MELODY SPOTTING IN RAW AUDIO RECORDINGS USING VARIABLE DURATION HIDDEN MARKOV MODELS

Aggelos Pikrakis, Sergios Theodoridis

Department of Informatics and Telecommunications, University of Athens
Panepistimioupolis, TYPA Buildings, 15784, Athens, Greece
phone: + (30) 2107275363, fax: + (30) 2107275337, email: {pikrakis,stheodor}@di.uoa.gr
web: www.di.uoa.gr/dsp

ABSTRACT

This paper presents a melody spotting system based on Variable Duration Hidden Markov Models (VDHMM's), capable of locating monophonic melodies in a database of raw audio recordings. The raw audio recordings may either contain a single instrument performing in solo mode, or an ensemble of instruments where one of the instruments has a leading role. The melody to be spotted is treated as a pattern and is first converted into a sequence of note durations and music intervals. Based on this representation, a VDHMM is constructed. For each raw audio recording in the database, a sequence of note durations and music intervals is extracted by means of a multipitch tracking algorithm. These sequences are subsequently fed as input to the VDHMM that models the melody to be located. The VDHMM employs an enhanced Viterbi algorithm, previously introduced by the authors, in order to account for pitch tracking errors and performance improvisations of the instrument players. It then suffices to post-process the best-state sequence generated by the enhanced Viterbi algorithm in order to locate occurrences of the melody in question. Our method has been successfully tested with a variety of cello recordings in the context of Western Classical music, as well as with Greek traditional multi-instrument recordings where clarinet has a leading role.

1. INTRODUCTION

Melody spotting can be defined as the problem of locating occurrences of a given melody in a set of music recordings. Depending on the origin and representation of the melody to be spotted, as well as the nature of the music recordings to be searched, several variations of the melody spotting problem can be encountered in the literature. For example, the melody serving as the "query data" may originate from a whistled or hummed tune [1], or from a MIDI keyboard [2]. The set of music recordings to be queried may consist of MIDI data [3], monophonic audio recordings [2] or polyphonic audio recordings [3]. Despite these variations most research effort has focused on the application of standard Hidden Markov Models to solve the above tasks [1], [2], [4].

In our approach, the melodic contour to be spotted is first transformed to a sequence of note durations and music intervals. Such a transformation is not restrictive, because it can be easily computed from MIDI data or a printed score. The resulting sequence is subsequently treated as a *pattern* and a Variable Duration Hidden Markov Model (VDHMM) is built in order to model it. Using VDHMM's makes it possible to account for variability of note durations and also permits to model variations of the pattern's sequence of music intervals. The resulting VDHMM is then fed with a feature sequence of note durations and music intervals that has been extracted from *a raw audio recording* by means of a multi-pitch tracking analysis model. We have focused on multi-pitch tracking algorithms because we want to treat in a unified manner both single-instrument recordings and multi-instrument recordings where one of the instruments has a leading role. The VDHMM generates a *best-state sequence by means of an enhanced Viterbi algorithm* which has been previously introduced by the authors [5]. The enhanced Viterbi algorithm is able to deal with pitch tracking errors stemming from the application of the multi-pitch algorithm to the raw audio recordings. Once the best-state sequence is generated, it can be further processed by a simple parser in order to locate instances of the musical pattern.

The novelty of our approach consists of the following:
a) a VDHMM is being employed to such problem for the first time, providing a noticeably enhanced performance in the system. This is because VDHMM allows the use of a robust, non-standard cost function for the Viterbi algorithm it presents.
b) A unified treatment of both monophonic and non-monophonic raw audio data.

Section 2 presents the pitch tracking procedure that is applied to the raw audio recordings. Section 3 describes the methodology with which the VDHMM is built in order to model the musical pattern. Section 4 describes the enhanced Viterbi algorithm and the post-processing stage that is applied on the best-state sequence. Implementation and experiment details are given in Section 5 and finally conclusions are drawn in Section 6.

2. FEATURE EXTRACTION FROM RAW AUDIO RECORDINGS

The goal of this stage is to convert a raw audio recording into a sequence of music intervals without discarding note durations. As it will be later explained, the use of music intervals ensures invariance to transposition of melodies, while note durations preserve information related to rhythm. This type of intervalic representation is an option between other standard music representation approaches (e.g. [6]).

At first, a sequence of fundamental frequencies is extracted from the raw audio recording. For the task of fundamental frequency tracking, we used Tolonen's multipitch analysis model [7]. Tolonen's method splits the audio recording into a number of frames by means of a moving window technique and extracts a set of pitch candidates from each frame. In our experiments, we always choose the strongest pitch candidate as the fundamental frequency of the frame. For single instrument recordings, this is the obvious choice, however for audio recordings consisting of an ensemble of instruments, where one of the instruments has a leading role, this choice does not guarantee that the extracted fundamental frequency coincides with the pitch of the leading instrument. Although this can distort the extracted sequence of fundamentals, such errors can be dealt with by the enhanced Viterbi algorithm of Section 4.

Without loss of generality, let $\mathbf{F} = \{f_1, f_2, \dots, f_N\}$ be the sequence of extracted fundamentals, where N is the number of frames that the audio recording is split into. Each fundamental frequency is in turn quantized to the closest half-tone frequency on a logarithmic frequency axis and, finally, the difference of the quantized sequence is calculated. The frequency resolution adopted at the quantization step can be considered as a parameter to our method, i.e., it is also possible to adopt quarter-tone resolution, depending on the nature of the signals to be classified. For micro-tonal music, as is the case with Greek Traditional Music, quarter-tone resolution is a more reasonable choice.

Therefore, in order to imitate certain aspects of the human auditory system, which is known to analyze an audio pattern on a logarithmic frequency axis, each f_i is mapped to a positive number, say k, equal to the distance (measured in half-tone units) of f_i from f_s (the lowest fundamental frequency of interest, e.g. $A1 = 55Hz$), i.e., $k = round(12\log_2 \frac{f_i}{f_s})$, where $round(\cdot)$ denotes the roundoff operation. As a result, sequence $\mathbf{F}$ is mapped to the sequence $\mathbf{L} = \{l_i;\ i = 1,\ldots,N\}$, where $l_i \in [0, l_{max}]$ (l_{max} is some maximum value).

It is now straightforward to compute $\mathbf{D}$, the sequence of music intervals (frequency jumps) and note durations, from sequence $\mathbf{L}$. This is achieved by calculating the difference of $\mathbf{L}$, i.e.,

$$\mathbf{D} = \{d_i = l_{i+1} - l_i;\ i = 1,\ldots,N-1\}$$

We assume that the d_i's fall in the range $[-G, G]$, where G is the maximum allowable music interval. In the rest of this paper, we will refer to d_i's as "symbols" and to $\mathbf{D}$ as the "symbol sequence". This is because the range of values of the d_i's can be considered as an alphabet of $2G+1$ discrete symbols.

It is worth noticing that, most of the time, l_{i+1} is equal to l_i, since each note in an audio recording is very likely to span more than one consecutive frames. As a result, $d_i = 0$ for most of the frames (i's). Therefore, in the general case, we can rewrite $\mathbf{D}$ as

$$\mathbf{D} = \{\mathbf{0}_{z_1}, m_1, \mathbf{0}_{z_2}, m_2, \ldots, \mathbf{0}_{z_{N-1}}, m_{N-1}, \mathbf{0}_{z_N}\} \qquad (1)$$

where $\mathbf{0}_{z_k}$ stands for z_k successive zeros (i.e., zero valued d_i's) and each m_i is a non-zero d_i. The structure of the sequence $\mathbf{D}$, as shown in equation (1), reveals the fact that $\mathbf{D}$ can actually be considered to consist of *subsequences of zeros* separated by *non-zero values* (the m_i's), with each m_i denoting a music interval, i.e., *the beginning of a new note*. The physical meaning of a subsequence of zeros is that it represents a steady musical note. The length of the subsequence, measured in frames, is actually the *note duration*.

3. MODELING THE MELODY TO BE SPOTTED BY MEANS OF A VARIABLE DURATION HMM

We now turn our attention to the representation of the melody to be spotted. Following the approach adopted in Section 2, the melody will, also, first be mapped to a sequence of music intervals and note durations. Let $\mathbf{M_s} = \{(fr_1, t_1), (fr_2, t_2), \ldots, (fr_M, t_M)\}$ be a melody consisting of M notes, where for each pair (fr_i, t_i), fr_i is the pitch of the $i-th$ note (measured in Hz) and t_i is the respective note duration (measured in seconds). Each fr_i can also be quantized to the closest half-tone frequency, say lr_i. As a result, $\mathbf{M_s}$ is mapped to the sequence $\mathbf{L_s} = \{(lr_i, t_i);\ i = 1,\ldots,M\}$ where lr_i lies again in the range 0 to l_{max}. In addition, the $i-th$ note duration is mapped to a sequence of z_i zeros, say $\mathbf{O}_{z_i}$, where $z_i = round(t_i/step)$, with *step* being the step of the moving window technique that was used for the raw audio recording (measured in seconds). $\mathbf{M_s}$ can now be written, following equation (1), as

$$\mathbf{D_s} = \{\mathbf{0}_{z_1}, mr_1, \mathbf{0}_{z_2}, mr_2, \ldots, \mathbf{0}_{z_{M-1}}, mr_{M-1}, \mathbf{0}_{z_M}\} \qquad (2)$$

where $mr_i = lr_{i+1} - lr_i$. Taking $\mathbf{D_s}$ as a starting point, a VDHMM is built for the melody to be spotted. Specifically:

- One state is created for each subsequence of zeros O_{z_k}, $k = 1,\ldots,M$. These are the Z-states, $Z_1,\ldots,Z_M$. Each Z-state only emits zeros with probability equal to one.
- For each mr_i, $i = 1,\ldots,M-1$, a separate state is created. These are the S-states, $S_1,\ldots,S_{M-1}$. Each S-state only emits the respective mr_i with probability equal to one.
- The state duration for each Z-state is modeled by a Gaussian probability density function, namely, $p_{Z_i}(\ ) = \mathscr{G}(\ ,\ _{Z_i},\ ^2_{Z_i})$. The values of $\ _{Z_i}$ and $\ _{Z_i}$ depend on the allowable tempo fluctuation and time elasticity, due to performance variations of the instrument players. By adopting different Z-states, we allow a different state duration model for each note, something that is dictated by the nature of real world signals.
- Although this HMM is expected to emit only one symbol each time an S-state is visited, it is still useful, for reasons that will be explained below (see section 4), to adopt a Gaussian probability density function for each S-state as well, namely, $p_{S_i}(\ ) = \mathscr{G}(\ ,\ _{S_i},\ ^2_{S_i})$.
- This is a left-to-right model, where each Z-state, Z_i, is followed by an S-state, S_i, and each S_i is definitely followed by Z_{i+1}. It must pointed out that, according to this approach, each note of the melody corresponds to a pair of states, namely an S-state followed by a Z-state, with the exception, of course, of the first note (Figure 1).

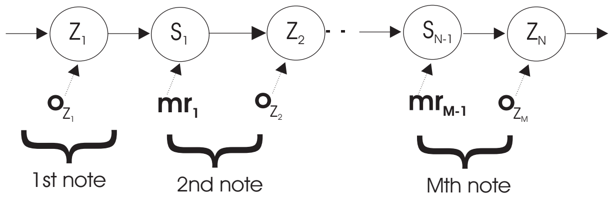

Figure 1: Mapping melody to a VDHMM

- A third type of state is added, both in the beginning and in the end of the VDHMM of Figure 1, which we call the *end-state*. Each end-state is allowed to emit any music interval (symbol) with equal probability. If the end states are named E_1 and E_2, the successor to E_1 can be either Z_1 or $E2$ and E_2 is now the rightmost state of the model. Furthermore the following state transitions are allowed to take place: $E_1 \rightarrow Z_1$, $E_1 \rightarrow E_2$ and $E_2 \rightarrow E_1$. The state duration for the end states is modeled by a uniform probability density function with a maximum state duration equal to $\simeq 1$ seconds. This completes a basic version of the VDHMM (shown in Figure 2).

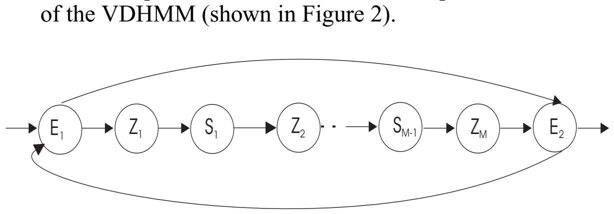

Figure 2: Basic version of the VDHMM

We have now reached the point where this basic version of the VDHMM can be used as a melody spotter. If the sequence of music intervals that has been extracted from the raw audio recording is fed as input to this HMM and the Viterbi algorithm is used for the calculation of the best-state sequence, the HMM is expected to iterate between the end-states, E_1 and E_2, until the melody is encountered. Then the HMM will go through the sequence of Z-states and S-states modeling the music intervals of the melody, will then jump to E_2 and will start again iterating between the end-states, until one more occurrence of the melody is encountered or the end of the feature sequence is reached. After the whole feature sequence is processed, the end-states can be removed from the best-state sequence and the remaining state subsequences (if any) will correspond to occurrences of the melody in the raw audio recording (this is equivalent to a simple parsing algorithm).

The VDHMM described so far only works for exact matches of the melody to be spotted in the raw audio recording, i.e. only note durations are allowed to vary according to the Gaussian pdf's that model the state duration. However, if certain state transitions are added, the VDHMM of Figure 2 can also deal with the cases of missing notes and repeating subpatterns. Specifically:

- Missing notes can be accounted for, if certain additional state transitions are permitted. Without loss of generality, we assume that no more than one successive note can be missing. Following the notation that we have so far adopted, if the i-th note is expected to be absent, then a transition from Z_{i-1} to S_i, denoted as $Z_{i-1} \rightarrow S_i$, should also be made possible. This is because the i-th note corresponds to the pair of states $\{S_{i-1}, Z_i\}$ and similarly, the (i+1)-th note starts at state S_i, whereas the (i-1)-th note ends at state Z_{i-1}.

- In the same manner, accounting for successive repetitions of a sub-pattern of the prototype, leads to permitting backward state transitions to take place. For instance, if notes $\{i, i+1, \ldots, i+K\}$ are expected to form a repeating pattern, then clearly, the backward transition $Z_{i+K} \rightarrow S_{i-1}$ must be added. This is again because the (i+K)-th note ends at state Z_{i+K}, whereas the i-th note starts at state S_{i-1}.

Furthermore, it is also possible to relax the constraint that each S-state emits only one symbol. This makes it possible to locate, in the raw audio recording, variations of the melody where certain music intervals are higher or lower than those defined in the melody. For example, state S_i may also emit symbols $mr_i + 1$ and $mr_i - 1$. This is desirable if one is unsure of the exact score of the melody to be searched, or if one wishes to locate with a single search variations of the melody. Following the above discussion, for a melody consisting of a sequence of M notes, the respective HMM consists of $S = 2 + M + M - 1 = 2M + 1$ states.

4. THE ENHANCED VITERBI ALGORITHM

Translated in the HMM terminology, let $\mathscr{H} = \{ \ , A, B, \mathscr{G}\}$ be the resulting variable duration HMM, where $_{Sx1}$ is the vector of initial probabilities, $A_{S\times S}$ is the state transition matrix and $B_{(2G+1)\times S}$ is the symbol probability matrix (G is the maximum allowed music interval). Regarding the $\mathscr{G}_{S\times 2}$ matrix, the first element of the i-th row is equal to the mean value of the Gaussian function modeling the duration of the i-th state and the second element of the i-th row is the standard deviation of the respective Gaussian. For the VDHMM of Figure 2, both Z_1 and E_1 can be the first state, suggesting that $(1) = \ (2) = 0.5$ and $(i) = 0$, $i = 3 \ldots S$. In addition, A is upper triangular with each element of the first diagonal being equal to one and all other elements of A have zero values, unless backward transitions are possible, as is the case when modeling repeating subpatterns. Finally, for the Z-states, each column of B has only one element with value equal to 1, $B_{Z_i}(d_s = 0) = 1$ (and all other elements are zero valued) and similarly, for each S-state, $B_{S_i}(d_s = mr_i) = 1$ and all other elements are zero valued, unless of course, a S-state is allowed to emit more than one music intervals.

In practice, sequence $\mathbf{D}$, which has been extracted from a raw audio recording, suffers from a number of pitch-tracking errors. Such errors are more frequent when dealing with multi-instrument recordings and appear as subsequences of symbols of $\mathbf{D}$ whose sum is equal to zero or to a mr_i of the pattern to be located (for a study of pitch-tracking errors see [5], [9]). If $\mathscr{H}$ employs a standard Viterbi algorithm for the calculation of the best-state sequence, a melody spotting failure will result, as $\mathscr{H}$ will only iterate between the end-states. This can be accommodated if the enhanced Viterbi algorithm that has been introduced by the authors in [5] is adopted. A detailed description of the algorithm can be found in [5]. Basically, the essence of this algorithm is to be able to account for all possible pitch-tracking errors (e.g. pitch doubling errors) by incorporating them in the cost function. In this paper, we will only summarize the equations for the calculation of the best-state sequence.

In order to proceed further, certain definitions must first be given. For a given observation sequence $\mathbf{D} = \{d_1 d_2 \ldots d_N\}$ and a discrete observation VDHMM $\mathscr{H}$, let us define the forward variable $a_t(j)$ as in [8], i.e.,

$$a_t(j) = P(d_1 d_2 \ldots d_t,\ state\ j\ ends\ at\ t|\mathscr{H}), j = 1 \ldots S \qquad (3)$$

that is $a_t(j)$ stands for the probability that the model finds itself in the j-th state after the first t symbols have been emitted. It can be shown that ([8]),

$$a_t(j) = \max_{1\leq \ \leq T, 1\leq i\leq S, i\neq j} [\ _t(i, \ ,j)] \qquad (4)$$

$$_t(i, \ ,j) = a_{t-}\ (i) A_{ij} p_j(\ ) \qquad \prod_{s=t- \ +1}^{t} B_j(d_s) \qquad (5)$$

where is the time duration variable, T is its maximum allowable value within any state, S is the total number of states, A is the state transition matrix, p_j is the duration probability distribution at state j and B is the symbol probability matrix. In other words, the probability of a path ending its state sequence at state j, depends on all possible ways to have reached state j, including the possibility of remaining at state j for successive time instances. We have already made the assumption that, p_j, follows a Gaussian probability density function. Equations (4) and (5) suggest that there exist $(S \times T - T)$ candidate arguments, $_t(i, \ ,j)$, for the maximization of each quantity $a_t(j)$. In order to retrieve the best state sequence, i.e., for backtracking purposes, the state that corresponds to the argument that maximizes equation (4) has to be stored in a two-dimensional array , as (j,t).
Therefore, $(j,t) = \arg\max[\ _t(i, \ ,j)], 1 \leq \ \leq T, 1 \leq i \leq S, i \neq j$
In addition, the number of symbols spent on state j is stored in a two-dimensional matrix c, as $c(j,t)$. We notice that in equation (4), each candidate argument, $_t(i, \ ,j)$, refers to symbols of the observation sequence and this is why the product $^{t}_{s=t- \ +1} B_j(d_s)$ is calculated in (5). If the value of $^{t}_{s=t- \ +1} d_s$ is equal to zero, this indicates a possible *pitch tracking error* cancellation. Thus, if these successive symbols add to zero, one must take into consideration that the symbols $d_{t- \ +1}, \ldots, d_{t-1}, d_t$, could be the result of a pitch tracking error, and must be replaced by a zero that lasts for successive time instances. Of course, since one cannot be sure that this cancellation is the correct action, it is left to the optimal option process to decide, by providing the cancellation option as an extra argument in the optimization. This is quantified by considering $(S\mathrm{x}T - T)$ additional $\hat{\ }$ arguments to augment equation (4), namely

$$\hat{\ }_t(i, \ ,j) = a_{t-}\ (i) A_{ij} p_j(\ ) \prod_{s=t- \ +1}^{t} B_j(d_s = 0) \qquad (6)$$

Therefore, for the Z-states equations (4) and (5) become

$$a_t(j) = \max_{1\leq \ \leq T, 1\leq i\leq S, i\neq j} [\ _t(i, \ ,j), \hat{\ }_t(i, \ ,j)] \qquad (7)$$

$$_t(i, \ ,j) = a_{t-}\ (i) A_{ij} p_j(\ ) \prod_{s=t- \ +1}^{t} B_j(d_s) \qquad (8)$$

$$\hat{\ }_t(i, \ ,j) = a_{t-}\ (i) A_{ij} p_j(\ ) \prod_{s=t- \ +1}^{t} B_j(d_s = 0), \qquad (9)$$

$$if \sum_{s=t- \ +1}^{t} d_s = 0$$

Thus, *maximization is now computed over all and $\hat{\ }$ quantities.* It is also worth noticing, that, if (7) is maximized by a $\hat{\ }$ argument, say $\hat{\ }_t(i, \ ,j)$, then the number of symbols spent at state j is equal to , as is the case with the standard VDHMM. In addition, for state j, state i is the winning predecessor state. If, in the end, it turns out that for some states of the best-state sequence, a symbol cancellation took place, it is useful to store this information in a separate two-dimensional matrix, s, by setting the respective $s(j,t)$ element equal to "1". Matrices and c are still used for backtracking purposes.

If $a_t(j)$ refers to an S-state, then a symbol summation is desirable, if the sum, $^{t}_{s=t- \ +1} d_s$ *is equal to the actual music interval associated with the respective S-state* of the VDHMM. In the same rationale as before, if this holds true, *the whole subsequence of symbols is treated as one symbol equal to the respective sum* and again, $(S\mathrm{x}T - T)$ additional $\hat{\ }$ arguments must be computed for $a_t(j)$, according to the following equation:

$$\hat{\ }_t(i, \ ,j) = a_{t-}\ (i) A_{ij} p_j(\ ) B_j(\sum_{s=t- \ +1}^{t} d_s) \qquad (10)$$

Therefore, for the S-states equations (4) and (5) become

$$a_t(j) = \max_{1\leq \ \leq T, 1\leq i\leq S, i\neq j} [\ _t(i,\ ,j), \hat{\ }_t(i,\ ,j)] \quad (11)$$

$$_t(i,\ ,j) = a_{t-}\ (i)A_{ij}p_j(\ ) \sum_{s=t-\ +1}^{t} B_j(d_s) \quad (12)$$

$$\hat{\ }_t(i,\ ,j) = a_{t-}\ (i)A_{ij}p_j(\ )B_j(\sum_{s=t-\ +1}^{t} d_s), \quad (13)$$

$$if B_j(\sum_{s=t-\ +1}^{t} d_s) > 0$$

Similar to the previous case, maximization is again computed over all   and $\hat{\ }$ quantities. If (11) is maximized by a $\hat{\ }$ argument, say $\hat{\ }_t(i,\ ,j)$, then the number of symbols spent at state j is equal to , as is the case with the standard variable duration model. In addition, state i is the predecessor of state j in the path. If a symbol summation took place, it is useful to store this information in the s matrix, that was previously introduced, by setting the respective $s(j,t)$ element equal to "1" (a zero indicates that no symbol summation took place). Matrices   and c are still used for backtracking purposes, as it is done in the standard VDHMM. The need to account for possible symbol summations reveals the fact that, although in the first place, the HMM was expected to spend one frame at each S-state, it turns out that a Gaussian probability density function must also be associated with each S-state. However, upon initializing the respective mean values and standard deviations for these Gaussians, T, the maximum allowable state duration, should have a smaller value for the S-states, compared to the Z-states.

5. EXPERIMENTS AND IMPLEMENTATION DETAILS

As it has already been mentioned, Tolonen's multipitch analysis model [7] was adopted as a fundamental frequency tracker for our experiments and certain parameter tuning was decided. Specifically: the length of the moving window was set equal to $50ms$ (each window was multiplied by a Hamming function) and a $5ms$ step was adopted between successive windows. This small step ensures that rapid changes in the signal are captured effectively by the pitch tracker, to the expense of increasing the length of the feature sequence. The pre-processing stage involving a pre-whitening filter was omitted. For the two channel filter bank, we used butterworth bandpass filters with frequency ranges $70Hz - 1000Hz$ and $1000Hz - 10KHz$. The parameter which controls frequency domain compression was set equal to 0.7. From each frame, the strongest candidate frequency returned by the model, was chosen as the fundamental frequency of the frame.

Our method was tested on two raw audio data sets: the first set consisted of *commercially available solo Cello recordings of J.S Bach's Six Suites for Cello (BWV 1007-1012)*, performed by seven different artists (namely Boris Pergamenschikow, Yo-Yo Ma, Anner Byslma, Ralph Kirshbaum, Roel Dieltiens, Peter Bruns and Paolo Beschi). The printed scores of these Cello Suites served as the basis to define (with the help of musicologists) a total of $\simeq 50$ melodies consisting of 3 to 16 notes. These melodies were manually converted to sequences of note durations and music intervals, following the representation adopted in Section 3. For the quantization step, half-tone resolution was adopted and an alphabet of 121 discrete symbols was used, implying music intervals in the range of $-60\ldots+60$ half-tones, i.e., $G = 60$. The duration of the Z-states of the resulting VDHMM's was tuned by permitting a 20% tempo fluctuation, in order to account for performance variations. The maximum state duration for the S-states was set equal to $25ms$. Depending on the pattern, e.g. for moving bass melodies, certain S-states were allowed to emit more than one music intervals, in order to be able to locate such pattern variations. The proposed method succeeded in locating approximately 95% of the pattern occurrences.

The second raw audio data set consisted of $\simeq 140$ *commercially available recordings of Greek Traditional music performed by an ensemble of instruments where Greek Traditional Clarinet has a leading role.* A detailed description of the music corpus can be accessed at http://www.di.uoa.gr/pikrakis/melody_spotter.html.

Due to the fact that Greek Traditional Music is micro-tonal, quarter-tone resolution was adopted (parameter G was set equal to 60 quarter-tones in this case). Although printed scores are not available for this type of music, following musicological advice, *we focused on locating twelve types of patterns that have been shaped and categorized in practice over the years in the context of Greek Traditional Music* (a description of the patterns can be found in [9]). These patterns exhibit significant time elasticity due to the improvisational attitude of the musicians and it was therefore considered appropriate to permit a 50% tempo fluctuation, when modeling the Z-states. In addition, the maximum state duration for the S-states was set equal to $50ms$, in order to account for the increased number of pitch-tracking errors due to the presence of multiple instruments. Our method successfully spotted 83% of the pattern occurrences. This reduced performance is mainly due to the fact that, despite the application of an enhanced Viterbi algorithm, the leading instrument's melodic contour can often be severely distorted in the extracted feature sequence of an audio recording, due to the presence of the accompanying instrument ensemble.

A prototype of our melody spotting system was initially developed in MATLAB and was subsequently ported to a C-development framework.

6. CONCLUSIONS

In this paper we presented a system capable of spotting monophonic melodies in a database of raw audio recordings. Both monophonic and non-monophonic raw audio data have been treated in a unified manner. A VDHMM has been employed for the first time as a model for the patterns to be spotted. Pitch tracking errors have been dealt with an enhanced Viterbi algorithm that results in noticeably enhanced performance.

REFERENCES

[1] Hsuan-Huei Shih et al., "Multidimensional Humming Transcription using a statistical approach for query by humming Systems," *Proceedings of ICASSP 2003*, 2003, Hong Kong.

[2] A. S. Durey and M. A. Clements, "Features for Melody Spotting Using Hidden Markov MOdels," *Proceedings of ICASSP 2002*, May 13-17, 2002, Orlando, Florida.

[3] M. Clause and F. Kurth, "A Unified Approach to Content-Based and Fault-Tolerant Music Recognition", *IEEE Transactions on Multimedia*, Vol. 6, No. 5, October 2004.

[4] A. S. Durey and M. A. Clements, "Melody Spotting Using Hidden Markov Models," *Proceedings of ISMIR 2001*, pp. 109-117, Bloomington, IN, October 2001.

[5] A. Pikrakis, S. Theodoridis and D. Kamarotos, "Classification of Musical Patterns using Variable Duration Hidden Markov Models", *Proceedings of EUSIPCO2004*, Sept. 2004, Vienna, Austria (also accepted for publication in the IEEE Transactions on Speech and Audio Processing).

[6] E. Cambouropoulos, "A General Pitch Interval Representation: Theory and Applications", *Journal of New Music Research*, vol. 25(3), September 1996.

[7] T. Tolonen and M. Karjalainen, "A Computationally Efficient Multipitch Analysis Model", *IEEE Transactions on Speech and Audio Processing*, Vol. 8(6), 2000

[8] L.R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proceedings of the IEEE*, Vol. 77, No. 2, 1989

[9] A. Pikrakis, S. Theodoridis, D. Kamarotos, "Recognition of Isolated Musical Patterns using Hidden Markov Models", *LNCS/LNAI 2445*, Springer Verlag, pp. 133-143, 2002.