EFFICIENT SEARCH OF MUSIC PITCH CONTOURS USING WAVELET TRANSFORMS AND SEGMENTED DYNAMIC TIME WARPING

Woojay Jeon and Changxue Ma
Motorola, Inc.
Schaumburg, Illinois, U.S.A.

ABSTRACT
We propose a method of music melody matching based on their "continuous" (or "time-frame-based") pitch contours. Most previous methods using frame-based contours either made limiting assumptions on the locations, musical scale, tempo, and/or rhythm of the queries in relations to the targets, or involved exhaustive dynamic time-warping procedures that were too computation-intensive for use in real scenarios. In the proposed method, variable-scale windowing and wavelet transformations are performed at an initial coarse search stage to efficiently match queries to targets. At the following fine search stage, we apply a novel segmented dynamic time warping (DTW) method for melody contours, computing a more accurate distance between the query and each of the candidate targets with less computation than traditional DTW. The method searches arbitrary target locations and explicitly adjusts for differences in tempo and musical scale between queries and targets as well as rhythmic inconsistencies within queries. At the same time, retrieval times are fast enough for use in real-world scenarios.

Index Terms— melody, qbh, query-by-humming, dynamic time warping, wavelets

1. INTRODUCTION
Music melody matching, usually embodied in a Query-by-Humming (QBH) application, is a content-based way of retrieving music data. Previous techniques searched melodies based on either their "continuous" (or "frame-based") pitch contours [2] or their note transcriptions [6]. The former are pitch values sampled at fixed, short intervals (10ms or so), such as the contours in Fig.1, while the latter are sequences of quantized, symbolic representations of melodies (such as "C4-D4-E4-G3-" or "Up-Up-Down-Same"). Note transcriptions usually provide fast and accurate matches [6] when the transcriptions of the melody are known (e.g. provided as MIDI files) or easy to obtain (e.g. if the source audio is monophonic). When the "main melody" must be automatically obtained from polyphonic source audio and is subject to errors, however, note transcriptions may segment and quantize dynamic pitch values too rigidly, compounding the effect of pitch extraction errors.

For this reason, using the "raw" frame-based pitch contours (which we call hereon "pitch contours") has been suggested in the past as giving more accurate match results [7]. The major drawback is that pitch contours hold much more data and therefore require much more computation than symbolic representations, especially when using the popular dynamic time warping (DTW) [5] to measure the similarity between two melodies [2, 1]. Although a number of methods achieved efficient melody matching performance using continuous contours, some limiting assumptions had to be made. In one study that achieved fast search times, the query and target had to have roughly similar tempo, and the starting locations of query melodies were limited to the beginning of specific music phrases [7]. Very little (if any) methods have been reported so far that can efficiently match frame-based pitch contours while adjusting for key, tempo, and rhythm and exhaustively search all target locations with no assumptions.

Striving toward this end, we propose a new method of indexing and searching pitch contours with two new developments compared to a previous method [3]: First, in an initial "coarse search" stage we apply variable-scale windowing on the query contours to compare them with fixed-length target contours segments while efficiently scanning over variable tempo differences and locations. Because the target segments are of fixed-length, we drastically reducing the storage space required in the previous method. Furthermore, by breaking the query contours into parts, we can potentially handle rhythmic inconsistencies more flexibly. Second, in the "fine search" stage, we apply a novel "segmented" DTW method for melody contours that calculates a more accurate similarity score between the query and each candidate target with more explicit consideration for rhythmic inconsistencies. We show how the segmented DTW is an approximation of the conventional DTW that sacrifices some accuracy but allows faster search times more suitable for practical application.

Experimental results using the MIREX 2006 QBH corpus and a preliminary "real-world" test set shows that the method improves performance without sacrificing too much speed compared to the previous method.

2. COARSE SEARCH
2.1. Previous search method
Assume a target pitch contour \( p(t) \) and a query pitch contour \( q(t) \) denoting log frequency values defined on the continuous time \( t \)-axis, as shown in Fig.1. The purpose is to compare \( q(t) \) with various segments of \( p(t) \) at various starting locations to find the best-matching segment(s) (e.g. the segment from \( t_0 \) to \( t_2 \)). In our previous work [3], the target contour was essentially divided into overlapping segments of varying length to account for differences in tempo between the query and the target. In order to directly compare segments using a simple Euclidean distance, each segment was normalized to have length 1. Also, to normalize differences in musical key, the mean was subtracted, since key transpositions result in linear translations of pitch contours along the log-frequency axis. For a segment of \( p(t) \) at \( t_0 \) with length \( T \), the time-normalized segment is

\[ p'(t) = p(Tt + t_0) \] (1)

on \( t \in [0, 1) \) and 0 elsewhere. The time-normalized, level(key)-normalized segment is defined as

\[ p'_k(t) = p'(t) - \int_0^1 p'(t) \, dt \] (2)
3. FINE SEARCH VIA SEGMENTED DTW

Dynamic time warping (DTW) [5] is very commonly used for matching melody sequences. In this section, we will begin by formulating an “optimal” DTW criterion for frame-based pitch contours, then derive a “segmented” DTW method for melody contours as an approximation of the optimal DTW, giving a formal mathematical treatment to how adjustments for key and tempo deviations can be made. Modified “fast” forms of DTW usually achieve faster speed by somehow reducing the amount of data (e.g., via smoothing). In the proposed case, we essentially divide the melody into partitions, treating each partition as a rhythmically consistent unit.

3.1. Problem formulation

Assume a query contour \( q(t) \) and target contour \( p(t) \), each defined on a bounded interval on the continuous \( t \)-axis. Assume we sample the contours at equal rates and obtain the sets of samples \( Q = \{ q_1, q_2, \ldots, q_Q \} \) and \( P = \{ p_1, p_2, \ldots, p_P \} \), where \( |Q| \) and \( |P| \) represent the cardinality of \( Q \) and \( P \), respectively. Following [5], the distance between \( Q \) and \( P \) according to the warping functions \( \phi_q(\cdot) \) and \( \phi_p(\cdot) \) where the total number of warping operations is \( R \) is

\[
D(Q, P; \phi_q, \phi_p, b) = \sum_{i=1}^{R} d(\phi_q(i), \phi_p(i) : b(i))
\]

The extra parameter \( b(i) \) is a bias factor indicating the difference in key between the query and target. If we know the target is sung at one octave higher than the query, for example, we can add 1 to all members in \( Q \) to make it directly comparable to \( P \) assuming \( \log_2 \) frequencies. We define the distance function as simply the squared difference between the target pitch and the biased query pitch:

\[
d(\phi_q(i), \phi_p(i) : b(i)) = \left[ q(\phi_q(i)) + b(i) - p(\phi_p(i)) \right]^2
\]

It is reasonable to assume that the bias \( b(i) \) remains roughly constant with respect to \( i \). That is, every singer should not deviate too much off-key when singing a song (otherwise, the singer must be regarded as singing a different song, as far as the melody is considered), although he is free to choose whatever key he wishes. We can constrain it to be tied to an overall bias \( b \) as follows:

\[
b(i) = b + \delta_i,
\]

where

\[
\delta_i = \arg\min_{\delta:|b|<\Delta} \left[ q(\phi_q(i)) + b + \delta - p(\phi_p(i)) \right]^2
\]

\( \Delta \) defines the maximum deviation of \( b(i) \) from \( b \).

Hence, the goal is to find the warping functions and the overall bias value that will minimize the overall distance between \( P \) and \( Q \):

\[
D^* = \min_{\phi_q, \phi_p, b} D(Q, P; \phi_q, \phi_p, b)
\]

DTW [5] can be used to solve this equation. However, this would be extremely computationally intensive [2]. If the set \( B = \{ b_1, b_2, \ldots, b_{|B|} \} \) denoted the set of all possible values of \( b \), we would essentially have to compute costs for all coordinates in a three-dimensional \( |Q| \times |P| \times |B| \) space.
3.2. Segmented dynamic time warping

We now propose a “segmented” DTW method that approximates (7). First, we partition the warping sequence into \( N \leq R \) parts, defined by a monotonically increasing sequence of integers \( \theta_1, \ldots, \theta_{N+1} \) where \( \theta_1 = 0 \) and \( \theta_{N+1} = R \). We rewrite (4) as

\[
D = \sum_{i=1}^{N} \sum_{i=	heta_{i+1}}^{\theta_i+1} d(\phi_q(i), \phi_p(t) ; b + \delta_i)
\]

The first approximation is to assume that the \( \delta_i \)'s are constant within each partition, i.e.,

\[
\delta_i = \delta_s \quad (\theta_s + 1 \leq i \leq \theta_{s+1})
\]

Next, for later convenience we approximate the partial summations above as integrals, assuming that \( \phi_q(i) \) and \( \phi_p(i) \) are defined on the continuous-time \( t \)-axis as well as the discrete-time \( i \)-axis.

\[
D \approx \sum_{s=1}^{N} \int_{\theta_s}^{\theta_{s+1}} d(\phi_q(t), \phi_p(t) ; b + \delta_s) dt
\]

The third approximation is that the warping functions \( \phi_q \) and \( \phi_p \) are straight lines within each partition, bounded as such:

\[
\begin{align*}
\phi_q(t) &= q_{start,s}, \quad \phi_q(t) = q_{end,s} \\
\phi_p(t) &= p_{start,s}, \quad \phi_p(t) = p_{end,s}
\end{align*}
\]

This results in the following warping functions:

\[
\begin{align*}
\phi_q(t) &= \frac{q_{end,s} - q_{start,s}}{\theta_{s+1} - \theta_s} (t - \theta_s) + q_{start,s} \\
\phi_p(t) &= \frac{p_{end,s} - p_{start,s}}{\theta_{s+1} - \theta_s} (t - \theta_s) + p_{start,s}
\end{align*}
\]

Substituting this into (10) and applying (5), we get

\[
D = \sum_{s=1}^{N} (\theta_{s+1} - \theta_s) \int_0^1 (q'_s(t) + b + \delta_s - p'_s(t))^2 dt
\]

where \( q'_s(t) \) and \( p'_s(t) \) are essentially the time-normalized versions of \( q(t) \) and \( p(t) \) in partition \( s \) as in (1):

\[
\begin{align*}
q'_s(t) &= q \{ (q_{end,s} - q_{start,s}) t + q_{start,s} \} \\
p'_s(t) &= p \{ (p_{end,s} - p_{start,s}) t + p_{start,s} \}
\end{align*}
\]

In (13), we set the weight factor to be the proportion of the query occupied by the partition.

\[
w_s \triangleq \theta_{s+1} - \theta_s = \frac{q_{end,s} - q_{start,s}}{Q(t) - q_{start,1}}
\]

In (6), we set \( \delta_i \) such that it minimizes the overall cost in segment \( s \):

\[
\delta_s = \arg \min_{|\delta|} \int_0^1 (q'_s(t) + b + \delta - p'_s(t))^2 dt
\]

Since the integral in the above equation is quadratic with respect to \( \delta \), the solution can be easily found to be

\[
\delta_s = \begin{cases} 
\xi_s & \text{if } -\delta \leq \xi_s \leq \delta \\
-\delta & \text{if } \xi_s < -\delta \\
\delta & \text{if } \xi_s > \delta
\end{cases}
\]

where

\[
\xi_s = \int_0^1 (p'_s(t) - q'_s(t) - b) dt
\]

There still remains the problem of finding \( b \). We set it to the value that minimizes the cost for the first segment with \( \delta_1 \) set to 0:

\[
b = \arg \min_b \int_0^1 (q'_s(t) + b - p'_s(t))^2 dt = \int_0^1 (p'_s(t) - q'_s(t) - b) dt
\]

In (11), we assume that the query boundary points \( q_{start,s} \) and \( q_{end,s} \) are provided to us by some query segmentation rule. The overall optimization criterion can now be summarized as

\[
D^* = \min_{\phi_p} \sum_{s=1}^{N} w_s \int_0^1 (q'_s(t) + b + \delta_s - p'_s(t))^2 dt
\]

where \( \phi_p \) is completely defined by the set of target contour boundary points, \( \{p_{start,1}, \ldots, p_{start,N}\} \) and \( \{p_{end,1}, \ldots, p_{end,N}\} \). All other variables in (20) depend on either \( \phi_p \) or preset constants. Compared to the original “optimal” criterion in (7), the problem has been reduced to optimizing only \( 2N \) variables that define the target contour boundary points.

3.3. Segmented DTW via level-building

(20) can be solved using level-building [5]. Each query segment \( Q_s = \{q_i : q_{start,s} \leq i \leq q_{end,s}\} \) is preset according to some heuristic query segmentation rule. The target pitch sequence is treated as a sequence of observed features to be aligned with the given sequence of query segments. To allow flexibility in aligning the target contour to the query segments, we do not impose \( p_{end,s} \) to be equal to \( p_{start,s+1} \). Since there are \( 2N \) boundary points to be determined, the level-building is done on \( 2N \) levels. Level \( 2s - 1 \) allows \( p_{start,s} \) to deviate from \( p_{end,s-1} \) over some range, while level \( 2s \) determines \( p_{end,s} \) subject to the constraint

\[
p_{start,s+1} + \alpha_{min} (q_{end,s} - q_{start,s}) \leq p_{end,s} \leq p_{start,s-1} + \alpha_{max} (q_{end,s} - q_{start,s})
\]

where \( \alpha_{min} \) and \( \alpha_{max} \) are heuristically set based on the estimated range of tempo difference between the query and target (this can be obtained from the coarse search stage). Fig. 3 shows an example where the query is divided into three segments of equal length, and the target’s boundary points are subject to the following constraints:

\[
\begin{align*}
1 & \leq p_{start,s} \leq 3 & s = 1 \\
1 & \leq p_{start,s} \leq p_{end,s} & s > 1 \\
1 & \leq p_{start,s} \leq p_{end,s} + 4 & s \geq 1
\end{align*}
\]

As shown in the figure, it is possible for the resulting optimal target segments to overlap one another (\( p_{start,2} < p_{end,3} \)). The bias factor \( b \)}
Table 1. Results for MIREX 2006 Test Set

<table>
<thead>
<tr>
<th>Method</th>
<th>MRR</th>
<th>Top-n Hit Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous</td>
<td>0.789</td>
<td>75.7 80.6 82.7 84.8 86.7</td>
</tr>
<tr>
<td>Proposed</td>
<td>0.838</td>
<td>82.4 84.5 85.3 86.2 88.7</td>
</tr>
</tbody>
</table>

in (19) is calculated at the second level and is propagated up the succeeding levels. The “time-normalized” integrals in (17) and (20) can be efficiently computed using the wavelet coefficients of the time-normalized signals in (1). The coefficients for the query segments, in particular, can be precomputed and stored for repeated use. All single path costs at odd-numbered levels are set to 0, and path costs are only accumulated at even-numbered levels to result in (20).

Note that if we set \( N = 1 \), \( q_{\text{tar},1} = 1 \), and \( q_{\text{end},1} = |Q| \), the problem essentially becomes the same as the previous method [3] where we simply matched the whole query segment with varying portions of the target. On the other hand, if we set \( N = |Q| \) and \( q_{\text{tar},s} = q_{\text{end},s-1} = s \), the problem becomes essentially identical to the “optimal” DTW in (7). Hence, the segmented DTW method proposed here is a compromise between computational efficiency and search flexibility (and accuracy).

### 4. EXPERIMENT AND RESULTS

To extract the dominant \( f_0 \) (fundamental frequency) contours from source audio, we used known subharmonic summation and dynamic programming techniques. The coarse search method described in Section 2.2 was used to create an initial list of match candidates. The segmented DTW method was then used to re-rank the list in order of relevance to produce the final results. Two experiments were conducted. The first was using the MIREX 2006 QBHS test set (see [7]). All target data in this set are monophonic MIDI, so we converted them to WAV and then extracted the \( f_0 \). Each song in the database was 29.9s long on average (17 hours total for the database of 2,048 songs). Tab.1 shows the Mean Reciprocal Rank (MRR) and the top-n hit rate of the search results using the previous method [3] and the proposed method (coarse search followed by segmented DTW). The MRR is defined as \( \frac{1}{n} \sum_{i=1}^{n} 1/r_i \), where \( n \) is the number of queries and \( r_i \) is the rank of the correct target in the list of returned results for the \( i \)th query. The top-n hit rate is the rate at which \( r_i \leq n \). The proposed method improves the previous method by 0.05 points for the MRR and about 7 percent points for the top-1 hit rate. State-of-the-art performance in this test is around 0.929 MRR [7], but the corresponding method limited target locations to the beginning of music phrases. Since all queries in the MIREX 2006 test start at the beginning of target songs, this would favor such a method over the proposed system which exhaustively searches all locations and therefore a much wider search space. To illustrate this point, we constrained the proposed method such that queries could only start at the beginning of songs, and achieved an MRR of 0.921, which is very close to the state-of-the-art. An MRR of 0.900 was achieved in other work [8] but statistical assumptions on query locations were still made, and note transcriptions were used at initial filtering stages. To the best of our knowledge, there are no reported results for this test from systems that efficiently use frame-based contours while explicitly allowing tempo, key, and location differences. Average search time per query was 0.15s and 0.64s for the previous and proposed methods, respectively, on a single 3.1GHz CPU core.

All target sources in the MIREX 2006 test set are monophonic MIDI sources. Unfortunately, a similar test set using polyphonic sources is not yet available, so we used a preliminary “real-world” test environment similar to that described in [3]. The database consists of 613 acoustic recordings of songs – the majority being commercial pop songs – with instrumental accompaniment, 37 hours in total length (average 3.6 minutes per song). 102 queries, each about 5~12 seconds long and occurring at random locations within the corresponding songs, were obtained from six non-professional singers. We observed a 0.06 improvement in MRR and 9 percent point improvement in top-1 hit rate when using the proposed method.

### 5. CONCLUSION AND FUTURE WORK

In this study, we proposed a melody-based music search system that uses frame-based pitch contours and adjusts for singer-dependent variations such as tempo, pitch scale, target location, and rhythm while maintaining computational efficiency. In a coarse search stage, variable-length windowing was applied on query pitch contours, followed by normalization, to search for similar segments in a set of targets. Wavelet coefficients were used to efficiently store, index, and match contour segments. In a fine search stage, a novel segmented dynamic time-warping method for melody contours was used to compute a more accurate distance between the query and each candidate target, adjusting for rhythmic inconsistencies in the query that typically occur among untrained singers. Results on the MIREX 2006 test set and a preliminary polyphonic audio test set showed that the proposed method provides better accuracy than a previous method. With a more comprehensive database of queries and polyphonic music, we plan to more rigorously assess the use of the method in practical scenarios. We also hope that the insights gained in our studies will help further the advancement of other content-based music retrieval methods.

### 6. REFERENCES


