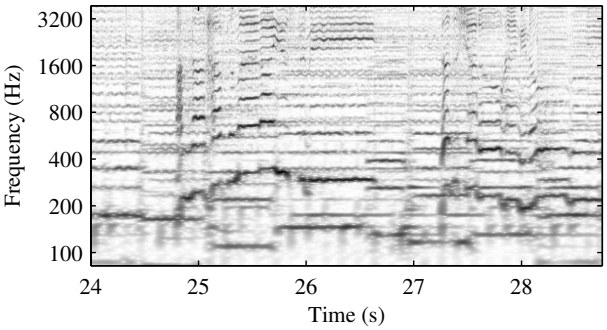
In practice, no more than 31 wavelet coefficients ($j = 0$ to $j = -4$ in $\mathcal{W}$) are usually sufficient to represent a melody segment with length 7s sampled every 10ms. In such a case, the first 7 coefficients ($j = 0$ to $j = -2$ in $\mathcal{W}$) can be indexed in the K-D tree, while the full 31 coefficients are used in the linear rescoring stage.

In terms of computational complexity, if dynamic programming(DP) were used to search for a query of length L_q[frames] in a target of length L_t[frames], scores would have to be calculated for $L_q L_t$ coordinates (assuming no

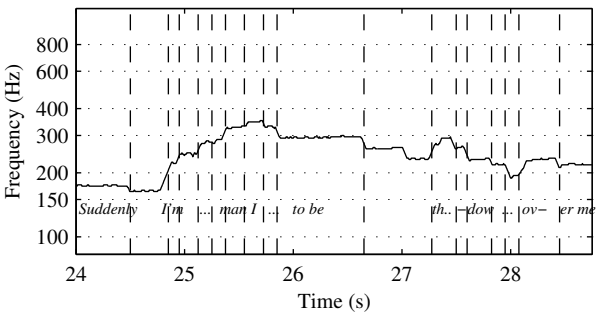
pruning). If a linear search were used with the proposed method, we would need to compute only $kL_t (k << L_q)$ vector distances, where k is essentially a constant since the number of wavelet coefficients (as in Fig.3) increases linearly with L_t. The addition of a K-D tree further reduces this number drastically, making the computational gains of the proposed method even more apparent.

3. EXPERIMENT

3.1 Pitch Contour Extraction



(a) Log-frequency spectrum

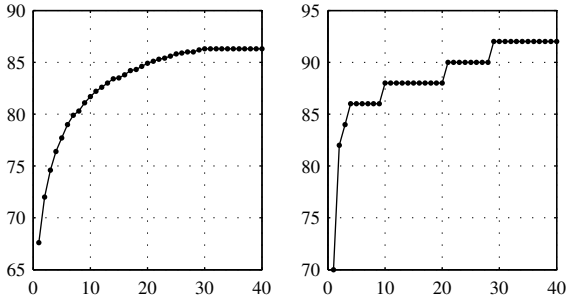


(b) Dominant f_0 contour

Figure 5. (a) Log magnitude of log-frequency spectrum (dark is high) from 82.4 Hz to 3.84 kHz of a segment of *Yesterday* by *The Beatles*. Frequency components of both voice and instrumental accompaniment are clearly visible. (b) Pitch contour of segment with hand-marked note boundaries (broken vertical lines) and corresponding lyrics in select locations (full lyrics are "Suddenly, I'm not half the man I used to be, there's a shadow hanging over me")

A simple method based on known techniques was used to obtain dominant f_0 contours from music recordings. The Constant-Q Transform [8] of each music signal was taken to obtain spectral components on a log-frequency scale. Fig.5(a) shows the spectrogram for a segment of *Yesterday* by *The Beatles*. Next, we assigned scores for each (t, f) on the time-frequency plane by computing weighted sums of the spectral components at harmonics of f [9]. After limiting the range of the dominant pitch via some heuristics, we applied dynamic programming on the $t - f$ plane of scores to obtain a continuous pitch contour [10] that maximizes the sum of pitch scores along its path. Fig.5(b) shows the pitch contour obtained for the *Yesterday* example. While the overall structure of the contour reflects the vocal melody of this part of the song, we can notice that in the non-vocal sections between "Suddenly" and "I'm" and between "to be" and "there's", the dynamic programming picked up the pitch of the strings in the background. Inflections in vocal pitch inevitably produced during singing, and other minor deviations from what is probably the "true" music score are also reflected in the continuous contour. However, we made no attempt to identify and compensate for any such deviations or discriminate between vocal and non-vocal sections, and directly used the whole pitch contour from every target in the database in our experiments.

3.2 QBH Test



(a) MIREX 2006 test set (b) "Real-world" test set

Figure 6. Search performance for (a) MIREX 2006 test set and (b) "real-world" test set. The vertical axis represents the inclusion rate(%), and the horizontal axis is the number of search results.

Two experiments were conducted: the first on monophonic music to validate our method with existing QBH tasks, and the second on polyphonic music to make a preliminary assessment of its use in real-world scenarios. For both experiments, the dominant f_0 contour was automatically extracted from query and target data using the aforementioned method. Contours were sampled every 10ms.

For the first experiment, we used the MIREX 2006 QBSH test set (see description in [2]). All target data in this set are monophonic MIDI data, so we first converted them to WAV format. Each song in the database was 29.9s long on average (17 hours total for the database of 2,048 songs). Fig.6(a) shows the inclusion rate for varying number of search results, i.e., the rate at which the correct melody was ranked within the top n of all returned results. For $n = 20$, the inclusion rate was 84.9%, which is significantly lower than the state-of-the-art [2], 96.4%. Note, however, that the latter system constrained the queries to occur at only the beginning of music phrases. Since almost all queries in the MIREX 2006 test set start at the beginning of their targets, such a data set would greatly favor systems with such constraints. Our proposed system, on the other hand, made no assumptions on starting locations and exhaustively searched over all possible locations, limited only by the wavelet parameters. Also, tempo variation is taken into account from the very beginning of the search, not just at the latter fine search stage. Hence,

the search space was larger, which resulted in more room for confusion. At the same time, the search time for each query was usually less than one second on a 3.2GHz processor depending on system parameters.

For the second experiment, we used a "real-world" database consisting of 613 acoustic recordings of songs with instrumental accompaniment, totaling around 37 hours of audio (average 3.6 minutes per song). 155 of the songs were from the RWC Music Database [11], and the rest were commercially-distributed pop songs. A preliminary set of queries were obtained from six non-professional singers – three male, and three female. Each person was asked to sing several easy and well-known songs including "Happy Birthday," "The Alphabet Song," and "Are You Sleeping, Brother John?" from which query segments at random locations were extracted. Each query was 5$\sim$12 seconds long, and there were a total 50 queries. We informally verified that the songs in the target database corresponding to these queries had reasonably clear dominant f_0's, but there were still noticeable errors in the f_0 extraction due to instrumental accompaniment, like in the example in Fig. 5(b). Fig. 6(b) shows the inclusion rate for a varying number of search results. The inclusion rate was 86% for $n = 5$ and 88% for $n = 20$, which seems similar to that of another state-of-the-art system [12] that also allows queries to begin at random locations but uses a MIDI database. We are cautious in directly comparing the performance of the two systems, however, because they differ in experimental setup. Nevertheless, our results are promising because we used a database of polyphonic recordings instead of MIDI data. Larger data sets and larger numbers of queries will have to be used in the future to more rigorously assess real-world performance.

4. CONCLUSION AND FUTURE WORK

We have proposed an efficient method of indexing and matching music melodies based on their continuous pitch contours while allowing partial matches at arbitrary locations using redundant wavelet transformations. By directly comparing continuous pitch contours instead of their note transcriptions as in most existing methods, we avoid the compounding of transcription errors. On the other hand, our method is also computationally efficient because it uses the mean squared sum between fixed vectors instead of dynamic programming, while at the same time being able to adjust for differences in tempo and key. Experiments were conducted on both existing monophonic MIDI databases and preliminarily on real-world recordings with instrumental accompaniment to show that the system can be practically applied, even when using a simple mean squared distance measure between key- and time-normalized contour segments. While the system still depends on reliable dominant pitch extraction, minor pitch tracking errors did not hurt performance because the overall pitch and rhythm structure of contours was compared. One trade-off for the system's efficiency is that it does not explicitly account for rhythmic variations within queries as do techniques based on string-matching. Much work is being done in the MIR community toward model-based symbolic representations that allow a more modular framework for indexing and search, such as via HMMs, and we plan to leverage the insights gained in our work to this end.

5. ACKNOWLEDGMENTS

Thanks to Lei Wang for kindly sharing the data for the MIREX 2006 QBH experiment used in [2].

6. REFERENCES

[1] D. Mazzoni and R. B. Dannenberg. Melody matching directly from audio. In *Proc. ISMIR*, 2001.

[2] L. Wang, S. Huang, S. Hu, J. Liang, and B. Xu. Improving searching speed and accuracy of query by humming system based on three methods: Feature fusion, candidates set reduction and multiple similarity measurement rescoring. In *Proc. INTERSPEECH*, pages 2024–2027, 2008.

[3] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Comm. ACM*, 18, 1975.

[4] L. Guo, X. He, Y. Zhang, and Y. Lu. Content-based retrieval of polyphonic music objects using pitch contour. In *IEEE Int. Conf. Acoust., Speech. Signal Processing*, pages 2205–2208, 2008.

[5] I. Daubechies. *Ten Lectures on Wavelets*. SIAM: Society for Industrial and Applied Mathematics, 1992.

[6] Q. M. Tieng and W. W. Boles. Complex daubechies wavelet based affine invariant representation for object recognition. In *IEEE ICIP*, pages 198–202, 1994.

[7] F. Farahani, P.G. Georgiou, and S.S. Narayanan. Speaker identification using supra-segmental pitch pattern dynamics. In *IEEE Int. Conf. Acoust., Speech. Signal Processing*, 2004.

[8] J. C. Brown and M. S. Puckette. An efficient algorithm for the calculation of a constant Q transform. *IEEE Trans. Audio, Speech, and Language Processing*, 92:2698–2701, 1992.

[9] D. J. Hermes. Measurement of pitch by subharmonic summation. *J. Acoust. Soc. Am.*, 83(1):257–264, 1988.

[10] B. Secrest and G. Doddington. An integrated pitch tracking algorithm for speech systems. In *IEEE Int. Conf. Acoust., Speech. Signal Processing*, April 1983.

[11] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC music database: Popular, classical, and jazz music databases. In *Proc. ISMIR*, pages 287–288, 2002.

[12] E. Unal, E. Chew, P.G. Georgiou, and S.S. Narayanan. Challenging uncertainty in query by humming systems: A fingerprinting approach. *IEEE Trans. ASLP*, 16, 2008.