A METHOD FOR VISUALIZING THE PITCH CONTENT OF POLYPHONIC MUSIC SIGNALS

Anssi Klapuri
Department of Signal Processing, Tampere University of Technology
anssi.klapuri@tut.fi

ABSTRACT

This paper proposes a method for visualizing the pitch content of polyphonic music signals. More specifically, a model is proposed for calculating the salience of pitch candidates within a given pitch range, and an optimization technique is proposed to find the parameters of the model. The aim is to produce a continuous function which shows peaks at the positions of true pitches and where spurious peaks at multiples and submultiples of the true pitches are suppressed. The proposed method was evaluated using synthesized MIDI signals, for which it outperformed a baseline method in terms of precision and recall. A straightforward visualization technique is proposed to render the pitch salience function on the traditional staves when the musical key and barline information is available.

1. INTRODUCTION

Pitch analysis of polyphonic music is a challenging task where computational methods have not yet achieved the accuracy and flexibility of trained musicians. Several different approaches have been proposed towards solving the problem. Some methods are based on a statistical model of the input signal [1], whereas some others model the human auditory system [2]. Joint detection of multiple pitches has been proposed [3], contrasted by techniques which carry out iterative pitch detection and cancellation [4]. Some methods are based on unsupervised learning [5] and some others on supervised classification [6]. These examples illustrate the remarkable variety of methods that have arisen in an attempt to mimic the human ability to make sense of complex sound mixtures. A nice review of multipitch detection algorithms can be found in [7].

A drawback of many of the existing multipitch analysis methods is that they produce a discrete set of detected pitch values (or, fundamental frequencies, F0s [1]) instead of a continuous detection function that would show the likelihoods of all possible pitch values within the pitch range of interest. For pitch content visualization and acoustic feature extraction purposes, a continuous detection function is often more desirable since it allows the human eye or a subsequent post-processing algorithm to pick the interesting features from the detection function and to decide which peaks correspond to true pitches.

A fundamental difficulty in computing such detection functions (think of the autocorrelation function for example) is that they do not show a peak only at the position of the true pitch, but also at twice and half the correct pitch, and often at all multiples and submultiples of it. This ambiguity is particularly challenging in multipitch detection where the detection function easily becomes congested with spurious peaks due to the ambiguity associated with each component sound.

Various techniques have been proposed to suppress the extraneous peaks in a detection function. For example, it has been proposed to detect F0s either iteratively or jointly and to cancel all the spurious peaks that are already explained by the detected F0s [3,4]. However, these methods produce only a discrete set of F0s. Karjalainen and Tolonen proposed a method which produces an entire detection function, where the spurious peaks were suppressed using an "enhancing" procedure [2].

In this paper, we propose a model for calculating the salience (or, strength) of all pitch values within a given range of interest, and investigate a numerical optimization technique to find the model parameters so that the truly existing pitch frequencies are indicated with peaks that tend towards unity value and spurious peaks are forced towards zero. We also propose a visualization technique, where the computed pitch salience is rendered on the staves of common musical notation. This allows people who are able to read the music notation to play directly from the visualization, or to use it to study performance nuances, such as pitch glides, vibrato, and expressive timing.

2. METHOD

In the proposed method, an audio signal is first blocked into frames which are short-time Fourier transformed. The spectra are whitened (see Sec. 2.1) and the noise floor in the spectrum due to drums and other non-pitched sounds is estimated (Sec. 2.2). The whitened spectrum and the noise spectrum are used in the pitch salience model (Secs. 2.3 and 2.4). These steps are now explained in more detail.

[1] The terms pitch and F0 are used here interchangeably.

2.1 Level normalization and spectral whitening

The time-domain audio signal $x(n)$ is blocked into partly-overlapping analysis frames that are windowed using the Hamming window. The signal within each frame is level-normalized to unity variance, zero-padded to twice its length, and then discrete Fourier transformed to obtain the magnitude spectrum $X_t(k)$ in frame t. Each frame is processed independently, therefore we drop the frame index t in the following for convenience.

Spectral whitening, or flattening, is applied on $X(k)$ in order to suppress timbral information and thereby make the subsequent pitch analysis more robust to various sound sources. This is achieved by calculating power σ_c^2 of the signal within narrow frequency bands c and by scaling the signal within each band by $g_c = \sigma_c^{\nu-1}$, where $\nu = 0.16$ is a parameter determining the amount of whitening. Center frequencies f_c of the subbands are distributed uniformly on the critical band scale, $f_c = 229(10^{(0.33c+1)/21.4} - 1)$, and each subband $c = 1, \ldots, 96$ has a triangular power response extending from f_{c-3} to f_{c+3}. The resulting whitened magnitude spectrum is denoted by $Y(k)$.

2.2 Noise estimation

From the viewpoint of pitch analysis, the sounds of drums and the beginning transients of many pitched instruments are considered as "noise". Several methods have been proposed in the literature for estimating the "noise" (stochastic spectral component) in music (see [3] for review). Perhaps the most widely used is the sinusoids plus noise model, where sinusoidal components are detected and subtracted in the frequency domain, and the residual is considered as coloured (filtered) white noise. Another, quite robust method is to calculate a moving median at local regions of the magnitude spectrum.

Here the emphasis is laid on the computational efficiency and on the robustness of the method against spectral peaks which are assumed to correspond to pitched sounds and should not affect the estimate. The proposed noise spectrum estimation method consists of the following steps. First, a moving average $N'(k)$ over the whitened spectrum $Y(k)$ is calculated as

$$N'(k) = \frac{1}{u_k - l_k} \sum_{k'=l_k}^{u_k} Y(k') \tag{1}$$

where l_k and u_k define the lower and upper boundaries of the critical-band subbands within which $N'(k)$ is calculated. Note that (1) can be computed very efficiently by first calculating cumulative sum $\bar{Y}(k)$ over $Y(k)$ and then $N'(k) = \bar{Y}(u_k) - \bar{Y}(l_k - 1)$. The band boundaries l_k and u_k can be pre-stored.

To make the noise estimate immune to spectral peaks, another moving average is calculated, but including in the averaging only frequency bins for which $Y(k) < N'(k)$. The resulting noise spectrum estimate is denoted by $N''(k)$. Performing one more local averaging of $N''(k)$ (by substituting $N''(k)$ in place of $Y(k)$ in (1)) produces the final noise spectrum estimate $N(k)$.

The presented noise estimation procedure is besides simple, also computationally efficient. If the input signal consists of coloured white noise (without pitched sounds), it can be shown that $\mathsf{E}[N(k)] = 0.61\mathsf{E}[N_0(k)]$, where $\mathsf{E}[\cdot]$ denotes expectation and $N_0(k)$ is the true noise spectrum being estimated. In other words, the estimated spectrum depends linearly on the true spectrum. In practice, however, the input signal may contain pitched sounds which affect the estimate and the scaling factor can be anything between 0.61 and about 1.0. In our case, the subsequent optimization process explained below takes care of finding the linear scaling factor for $N(k)$, therefore the proposed noise estimation method is well suited here.

2.3 Pitch salience model

For convenience, the whitened spectrum $Y(k)$ and the estimated noise spectrum $N(k)$ are stored as columns in matrix $\mathbf{Y}$, together with an all-one "spectrum":

$$\mathbf{Y} = \begin{bmatrix} Y(0) & N(0) & 1 \\ Y(1) & N(1) & 1 \\ \vdots & \vdots & \vdots \\ Y(K-1) & N(K-1) & 1 \end{bmatrix} \tag{2}$$

Let us also define basis functions

$$a_m(k) = [\log(k+1)]^{m-1} \tag{3}$$

where k denotes the frequency index and $m = 1, 2, \ldots, M$ indexes the basis functions. This is a polynomial basis on the log-frequency scale. For convenience, the bases are collected as columns in matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_M]$.

The columns of $\mathbf{Y}$ are linearly combined into a single spectrum $Z(k)$ according to the following linear model:

$$Z(k) = (\mathbf{AW}.\times\mathbf{Y})\,\mathbf{1} \tag{4}$$

where $.\times$ denotes element-wise multiplication, $\mathbf{W}$ is a matrix of size $(M \times 3)$ that contains the model parameters, and $\mathbf{1}$ is an all-one vector of length 3 (the number of columns in $\mathbf{Y}$). Note that the product $\mathbf{AW}$ is a matrix of size $(K \times 3)$, the same size as $\mathbf{Y}$. The three columns define frequency responses for the three columns of $\mathbf{Y}$, before they are summed (by multiplying with $\mathbf{1}$) to obtain the final spectrum $Z(k)$. The first column of $\mathbf{AW}$ defines the frequency response of the whitened spectrum $Y(k)$. The second column defines the frequency response of the noise spectrum $N(k)$ and allowing it to be negative leads to noise subtraction from the final spectrum $Z(k)$. The third column of $\mathbf{AW}$ is multiplied by the all-one spectrum in $\mathbf{Y}$ and allows an additive frequency-dependent curve to be added to the final spectrum $Z(k)$.

Crucial for the model are the parameters in matrix $\mathbf{W}$. The M parameters in column i of $\mathbf{W}$ (together with the fixed basis functions $\mathbf{A}$) determine the frequency response of column i in $\mathbf{Y}$. The basis functions are necessary in order to be able to represent the frequency responses with only a few parameter values. An algorithm for learning the parameters will be described in Sec. 2.4.

A harmonic transform is applied on the spectrum $Z(k)$ to obtain a "raw" salience function $r(\tau)$ which indicates the strength of pitch period candidates τ:

$$r(\tau) = \sum_{h=1}^{H} Z(k_{\tau,h}). \quad (5)$$

The period τ corresponds to the F0 value f_s/τ, where f_s denotes the sampling rate. The frequency bin $k_{\tau,h}$ corresponds to the h:th harmonic (integer multiple) of the F0 and is determined by the largest value of $Z(k)$ in the vicinity of the frequency hK/τ. More exactly, the maximum is found in range $\lfloor hK/(\tau + \Delta\tau/2) \rceil, \ldots, \lfloor hK/(\tau - \Delta\tau/2) \rceil$, where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer, K is the length of the Fourier transform, and $\Delta\tau = 0.5$ denotes the spacing between successive period candidates τ. The number of harmonic partials $H = 20$.

The harmonic transform (5) is motivated by the Fourier theorem which states that a periodic signal can be represented with spectral components at integer multiples of the inverse of the period. Pitch perception, in turn, is closely linked to the time-domain periodicity of sounds.

The function $r(\tau)$ contains peaks at the positions of true pitch periods, but it requires further processing to suppress peaks that often occur at integer (sub)multiples of the true period(s). The method proposed in the following bears resemblance to the "enhancing" technique of Karjalainen et al. [2] which suppresses the peaks at integer multiples of the true period(s) in the autocorrelation function (ACF). They clipped the ACF to positive values, scaled it to twice its length, and subtracted the result from the original clipped ACF. This was repreated for time-scaling factors up to about five to suppress the peaks occurring at integer multiples of the true period(s).

The method proposed in the following is a generalization of the above idea. First, let us create scaled versions of $r(\tau)$. The original function $r(\tau)$ is scaled by a factor j by inserting zeros between the original samples, lowpass filtering the result with cutoff frequency $\frac{1}{2} f_s/j$ and multiplying the filtered signal by j. The resulting signal, denoted by $r_j(\tau)$, is finally truncated to the same length as $r(\tau)$. Stretched versions with scaling factors $j = 2, 3, \ldots, J$ are calculated (here $J = 5$).

Secondly, we calculate shrunk versions of $r(\tau)$ by scaling with factor $1/j$. This is done by lowpass filtering $r(\tau)$ with cutoff frequency $\frac{1}{2} f_s/j$ and then copying every j:th sample of $r(\tau)$ to a signal denoted by $r_{1/j}(\tau)$. Since the length of $r_{1/j}(\tau)$ is only r:th fraction of $r(\tau)$, new values have to be calculated for long periods τ using (5) in order that the shrunk function would be of the same length as the original $r(\tau)$.

For convenience, the strecthed and shrunk versions of $r(\tau)$ are stored as columns in matrix $\mathbf{R}$,

$$\mathbf{R} = \left[\mathbf{r}, \mathbf{1}, \mathbf{r}_2, \mathbf{r}_3, \ldots, \mathbf{r}_J, \mathbf{r}_{1/2}, \mathbf{r}_{1/3}, \ldots, \mathbf{r}_{1/J}\right] \quad (6)$$

where we have denoted $\mathbf{r} \equiv r(\tau)$, $\mathbf{r}_2 \equiv r_2(\tau)$, and so on for convenience, and $\mathbf{1}$ denotes an all-one vector.

Let us define basis functions

$$b_n(\tau) = [\log(\tau + 1)]^{n-1} \quad (7)$$

where τ is the period and $n = 1, 2, \ldots, N$ indexes the basis functions. This is a polynomial basis on the log-period scale. For convenience, the bases are collected as columns in a matrix $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N]$.

The final pitch salience function $s(\tau)$ is calculated as a linear function of the columns of $\mathbf{R}$:

$$s(\tau) = (\mathbf{BV} . \times \mathbf{R})\, \mathbf{1} \quad (8)$$

where $\mathbf{V}$ is a matrix of size $(N \times 2J)$ that contains the model parameters, and $\mathbf{1}$ is an all-one vector of length $2J$ (the number of columns in $\mathbf{R}$). The product $\mathbf{BV}$ gives a matrix of the same size as $\mathbf{R}$. Its columns define period-dependent weights for the columns of $\mathbf{R}$, that is, for the original raw salience $\mathbf{r}$ and its stretched and shrunk versions, $\mathbf{r}_j$ and $\mathbf{r}_{1/j}$. For example, setting small negative weights for the columns that correspond to $\mathbf{r}_2$ and $\mathbf{r}_{1/2}$ implements suppression of the peaks that occur an octave above and below each true pitch period. The N parameters in column j of $\mathbf{V}$ (together with the fixed basis functions $\mathbf{B}$) determine the period-dependent weights for column j in $\mathbf{R}$. Finally, multiplication with $\mathbf{1}$ is equivalent to summing over the columns and yields $s(\tau)$.

The proposed pitch salience model is now fully defined, except for the two parameter matrices $\mathbf{W}$ and $\mathbf{V}$ in (4) and (8), respectively.

2.4 Algorithm for learning the parameters W and V

The described pitch salience model may look quite complicated at a first sight, therefore we start from a simplified case to develop an intuition how the model works. Let us set the parameters $\mathbf{W}$ in (4) to zero for all except the first column which corresponds to the first column of $\mathbf{Y}$, and let us set the values in the first column so that $Z(k) \approx \frac{K}{k} Y(k)$ where K is the Fourier transform length. Furthermore, let us set the parameters $\mathbf{V}$ in (8) to zero for all except the first column (which correspond to the first column of $\mathbf{R}$), and set the values so that $s(\tau) \approx \frac{1}{\tau} r(\tau)$. Substituting $r(\tau)$ from (5), the overall model becomes

$$s(\tau) \approx \frac{1}{\tau} \sum_{h=1}^{H} \frac{K}{k_{\tau,h}} Y(k_{\tau,h}) \approx \sum_{h=1}^{H} \frac{1}{h} Y(k_{\tau,h}). \quad (9)$$

where the latter equivalence is because $k_{\tau,h} \approx hK/\tau$. In other words, salience is computed as $\frac{1}{h}$-weighted sum of the partial amplitudes in the whitened spectrum $Y(k)$, which is a reasonable (although simplistic) way of computing pitch salience.

The above simplified model is actually exactly how the parameters $\mathbf{W}$ and $\mathbf{V}$ are initialized in the learning algorithm to be described here. The simple model (9) is a good starting point, from where we iteratively refine the values.

An overview of the learning algorithm is as follows:

0) Matrices $\mathbf{W}$ and $\mathbf{V}$ are initialized to values that correspond to the simple model (9).

1) Matrix $\mathbf{W}$ is updated, keeping $\mathbf{V}$ fixed.

2) Matrix $\mathbf{V}$ is updated, keeping $\mathbf{W}$ fixed.

3) Steps 1 and 2 are repeated until $\mathbf{W}$ and $\mathbf{V}$ converge.

In practice, it was found to be sufficient to repeat the steps 1 and 2 just a couple of times.

The exact goal of the optimization is to find such parameters $\mathbf{W}$ and $\mathbf{V}$ that the salience function $s(\tau)$ is as close as possible to unity value at points τ that correspond to true pitch periods, and as close as possible to zero at next-largest peaks that correspond to "false" pitch periods. The steps are now described in more detail.

Initialization. As already mentioned, $\mathbf{W}$ and $\mathbf{V}$ are initialized to values that correspond to the simple model in (9). Matrix $\mathbf{W}$ is initialized so that $Z(k) \approx \frac{K}{k} Y(k)$ and matrix $\mathbf{V}$ so that $s(\tau) \approx \frac{1}{\tau} r(\tau)$. The initial values in the first column of $\mathbf{W}$ are calculated by least-squares fit $\mathbf{w}_1 = (\mathbf{A}^\mathsf{T}\mathbf{A})^{-1}\mathbf{A}^\mathsf{T}\alpha$, where vector $\alpha(k) = K/(k+\epsilon)$ denotes the target function and regularization using $\epsilon \approx 50$ is needed to avoid fitting only the largest values near the zero frequency. Similarly, the initial values in the first column of $\mathbf{V}$ are calculated by $\mathbf{v}_1 = (\mathbf{B}^\mathsf{T}\mathbf{B})^{-1}\mathbf{B}^\mathsf{T}\beta$, where vector $\beta(k) = 1/(\tau + \epsilon)$ denotes the target function and $\epsilon \approx 50$ is again needed to avoid fitting only the largest values near the zero period.

Updating $\mathbf{W}$. In order to learn better values for $\mathbf{W}$, some training material is neeeded. For this purpose, we mixed samples from 32 musical instruments with equal mean-square levels. Random mixtures up to six simultaneous sounds were generated using the McGill University Master Samples (MUMS) database.

For each training instance g, $g = 1, 2, \ldots, G$, the following operations are performed:

1.a) The salience function $s(\tau)$ is calculated using (8) and the current parameters $\mathbf{W}$ and $\mathbf{V}$.

1.b) From $s(\tau)$, we record the exact period values of the P annotated true pitches in training instance g. In addition, we record the period values of $10-P$ next-largest "false" peaks in $s(\tau)$. The peak periods are denoted by τ_p, $p = 1, \ldots, 10$, and the types of the peak by $\phi_p = [1, \ldots, 1, 0, \ldots, 0]$ where 1 indicates true peaks and 0 the false ones.

1.c) Parameter-specific salience functions $s_{m,i}(\tau)$ are calculated using (8) and current $\mathbf{V}$ and special $\mathbf{W}$ which has value 1 at position $[\mathbf{W}]_{m,i}$ and 0 elsewhere.

1.d) For each true or false peak $p = 1, \ldots, 10$, the value of $s_{m,i}(\tau_p)$ is stored in matrix $\mathbf{Q}$ on row $p+10(g-1)$ and column $m+(i-1)M$. The peak type ϕ_p is stored in vector $\mathbf{c}$ on row $p + 10(g-1)$.

After all instances g have been processed and the corresponding values stored in matrix $\mathbf{Q}$ and vector $\mathbf{c}$, updated parameters $\mathbf{w}$ are obtained by least-squares estimation

$$\mathbf{w} = (\mathbf{Q}^\mathsf{T}\mathbf{Q})^{-1}\mathbf{Q}^\mathsf{T}\mathbf{c}. \quad (10)$$

The corresponding matrix $\mathbf{W}$ is obtained by storing the $3M$ values in vector $\mathbf{w}$ to the three columns of $\mathbf{W}$. Equation (10) finds parameters which satisfy $s(\tau) \approx 1$ at the positions of "true" peaks, and $s(\tau) \approx 0$ for the false ones.

Updating $\mathbf{V}$. Updating the matrix $\mathbf{V}$ is analogous to above. For each training instance g, the following operations are performed:

2.a) and 2.b) are identical to 1.a) and 1.b), respectively.

2.c) Parameter-specific salience functions $s_{n,j}(\tau)$ are calculated using (8) and current $\mathbf{W}$ and special $\mathbf{V}$ which has value 1 at position $[\mathbf{V}]_{n,j}$ and 0 elsewhere.

2.d) For each true or false peak $p = 1, \ldots, 10$, the value of $s_{n,j}(\tau_p)$ is stored in matrix $\mathbf{O}$ on row $p+10(g-1)$ and column $n+(j-1)N$. The peak type ϕ_p is stored in vector $\mathbf{d}$ on row $p + 10(g-1)$.

After all cases g have been processed and the corresponding values stored in matrix $\mathbf{O}$ and vector $\mathbf{d}$, updated parameters $\mathbf{v}$ are obtained by least-squares estimation $\mathbf{v} = (\mathbf{O}^\mathsf{T}\mathbf{O})^{-1}\mathbf{O}^\mathsf{T}\mathbf{d}$. The corresponding matrix $\mathbf{V}$ is obtained by storing the $2JN$ values in vector $\mathbf{v}$ to the $2J$ columns of $\mathbf{V}$.

3. RESULTS

Figure 1 shows some example salience functions calculated for random sound mixtures using the proposed method (right panels) and, for comparison, for a baseline method (left panels). As a baseline method, we chose the salience function proposed in [4, Eq. (3)]. [2] The baseline method is practically identical to the simple model (9) which is used to initialize the parameter learning process here.

The two panels on top of Figure 1 show the output of the baseline and the proposed method for a single harmonic sound. The true pitch period is marked with a circle and the remaining largest false peaks are indicated with crosses. The proposed method is effective in suppressing the extraneous peaks to zero level (indicated by the horizontal line) and in forcing the true peak towards unity value. For curiosity, the next two panels show the outputs of the baseline and the proposed system for a single sinusoidal component. The last four panels show the output of the baseline system and the proposed system for a random combination of two and four sounds. As the polyphony increases, the proposed method too shows many spurious peaks although its result is still considerably cleaner that the baseline method.

Figure 2 shows precision, recall, and F-measure for the proposed method (solid line) and for the baseline method (dashed line) using synthesized MIDI signals as test material. The results were calculated by fixing a threshold value T_0, picking all the peaks in all frames above the threshold, and then calculating the resulting precision $\pi = \frac{C(\text{corr.})}{C(\text{det.})}$, recall $\rho = \frac{C(\text{corr.})}{C(\text{ref.})}$, and F-measure $\varphi = 2\pi\rho/(\pi+\rho)$. Here $C(\cdot)$, denotes the count of correct pitches found (corr.), count of all pitches detected (det.), or count of pitches in the reference (ref.). By varying the threshold, different precision/recall tradeoffs were obtained.

[2] The subsequent iterative detection and cancellation process in [4] was not used here, since it would lead to a discrete set of F0 values.

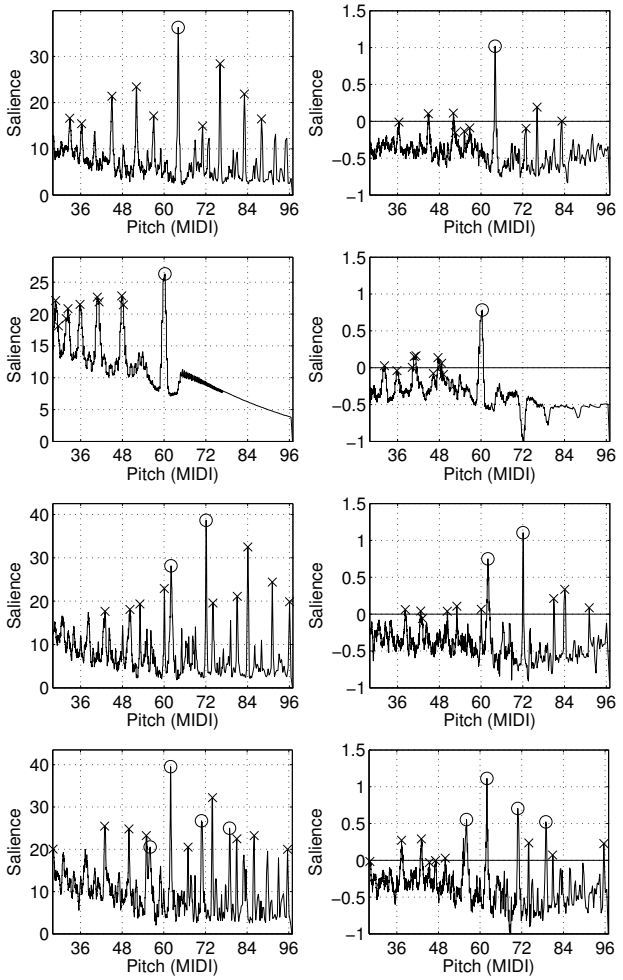

Figure 1. Example salience functions for the baseline method (left panels) and for the proposed method (right panels). The four cases from top to bottom represent 1) harmonic sound, 2) single sinusoidal component, 3) mixture of two harmonic sounds, and 4) mixture of four sounds. Peaks corresponding to the true pitch are circled.

The MIDI pieces were obtained by synthesizing random pieces from the RWC Pop and RWC Genre databases [8] and from midifarm.com. Synthesis of the MIDI files was used in order to ensure the correctness and synchronization between the synthesized file and the reference MIDI. Timidity software synthesizer and GeneralUser GS 1.4 soundfont were used for the synthesis. As can be seen in Fig. 2, the proposed method improves significantly over the baseline method. Here one should not pay too much attention on the absolute numerical values, since the polyphony of the pieces is quite high and especially the tails of long sounds can be very weak and difficult to detect.

4. APPLICATION TO PITCH VISUALIZATION

Figure 3 shows the computed salience $s(\tau)$ as a function of time for the piece No. 34 in RWC Popular Music database [8]. Here, the audio from the database was used instead of synthesizing from MIDI. The reference MIDI file is rendered on top of the salience function as boxes. In this

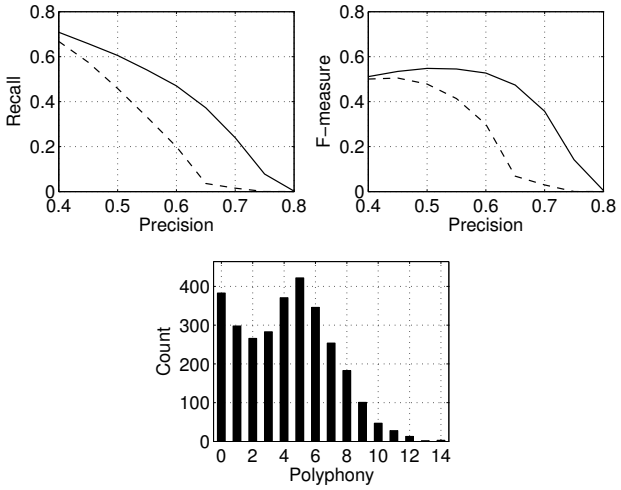

Figure 2. Precision, recall, and F-measure calculated for synthesized MIDI signals. Results are shown for the proposed method (solid line) and for the baseline method (dashed line). The third panel shows a histrogram of the number of concurrent sounds in the test data.

"piano roll" representation, the notes are arranged on the vertical axis and time flows from left to right.

Many people are not comfortable with reading music directly from a piano-roll. Therefore we propose here to map the data from the piano-roll to the traditional staves. This "fuzzy score" is very handy since it allows studying performance nuances, such as timing deviations and singing pitch glides and vibrato quite easily.

Figure 4 illustrates the mapping of different notes on the lines and spaces of the staves. Important to notice is that the note positions depend on the musical key of the piece, therefore key estimation is necessary to render the salience function on the staves. Here we used the key estimator from [9]. Secondly, the mapping is not linear: the distance between a line and its neighbouring space on the staves can be either one or two semitones. For this purpose, the piano-roll representation is "stretched" or "shrunk" to align with the staves.

Another requirement to make the score readable are bar lines which function as temporal anchors and make the timing of notes readable. Here we used the meter analysis method from [10]. The barlines are indicated with vertical lines and possible tempo changes appear as varying distances between the barlines.

Figure 5 shows the resulting "fuzzy score" representation for the same example that was shown in Fig. 3. The reference MIDI from the RWC database is drawn with circles on top of the salience. More examples of the computed fuzzy scores and the corresponding audio excerpts can be found at http://www.cs.tut.fi/sgn/arg/klap/ismir09/.

5. CONCLUSIONS

The proposed pitch salience model was shown to improve over the baseline method in terms of precision and recall when detecting multiple simultaneous pitches in synthe-

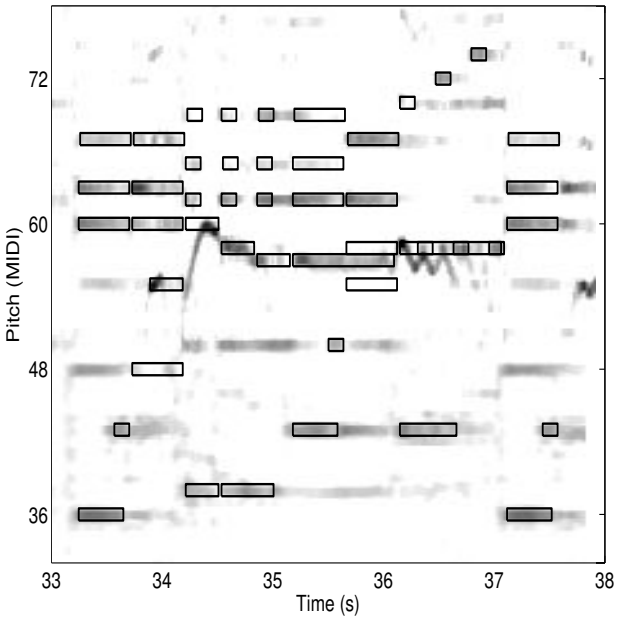

Figure 3. Computed pitch salience for an excerpt of piece No. 34 in RWC Pop database.

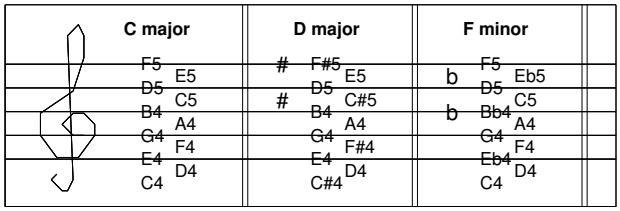

Figure 4. Mapping of pitch values on the staves positions in a few example musical keys.

sized MIDI files. This is due to the salience model which allows suppressing the peaks that occur at (sub)multiples of the true pitches in the salience function. The main advantage of the proposed method compared to many existing multipitch detection methods, however, is that it produces a continuous function that indicates the salience of all pitch candidates within a given range. This makes the proposed method particularly suitable for pitch content visualization. To this end, the proposed method was augmented with musical key and meter estimation methods which allow rendering the computed salience on the staves of common musical notation. [3]

6. REFERENCES

[1] H. Kameoka, T. Nishimoto, and S. Sagayama, "A multipitch analyzer based on harmonic temporal structured clustering," *IEEE Trans. Audio, Speech, and Language Proc.*, vol. 15, no. 3, pp. 982–994, 2007.

[2] M. Karjalainen and T. Tolonen, "Multi-pitch and periodicity analysis model for sound separation and auditory scene analysis," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Phoenix, USA, 1999.

[3] C. Yeh, *Multiple fundamental frequency estimation of polyphonic recordings*, Ph.D. thesis, University of Paris VI, 2008.

[4] A. Klapuri, "Multiple fundamental frequency estimation by summing harmonic amplitudes," in *Intl. Conf. on Music Information Retrieval*, Victoria, Canada, 2006.

[5] E. Vincent, N. Bertin, and R. Badeau, "Two nonnegative matrix factorization methods for polyphonic pitch transcription," in *MIREX'07 Extended Abstracts*, 2007.

[6] D. P. W. Ellis and G. Poliner, "Classification-based melody transcription," *Machine Learning*, vol. 65, no. 2–3, pp. 439–456, 2006.

[7] A. de Cheveigné, "Multiple F0 estimation," in *Computational Auditory Scene Analysis: Principles, Algorithms and Applications*, D. Wang and G. J. Brown, Eds. Wiley–IEEE Press, 2006.

[8] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Music genre database and musical instrument sound database," in *International Conference on Music Information Retrieval*, Baltimore, USA, Oct. 2003, pp. 229–230.

[9] M. Ryynänen and A. Klapuri, "Automatic transcription of melody, bass line, and chords in polyphonic music," *Computer Music Journal*, vol. 32, no. 3, pp. 72–86, 2008.

[10] A. Klapuri, A. Eronen, and J. Astola, "Analysis of the meter of acoustic musical signals," *IEEE Trans. Speech and Audio Processing*, vol. 14, no. 1, 2006.

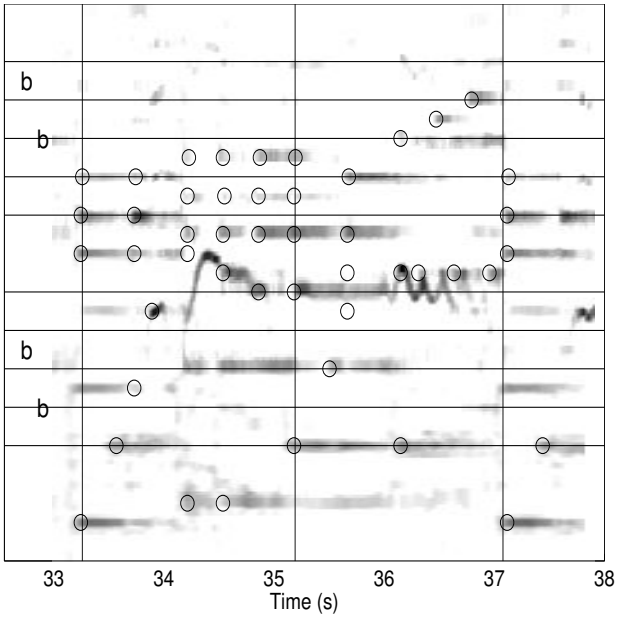

Figure 5. Computed pitch salience data rendered on the staves.

[3] This work was supported by the Academy of Finland, project 129657, (Finnish Centre of Excellence Program 2006-2011).