Efficient Search of Music Pitch Contours Using Wavelet Transforms and Segmented Dynamic Time Warping

Woojay Jeon

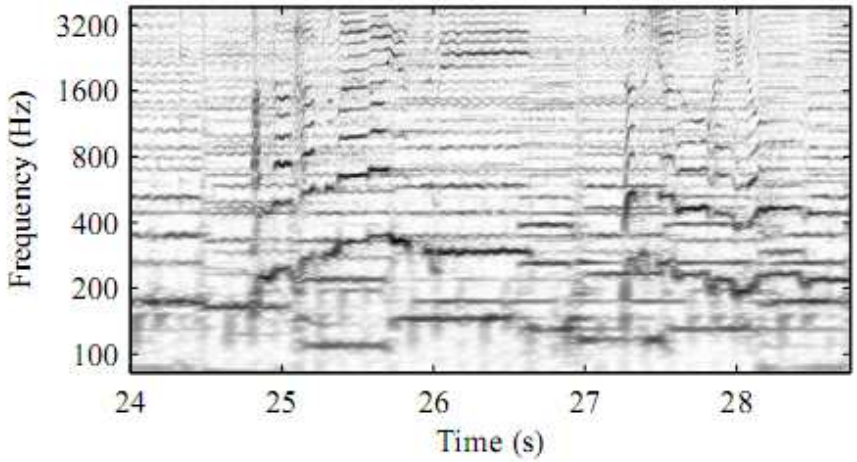
Changxue Ma

Content(Melody)-Based Music Search

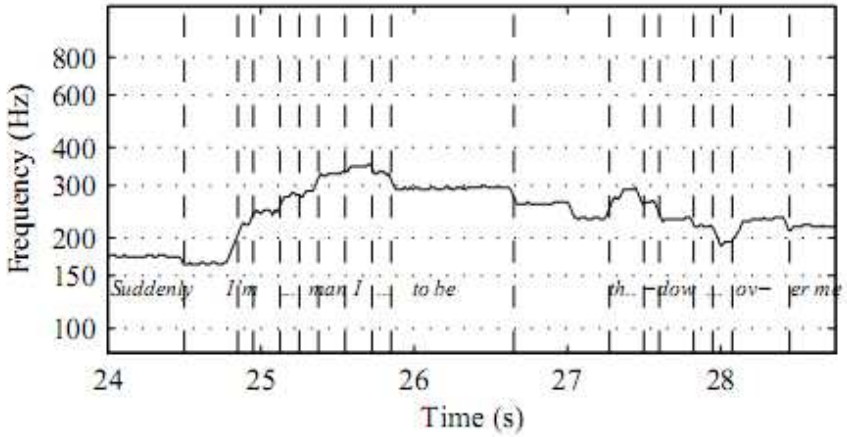
- Definition of “Main Melody”
  - “The single (monophonic) pitch sequence that a listener might reproduce if asked to whistle or hum a piece of polyphonic music, and that a listener would recognize as being the ‘essence’ of that music when heard in comparison” [Poliner 07]
- Our loose definition

  Melody = Dominant Pitch Contour = **Dominant Fundamental Frequency (f_0) Contour**
- Methods of representing melodies for music search
  - **“Note” (or “note-interval”) transcriptions** : C4-D4-E4-G3-G3, Up-Down-Repeat
  - **“Continuous” or “frame-based” pitch contour** : A smooth curve sampled at every 32ms, starting at 262 Hz, rising to 294 Hz, dropping to 392 Hz, and staying at 196 Hz

“Continuous” Dominant f_0 Contour



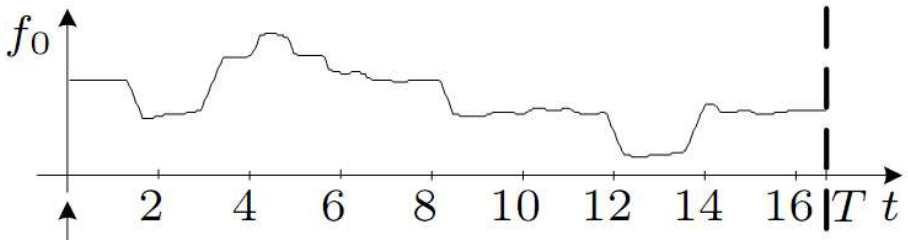
(a) Log-frequency spectrum



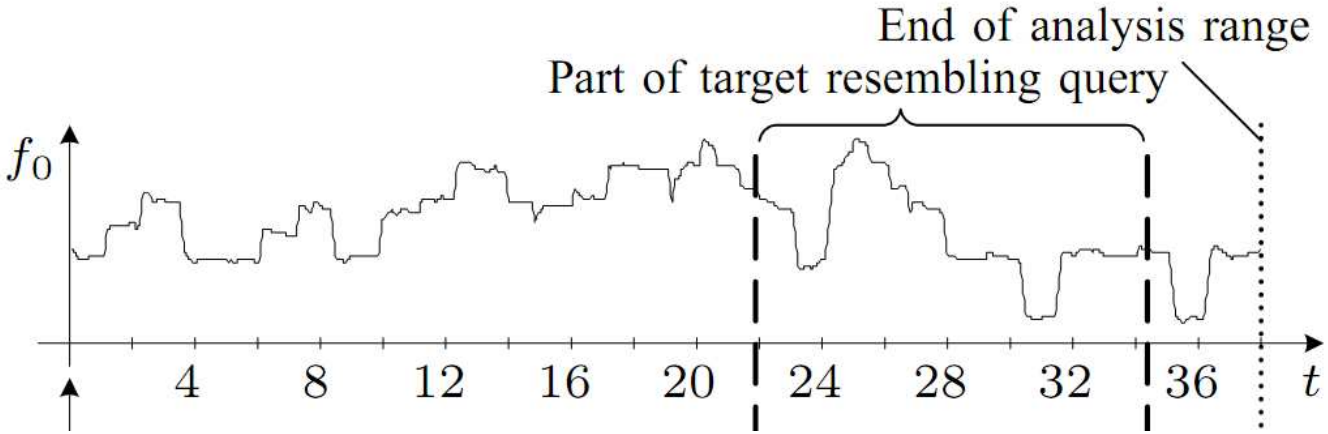
(b) Dominant f_0 contour

Content(Melody)-Based Music Search

- **Given an input query melody, search a set of target melodies to find the best match**



Query



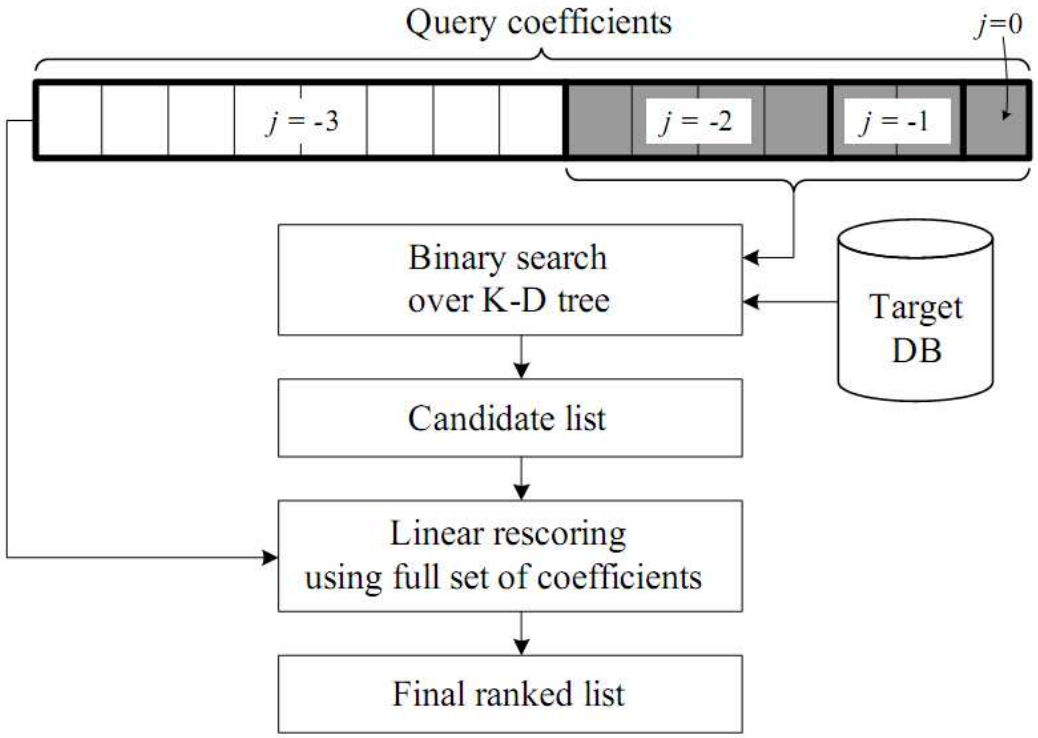
Target

Music Search

- Must be able to:
  - Search all possible starting locations of each target
  - Adjust for difference in speed (tempo)
  - Adjust for difference in key
  - Handle inconsistencies in rhythm and pitch
- **Dynamic Time Warping (DTW)**, fingerprinting, and hashing techniques have allowed efficient, effective search using **note transcription data**.
- **Continuous pitch contours** have been suggested to work better than note transcriptions [Mazzoni 01]
  - Note transcriptions can only be reliably obtained from monophonic music; with polyphonic music, note transcriptions easily break down
- The problem with continuous pitch contours

  → **Very computationally expensive**

Indexing Using Wavelet Coefficients



- Index melody fragments using a set of “time-normalized”, “key-normalized” wavelet coefficients.
- Use a K-D Tree to store the wavelet coefficients and search them efficiently
- Compares melodies too rigidly; Does not allow within-query changes in rhythm

Dynamic Time Warping

- Assume
  - Query sequence $Q = \{q_1, q_2, \cdots, q_{|Q|}\}$
  - Target Sequence $P = \{p_1, p_2, \cdots, p_{|P|}\}$
- Distance according to DTW with R warping operations is

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

$$d(\phi_q(i), \phi_p(i); b(i)) = [q\{\phi_q(i)\} + b(i) - p\{\phi_p(i)\}]^2$$

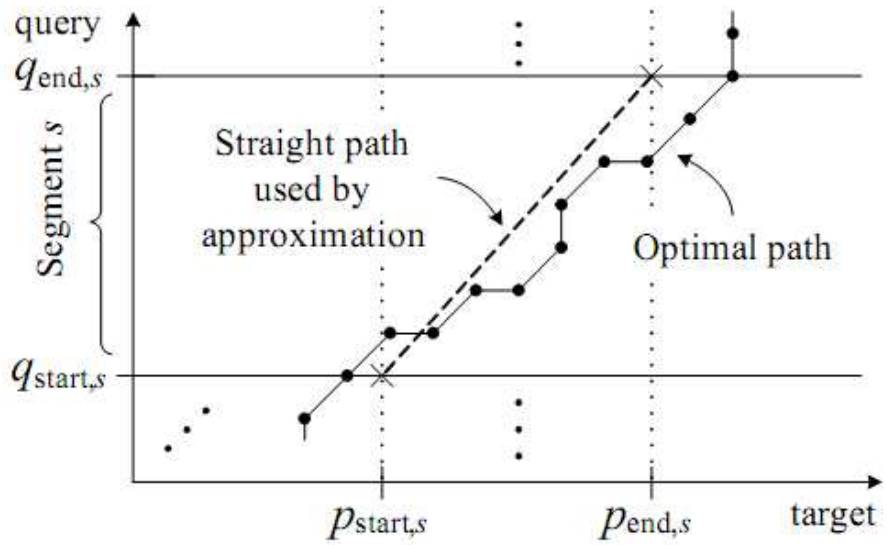
- $b(i)$ is a bias factor modeling the difference in musical key between Q and P. This can be constrained to remain roughly constant with respect to i

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

- Assuming $B = \{b_1, b_2, \cdots, b_{|B|}\}$, we essentially need to compute values over a $|Q| \times |P| \times |B|$ space

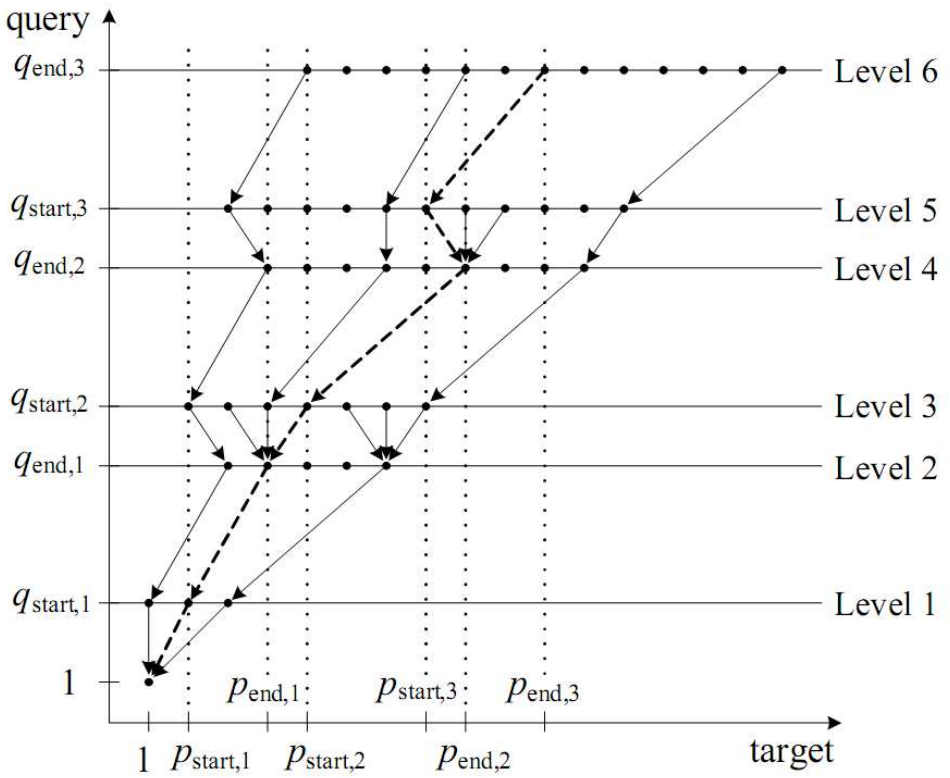
  → Huge Computation Cost

“Segmented” Dynamic Time Warping



- To speed up the DTW, partition the query into “segments”, treating each segment as a rhythmically consistent unit.
- This drastically reduces the amount of computation
- However, a number of mathematical issues exist that are specific to the music search problem.
- A framework is proposed for handling these issues

Implementation Using Level Building



- Because the partitioning may introduce too much rigidity into the system, “buffer zones” between query segments are employed to allow the matching target segments to overlap, providing greater flexibility in the DTW.
- A level-building scheme can be used to elegantly solve this problem.

Search Efficiency

TABLE III

MEAN AND STANDARD DEVIATION OF SEARCH TIME (S) PER QUERY FOR MIREX 2006 TEST SET. *THE MEAN SEARCH TIME FOR MQW+DTW IS AN ESTIMATION BASED ON COMPUTATIONAL COMPEXITY ANALYSIS.

	MTW	MQW	MQW+SDTW	MQW+DTW*
Mean (s)	0.15	0.48	0.64	154*
Std. (s)	0.053	0.27	0.27	N/A

- Parameters
  - *q* : length of query
  - *p* : length of target
  - *n* : number of query segments
  - *b* : number of possible key shifts
- Asymptotic search cost (does not include initial location search)
  - MTW (Multiscale Target Windowing) : O(1)
  - MQW (Multiscale Query Windowing) : O(*n*)
  - SDTW (Segmented DTW) : O(*np*)
  - DTW (“Brute-Force” DTW): O(*pqb*)

Experimental Results from MIREX 2006 Test

TABLE I
RESULTS FOR MIREX 2006 TEST SET

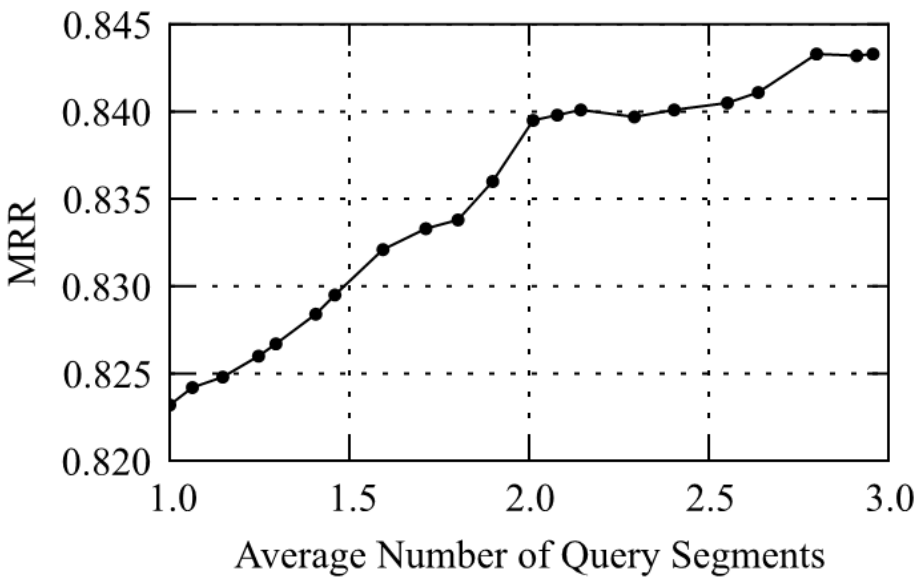
Method	MRR	Top-n Hit Rate (%)				
		1	3	5	10	20
MTW	0.789	75.7	80.6	82.7	84.8	86.7
MQW	0.818	79.2	83.4	85.0	86.5	88.7
MQW+SDTW	0.838	82.4	84.5	85.3	86.2	88.7

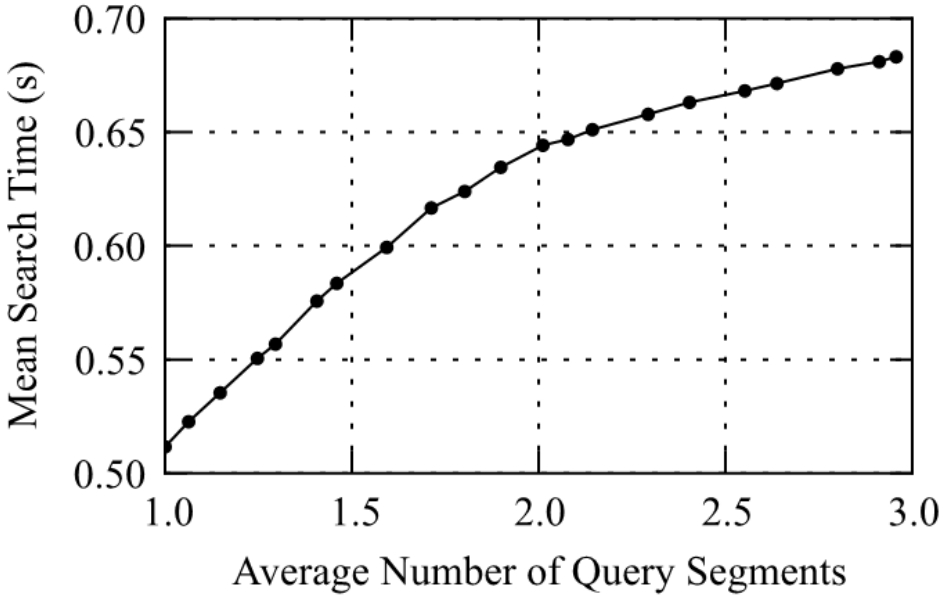
(MRR=0.921 when constraining all queries to start at the beginning of melodies)

TABLE II
RESULTS FOR POLYPHONIC MUSIC SET

Method	MRR	Top-n Hit Rate (%)				
		1	3	5	10	20
MTW	0.440	38.2	48.0	48.0	55.9	60.8
MQW	0.473	42.2	50.0	52.9	56.9	60.8
MQW+SDTW	0.500	47.1	51.0	53.0	57.8	60.8

MRR, Search Time vs. Number of Segments





- As the number of segments increase, the segmented DTW becomes closer to the “optimal” DTW, and both the MRR and mean search time increase.

The End

- Questions?