A COMPARISON OF PROBABILISTIC MODELS FOR ONLINE PITCH TRACKING

Umut Şimşekli
Boğaziçi University
Department of Computer Engineering
umut.simsekli@boun.edu.tr

A. Taylan Cemgil
Boğaziçi University
Department of Computer Engineering
taylan.cemgil@boun.edu.tr

ABSTRACT

In this study, we propose and compare two probabilistic models for online pitch tracking: Hidden Markov Model and Change Point Model. In our models each note has a certain characteristic spectral shape which we call spectral templates. Hence the system's goal is to find the note whose template is active given the audio data. The main focus on this work is the trade off between latency and accuracy of the pitch tracking system. We present the probabilistic models and the inference schemes in detail. Encouraging results are obtained from the experiments that are done on low-pitched monophonic audio.

1. INTRODUCTION

Pitch tracking is one of the most studied topics in the computer music field since it lies at the center of many applications. It is widely used in phonetics, speech coding, music information retrieval, music transcription, and interactive musical performance systems. It is also used as a pre-processing step in more comprehensive music analysis applications such as chord recognition systems.

Many pitch tracking methods have been presented in the literature. Klapuri proposed an algorithmic approach for multipitch tracking in [1]. Kashino et al. presented applied graphical models for polyphonic pitch tracking [2]. Cemgil presented generative models for both monophonic and polyphonic pitch tracking [3]. Orio et al. and Raphael proposed Hidden Markov Model based pitch tracking methods in [4] and [5] respectively. On the other hand, using nonnegative matrix factorization (NMF) methods become popular at various audio processing applications. Different types of NMF models were proposed and tested on polyphonic music analysis, [6], [7], [8].

In this study, we propose and compare two probabilistic models for online pitch tracking. Our probabilistic models are extensible to polyphonic pitch tracking by using factorial models [9] but we mainly focus on monophonic pitch tracking of low pitched instruments. The main concern of the work is reducing the pitch detection latency without compromising the detection quality. Here the term, latency is defined as the time difference between the true note onset and the time that the pitch tracker has computed its estimate. In our point of view, a pitch tracking method might have latency due to two reasons. The first reason is that the method cannot estimate the note accurately because it has not accumulated enough data yet. The second reason is the computational burden. With the increase of the computational power, the latter can be eliminated by using more powerful computers. Hence, in our work we will focus on decreasing the latency by increasing the accuracy at note onsets rather than trying to reduce the computational complexity. We will test our models on recordings of two low pitched instruments: tuba and bass guitar. This would be challenging since estimating low pitches in shortest time is a difficult problem.

The rest of the paper is organized as follows: in Section 2 and 3 we describe our approach and probabilistic models in detail. We describe our inference scheme in Section 4. The template learning procedure is described in Section 5. In section 6, we present our results on monophonic pitch tracking. Finally Section 7 concludes this paper.

2. APPROACH

In this study, we would like to infer a predefined set of pitch labels from streaming audio data. Our approach to this problem is model based. We will construct two probabilistic generative models that relate a latent event label to the actual audio recording, in this case audio is represented by the magnitude spectrum. We define $x_{\nu,\tau}$ as the magnitude spectrum of the audio data with frequency index ν and time index τ, where $\tau \in \{1, 2, ..., T\}$ and $\nu \in \{1, 2, ..., F\}$.

For each time frame τ, we define an indicator variable r_τ on a discrete state space D_r, which determines the label we are interested in. In our case D_r consists of note labels such as {C4, C#4, D4, D#4, ..., C6}. The indicator variables r_τ are hidden since we do not observe them directly. For online processing, we are interested in the computation of the following posterior quantity, also known as the filtering density [1] :

$$p(r_\tau | x_{1:F,1:\tau}).$$

Similarly, we can also compute the most likely label tra-

[1] Note that we use MATLAB's colon operator syntax in which $(1 : F)$ is equivalent to $[1, 2, 3, ..., F]$.

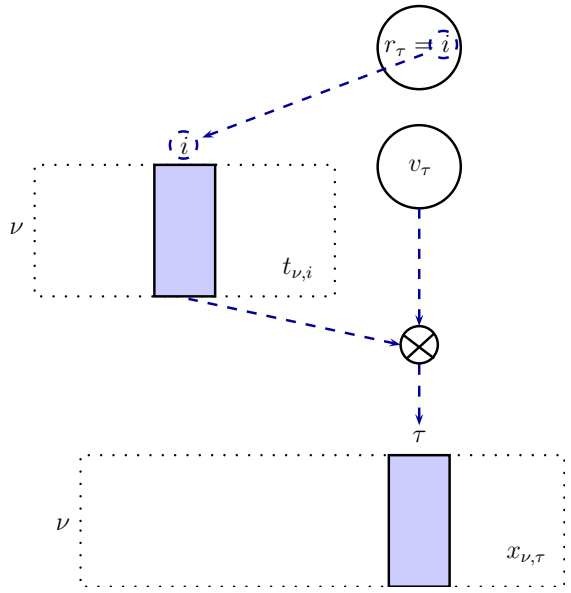


Figure 1. The block diagram of the probabilistic models. The indicator variables, r_τ choose which template to be used. The chosen template is multiplied by the volume parameter v_τ in order to obtain the magnitude spectrum, $x_{\nu,\tau}$.

jectory given all the observations

$$r^*_{1:T} = \underset{r_{1:T}}{\operatorname{argmax}}\, p(r_{1:T}|x_{1:F,1:T}).$$

This latter quantity requires that we accumulate all data and process in a batch fashion. There are also other quantities, called "fixed lag smoothers" that between those two extremes. For example, at time τ we can compute

$$p(r_{\tau-L}|x_{1:F,1:\tau}),$$

where L is a specified lag and it determines the trade off between the accuracy and the latency. By accumulating a few observations from the future, the detection at a specific frame can be eventually improved by introducing a slight latency. Hence we have to fine-tune this parameter in order to have the optimum results.

3. MODELS

In our models, the main idea is that each event has a certain characteristic spectral shape which is rendered by a specific volume. The spectral shapes that we denote as *spectral templates* are denoted by $t_{\nu,i}$. The ν index is again the frequency index and the index i indicates the pitch labels. Here, i takes values between 1 and I, where I is the number of different spectral templates. The volume variables v_τ define the overall amplitude factor, by which the whole template is multiplied. An overall sketch of the model is given in Figure 1.

3.1 The Hidden Markov Model

Hidden Markov Models have been widely studied in various types of applications such as audio processing, natural language processing, and bioinformatics. Like in many computer music applications, HMMs have also been used in pitch tracking applications [4], [5].

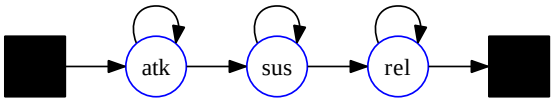


Figure 2. The prior structure of the indicator variable r_τ. Here atk, sus, and rel refers to the attack, sustain, and release parts of a note respectively. The first black square can be either the silence or a note release state. Similarly the second black square can be either a silence or a note attack state.

We define the probabilistic model as follows:

$$\begin{aligned} r_0 &\sim p(r_0) \\ r_\tau|r_{\tau-1} &\sim p(r_\tau|r_{\tau-1}) \\ v_\tau &\sim \mathcal{G}(v_\tau; a_v, b_v) \\ x_{\nu,\tau}|v_\tau, r_\tau &\sim \prod_{i=1}^{I} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i} v_\tau)^{[r_\tau = i]}, \end{aligned}$$

where the symbols $\mathcal{G}$ and $\mathcal{PO}$ represent the Gamma and the Poisson distributions respectively, where

$$\begin{aligned} \mathcal{G}(v; a, b) &= \exp((a-1)\log v - bv - \log\Gamma(a) + a\log(b)) \\ \mathcal{PO}(x; \lambda) &= \exp(x\log\lambda - \lambda - \log\Gamma(x+1)). \end{aligned}$$

Here we have Markovian prior on the indicator variables, r_τ which means r_τ depends only on $r_{\tau-1}$. We use three states to represent a note: one state for the attack part, one for the sustain part, and one for the release part. We also use a single state in order to represent silence. Figure 2 shows the Markovian structure in more detail.

In some recent work on polyphonic pitch tracking, Poisson observation model was used in the Bayesian non-negative matrix factorization models (NMF) [11]. Since our probabilistic models are similar to NMF models, we choose the Poisson distribution as the observation model. We also choose Gamma prior on v_τ to preserve conjugacy and make use of the scaling property of Gamma distribution.

In this probabilistic model we can integrate out analytically the volume variables, v_τ. It is easy to check that once we do this, provided the templates $t_{\nu,i}$ are already known, the model reduces to a standard Hidden Markov Model (HMM) with a Compound Poisson observation model.

3.2 The Change Point Model

In addition to the HMM, in the change point model (CPM), the volume parameter v_τ has a specific structure which depends on $v_{\tau-1}$ (i.e. staying constant, monotonically increasing or decreasing and etc.). But at certain unknown times, it jumps to a new value independently from $v_{\tau-1}$. We call these times as *"change points"* and the occurrence of a change point is determined by the relationship between r_τ and $r_{\tau-1}$. If $r_{\tau-1}$ jumps to a new value at time τ, in other words if r_τ is not equal to $r_{\tau-1}$, then a change point has occurred at time τ.

The formal definition of the generative model is given below:

$$
\begin{aligned}
v_0 &\sim \mathcal{G}(v_0; a_0, b_0) \\
r_0 &\sim p(r_0) \\
r_\tau | r_{\tau-1} &\sim p(r_\tau | r_{\tau-1}) \\
v_\tau | v_{\tau-1}, r_\tau, r_{\tau-1} &\sim \begin{cases} \delta(v_\tau - \theta(r_\tau) v_{\tau-1}), & r_\tau = r_{\tau-1} \\ \mathcal{G}(v_\tau; a_v, b_v), & r_\tau \neq r_{\tau-1} \end{cases} \\
x_{\nu,\tau} | v_\tau, r_\tau &\sim \prod_{i=1}^{I} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,i} v_\tau)^{[r_\tau = i]}.
\end{aligned}
$$

Here, $\delta(x)$ is the Kronecker delta function which is defined by $\delta(x) = 1$ when $x = 0$, and $\delta(x) = 0$ elsewhere. The $\theta(r_\tau)$ parameter determines the specific structure of the volume variables. Our selection of $\theta(r_\tau)$ is as follows:

$$
\theta(r_\tau) = \begin{cases} \theta_1, & \text{if } r_\tau \text{ is attack,} \\ \theta_2, & \text{if } r_\tau \text{ is sustain,} \\ \theta_3, & \text{if } r_\tau \text{ is release.} \end{cases}
$$

$\theta(r_\tau)$ gives flexibility to the CPM since we can adjust it with respect to the instrument whose sound would be processed (i.e. we can select $\theta(r_\tau) = 1$ for woodwind instruments by assuming the volume of a single note would stay approximately constant). Figure 3 visualizes example templates and synthetic data which are generated from the CPM.

4. INFERENCE

4.1 Inference on the Hidden Markov Model

As we mentioned in Section 3.1, we can integrate out analytically the volume variables, v_τ. Hence, given that the $t_{\nu,i}$ are already known, the model reduces to a standard Hidden Markov Model (HMM) with a Compound Poisson observation model as shown below:

$$
\begin{aligned}
&p(x_{1:F,1:\tau} | r_\tau = i) \\
&\quad = \int dv_\tau \; \exp\Big(\sum_{\nu=1}^{F} \log \mathcal{PO}(x_{\nu,\tau}; v_\tau t_{\nu,i}) \\
&\qquad + \log \mathcal{G}(v_\tau; a_v, b_v)\Big) \\
&\quad = \frac{\Gamma(X_\tau + a_v)}{\Gamma(a_v) \prod_{\nu=1}^{F} \Gamma(x_{\nu,\tau} + 1)} \frac{{b_v}^{a_v} \prod_{\nu=1}^{F} t_{\nu,i}^{x_{\nu,\tau}}}{(T_i + b_v)^{X_\tau + a_v}}.
\end{aligned}
$$

Since we have standard HMM from now on, the inference algorithm can be made to run very fast without any approximation. We can run the well-known forward algorithm in order to compute the filtering density or fixed lag versions with a few backward steps for real-time applications. Also we can estimate the most probable state sequence by running the Viterbi algorithm.

4.2 Inference on the Change Point Model

While making inference on the CPM, our task is finding the posterior probability of the indicator variables, r_τ and volume variables v_τ. If the state space of v_τ, D_v was discrete, then the CPM would reduce to an ordinary HMM on $D_r \times D_v$. However when D_v is continuous, which is our case, an exact forward backward algorithm cannot be implemented in general. This is due to the fact that the prediction density $p(r_\tau, v_\tau | x_{1:F,\tau})$ needs to be computed by integrating over $v_{\tau-1}$ and summing over $r_{\tau-1}$. The summation over $r_{\tau-1}$ renders the prediction density a mixture model where the number of mixture component grow exponentially with τ. In this section we will describe the implementation of exact forward backward algorithm for the CPM and the pruning technique that we use for real-time applications.

The forward backward algorithm is a well known algorithm for computing the marginals of form $p(r_\tau, v_\tau | x_{1:F,\tau})$. We define the following forward messages:

$$
\begin{aligned}
\alpha_{0|0}(r_0, v_0) &= p(r_0, v_0) \\
\alpha_{\tau|\tau-1}(r_\tau, v_\tau) &= p(r_\tau, v_\tau, x_{1:F,1:\tau-1}) \\
\alpha_{\tau|\tau}(r_\tau, v_\tau) &= p(r_\tau, v_\tau, x_{1:F,1:\tau}),
\end{aligned}
$$

where $\tau \in \{1, 2, ..., T\}$. These messages can be computed by the following recursion:

$$
\begin{aligned}
\alpha_{\tau|\tau-1}(r_\tau, v_\tau) &= \sum_{r_{\tau-1}} \int dv_{\tau-1} \, p(r_\tau, v_\tau | r_{\tau-1}, v_{\tau-1}) \\
&\qquad \alpha_{\tau-1|\tau-1}(r_{\tau-1}, v_{\tau-1}) \\
\alpha_{\tau|\tau}(r_\tau, v_\tau) &= p(x_{1:F,\tau} | r_\tau, v_\tau) \alpha_{\tau|\tau-1}(r_\tau, v_\tau).
\end{aligned}
$$

We also define the backward messages and recursions similarly:

$$
\begin{aligned}
\beta_{T|T}(r_T, v_T) &= p(x_{1:F,T} | r_T, v_T) \\
\beta_{\tau|\tau+1}(r_\tau, v_\tau) &= p(x_{1:F,\tau+1:T} | r_\tau, v_\tau) \\
&= \sum_{r_{\tau+1}} \int dv_{\tau+1} \, p(r_{\tau+1}, v_{\tau+1} | r_\tau, v_\tau) \\
&\qquad \beta_{\tau+1|\tau+1}(r_{\tau+1}, v_{\tau+1}) \\
\beta_{\tau|\tau}(r_\tau, v_\tau) &= p(x_{1:F,\tau:T} | r_\tau, v_\tau) \\
&= p(x_{1:F,\tau} | r_\tau, v_\tau) \beta_{\tau|\tau+1}(r_\tau, v_\tau),
\end{aligned}
$$

where $\tau \in \{1, 2, ..., T-1\}$. Moreover, the posterior marginals can simply be obtained by multiplying the forward and backward messages:

$$
\begin{aligned}
p(r_\tau, v_\tau | x_{1:F,1:T}) &\propto p(x_{1:F,1:T}, r_\tau, v_\tau) \\
&= p(x_{1:F,1:\tau-1}, r_\tau, v_\tau) \\
&\qquad p(x_{1:F,\tau:T} | r_\tau, v_\tau, \cancel{x_{1:F,1:\tau-1}}) \\
&= \alpha_{\tau|\tau-1}(r_\tau, v_\tau) \beta_{\tau|\tau}(r_\tau, v_\tau).
\end{aligned}
$$

Due to the fact that r is discrete and v is continuous random variables, in the CPM, we have to store α and β messages as mixtures of Gamma distributions. In order to achieve ease of implementation, we can represent the Gamma mixture

$$
p(v_\tau | r_\tau = i, .) = \sum_{m=1}^{M} \exp(w_m) \mathcal{G}(v_\tau; a_m, b_m),
$$

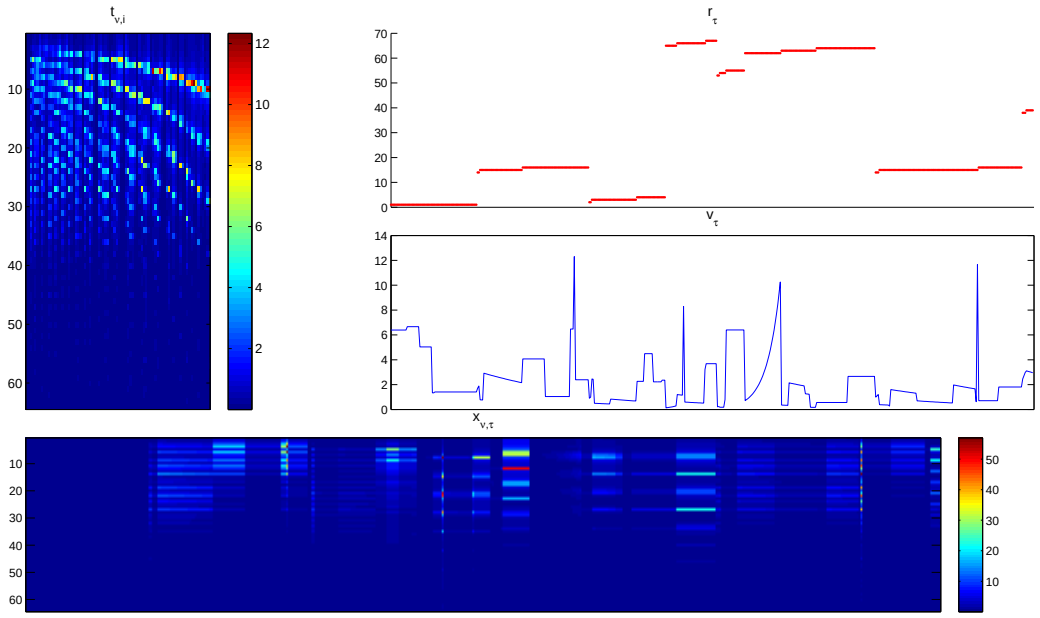


Figure 3. Spectral templates and synthetic data generated from the CPM. It can be observed that the templates implicitly capture the harmonic structure of the signals. The topmost right figure shows a realization of the indicator variables r_τ and the second topmost figure shows a realization of the volume variables v_τ. Here we set $\theta_{1:3} = \{1.10, 0.99, 1.00\}$. With this parametrization, we force the volume variables to increase during the attack parts, slowly damp at the sustain parts and stay constant during the release parts of the notes. The θ parameters should be determined by taking the audio structure into account (i.e. $\theta(r_\tau)$ should be different for higher sustained sounds, percussive sounds, woodwinds, etc.).

as $\{(a_1, b_1, w_1, i), (a_2, b_2, w_2, i), ..., (a_M, b_M, w_M, i)\}$. This will be simply $M \times 4$ array of parameters.

4.2.1 *Forward Pass*

To start the forward recursion, we define

$$\begin{aligned}\alpha_{0|0}(r_0, v_0) &= p(r_0, v_0) \\ &= p(r_0)p(v_0) \\ &= \sum_i^I \exp(l_i)\mathcal{G}(v_0; a_0, b_0),\end{aligned}$$

where, $l_i = \log p(r_0 = i)$. As we mentioned earlier, we represent this message with the array representation of the Gamma mixtures:

$$(a^k_{0|0}, b^k_{0|0}, c^k_{0|0}, d^k_{0|0}) = (a_0, b_0, l_k, k),$$

where k = 1, 2, 3, ..., I denotes the index of the components in the Gamma mixture.

In the forward procedure, we have I Gamma potentials at time $\tau = 0$. Since we are dealing with the CPM, at each time frame, we would have two possibilities: there would be a change point or not. Hence, at $\tau = 1$, we would have I newly initialized Gamma potentials for the possibility of a change point and I Gamma potentials which we copy from the previous time frame, $\tau = 0$, in order to handle the case when a change point does not occur. Similarly, at $\tau = 2$, again we would have I newly initialized Gamma potentials to handle a change point and $2I$ Gamma potentials which we copy from $\tau = 1$. Note that we would have $(\tau + 1)I$ Gamma potentials at time frame τ. Figure 4 visualizes the forward procedure.

Derivation of the prediction step at time τ is as follows:

$$\begin{aligned}&\alpha_{\tau|\tau-1}(v_\tau, r_\tau) \\ &= \sum_{r_{\tau-1}} \int dv_{\tau-1}\, p(v_\tau, r_\tau | v_{\tau-1}, r_{\tau-1}) \\ &\quad \alpha_{\tau-1|\tau-1}(v_{\tau-1}, r_{\tau-1}) \\ &= \sum_{r_{\tau-1}} \int dv_{\tau-1}\, p(v_\tau | r_\tau, v_{\tau-1}, r_{\tau-1}) p(r_\tau | r_{\tau-1}) \\ &\quad \alpha_{\tau-1|\tau-1}(v_{\tau-1}, r_{\tau-1}) \\ &= \sum_{r_{\tau-1}} \int dv_{\tau-1}\, \big([r_\tau \neq r_{\tau-1}]\mathcal{G}(v_\tau; a_v, b_v) \\ &\quad + [r_\tau = r_{\tau-1}]\delta(v_\tau - \theta(r_\tau)v_{\tau-1})\big) p(r_\tau | r_{\tau-1}) \\ &\quad \alpha_{\tau-1|\tau-1}(v_{\tau-1}, r_{\tau-1}).\end{aligned}$$

The first I potentials that handle the change point case become

$$(a^k_{\tau|\tau-1}, b^k_{\tau|\tau-1}, c^k_{\tau|\tau-1}, d^k_{\tau|\tau-1}) = (a_v, b_v, c', k)$$

for k = 1, 2,..., I. Here $a_{ij} = p(r_\tau = i | r_{\tau-1} = j)$ and

$$c' = \log \left(\sum_{\substack{j=1 \\ j \neq k}}^{I} a_{ij} \sum_{m=1}^{\tau I} [d^m_{\tau-1|\tau-1} = j] \exp(c^m_{\tau-1|\tau-1}) \right).$$

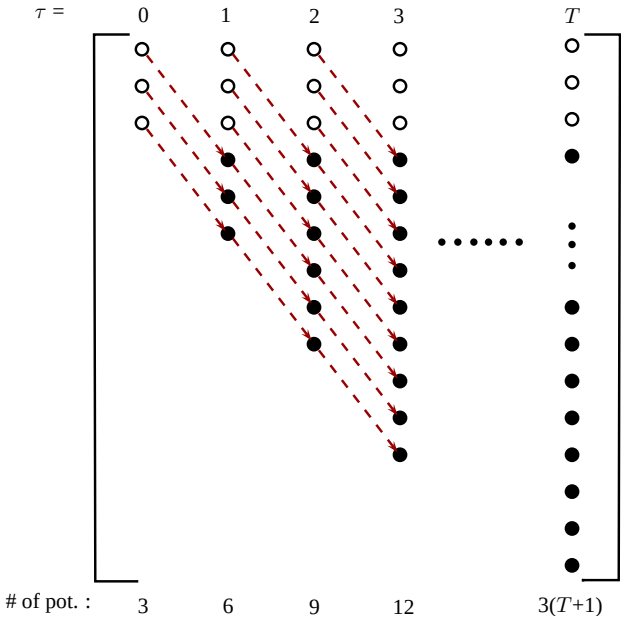


Figure 4. The forward procedure for the CPM where the number of templates, I is 3. The empty circles represent the Gamma potentials that handle the occurrence of a change point and the filled ones handle the other case. The inheritance structure between the time frames are shown with arrows.

We also have τI Gamma potentials which are inherited from the time frame $\tau - 1$:

$$(a^k_{\tau|\tau-1}, b^k_{\tau|\tau-1}, c^k_{\tau|\tau-1}, d^k_{\tau|\tau-1}) = (a^{k-I}_{\tau-1|\tau-1}, \frac{b^{k-I}_{\tau-1|\tau-1}}{\theta(d^{k-I}_{\tau-1|\tau-1})}, c', d^{k-I}_{\tau-1|\tau-1})$$

for k = I+1, I+2,..., $(\tau + 1)I$, where

$$c' = \left(\sum_{i=1}^{I} [d^{k-I}_{\tau-1|\tau-1} = i] \log a_{ii}\right) + c^{k-I}_{\tau-1|\tau-1}$$

Once we compute the predictive distributions, we have to update the Gamma potentials as we acquire the observations:

$$\begin{aligned}
&\alpha_{\tau|\tau}(v_\tau, r_\tau = i) \\
&= p(x_{1:F,1:\tau}, v_\tau, r_\tau = i) \\
&= \alpha_{\tau|\tau-1}(v_\tau, r_\tau = i) p(x_{1:F,\tau}|v_\tau, r_\tau = i) \\
&= \sum_{m=1}^{(\tau+1)I} [d^m_{\tau|\tau-1} = i] e^{(c^m_{\tau|\tau-1})} \mathcal{G}(v_\tau; a^m_{\tau|\tau-1}, b^m_{\tau|\tau-1}) \\
&\quad \prod_{\nu=1}^{F} \prod_{j=1}^{I} \mathcal{PO}(x_{\nu,\tau}; t_{\nu,j} v_\tau)^{[r_\tau = i]}.
\end{aligned}$$

Hence the update equation requires multiplication of Gamma and Poisson potentials. A nice property is that the product is also a Gamma potential, as derived in the Appendix. The updated Gamma potentials are as follows:

$$(a^k_{\tau|\tau}, b^k_{\tau|\tau}, c^k_{\tau|\tau}, d^k_{\tau|\tau}) = (a', b', c', d')$$

for k = 1, 2,..., $(\tau + 1)I$. Here

$$\begin{aligned}
a' &= a^k_{\tau|\tau-1} + \sum_{\nu=1}^{F} x_{\nu,\tau} \\
b' &= b^k_{\tau|\tau-1} + \sum_{i=1}^{I} [d^k_{\tau|\tau-1} = i] \sum_{\nu=1}^{F} t_{\nu,i} \\
c' &= c^k_{\tau|\tau-1} + \sum_{i=1}^{I} [d^k_{\tau|\tau-1} = i] g(a^k_{\tau|\tau-1}, b^k_{\tau|\tau-1}, x, t) \\
d' &= d^k_{\tau|\tau-1}.
\end{aligned}$$

4.2.2 *Backward Pass*

The backward pass is initialized as follows:

$$\begin{aligned}
\beta_{T|T+1}(v_T, r_T) &= 1 \\
(\hat{a}^k_{T|T+1}, \hat{b}^k_{T|T+1}, \hat{c}^k_{T|T+1}, \hat{d}^k_{T|T+1}) &= (1, 0, 0, k),
\end{aligned}$$

for k = 1, 2,..., I. Here the Gamma potential, $(1, 0, 0, k)$ is the improper Gamma distribution where

$$(a, b, c, k) \times (1, 0, 0, k) = (a, b, c, k),$$

for any a, b, and c.

Similar to the forward pass, we derive the backward recursion as follows:

$$\begin{aligned}
&\beta_{\tau|\tau+1}(v_\tau, r_\tau) \\
&= \sum_{r_{\tau+1}} \int dv_{\tau+1}\, p(v_{\tau+1}, r_{\tau+1}|v_\tau, r_\tau) \\
&\quad \beta_{\tau+1|\tau+1}(v_{\tau+1}, r_{\tau+1}) \\
&= \sum_{r_{\tau+1}} \int dv_{\tau+1}\, p(v_{\tau+1}|r_{\tau+1}, v_\tau, r_\tau) p(r_{\tau+1}|r_\tau) \\
&\quad \beta_{\tau+1|\tau+1}(v_{\tau+1}, r_{\tau+1}) \\
&= \sum_{r_{\tau+1}} \int dv_{\tau+1}\, ([r_{\tau+1} \neq r_\tau] \mathcal{G}(v_{\tau+1}; a_v, b_v) \\
&\quad + [r_{\tau+1} = r_\tau] \delta(v_{\tau+1} - \theta(r_\tau) v_\tau)) \\
&\quad p(r_{\tau+1}|r_\tau) \beta_{\tau+1|\tau+1}(v_{\tau+1}, r_{\tau+1}).
\end{aligned}$$

The backward recursions works very similar to the forward recursions, where we have I potentials at time T. At time $T - 1$, we would have $2I$ Gamma potentials where the first I potentials handle the case of a change point and the remaning I potentials handle the opposite case which is the same case in the forward pass. Note that, in the backward pass we would have τI Gamma potentials at time $(T - \tau)$ as opposed to the forward pass.

4.2.3 *The Pruning Procedure*

One disadvantage of this model is that the need for the computational power increases as τ increases and exact inference becomes impossible after a couple of steps. In order to eliminate this problem we developed a pruning technique for the CPM as an approximate inference scheme. In the standard pruning algorithms, at time τ, we would sort the Gamma potentials with respect to their mixture coefficients $c^k_{\tau|\tau}$, keep the N best potentials, and discard the

rest of them. However, with this scheme, we may discard the first, immature potentials in the mixture since they have been recently inserted to the mixture.

In this study we propose a different pruning scheme for the CPM. As opposed to the standard pruning methods, we always keep the first N_{keep} Gamma potentials without taking into account their mixture coefficients, where $0 \leq N_{keep} << N$. Then we apply the standard pruning algorithm to the rest of the potentials, i.e. we select the $(N - N_{keep})$ best Gamma potentials.

5. TRAINING AND PARAMETER LEARNING

Since we have constructed our inference algorithms with the assumption of the templates $t_{\nu,i}$ to be known, we have to train the spectral templates at the beginning. In this study we utilized the EM algorithm for this purpose. This algorithm maximizes the log-likelihood iteratively as follows:

$$\begin{aligned}&\text{E-step :}\\&q(v_{1:T}, r_{1:T})^{(n)} = p(v_{1:T}, r_{1:T}|x_{1:F,1:T}, t_{1:F,1:I}^{(n-1)})\\&\text{M-step :}\\&t_{1:F,1:I}^{(n)} = \underset{t_{1:F,1:I}}{\operatorname{argmax}} \left\langle \mathcal{B}^{(n-1)} \right\rangle_{q(v_{1:T}, r_{1:T})^{(n)}}\end{aligned}$$

where

$$\mathcal{B}^{(n)} \quad = \quad p(v_{1:T}, r_{1:T}, x_{1:F,1:T}|t_{1:F,1:I}^{(n)}).$$

The E-step can be computed via the methods which we described in Section 3.1 and 3.2. The M-step for the models is computed as follows:

$$t_{\nu,i}^{(n)} \quad = \quad \frac{\sum_{\tau=1}^{T} \langle [r_\tau = i] \rangle^{(n)} x_{\nu,\tau}}{\sum_{\tau=1}^{T} \langle [r_\tau = i] v_\tau \rangle^{(n)}}.$$

Intuitively, we can interpret this result as the weighted average of the normalized audio spectra with respect to v_τ.

6. EXPERIMENTS AND RESULTS

In order to evaluate the performance of the probabilistic models on pitch tracking, we have conducted several experiments. As mentioned earlier, in this study we focus on the monophonic pitch tracking of low-pitched instruments.

In our experiments we used the electric bass guitar and tuba recordings of the RWC Musical Instrument Sound Database. We first trained the templates offline, and then we tested our models by utilizing the previously learned templates. At the training step, we run the EM algorithm for each note where we use short isolated recordings. On the whole, we use 28 recordings for bass guitar (from E2 to G4) and 27 recordings for tuba (from F2 to G4). The HMM's training phase lasts approximately 30 seconds and the CPM's lasts approximately 2 minutes. At the testing step, we rendered monophonic MIDI files to audio by using the RWC recordings. The total duration of the test files are approximately 4 minutes. At the evaluation step, we compared our estimates with the ground truth which is obtained from the MIDI file. In both our models we used 46 ms. long frames at 44.1 kHz sampling rate.

In our point of view, the main trade-off of these pitch tracking models is between the latency and the accuracy. We can increase the accuracy by accumulating the data, in other words increasing the latency. However after some point the pitch tracking system would be useless due to the high latency. Hence we tried to find the optimum latency and accuracy by adjusting the "lag" parameter of the fixed-lag viterbi path which is defined as:

$$r_\tau^* = \underset{r_\tau}{\operatorname{argmax}}\, p(r_{1:\tau+L}|x_{1:F,1:\tau+L}),$$

where L is the number of audio frames to be accumulated.

As evaluation metrics, we used the recall rate, the precision rate, the computational complexity and the note onset latency. The recall rate, the precision rate and the computational complexity are defined as:

$$\begin{aligned}\text{recall} &= \frac{\text{num. of correct notes}}{\text{num. of true notes}},\\ \text{precision} &= \frac{\text{num. of correct notes}}{\text{num. of transcribed notes}},\\ \text{complexity} &= \frac{\text{running time of the method}}{\text{duration of the test file}},\end{aligned}$$

and we define the note onset latency as the time difference between the pitch tracker's estimate and the ground truth, without considering the label of the estimate. The evaluation results are shown in Figure 5.

We also compared the performance of our models with the YIN algorithm [10]. We used the *aubio* implementation and tuned the onset threshold parameter. The results are shown in Table 1.

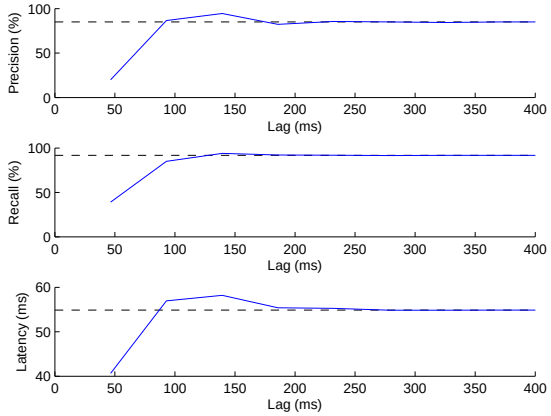
	Rec. (%)	Prec. (%)	Lat (ms)	Comp.
YIN	43.43	9.40	58.74	1.33
HMM	91.72	85.02	54.89	0.02
CPM	97.37	93.59	49.09	0.05

Table 1. The comparison of our models with the YIN algorithm. Here, *Rec*, *Prec*, *Lat* and *Comp* stand for the recall rate, the precision rate, the latency and the computational complexity respectively. The CPM performs better than the others. Moreover, the HMM would also be advantageous due to its cheaper computational needs.

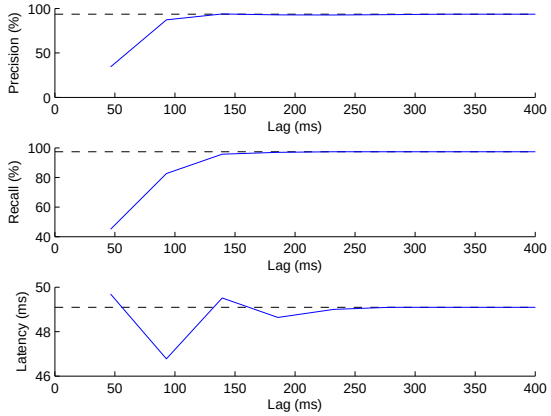
7. DISCUSSIONS AND CONCLUSION

In this study we presented and compared two probabilistic models for online pitch tracking. The main focus was on the trade off between the latency and the accuracy of the proposed pitch detection models.

Apart from the previous works that aimed to develop instrument-independent pitch tracking systems, our approach is based on modeling of a specific musical instrument's spectral structure. Our systems can be optimized for any



(a) HMM performance



(b) CPM performance.

Figure 5. The overall performance of the probabilistic models on low-pitched audio. The dashed lines represent the offline processing results. The total latency of the system is the sum of the lag and the latency at the note onsets.

instrument with a quick training procedure. Besides, this flexible template matching framework can also be used for various types of applications such as acoustic event detection.

Despite testing our probabilistic models on monophonic data, the models are extensible to more complicated scenarios such as polyphony. This kind of extensions require more complex inference schemes, but fortunately there exists powerful state-of-the-art inference methods. Moreover, we can also combine the proposed models with different kinds of probabilistic models for deeper music analysis schemes like joint pitch-tempo tracking.

One limitation of the CPM is that it has the same damping coefficient (θ) for all frequency components in the spectrum. This assumption is limiting since each partial of a note evolves differently over time. As a natural next step of our work is to construct probabilistic models that have frequency dependent damping coefficients.

8. APPENDIX

The update step of a single Gamma potential is derived as follows:

$$
\begin{aligned}
& \log\left(\exp(c)\mathcal{G}(v_\tau;a,b)\prod_{\nu=1}^{F}\mathcal{PO}(x_{\nu,\tau};t_{\nu,i}v_\tau)\right) \\
= \;& c+\log\mathcal{G}(v_\tau;a,b)+\sum_{\nu=1}^{F}\log\mathcal{PO}(x_{\nu,\tau};t_{\nu,i}v_\tau) \\
= \;& c+(a-1)\log v_\tau - bv_\tau - \log\Gamma(a)+a\log(b) \\
& +\sum_{\nu=1}^{F}\left(x_{\nu,\tau}\log t_{\nu,i}v_\tau - t_{\nu,i}v_\tau - \log\Gamma(x_{\nu,\tau}+1)\right) \\
= \;& (a+X_\tau-1)\log v_\tau-(b+T_i)v_\tau+c \\
& -\log\Gamma(a)+a\log(b)+\sum_{\nu=1}^{F}x_{\nu,\tau}\log t_{\nu,i} \\
& -\sum_{\nu=1}^{F}\log\Gamma(x_{\nu,\tau}+1) \\
= \;& (a+X_\tau-1)\log v_\tau-(b+T_i)v_\tau \\
& -\log\Gamma(a+X_\tau)+(a+X_\tau)\log(b+T_i) \\
& +c+g(a,b,x,t) \\
= \;& c'+\log\mathcal{G}(v_\tau;a',b'),
\end{aligned}
$$

where

$$
\begin{aligned}
X_\tau &= \sum_{\nu=1}^{F}x_{\nu,\tau} \\
T_i &= \sum_{\nu=1}^{F}t_{\nu,i} \\
a' &= a+X_\tau \\
b' &= b+T_i \\
c' &= c+g(a,b,x,t)
\end{aligned}
$$

and

$$
\begin{aligned}
g(.) = \;& \log\Gamma(a+X_\tau)-(a+X_\tau)\log(b+T_i) \\
& -\log\Gamma(a)+a\log(b)+\sum_{\nu=1}^{F}x_{\nu,\tau}\log t_{\nu,i} \\
& -\sum_{\nu=1}^{F}\log\Gamma(x_{\nu,\tau}+1)
\end{aligned}
$$

9. ACKNOWLEDGEMENTS

We would like to thank to the three anonymous reviewers for their useful feedback. This work is supported by Bogazici University research fund BAP 09A105P. The work of Umut Şimşekli is supported by the MS scholarship from the Scientific and Technological Research Council of Turkey (TÜBİTAK).

10. REFERENCES

[1] A. Klapuri, “Multipitch analysis of polyphonic music and speech signals using an auditory model,” *IEEE*

Transactions on Audio, Speech & Language Processing, vol. 16, no. 2, pp. 255–266, 2008.

[2] K. Kashino, K. Nakadai, T. Kinoshita, and H. Tanaka, "Application of bayesian probability network to music scene analysis," 1998.

[3] A. T. Cemgil, *Bayesian Music Transcription*. PhD thesis, Radboud University of Nijmegen, 2004.

[4] N. Orio and M. S. Sette, "An hmm-based pitch tracker for audio queries," in *ISMIR*, 2003.

[5] C. Raphael, "Automatic transcription of piano music," in *ISMIR*, 2002.

[6] E. Vincent, N. Bertin, and R. Badeau, "Harmonic and inharmonic nonnegative matrix factorization for polyphonic pitch transcription," in *ICASSP*, 2008.

[7] N. Bertin, C. Févotte, and R. Badeau, "A tempering approach for itakura-saito non-negative matrix factorization. with application to music transcription," in *ICASSP'09*, 2009.

[8] A. Cont, "Realtime multiple pitch observation using sparse non-negative constraints," in *in International Conference on Music Information Retrieval*, 2006.

[9] A. T. Cemgil, "Sequential inference for Factorial Changepoint Models," in *Nonlinear Statistical Signal Processing Workshop*, (Cambridge, UK), IEEE, 2006.

[10] A. de Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *J. Acoust. Soc. Am.*, vol. 111, pp. 1917–1930, 2002.

[11] A. T. Cemgil, "Bayesian inference in non-negative matrix factorisation models," *Computational Intelligence and Neuroscience*, no. Article ID 785152, 2009.