A NOVEL HMM APPROACH TO MELODY SPOTTING IN RAW AUDIO RECORDINGS

Aggelos Pikrakis and Sergios Theodoridis
Dept. of Informatics and Telecommunications
University of Athens
Panepistimioupolis, TYPA Buildings
15784, Athens, Greece
phone: + (30) 2107275363
fax: + (30) 2107275337
{pikrakis, stheodor}@di.uoa.gr

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 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 presented to the system as a sequence of note durations and music intervals. In the sequel, this sequence is treated as a pattern prototype and based on it, a VDHMM is constructed. The probabilities of the associated VDHMM are determined according to a set of rules that account (a) for the allowable note duration flexibility and (b) with possible structural deviations from the prototype pattern. In addition, for each raw audio recording in the database, a sequence of note durations and music intervals is extracted by means of a multi pitch tracking algorithm. These sequences are subsequently fed as input to the constructed VDHMM that models the pattern 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. For each audio recording in the database, the best-state sequence generated by the enhanced Viterbi algorithm is further post-processed in order to locate occurrences of the melody which is searched. 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, in which clarinet has a leading role.

Keywords: Melody Spotting, Variable Duration Hidden Markov Models.

1 INTRODUCTION

Melody spotting can be defined as the problem of locating occurrences of a given melody in a database 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 practice. Most research effort has focused on comparing sung (or hummed) queries to MIDI data [1,2,3,4,5] in the context of the so-called "Query-by-Humming" systems. Such systems mainly employ Dynamic Time Warping techniques (variations of the Edit Distance) for melody matching, in order to account for pitch and tempo errors that are usually inherent in any real hummed tune.

In an effort to circumvent the need for MIDI metadata in the database, certain researchers have proposed using standard Hidden Markov Models for locating monophonic melodies in databases consisting of raw audio data. In [6] and [7] the database consists of recordings of a single instrument performing in solo mode, whereas in [8] the case of studio recordings of operas, that contain a leading vocalist, is treated. In [6–8], the input to the system is assumed to be a symbolic representation of the melody to be searched (e.g., a MIDI-like representation). This assumption leads to a different melody matching philosophy, when compared with "Query-by-Humming" systems. The term "Query-by-Melody" is often used in order to describe the functionality of systems like those proposed in [6–8].

In our approach, the melody to be spotted is also assumed to be available in a symbolic format, e.g., a MIDI like representation. This type of representation makes it possible to convert the melody to be searched to *a sequence of note durations and music intervals (time - music interval representation)*. This 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 (feature) sequences of note durations and music intervals that have been extracted from *the raw audio recordings* 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 in which one of the instruments has a leading role. For each feature sequence, the VDHMM generates a *best-state sequence by means of an enhanced Viterbi algorithm*, which has been previously introduced by the authors [9]. 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 a best-state sequence is generated, it can be further processed by a simple parser in order to locate instances of the musical pattern. For each detected occurrence of the melody in question, a recognition probability is also returned, thus allowing for sorting the list of results.

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, provided that in the non-monophonic case, an instrument has a leading role.

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 melody to be spotted. 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 each raw audio recording in the database to a sequence of music intervals without discarding note durations. 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. [10]).

At first, a sequence of fundamental frequencies is extracted from the audio recording using Tolonen's multi-pitch analysis model [11]. 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 efficiently dealt with by the enhanced Viterbi algorithm of Section 4.

Without loss of generality, let $\mathbf{F} = \{f_1, f_2, \ldots, f_N\}$, be the sequence of extracted fundamentals, where N is the number of frames into which the audio recording is split. 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 of Greek Traditional Music, quarter-tone resolution is a more reasonable choice.

Each f_i is then mapped to a positive number, say k, equal to the distance of f_i from f_s (the lowest fundamental frequency of interest, $A_1 = 55Hz$ in our experiments). For half-tone resolution, $k = round(12 \log_2 \frac{f_i}{f_s})$, where $round(\cdot)$ denotes the roundoff operation. As a result, $\mathbf{F}$ is mapped to sequence $\mathbf{L} = \{l_i; i = 1 \ldots N\}$, where $l_i \in [0, l_{max}]$. It is now straightforward to compute $\mathbf{D}$, the sequence of music intervals and note durations, from $\mathbf{L}$. 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 $d_i \in [-G, G]$, 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".

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. Therefore, we can rewrite $\mathbf{D}$ as

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

where $\mathbf{0}_{z_k}$ stands for z_k successive zeros and each m_i is a non-zero d_i. As a result, $\mathbf{D}$ consists 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 the duration of a musical note.*

3 MODELING THE MELODY TO BE SPOTTED BY MEANS OF A VDHMM

We now turn our attention to the representation of the melody to be spotted. Following the notation adopted in equation (1), the melody will also first be represented as a sequence of music intervals and note durations. Without loss of generality, let

$$\mathbf{M_p} = \{(fr_1, t_1), (fr_2, t_2), \ldots, (fr_M, t_M)\}$$

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). This time-frequency representation is not restrictive, as it can be computed in a straightforward manner from data stored in symbolic format (e.g., MIDI). Following the approach adopted in Section 2, each fr_i can also be quantized to the closest half-tone frequency, say lr_i. As a result, $\mathbf{M_p}$ is mapped to $\mathbf{L_p} = \{(lr_i, t_i); i = 1 \ldots M\}$, where $lr_i \in [0, l_{max}]$ and t_i is still measured in seconds. 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 also used for the raw audio recordings (measured

in seconds). $\mathbf{M_p}$ can now be written as

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

where $mr_i = lr_{i+1} - lr_i$. Taking equation (2) as a starting point, a VDHMM can now be built for the melody to be spotted. Before proceeding, it has to be noted that, with the exception of the first note of the melody (which has been mapped to a sequence of zeros), each note corresponds to a non-zero symbol followed by a sequence of zeros. The VDHMM is thus built according to the following set of rules:

(I) 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. Therefore, each note duration is modeled by a Z-state.

(II) The state duration for each Z-state is modeled by a Gaussian probability density function, namely, $p_{Z_i}(\tau) = \mathcal{G}(\tau, \mu_{Z_i}, \sigma^2_{Z_i})$. The values of μ_{Z_i} and σ_{Z_i} depend on the allowable tempo fluctuation and time elasticity, due to performance variations of the instrument players. By adopting different zero-states, we allow a different state duration model for each note, something that is dictated by the nature of real world signals.

(III) For each mr_i, $i = 1 \ldots M-1$, marking the beginning of a note, 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.

(IV) 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 a non-zero state followed by a zero-state, with the exception, of course, of the first note (figure 1).* In addition, for a melody consisting of a sequence of M notes, the respective HMM consists of $S = 2 + M + M - 1 = 2M + 1$ states.

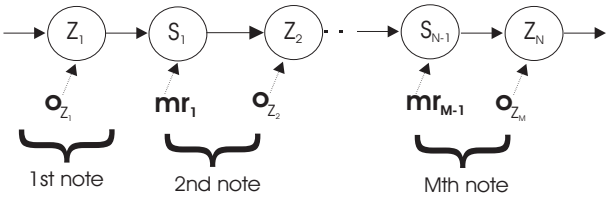

Figure 1: Mapping melody to a VDHMM

(V) 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), as well as zeros, 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. As a result, 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).

We have now reached the point where this basic version of the VDHMM can be used as a melody spotter. This is because, if the sequence of music intervals, that has been extracted from the raw audio recording (equation (1)), is fed as input to this VDHMM and the Viterbi algorithm is used for the calculation of the best-state sequence, the VDHMM is expected to iterate between the end-states, E_1 and E_2, until the melody is encountered. Then, the VDHMM will go through the sequence of Z-states and S-states modeling the music intervals of the melody, until it jumps 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.

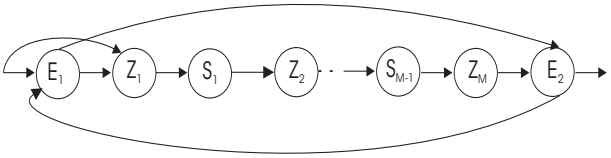

Figure 2: Basic version of the VDHMM

After the whole feature sequence of the raw audio recording is processed, a simple parser can post-process the best-state sequence and any state subsequences corresponding to occurrences of the melody can be easily located. This is because, whenever an instance of the melody is detected, the VDHMM will go through a sequence of states consisting only Z-states and S-states. It is therefore straightforward to locate such sequences of states with a simple parser (like in a simple string-matching situation).

The VDHMM described so far is only suitable 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 sub-patterns, by extending the aforementioned set of rules. Specifically:

(VI) Missing notes can be accounted for, if certain additional state transitions are permitted. For example, 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} (figure 3).

(VII) 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} (figure 3).

Missing notes and repeated sub-patterns are particularly useful to model, when dealing with music where improvisation of the instrument players is a common phenomenon, like in the case of Greek Traditional Clarinet performing a leading role while accompanied by an ensemble of instruments.

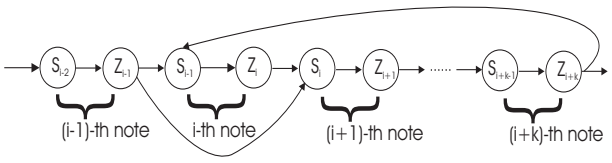

Figure 3: $Z_{i-1} \rightarrow S_i$ accounts for a possibly missing i-th note. $Z_{i+K} \rightarrow S_{i-1}$ accounts for a repeating sub-pattern of $k+1$ notes

Furthermore, it is also possible to relax the constraint that each S-state emits only one symbol, if one is unsure of the exact score of the melody to be searched, or if one wishes to locate variations of the melody with a single search. For example, state S_i could also be allowed to emit symbols mr_i+1 or mr_i-1.

4 THE ENHANCED VITERBI ALGORITHM

Translated in the HMM terminology, let $\mathcal{H} = \{\pi, A, B, \mathcal{G}\}$ be the resulting VDHMM, where π_{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 $\mathcal{G}_{S\times 2}$ 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 is the standard deviation of the respective Gaussian. For the VDHMM of figure (2):
(a) Both Z_1 and E_1 can be the first state, suggesting that $\pi(1) = \pi(2) = 0.5$ and $\pi(i) = 0$, $i = 3 \ldots S$.
(b) A is upper triangular with each element of the first diagonal being equal to one. All other elements of A have zero values, unless backward transitions are possible, as is the case when modeling repeating sub-patterns.
(c) 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 which case all allowable emissions can be set to be equiprobable).

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, where one of the instruments has a leading role.* This can be seen in figure (4), where pitch-tracking errors have been marked in circles. In the feature sequence of the audio recording, such errors are likely to appear as subsequences of symbols 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 [12]).

If $\mathcal{H}$ employs a standard Viterbi algorithm for the calculation of the best-state sequence, a melody spotting failure will result, as $\mathcal{H}$ will only iterate between the end-states. This can be accommodated if the enhanced Viterbi algorithm that has been introduced by the authors in [9] is adopted. In this paper, we will only summarize the equations for the calculation of the best-state sequence.

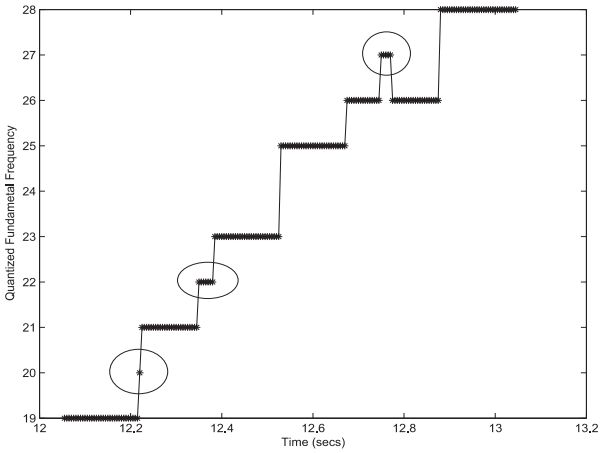

Figure 4: Pitch tracking results from an audio recording where a cello instrument performs in solo mode. Errors have been marked in circles

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 of the Viterbi algorithm.

As an example, consider the feature sequence $\mathbf{D_t} = \{\mathbf{0}_{z_1}, +1, \mathbf{0}_{z_2}, +1, \mathbf{0}_{z_3}, +1, \mathbf{0}_{z_4}, +1, \mathbf{0}_{z_5}, +2, \mathbf{0}_{z_6}, +1, \mathbf{0}_{z_7}, +1, \mathbf{0}_{z_8}, -1, \mathbf{0}_{z_9}, +2, \mathbf{0}_{z_{10}}\}$ of figure (4), which can be considered as a variation of the prototype $\mathbf{D}_p = \{\mathbf{0}_{z_{p1}}, +2, \mathbf{0}_{z_{p2}}, +2, \mathbf{0}_{z_{p3}}, +2, \mathbf{0}_{z_{p4}}, +1, \mathbf{0}_{z_{p5}}, +2, \mathbf{0}_{z_{p6}}\}$. If $\mathbf{D_t}$ is given as input to a VDHMM built for $\mathbf{D_p}$, a melody spotting failure will occur, which is clearly undesirable.

On the other hand, careful observation of $\mathbf{D}_t$ reveals that, m_7 (the 7th music interval), which is equal to 1 and m_8, which is equal to -1, cancel out. In addition, $m_1 + m_2 = 2$, *which is the respective music interval of the prototype pattern that is modeled by the VDHMM*. Similarly, $m_3 + m_4 = 2$ (*which is again the respective music interval of the prototype*).

These observations lead us to the idea that one can *enhance the performance of the VDHMM, by inserting in the model a mechanism capable of deciding which symbol cancellation/summations are desired.* For example, regarding sequence $\mathbf{D}_t$:
(a) if $+1$ and -1 are canceled out, the subsequence $\{\mathbf{0}_{z_7}, 1, \mathbf{0}_{z_8}, -1, \mathbf{0}_{z_9}\}$ can be replaced by a single subsequence of zeros, $\mathbf{0}_{z_7+z_8+z_9+2}$. This, in turn, suggests that if a modified version of $\mathbf{D}_t$, say $\hat{\mathbf{D}}_t$, is generated by taking into account the aforementioned symbol cancellation, $\hat{\mathbf{D}}_t$ would possess a structure closer to the prototype $\mathbf{D}_p$.
(b) Concerning symbols m_1 and m_2, which sum to $+2$, it is desirable to treat subsequence $\{+1, \mathbf{0}_{z_2}, +1\}$ as one symbol, equal to $+2$. Similarly, concerning symbols m_3 and m_4, which sum to $+2$, it is desirable to treat subsequence $\{+1, \mathbf{0}_{z_4}, +1\}$ as one symbol equal to $+2$.

If these transformations are applied to the original feature sequence $\mathbf{D}_t$, the new sequence $\hat{\mathbf{D}}_t$ becomes $\hat{\mathbf{D}}_t = \{\mathbf{0}_{z_1}, +2, \mathbf{0}_{z_3}, +2, \mathbf{0}_{z_5}, +2, \mathbf{0}_{z_6}, +1, \mathbf{0}_{z_7+z_8+z_9+2}, +2, \mathbf{0}_{z_{10}}\}$, which is likely to be different from $\mathbf{D}_p$ only in

the number of zeros separating the non-zero valued symbols (depending on the observed tempo fluctuation).

In order to present in brief the equations for the enhanced Viterbi algorithm, certain definitions must first be given. For an observation sequence $\mathbf{D} = \{d_1 d_2 \dots d_N\}$ and a discrete observation VDHMM $\mathcal{H}$, let us define the forward variable $a_t(j)$ as in [13], i.e.,

$$a_t(j) = P(d_1 d_2 \dots d_t,\ state\ j\ ends\ at\ t|\mathcal{H}), j = 1 \dots S \quad (3)$$

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 ([13]),

$$a_t(j) = \max_{1\le\tau\le T, 1\le i\le S, i\neq j} [\delta_t(i,\tau,j)] \quad (4)$$

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

where τ 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. Equations (4) and (5) suggest that there exist $(S \times T - T)$ candidate arguments, $\delta_t(i,\tau,j)$, 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), is stored in a two-dimensional array ψ, as $\psi(j,t)$. Therefore, $\psi(j,t) = \arg\max[\delta_t(i,\tau,j)], 1 \le \tau \le T, 1 \le i \le S, i \neq j$ In addition, the number of symbols spent on state j is stored in a two-dimensional matrix c, as $c(j,t)$.

It is important to notice that, if $\sum_{s=t-\tau+1}^{t} d_s = 0$, this indicates a possible pitch tracking error cancellation. Thus, one must also take into consideration that the symbols $\{d_t, d_{t-1}, \dots, d_{t-\tau+1}\}$ could be the result of a pitch tracking error, and must be replaced by a zero that lasts for τ successive time instances. This is quantified by considering, for the Z-states, (Sx$T-T$) additional $\hat{\delta}$ arguments to augment equation (4), namely

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

Thus, *maximization is now computed over all δ and $\hat{\delta}$ quantities*. If maximization occurs for a $\hat{\delta}$ argument, say $\hat{\delta}_t(i,\tau,j)$, then the number of symbols spent at state j is equal to τ, as is the case with the standard VDHMM. 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".

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

$$\hat{\delta}_t(i,\tau,j) = a_{t-\tau}(i) A_{ij} p_j(\tau) B_j(\sum_{s=t-\tau+1}^{t} d_s) \quad (7)$$

Similar to the previous case, maximization is again computed over all δ and $\hat{\delta}$ quantities. 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, namely $p_{S_i}(\tau) = \mathcal{G}(\tau, \mu_{S_i}, \sigma^2_{S_i})$, must also be associated with each S-state.

After the whole feature sequence of the raw audio recording is processed, a simple parser can post-process the best-state sequence and any state subsequences corresponding to occurrences of the melody can be easily located. This is because, whenever an instance of the melody is detected, the VDHMM will go through a sequence of states consisting only of Z-states and S-states. It is therefore straightforward to locate such sequences of states with a simple parser (like in a simple string-matching situation).

4.1 Computational cost related issues

The proposed enhanced Viterbi algorithm leads to increased recognition accuracy to the expense of increasing the computational cost, due to the fact that the $\hat{\delta}_t(i,\tau,j)$ arguments need also be computed. However, it is possible to reduce the computational cost, if the following assumptions are adopted:

(a) *A Z-state may only emit sequences of symbols (d_i's) that start and end with a zero-valued d_i.* This suggests that for the Z-states, the emitted symbol sequence must be of the form $\{\mathbf{0}_{z_k}, m_k, \dots, m_{l-1}, \mathbf{0}_{z_l}\}, l >= k$. If $l = k$ then only one zero-valued subsequence has been emitted. As a result, for the Z-states, the respective equations need only be computed when the following hold: $d_t = 0$, $d_{t+1} \neq 0$, $d_{t-\tau+1} = 0$ and $d_{t-\tau} \neq 0$

(b) In a similar manner, *a S-state may only emit sequences of symbols (d_i's) that start and end with a non-zero d_i.* Equivalently, for the S-states, the emitted symbol sequence must be of the form $\{m_k, \mathbf{0}_{z_{k+1}}, \dots, m_l\}$, $l >= k$. If $l = k$ then only one non-zero d_i has been emitted. As a result, for the S-states, the respective equations need only be computed when the following hold: $d_t \neq 0$, $d_{t+1} = 0$, $d_{t-\tau+1} \neq 0$ and $d_{t-\tau} = 0$.

5 EXPERIMENTS

As it has already been mentioned, Tolonen's multipitch analysis model [11] was adopted as a pitch tracker for our experiments and the following parameter tuning was decided: the moving window length 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 $40ms$. 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 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. 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 [12]). These patterns exhibit significant time elasticity due to improvisations in the performance of musicians and it was therefore considered appropriate to permit a 50% tempo fluctuation, when modeling the Z-states. In this set of raw audio data, our method successfully spotted 83% of the pattern occurrences. This 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] Ning Hu and Roger B. Dannenberg, "A Comparison of Melodic Database Retrieval Techniques using Sung Queries", *Proceedings of the Joint Conference on Digital Libraries (JCDL'02)*, pp. 301-307, July 13-17, 2002, Portland, Oregon, USA.

[2] Ning Hu, Roger B. Dannenberg and Ann L. Lewis, "A Probabilistic Model of Melodic Similarity", *Proceedings of the International Computer Music Conference (ICMC'02)*, Gotheborg, Sweden, September 2002.

[3] Yongwei Zhu and Mohan Kankanhali, "Music Scale Modeling for Melody Matching", *Proceedings of the ACM MM'03*, November 2-8, Berkeley, California, USA.

[4] V. Lavrenko and J. Pickens, "Polyphonic Music Modeling with Random Fields", *Proceedings of the ACM MM'03*, November 2-8, Berkeley, California, USA.

[5] N.Kosugi et al, "SoundCompass: A practical Query-by-Humming System", *Proceedings of the ACM SIGMOD 2004*, June 13-18, 2004, Paris France.

[6] 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.

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

[8] S. S. Shwartz et al, "Robust Temporal and Spectral Modeling for Query By Melody", *Proceedings of SIGIR'02*, August 11-15, 2002, Tampere, Finland.

[9] A. Pikrakis, S. Theodoridis and D. Kamarotos, "Classification of Musical Patterns using Variable Duration Hidden Markov Models", Proceedings of the 12th European Signal Processing Conference (EUSIPCO-2004), Vienna, Austria, September 2004.

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

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

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

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