

A SYSTEM FOR RECOGNITION OF HUMMED TUNES

Tom Brøndsted, Søren Augustensen, Brian Fisker, Christian Hansen, Jimmy Klitgaard, Lau W. Nielsen, Thomas Rasmussen

Center for PersonKommunikation,
Institute of Electronic Systems
University of Aalborg
tb@cpk.auc.dk

ABSTRACT

This paper describes the prototype of a system for recognition of hummed tunes. The prototype takes as input an unknown hummed tune, turns it into segments representing the distinct notes of the tune, detects the pitch of the segments and turns them into a simple note-like symbolic representation to be matched against a reference database of known tunes. The system returns a list of tunes sorted according to similarity to the hummed tune. Each entry in the list contains information like title, composer, author, and a link to a midi-file enabling the user to ensure himself that he is reading the data of the song that he has hummed.

Testing with seven persons humming 46 songs in total, has shown a successful recognition rate of 75.6% with a search space of 39925 songs.

1. INTRODUCTION

The system described in this paper can be considered an electronic counterpart to encyclopedic books like the classical "A Dictionary of Musical Themes" by Barlow & Morgenstein [1] or "The Directory of Tunes and musical Themes" by Denys Parsons [2] allowing users to look up musical themes via note-like textual representations. However, rather than providing the user with actual information about a huge number of musical themes, the system is concerned with the exploration of core technologies involved when the textual representation is replaced by a hummed sound signal: Segmentation, pitch extraction, encoding, and pattern matching. Consequently, the database only contains a small number of "real" musical themes (songs) whereas the rest has been randomly generated to simulate a large search space.

A fully functional prototype has been established. However, the prototype has the following limitations:

- (1) The user has to hum "baba" to ease segmentation (which utilizes the closure phase of 'b').
- (2) He/she must hum the entire tune (an entire stanza of a song)
- (3) The pattern matching algorithm has not yet been optimized for speed.

An experimental version has been developed not requiring the input signal to match an entire reference pattern (allowing the user just to hum "the beginning" of a tune).

2. PROTOTYPE

On the face of it, the problem of recognizing a hummed tune can be viewed as a matter of extracting the pitch contour of the tune and compare it to the contours of a set of known tunes using pattern matching algorithms such as Dynamic Time Warping. However, for a number of reasons this solution is not considered optimal for the application being discussed. The problem of maintaining and enlarging the reference database (i.e. the set of known tunes) is very time-consuming if each tune has to be hummed (perhaps even by a number of representative singers), recorded, and converted to pitch contours. Further, the system may become very sensitive to variations caused by a user not humming perfectly in tune or humming very slowly or very fast. And finally, the direct comparison of pitch contours requires an extremely powerful computer, especially if the reference database is large.

If the pitch contour can be represented in another, more simple way, it will reduce the time used for pattern matching. This can be accomplished by using an encoder to transform the pitch contours into a format conforming to the notes found in a traditional music score or even simpler: the length (value) of each note may be disregarded so that the format only captures the sequence of note qualities in the tune. This simplified note format is in fact used for textual lookup in the dictionary by Barlow & Morgenstein [1] mentioned above. Further simplifications are found in music dictionaries like Parsons' book [2] (known as the "up-down book"!) where the pitch quality of each note is stated relatively to the previous one as "up" or "down" or "same". This latter dictionary addresses ordinary users, whereas the first one presupposes some knowledge in music theory (music reading, transposing themes etc.).

An encoding scheme transforming the pitch contours into sequences of "symbols" solves the problems mentioned above, however introduces a new one: information omitted in the simplifying scheme may be non-redundant causing two or more different tunes to be assigned the same sequence of note-like symbols. However, so far this problem is considered to be of merely theoretical nature. The two dictionaries mentioned above document that the simplified scheme works in practice. Further, as the system employs an n-best pattern matching algorithm

(similar to Web search engines like Google or AltaVista) the user will hardly ever experience the problem as a limitation.

2.1. Architecture

The architecture of the prototype is shown in Figure 1 below.

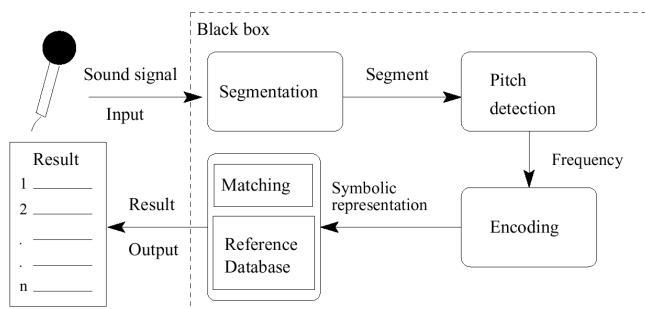


Figure 1. Architecture of prototype

The sound signal is segmented into notes each of which is assigned a pitch value used for the encoding into a symbol. The sequence of symbols is matched against a reference database resulting in a scored n -best list.

2.1.1. Segmentation

Segmentation into notes is based on a dynamically calculated threshold applied to the smoothed amplitude contour of the signal. As shown in Figure 2 the segmentation removes the less sonorous (unvoiced) parts of the signal (the closure-phase of 'b' and pauses) passing only voiced segments ('a's) on to the pitch detection¹:

The segmentation scheme works well and only produces few insertions and deletions provided that the Signal to Noise Ratio is sufficiently high and the user hums on "ba" as instructed by the system. In fact, informal tests give evidence that users can hum on any plosive (b,d,g,p,t,k) followed by any vowel, whereas humming on a sonorous consonant+vowel ("lala") or especially singing the actual words of the tune result in a decreasing performance.

Alternative segmentation methods based on the pitch contours (i.e. applied after pitch detection) allowing a more free humming mode have been considered. The major drawback of these alternatives is their inability to segment many repetitions of the same note (e.g. as found in the refrain of "Jingle Bells"). In such cases, segmentation based on the amplitude contour is unavoidable.

¹ Note that the sound 'b' like 'd' and 'g' are unvoiced in Danish whereas their English counterparts are voiced. The system reported in this paper may be fine-tuned for Danish users, though we doubt that it would perform significantly worse if tested on English ones.

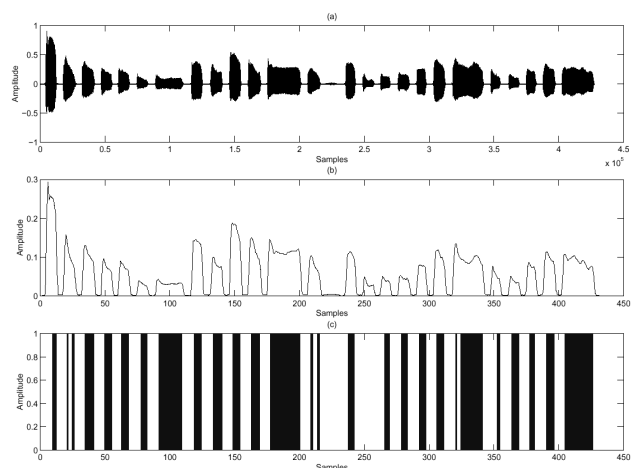


Figure 2. Segmentation module: (a) waveform, (b) smoothed amplitude contour, (c) detected notes (colored areas)

2.1.2. Pitch detection

A number of pitch detection algorithms have been explored including both time domain based algorithms (zero crossing, autocorrelation, and the average magnitude difference function) and frequency domain methods (Fourier transform, harmonic product spectrum, power spectrum). In the literature, the various methods are discussed mainly in the context of speech processing and the problem of extracting so-called intonation contours [3] [4]. However, many problems encountered in speech like lanryngelazation ("creaky voice") hardly ever occur in hummed sound signals and we have not observed the typical halving/doubling errors in the pitch detection module utilized in the present system. We ascribe this to the following reasons:

- (1) Most intonation classification systems operates with only *two* states of the glottis, voiced (regular vibrations of the vocal folds) and unvoiced (no vibrations) whereas real speech has at least a third state: *irregular* vibrations of the vocal folds. We feel that this third state which causes many problems experienced by pitch-extraction algorithms is rare in normal hummed signals².
- (2) In the present system, the segmentation module (sec. 2.1.1) replaces the voice detector of an intonation classification module. However, whereas it is critical if a voice detector disregards too much of the signal, the segmentation module has been optimized for only passing cleanly voiced segments on to the pitch detection. Hence, we do not encounter the problems found in the transitions between voiced and unvoiced segments.

² Danish has a glottal stop ("stød") mostly realized as irregular vibrations of the vocal folds (as in "she eats" spoken fluently and yet opposed to "sheets"). It is well known that this prosodic feature cannot be pronounced when singing and that good song texts avoid words with "stød". For further discussion, see [5].

As we don't think the choice of pitch detection algorithm is very critical in the system described, we have chosen a standard approach based on autocorrelation. The autocorrelation of a block of sampled data is calculated as

$$R_k = \sum_{i=1}^{N-k} S_i S_{i+k}, \quad k = 0, 1, 2, \dots, N - 1 \quad (1)$$

where R_k is the correlation function for the k th lag, N is the number of data points in the block and S_i is the i th point (cf. [6]).

2.1.3. Encoding

The prototype of the system utilizes the simple DUR-representation found in books like [2], i.e. each segment (note) is classified as either D (down), U (up), R (repeated) denoting whether its pitch quality is lower, higher or the same as the previous one, see Figure 3C. The threshold is set to a semitone.


- (A) 
- (B) cdeccecefegefegggagfecgagfcccGccGcc
- (C) *UUDRUUDUUUDUURUDDDDUDDDDRDURDU
- (D) *UUDRUUDUUUDUURUDDDDUDDDDRDURDU
*RRRRRRRRRLSRLSRRRLRSRRRLRRRLSRL

Figure 3: "Brother John" in the notation of (A) a traditional score, (B) textual notes in the style of [1], (C) simplified contour symbols in the style of [2], and (D) simplified contour symbols enriched with simplified duration symbols.

An alternative representation taking more precise intervals into account has been considered: unison, minor/major second up or down, minor/major third up or down etc. Such intervals can be derived automatically from the Barlow & Morgenstern style of representation (Figure 3B). However, so far experiments have only been conducted with simple enrichment of the DUR-representation with duration information. The experimental version being developed also classifies the value (length) of the notes using the simple scheme: L (longer), S (shorter), R (repeated value), cf. the notation style of Figure 3D.

The crucial question when modifying the encoding and representation scheme is, of course, how to modify the local cost function of the matching routine (see below).

2.1.4. Matching

The matching module compares the DUR representation generated by the encoder to the DUR representations of known songs in the reference database using a Dynamic Programming (DP) algorithm. The DP scheme is very similar to Minimal Edit Distance algorithms or the Dynamic Time Warping (DTW) algorithms typically used in speaker dependent speech recognition [7] [8] [9]. The encoded representation is considered

a "noisy" version of a reference pattern and the module finds the n -best matches with the lowest distance paths aligning the input pattern to the reference templates. The results are presented to the user in the form of a scored list of song titles. Each title has a link to a midi-file allowing the user to check if the correct tune has been found.

The DP algorithm operates on the DUR representation passed on by the encoder and, in turn, each of the DUR template patterns found in the database. A local cost matrix is computed calculating the distance between each symbol of the encoding vector and each symbol in the reference vector. The global path-cost calculation is based on equation (2):

$$D_A(i, j) = \min_{(i', j')} [D_A(i', j') + d((i', j'), (i, j))] \quad (2)$$

stating that the global cost of a given node $D_A(i, j)$ in the matrix is equal to the minimum of the global cost at the previous node $D_A(i', j')$ plus the cost of moving from that node $d((i', j'), (i, j))$. The function d is defined as the local cost of the current node (d_N) times the cost of the transition (d_T) i.e. $d_N(i, j) \times d_T((i', j'), (i, j))$, where the d_T function denotes the best of three transitions as shown in Figure 4.

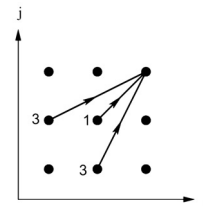


Figure 4: Path-cost for different transitions

The three transitions conform to the insertion, substitution, and deletion operations of a Minimal Edit Distance algorithm.

The algorithm can easily be modified to allow the test pattern just to match "the beginning" of a reference pattern. Assuming that users may typically stop at specific notes in a tune, e.g. after a bar in the notation of a traditional score (but *not* within a bar), we would simply have to extend the notation of template patterns with markers at possible end notes (e.g. at note 4, 8, 11, 14 etc. in Figure 3). However, matching against small pattern fragments obviously magnifies the problem of ambiguity addressed in the section 2. This approach presupposes a more detailed notation style than the DUR-representation used in the prototype (e.g. styles conforming to Figure 3B or 3D).

Experiments have been conducted with the local cost function (comparable to the Euclidean distance calculation in standard DTW speech recognition) assuming that the D/U cost should be higher than D/R and U/R; however, the test results reported below are produced by the prototype that only examines if the two symbols being compared are equal or not.

Likewise, preliminary experiments with enriched notation styles conforming to Figure 3B and 3D have been conducted. The number of semitones dividing two intervals is the obvious local

cost measure in case of 3B. In case of 3D, local cost functions for duration and melodic contours can be weighted.

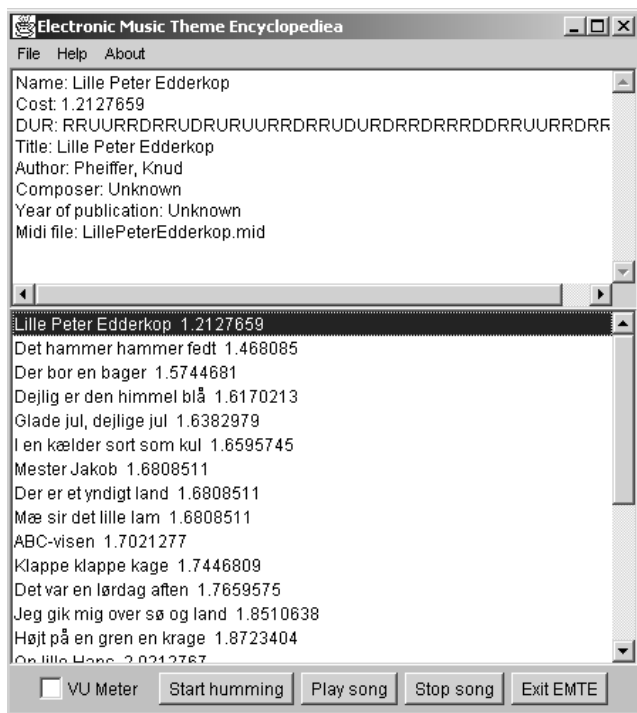


Figure 5: User Interface of prototype with an n -best list in the lower frame and the data of the selected song in the upper frame.

3. TEST AND EVALUATION

The prototype has been implemented and tested using a database of 46 songs hummed on “ba” by 4 female and 3 male test persons. The part of the database used during the implementation and tuning of the system has been kept apart from the data used for testing. The reference database consists of 25 DUR representations of real songs supplemented by a number of pseudo songs generated by a random function. The generated songs are all between 10 and 40 symbols long (like the 25 real songs). The 25 DUR representations of real tunes have been derived semi-automatically from available midi-files³. The prototype has a successful recognition rate of 75.6% with a search space of 39925 songs when the presence of the correct tune under the top-5 candidates is scored as a correct recognition.

Informal tests indicate that users can easily adjust themselves to the system and hum very distinct notes. If the test persons had been instructed to hum very distinct “ba”s forgetting their

³ Note that it is far from trivial to derive melodic templates from multi-tracked midi-files automatically. However, as far as this is possible the methods discussed in the present paper can also be viewed as core technologies of a general, web-based midi-file finder.

“musical appreciation”, the recognition rate would undoubtedly have been significantly better.

4. CONCLUSION

The most obvious components of the system to improve are: (1) The encoding (including classification of node intervals in the symbolic representation) and (2) the pattern matching (modification of the local cost function; modifications of the DP allowing users just to hum “the beginning” of a tune, optimization for speed). These improvements are taken into account in a new version being designed.

The user interface of the prototype is minimalistic (Figure 5) centered round the microphone input. Users may experience it a limitation that they have to adapt to the system and hum distinct “ba”s. However, this limitation is compensated for by the simplicity, reliability, and robustness of the system.

5. REFERENCES

- [1] Barlow, S Harold and Morgenstern, S.: *A Dictionary of Musical Themes*. Crown Publishers New York, 1948 (revised and reprinted Faber & Faber 1983)
- [2] Parsons, Denys: *Directory of Tunes and Musical Themes*, Spencer Brown 1975
- [3] Rabiner, L.R. and Schafer, R.W.: *Digital Processing of Speech Signals*. Prentice-Hall, 1st edition, 1978
- [4] Hess, W.: *Pitch Determination of Speech Signals*. Springer-Verlag, 1983
- [5] Brøndsted, T.: Intonation Contours “distorted” by Tone Patterns of Stress Groups and Word Accents. In: A. Botinis, G. Kouroupetoglou, G. Carayiannis (eds.): *Intonation: Theory, Models and Applications*, Athens (Athanasopoulos) 1997
- [6] Kendall, W.B.: A New Algorithm for Computing Correlations. *IEEE Transactions on Computers*, January 1974
- [7] Deller, Jr. John R., Hansen, John H.L., Proakis, John G. *Discrete-Time Processing of Speech Signals*, chapter 11, pages 623-634. IEEE Press, 1993
- [8] Rabiner, L. R. , Juang, B.H. *Fundamentals of Speech Recognition*, chapter 4 and 7, pages 200-232 and 416-420. Prentice-Hall, 1993
- [9] Wrigley, Stuart N. *Speech Recognition by Dynamic Time Warping*, 1998. URL <http://www.dcs.shef.ac.uk/~stu/com326>. Visited April 26 th 2001