FAST MIR IN A SPARSE TRANSFORM DOMAIN

Emmanuel Ravelli
Université Paris 6
ravelli@lam.jussieu.fr

Gaël Richard
TELECOM ParisTech
gael.richard@enst.fr

Laurent Daudet
Université Paris 6
daudet@lam.jussieu.fr

ABSTRACT

We consider in this paper sparse audio coding as an alternative to transform audio coding for efficient MIR in the transform domain. We use an existing audio coder based on a sparse representation in a union of MDCT bases, and propose a fast algorithm to compute mid-level representations for beat tracking and chord recognition, respectively an onset detection function and a chromagram. The resulting transform domain system is significantly faster than a comparable state-of-the-art system while obtaining close performance above 8 kbps.

1 INTRODUCTION

Music recordings are now widely available in coded format. The reason is that state-of-the-art audio coders such as MP3 [6] or AAC [7] are able to reduce the size of a PCM audio signal more than 10 times, while guaranteeing a near-transparent quality. Consequently, such technology allows users to easily exchange and store music on mobile devices and networks.

On the other hand, state-of-the-art audio indexing algorithms such as beat tracking [8, 2] and chord recognition [1, 10] are designed to process PCM audio signals. Consequently, to use them with coded audio, one has to decode to PCM first and then apply the audio indexing algorithm on the PCM signal (*Processing in the time domain*, see Fig. 1). To save computational cost, which is often required e.g. when using such algorithms with mobile devices or on very large databases, it would be more efficient to design audio indexing algorithms that work directly with the coded data.

There are two ways to process coded data depending on which stage of the decoding process we are working on (see Fig. 1). The first way is to use directly the bitstream, this approach is called *processing in the compressed domain*. The second way is to use the transform representation, this approach is called *processing in the transform domain*. The first approach is faster as we avoid the cost of decoding the transform representation, however, for certain cases the information available in the bitstream is not sufficiently explicit, and it it thus necessary to use the transform domain representation.

We consider in this paper two audio indexing applications, beat tracking [8, 2] and chord recognition [1, 10].

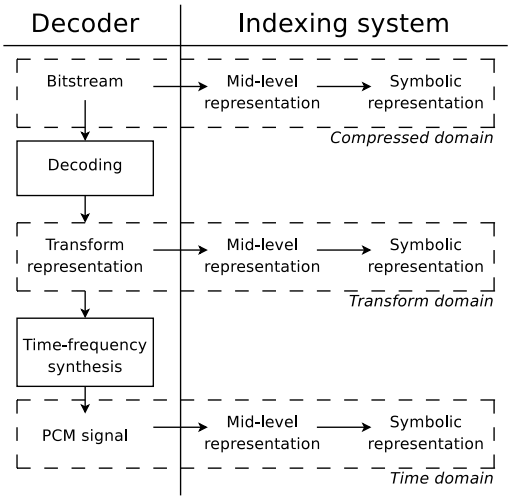

Figure 1. *Block diagram of a common audio decoder and three possible audio indexing systems.*

Beat tracking has already been investigated using MP3 audio files in the transform domain [13] and in the compressed domain [14]. However, no work related to chord recognition using coded data has been found in the literature. This may be due to the limited frequency resolution of the time-frequency analysis used in state-of-the-art transform audio coders such as MP3 [6] and AAC [7].

Due to this limitation of the transform based coders, it is interesting to consider other kinds of audio coders, such as parametric coding (e.g. [3]) or sparse representation based coding (e.g. [12]). We have chosen here to use a new prototype audio coder based on a union of MDCT bases [12], which has the advantage to provide a sparse representation which has both precise time and frequency resolution, contrary to transform based coders. This coder, available for testing in open source [1], has also the interesting property of fine-grain scalability.

We show in this paper that this new signal representation approach is not only useful for audio coding, but also for audio indexing. We propose a fast method to calculate, in the transform domain, mid-level representations similar to those used in state-of-the-art systems, namely an onset

[1] `http://www.emmanuel-ravelli.com/downloads.html`

detection function and a chromagram. The onset detection function and the chromagram are then used to perform respectively beat tracking and chord recognition using same machine learning systems as used in state-of-the-art systems [2, 1].

The remainder of this paper is as follows. In section 2, we describe the signal representation used in [12]. In section 3, the details of the calculation of the mid-level representations are given. In section 4, the machine learning systems we use to perform beat tracking and chord recognition are briefly described. Finally, section 5 gives the performance evaluation, section 6 discuss about computation times, and we conclude in section 7.

2 SIGNAL REPRESENTATION IN A UNION OF MDCT BASES

In state-of-the-art audio coders such as AAC [7], the Modified Discrete Cosine Transform (MDCT) is used. While such coders allow transparent quality at high bitrates, they give limited performance at low bitrates. In [12], Ravelli et al proposed a generalization of the transform coding approach where the signal is decomposed in a union of MDCT bases with different scales. The results showed that this new approach allows improved performance at low bitrates. We briefly present in the following the signal model and the decomposition algorithm.

2.1 Signal model

The signal is modeled using a union of 8 MDCT bases, where the window length ranges from 128 to 16384 samples (i.e. from 2.9 to 370 ms) in powers of 2. The smallest windows are needed to model very sharp attacks while larger windows are useful for modeling long stationary components. The signal $f \in \mathbb{R}^N$ is then decomposed as a weighted sum of functions $g_\gamma \in \mathbb{R}^N$ plus a residual of negligible energy r

$$f = \sum_{\gamma \in \Gamma} \alpha_\gamma g_\gamma + r \tag{1}$$

where α_γ are the weighting coefficients. The set of functions $\mathcal{D} = \{g_\gamma, \gamma \in \Gamma\}$ is called the dictionary and is a union of M MDCT bases (called blocks). The functions g, called atoms are defined as:

$$g_{m,p,k}(n) = w_m(u). \sqrt{\frac{2}{L_m}} \cos\left[\frac{\pi}{L_m}\left(u + \frac{1+L_m}{2}\right)\left(k + \frac{1}{2}\right)\right] \tag{2}$$

where $u = n - pL_m - T_m$ and m is the block index, p is the frame index, k is the frequency index, L_m is the half of the analysis window length of block m (defined as power of two $L_m = L_0 2^m$), P_m is the number of frames of block m, T_m is a time offset introduced to "align" the windows of different lengths ($T_m = \frac{L_m}{2}$) and $w_m(u)$ is the sine window defined on $u = 0, .., 2L_m - 1$.

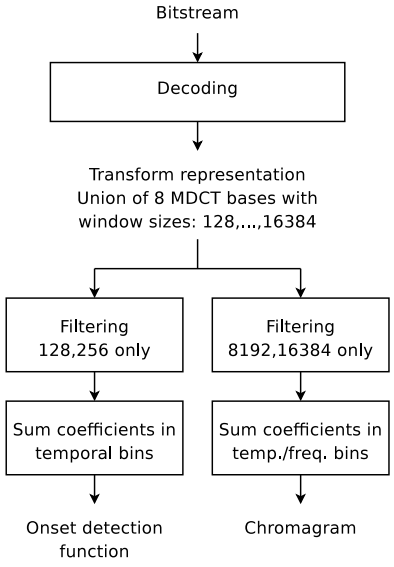

Figure 2. *Block diagram of the proposed system for the calculation of the mid-level representations.*

2.2 Decomposition algorithm

The signal is decomposed using Matching Pursuit (MP). MP [11] is an iterative algorithm which select at each iteration the atom in the dictionary that is the most correlated with the residual; subtracts the selected atom from the residual; and iterates until a stopping condition (e.g. target SNR) is met. The decomposition algorithm has been implemented in the Matching Pursuit ToolKit (MPTK) framework [9], which is to date the fastest available generic implementation of Matching Pursuit for audio signals.

3 CALCULATION OF MID-LEVEL REPRESENTATIONS

The signal representation used in the audio coder of [12] is based on a union of 8 MDCT bases with analysis window sizes from 128 to 16384 samples. We have remarked that high amplitude atoms with small window sizes (128 and 256) are often located around attacks; consequently, we can build a very cheap onset detection function by filtering the decomposition such that we keep only small window sizes atoms, and then sum the absolute value of the coefficients in temporal bins to construct a downsampled signal with peaks located at attacks. We have also remarked that strong tonal components are modeled by the large window sizes atoms (8192 and 16384) with a precise frequency resolution. Consequently, it is possible to build a very cheap chromagram by

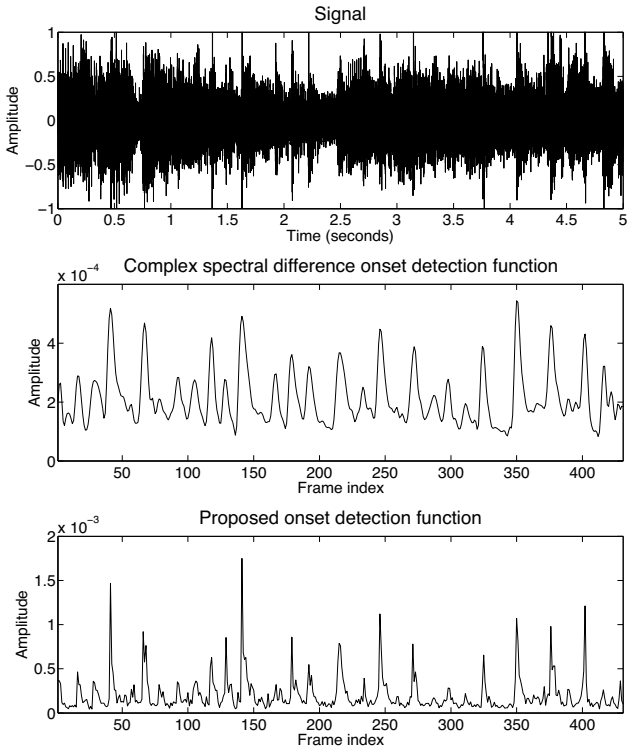

Figure 3. *A 5 seconds signal of rock music; the complex spectral difference onset detection function of [2]; the proposed onset detection function.*

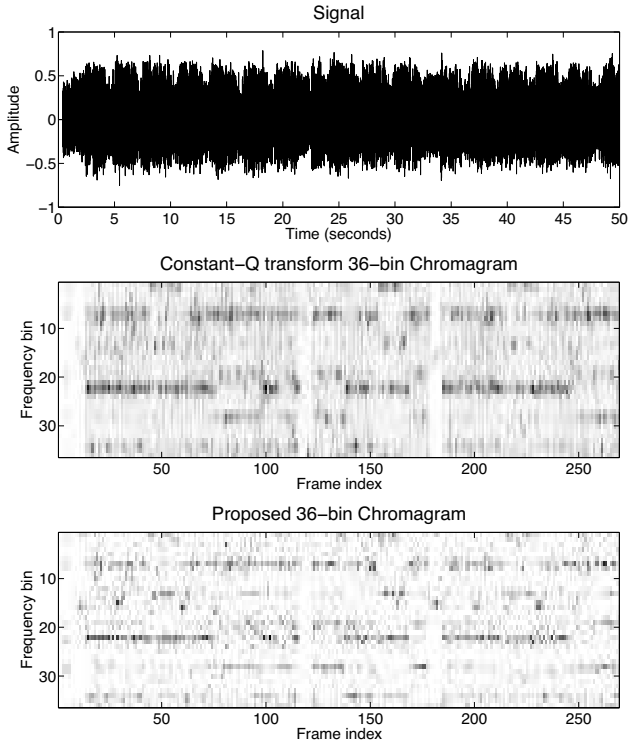

Figure 4. *A 50 seconds signal of rock music; the constant-Q transform 36-bin chromagram of [1]; the proposed 36-bin chromagram*

summing the absolute value of the coefficients of the largest atoms in time/frequency bins. The complete system is in Fig. 2 and we describe it in more details in the following.

3.1 Onset detection function

The onset detection function Γ is computed on a frame-by-frame basis. The length of one frame is defined such that the corresponding time resolution is the same as in [2] and [8] which is 11.6 ms and it is equivalent to $t_{DF} = 512$ samples at 44.1 kHz. The function $\Gamma(q)$ at frame q is thus defined as

$$\Gamma(q) = \sum_{m,p,k} |\alpha_{m,p,k}| \tag{3}$$

where we sum only the atoms satisfying the following two conditions:

- the window size is 128 or 256 samples

$$m < 2 \tag{4}$$

- the center is in the temporal support of the frame q

$$floor\left(\frac{(p+1)L_m + T_m}{t_{DF}}\right) = q. \tag{5}$$

Fig. 3 shows the onset detection function obtained with a 5 seconds signal of rock music, and as a reference the onset detection function of [2].

3.2 Chromagram

The chromagram is computed on a frame-by-frame basis too. In [1], the time resolution is 92.9 ms, while in [10] the time resolution is the double 185.8 ms. We decide to use here a time resolution of 185.8 ms as this is the hop size of the MDCT with largest window size. This is equivalent to a frame size of $t_{CH} = 8192$ samples at 44.1 kHz. The Chromagram $CH(q, b)$ at frame q and frequency bin b is thus defined as

$$CH(q, b) = \sum_{m,p,k} |\alpha_{m,p,k}| \tag{6}$$

where we sum only the atoms satisfying the following three conditions:

- the window size is 8192 or 16384 samples

$$m \geq 6 \tag{7}$$

- the center is in the temporal support of the frame q

$$floor\left(\frac{(p+1)L_m + T_m}{t_{CH}}\right) = q. \quad (8)$$

- the frequency value k maps to the frequency bin b of the chromagram

$$\text{mod}\left(\text{round}\left(B\ \text{log2}\left(\frac{22050\ \text{k}/L_m}{f_{min}}\right)\right), B\right) = b \quad (9)$$

with $B = 36$ the number of bins per octave and f_{min} is the minimum frequency.

Fig. 4 shows the chromagram obtained with a 50 seconds signal of rock music, and as a reference the chromagram of [1].

4 MACHINE LEARNING SYSTEMS

After calculation of the mid-level representation, it is passed into a machine learning system to produce a symbolic representation. The onset detection function produces a sequence of beats (frame index of the detected beats) and the chromagram produces a sequence of chords (one detected chord per frame). We use the same machine learning as in other works in order to compare only the mid-level representations in the evaluation of the final system. We describe briefly in the following the machine learning systems.

4.1 Beat tracking

The same system as in [2] is used. The onset detection function is first post-processed using an adaptive moving average threshold. Then the onset detection function is partitioned into overlapping frames to allow variable tempo. In each frame, the unbiased autocorrelation function of the onset detection function is calculated. The autocorrelation function is then passed into a shift-invariant context-dependant comb filterbank in order to estimate the tempo of the current frame. Finally, a beat train at the estimated tempo is built and aligned with the current frame by passing the detection function into a tuned context-dependant comb filterbank.

4.2 Chord recognition

The same system as in [1] is used: the 36-bin chromagram is first circularly shifted according to the estimated tuning of the piece, low-pass filtered, and mapped to a 12-bin chromagram by simply summing within semitones. Then, the Expectation Maximization (EM) algorithm is used to train the initial states probabilities and the transition matrix of an Hidden Markov Model (HMM). Finally, the sequence of chords is estimated using the Viterbi algorithm with the chromagram and the trained HMM.

5 EVALUATION

We evaluate in the following the performance of our proposed system, in comparison with the reference systems [2, 1]. As the coding/decoding process is lossy, we also study the influence of the codec bitrate on the system performance. The audio files are coded with the coder described in [12], using the standard matching pursuit with a target SNR of 60 dB, and the bitplane encoder with a bitrate of 128 kbps. As we use an embedded coding method, the decoder simply truncates the bitstream to obtain lower bitrates.

5.1 Beat tracking

We compare the performance of our system with the system of Davies et al [2] on the same database provided by S. Hainsworth [4]. There are 222 files of several music genres. As we use the same beat tracking system, this evaluation compares only the detection function used in the system. Davies *et al.* use a complex spectral difference onset detection function, which is sensitive not only for percussive sounds but also for soft tonal onsets (contrary to our detection function). Results are in Fig. 5. We give four measures of beat accuracy, as proposed in [8] and used also in [2]: correct metrical level with continuity required (CML cont); correct metrical level with continuity not required (CML total); allowed metrical levels with continuity required (AML cont); allowed metrical levels with continuity not required (AML total). The performance of our system is close to the performance of the reference system at high bitrates (5-12% less) and even at 8 kbps, with a very bad quality of the synthesized audio, the performance is still high.

5.2 Chord recognition

We compare the performance of our system with the system of Bello et al [2]. We use the same evaluation database as in [1]. It consists of 2 albums of the Beatles: Please Please Me (14 songs) and Beatles for Sale (14 songs). The database has been annotated by C. Harte et al [5]. Only the chromagram calculation is different, the machine learning system is exactly the same. To calculate the 36-bin chromagram, Bello et al use a constant-Q transform on a downsampled PCM audio signal at 11.25 kHz, with a analysis window length of 8192 samples (32768 at 44.1 kHz, twice as ours). To obtain a relevant comparison, we have modified the time resolution of the Bello system such that it is the same as our system: 185.8 ms. Recognition rates (percentage of well detected frames) for both Beatles CD are in Fig. 6. The performance of our system is close to the performance of the reference system at high bitrates (5-7% less) and even at 4 kbps, with a very bad quality of the synthesized audio, the performance is still high.

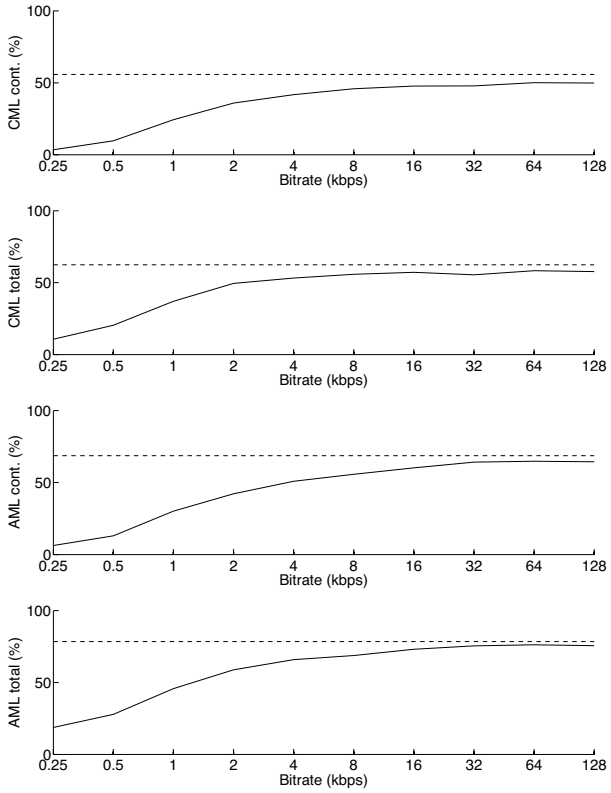

Figure 5. *Performance of the proposed beat tracking system in function of the decoding bitrate. The dotted line corresponds to the performance of the reference system (with the complex spectral difference onset detection function) on the original PCM audio.*

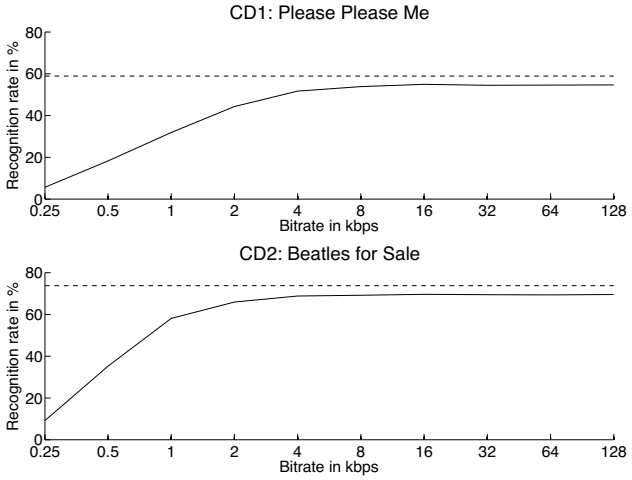

Figure 6. *Performance of the proposed chord recognition system in function of the decoding bitrate. The dotted line corresponds to the performance of the reference system (with the constant-Q transform chromagram) on the original PCM audio.*

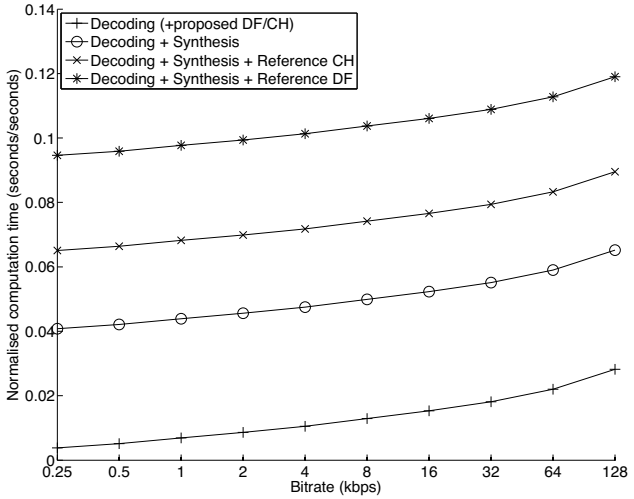

Figure 7. *Computation times of the different stages: bitstream decoding; signal synthesis; reference mid-level representations. The computation times for the proposed mid-level representations are not given as they are negligible in comparison with the bitstream decoding*

6 COMPUTATION TIME

As stated in the previous section, the performance of our transform domain systems is slightly lower than the one of the reference systems. However, working in the transform domain allows a huge gain in computation times.

As we are working with coded audio, there are two possibilities for a user to calculate mid-level representations from the coded audio. The first possibility is to decode the bitstream, synthesize the decoded transform representation, and calculate state-of-the-art mid-level representations on the synthesized PCM audio. This is the best approach in terms of performance. However, it requires several operations with a non negligible computational cost: the bitstream decoding; the transform representation synthesis, which requires 8 IMDCT; the mid-level representation, which requires a time-frequency analysis (FFT) plus additional computations. The second possibility is to compute mid-level representations from the transform representation as explained in the previous sections. This approach is a bit less efficient in terms of performance but it requires less operations that the first approach and thus is faster. Only two operations are required: the bitstream decoding; and the mid-level representation calculation, which involves only histogram-like calculations and has thus small computation cost as compared to the bitstream decoding cost.

Fig. 7 shows the computation times of the different operations: the bitstream decoding; the synthesis; the reference chromagram from PCM audio; and the reference onset detection function from PCM audio. The proposed mid-level

representations cost is not given as it is very small in comparison with the bitstream decoding. Results show that the computation of the proposed transform-domain mid-level representations is from 3 times (CH at high bitrates) to 24 times (DF at low bitrate) faster than the computation of the reference mid-level representations.

7 CONCLUSION

We have considered in this paper an alternative approach for audio indexing in the transform domain. While common approach use standard transform based coders, we show that using sparse representation allows to go beyond the limitations of the time-frequency resolution of the transform approach. We have chosen in this paper the audio coder of [12], which is based on a union of MDCT bases and we show that MIR in the sparse transform domain allows user to trade performance and computational complexity.

We have considered in this paper two applications, beat tracking and chord recognition. And we have proposed the calculation of two mid-level representations for these applications; an onset detection function and a chromagram. We have showed that the performance of the resulting systems are close to the performance of the reference systems above 8kbps. We have also shown that the computation time of the proposed systems depend mainly on the bitstream decoding module, as the cost of the proposed mid-level representations is small in comparison. Moreover, the saving of the synthesis and the time-frequency analysis allows a gain in the computation time from 3 to 24 times.

Given the results of this work, it would be interesting to consider in the future other audio indexing applications, such as e.g. musical genre classification, by using equivalent low level features such as e.g. MFCC. It would be also interesting to try other signal representations, such as MDCT with greater window sizes, or union of MCLT, and study the influence of such representations on the audio coding and audio indexing systems.

Acknowledgment

The authors would like to thank M. Davies and J. P. Bello for kindly providing their Matlab code, S. Hainsworth for providing his annotated database, and C. Harte for providing his annotations.

8 REFERENCES

[1] J. P. Bello and J. Pickens. A robust mid-level representation for harmonic content in music signals. In *Proc. Int. Conf. Music Inf. Retrieval*, pages 304–311, 2005.

[2] Matthew E. P. Davies and Mark D. Plumbley. Context-dependant beat tracking of musical audio. *IEEE trans. on audio, speech and lang. proc.*, 15(3):1009–1020, 2007.

[3] A.C. den Brinker, E.G.P. Schuijers, and A.W.J. Oomen. Parametric coding for high-quality audio. In *Proc. of the 112th AES Convention*, 2002. Paper 5554.

[4] S. Hainsworth. *Techniques for the Automated Analysis of Musical Audio*. PhD thesis, Dept. Eng., Cambridge University, 2004.

[5] C. Harte and M. Sandler. Automatic chord identification using a quantized chromagram. In *Proceedings of the 118th AES Convention*, May 2005.

[6] ISO/IEC, JTC1/SC29/WG11 MPEG. Information technology - coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s - part 3: Audio, IS11172-3 1992.

[7] ISO/IEC, JTC1/SC29/WG11 MPEG. Information technology - coding of audio-visual objects - part 3: Audio, IS14496-3 2001.

[8] Ansi P. Klapuri, Antti J. Eronen, and Jaakko T. Astola. Analysis of the meter of acoustic musical signals. *IEEE trans. on audio, speech and lang. proc.*, 14(1):342–355, 2006.

[9] S. Gribonval R. Krstulovic. MPTK: Matching pursuit made tractable. In *Proc. Int. Conf. on Acoustics, Speech, and Sig. Proc.*, volume 3, pages 496–499, 2006.

[10] K. Lee and M. Slaney. Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):291–301, 2008.

[11] S.G. Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. on Signal Processing*, 41(12):3397–3415, Dec. 1993.

[12] E. Ravelli, G. Richard, and L. Daudet. Extending fine-grain scalable audio coding to very low bitrates using overcomplete dictionaries. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA'07)*, pages 195–198, Oct. 2007.

[13] Ye Wang and Miikka Vilermo. A compressed domain beat detector using mp3 audio bitstreams. In *ACM Multimedia*, pages 194–202, 2001.

[14] Jia Zhu and Ye Wang. Pop music beat detection in the huffman coded domain. In *Proc. IEEE International Conference on Multimedia and Expo*, pages 60–63, 2007.