Audio signal representations for indexing in the transform domain

Emmanuel Ravelli, *Student Member, IEEE,* Gaël Richard, *Senior Member, IEEE,* and Laurent Daudet, *Member, IEEE*

***Abstract*—Indexing audio signals directly in the transform domain can potentially save a significant amount of computation when working on a large database of signals stored in a lossy compression format, without having to fully decode the signals. Here, we show that the representations used in standard transform-based audio codecs (e.g. MDCT for AAC, or hybrid PQF/MDCT for MP3) have a sufficient time resolution for some rhythmic features, but a poor frequency resolution, which prevents their use in tonality-related applications. Alternatively, a recently developed audio codec based on a sparse multi-scale MDCT transform has a good resolution both for time- and frequency-domain features. We show that this new audio codec allows efficient transform-domain audio indexing for 3 different applications, namely beat tracking, chord recognition and musical genre classification. We compare results obtained with this new audio codec and the two standard MP3 and AAC codecs, in terms of performance and computation time.**

***Index Terms*—Audio indexing - Time-frequency representations - Audio coding - Transform-domain indexing.**

I. Introduction

Digital audio has progressively replaced analog audio since the 80s and music is now widely stored and diffused in digital form. This revolution is mainly due to the spread of audio coding technologies, which allow to considerably reduce the amount of data necessary to represent a PCM audio signal with no (or little) loss in the perceived quality of the decoded signal. The basic principle of an audio coder is to use a time-frequency representation of an input PCM signal, which is then quantized with variable precision according to a psychoacoustic model such that the loss introduced by the quantization is minimally perceived. The first standardized MPEG audio codecs (MPEG-1 [1]), developed in the early 90s, employ a PQF filterbank (Polyphase Quadrature Filters) to decompose the sound in several subband signals. The third layer of MPEG-1 (MP3) also uses a MDCT transform (Modified Discrete Cosine Transform) which is applied on each subband signal to get better frequency resolution. MP3 is able to reduce the size of a PCM audio signal more than 6 times while guaranteeing a near-transparent quality. This property made it very attractive to the music listeners and it is now widely used. The most widespread successor of MP3, called Advanced Audio Coding (AAC), was first introduced in the MPEG-2 standard [2] in 1997 and included in the MPEG-4 standard [3] in 1999. AAC is based on a pure MDCT (without PQF filterbank), an improved encoding algorithm, and several additional coding tools (e.g. Temporal Noise Shaping, Perceptual Noise Substitution ...). Formal listening tests [4] showed that AAC is able to encode stereo music at 96 kbps with better quality than MP3 at 128 kbps, and with "indistinguishable quality" (in the EBU sense) at 128 kbps - as a comparison, the bitrate of a stereo PCM signal in CD format is 1411 kbps. MPEG-4 AAC is still considered as the state-of-the-art standard for (near-)transparent audio coding.

More recently, the digital revolution gave birth to another research domain known as automatic audio indexing, which allows to extract high-level features from digital audio. Examples of audio indexing tasks include beat tracking [5]–[8], chord recognition [9]–[12] and musical genre classification [13]–[15]. Audio indexing is useful for music information retrieval (MIR), a research domain that studies the problem of efficiently finding a given information in the ever increasing mass of digital music data. Most audio indexing systems are based on a time-frequency representation of an input PCM signal. This time-frequency representation is then used as an input to an indexing system that extract the desired high-level features (e.g. a sequence of beat positions for beat tracking, a sequence of chords for chord recognition, or a genre class for musical genre classification).

Though research in audio coding and in audio indexing has been conducted independently, the methods used in both areas share many similarities. In particular, they are both based on similar time-frequency representations. Then, one of the current challenges in audio signal processing would be to design a single time-frequency representation that could be useful for both audio coding and indexing. It would open the possibility of designing an audio indexing system that uses the internal representation of an audio codec, a case known as "transform-domain audio indexing". Given a coded file, a transform-domain audio indexing system does not decode the PCM signal but directly uses the internal time-frequency representation. The main interest of such a system is thus to reduce computational cost when processing coded files. This is useful e.g. for processing very large databases of coded files (see Fig. 1).

Transform-domain audio indexing has already been studied for the standard MPEG codecs: MPEG-1 Layer 1/2/3 and AAC. The study of Patel and Sethi [16] was probably the first to propose low-level audio features based on MPEG-1 compressed data. The basic principle is to use the internal MPEG-1 representation composed of the PQF subband signals, and to

This research was supported in part by the French GIP ANR under contract ANR-06-JCJC-0027-01 "DESAM", and in part by EPSRC grant D000246/1.

E. Ravelli and L. Daudet are with the UPMC - Univ. Paris 06, Institut Jean le Rond d'Alembert-LAM, 11 rue de Lourmel, 75015 Paris, France (email: ravelli@lam.jussieu.fr; daudet@lam.jussieu.fr).

G. Richard is with Institut TELECOM, Telecom ParisTech, LTCI-CNRS, 37-39 rue Dareau, 75014 Paris, France (email: gael.richard@enst.fr).

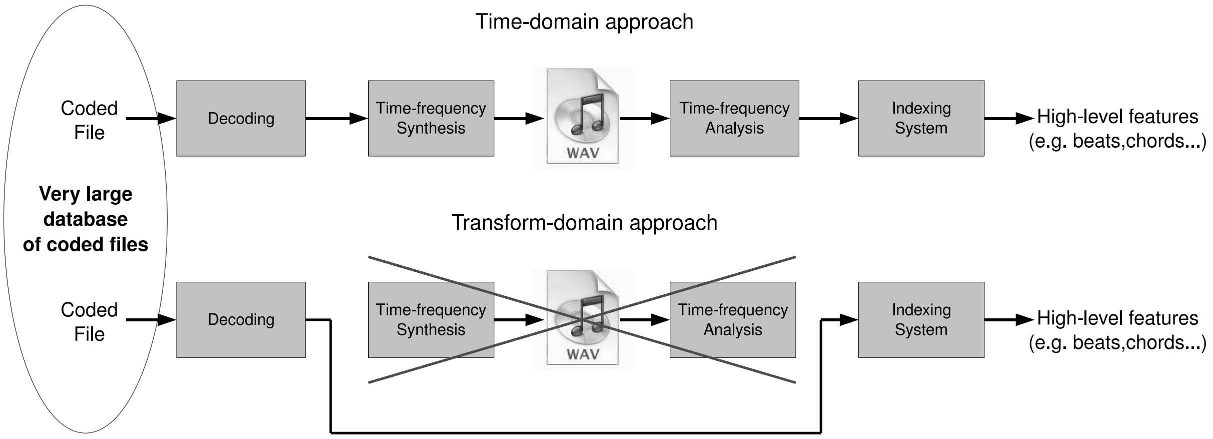

Fig. 1. Audio indexing on a very large database of coded files. Top: the traditional time-domain approach. Bottom: the transform-domain approach.

compute low-level features such as "signal energy", "pause rate", "band energy ratio"... These audio features were then combined with video features and used in a machine learning system in order to classify video clips. Other studies [17]–[23] follow a similar approach but propose different low-level audio features and consider different applications such as speech recognition [17], audio segmentation and classification [18], [19], [22], beat tracking [20], [23] and music summarization [21] (see also [24] for a review on MPEG-1 transform-domain indexing). One should note that the work of Wang et al. [20], [21], [23] and the work of [22] are in a way different from other works as they use MDCT coefficients instead of the PQF subband signals for the calculation of audio features. Indeed, these appproaches are targeted for the MP3 codec [20], [21], [23], and the AAC codec [22], [23]; they are not intented to work on the first two layers of the MPEG-1 codecs.

Despite the success of standard audio codecs for several transform-domain audio indexing applications, the internal time-frequency representation used in MP3 and AAC has limitations and in particular in terms of frequency resolution. This limitation prevents the use of such codecs for transform-domain audio indexing applications that need good frequency resolution such as chord recognition. To overcome this limitation, we propose the study of a new non-standard audio codec [25] that uses a sparse overcomplete transform composed of a union of 8 MDCT bases with different scales, allowing different time-frequency tradeoffs. This new audio codec, noted "8xMDCT" in this paper, uses simultaneously both very small and very large analysis windows allowing both good time and frequency resolution. We show in this paper that, contrary to the standard MP3 and AAC codecs, this new codec allows efficient transform-domain audio indexing for different applications including beat tracking, chord recognition and musical genre classification. For each application (beat tracking, chord recognition and musical genre classification) and each codec (MP3, AAC and 8xMDCT), we propose simple mid-level representations (a mid-level representation is an intermediate representation that emphasizes certain structures useful for a given application) that are computed in the transform-domain. We then integrate these mid-level representations in state-of-the-art audio indexing systems and evaluate their performance and computation times.

The remainder of the paper is as follows. In Section II, we briefly describe the coding/decoding process of the 3 considered codecs and present the used transform representations. In Section III, we propose simple and fast algorithms that compute mid-level representations based on the transform representations. In Section IV, we describe the state-of-the-art audio indexing systems and give results for the 3 considered applications. And finally, we conclude in Section V.

II. AUDIO CODECS

We consider in this paper three audio codecs: MPEG-1 Audio Layer 3 [1], MPEG-4 AAC LC [3] and the new 8xMDCT codec [25]. In this section, we briefly describe, for each codec, the coding/decoding process and the time-frequency representation used. It is important to note that we consider in this paper mono signals; we thus present in this section the mono version of each codec only.

A. MPEG-1 Audio Layer 3

1) Coding/decoding process: The MPEG-1 Audio Layer 3 [1] coding/decoding process is shown in Fig. 2. The input PCM signal is first passed through a 32-band PQF filterbank. Then, each subband signal is transformed with a time-varying MDCT and the resulting coefficients are processed in order to reduce the aliasing introduced by the PQF. Finally, the MDCT coefficients are scaled using scalefactors, non-linear quantized and Huffman coded. The MP3 decoder first recovers the MDCT coefficients with Huffman decoding, inverse quantization and scaling. Then, the coefficients are processed in order to revert the alias reduction performed in the coder, and inverse transformed using a time-varying inverse MDCT per subband. Finally the subband signals are passed trough a synthesis PQF, which produces the decoded PCM signal.

2) Signal representation: We use the MDCT representation for the transform-domain audio indexing. We get the MDCT coefficients produced by the decoder after inverse quantization/scaling, and just before the inverse alias reduction stage to avoid the aliasing introduced by the PQF (see Fig. 2). We have

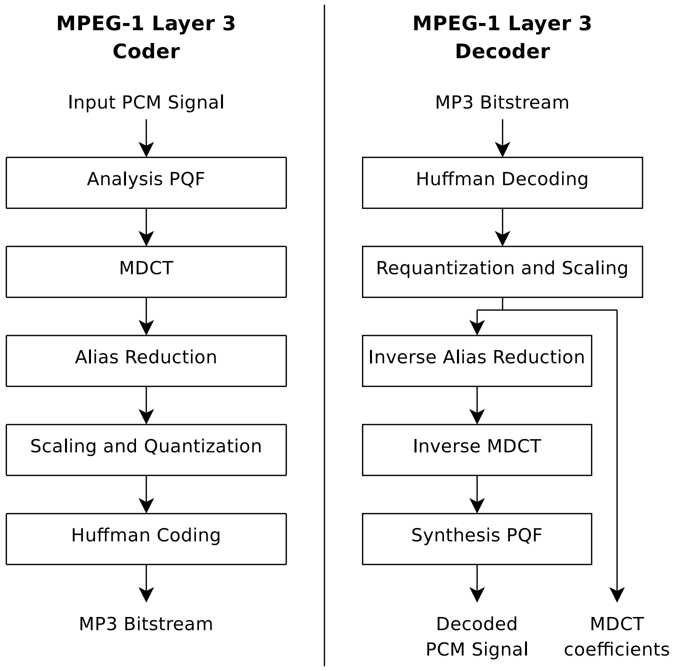

Fig. 2. Main operations in MPEG-1 Audio Layer 3 coding/decoding.

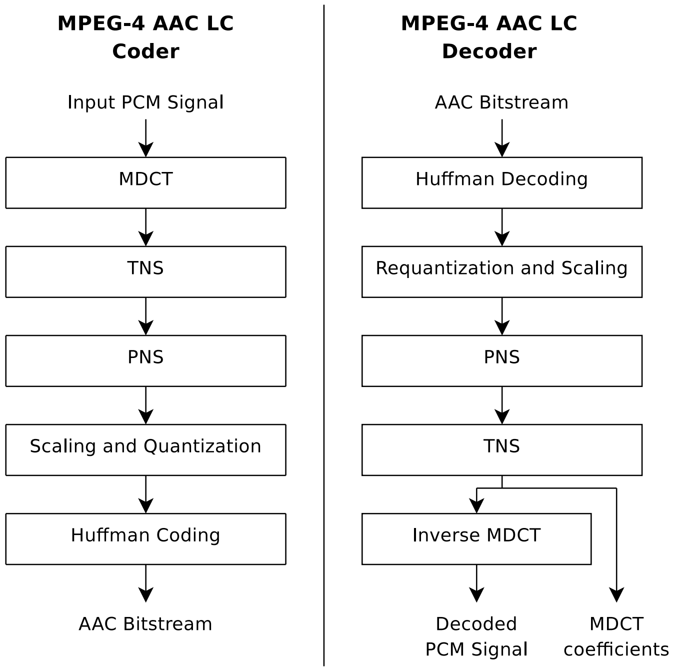

Fig. 3. Main operations in MPEG-4 AAC LC coding/decoding.

chosen this approach instead of the PQF subband signals (as in e.g. [19]) for two reasons. Firstly, the frequency resolution of the MDCT is higher ; and secondly, the computational cost is lower as the inverse MDCT operation is avoided. The time-varying MDCT used in the MPEG-1 Audio Layer 3 codec is based on a sine analysis window and two window sizes, one long of 36 samples (the long window has 3 possible shapes, one symmetric window and two asymmetric windows), and one short of 12 samples. The two long asymmetric windows are used for the transition between the symmetric long and short windows. The short windows are always selected by groups of 3 consecutive short windows. One frame (also called granule) is then composed by either one long window or 3 consecutive short windows. We assume that the same window sequence is used in each subband[1]. The MDCT coefficients of a "long-window frame" are noted $X_k^{\text{long}}(q)$ with $k = k_f 18 + k_s$ ($0 \leq k < K^{\text{long}} = 576$) is the frequency index ($0 \leq k_s < 32$ is the subband index and $0 \leq k_f < 18$ is the frequency index in one subband), and q is the frame index. The coefficients of a "short-window frame" are noted $X_{p,k}^{\text{short}}(q)$ with $0 \leq p < P^{\text{short}} = 3$ is the window index, $k = k_s 6 + k_f$ ($0 \leq k < K^{\text{short}} = 192$) is the frequency index for one short window ($0 \leq k_s < 32$ is the subband index and $0 \leq k_f < 6$ is the frequency index in one subband), and q is the frame index.

B. MPEG-4 AAC LC

1) Coding/decoding process: The MPEG-4 AAC LC coding/decoding process is shown in Fig. 3. Instead of the hybrid approach used in MP3, AAC uses a pure time-varying MDCT that is applied directly to the input PCM signal. The MDCT coefficients are then processed using two optional tools, Temporal Noise Shaping (TNS) and Perceptual Noise Substitution (PNS). TNS is based on the duality of time and frequency domain; it uses a prediction approach in the frequency domain that aims at shaping the quantization noise in the time domain; TNS is useful for e.g. pitched speech signals. PNS models the noisy-like components using a parametric approach; PNS is useful at low bitrates. It is important to note that most existing coders (e.g. Nero AAC [27] and iTunes AAC [28]) do not support the PNS tool. Finally, the MDCT coefficients in each block are scaled using scalefactors, non-linear quantized and Huffman coded. The AAC decoder first recovers the MDCT coefficients with Huffman decoding, inverse quantization and scaling. Then the coefficients are processed with the optional tools TNS and PNS. Finally the decoded PCM signal is synthesized using an inverse time-varying MDCT.

2) Signal representation: We use the MDCT representation for the transform-domain audio indexing. We get the MDCT coefficients produced by the decoder after the TNS stage and just before the inverse MDCT stage. The time-varying MDCT used in MPEG-4 AAC LC is similar to the one used in MP3 but with different window sizes. The long window has a length of 2048 samples and allows better frequency resolution than the MP3 long window (21.5 Hz for AAC, 38.3 Hz for MP3 at 44.1 kHz). The short window has a length of 256 samples and allows better time resolution than the MP3 short window (2.90 ms for AAC, 4.35 ms for MP3 at 44.1 kHz). Another difference with MP3 is the possible use of the Kaiser-Bessel Derived (KBD) window instead of the sine window, the window can be different for each frame (this choice is made by the coder). A frame is composed by either one long window or 8 consecutive short windows.

[1] We assume that the mixed block feature is not used, which is indeed the case in most coders such as LAME [26]

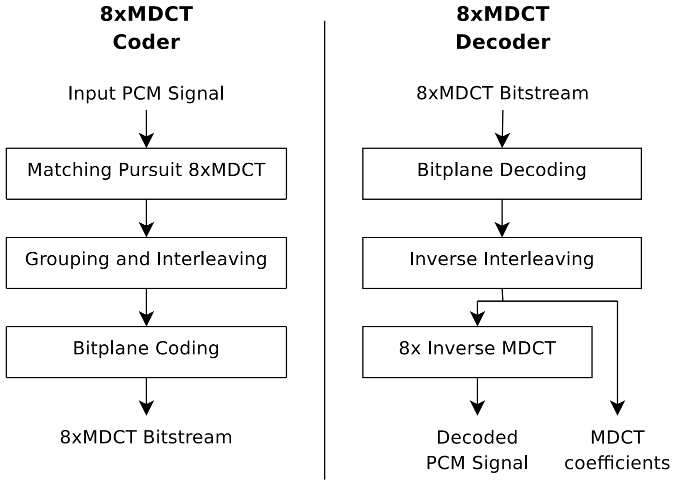

Fig. 4. Main operations in 8xMDCT coding/decoding.

The coefficients of a “long-window frame” are noted $X_k^{\text{long}}(q)$, where $0 \leq k < K^{\text{long}} = 1024$ is the frequency index and q is the frame index. The coefficients of a “short-window frame” are noted $X_{p,k}^{\text{short}}(q)$, where $0 \leq p < P^{\text{short}} = 8$ is the window index, $0 \leq k < K^{\text{short}} = 128$ is the frequency index for one short window and q is the frame index.

C. New 8xMDCT codec

1) Coding/decoding process: The 8xMDCT coding/decoding process is shown in Fig. 4 and described with more details in [25]. The input signal is first approximated using the Matching Pursuit (MP) algorithm over a union of 8 MDCT bases with the following window sizes 128, 256, 512, 1024, 2048, 4096, 8192, 16384 samples. These analysis windows allow corresponding time resolution 1.45, 2.90, 5.81, 11.6, 23.2, 46.4, 92.9, 186 ms and corresponding frequency resolution 344, 172, 86.1, 43.1, 21.5, 10.8, 5.38, 2.69 Hz. The MP algorithm is stopped when a target SNR is reached, generally high (above 50dB) to reach near-perfect reconstruction. It is important to note that the signal approximation is here performed globally on the whole signal, it is fundamentally different from the frame-by-frame analysis used in the MP3 and AAC coders. Once the signal has been approximated, the coefficients are grouped in frames; then in each frame, the coefficients are interleaved and coded using bitplane encoding. The 8xMDCT decoder first recovers the interleaved coefficients in each frame using bitplane decoding. Then the coefficients are de-interleaved and finally the decoded PCM signal is synthesized using 8 inverse MDCT. It is important to note that, contrary to the MP3 and AAC codec, 8xMDCT is a scalable codec; it means that given a sound file coded at high bitrate, the decoder can decode the file at any bitrate from very low to high bitrate just by truncating the bitstream of each frame.

2) Signal representation: We use the MDCT representation for the transform-domain audio indexing. We get the MDCT coefficients produced by the decoder after the inverse interleaving stage and just before the inverse MDCT stage. The MDCT coefficients are noted $X_{m,p,k}$, where m is the MDCT basis index ($0 \leq m < 8$), p is the window index of the m-th MDCT ($0 \leq p < P_m = 128 \times 2^{-m}$) and k is the frequency index ($0 \leq k < K_m = 128 \times 2^{m-1}$).

III. Mid-level representations

We propose in this section several mid-level representations that are computed in the transform domain using the MDCT coefficients of the three codecs presented in the previous section. We are interested here in three types of mid-level representations: onset detection function (for beat tracking), chromagram (for chord recognition), and MFCC-based features (for musical genre classification).

A. Onset detection function

Onset detection functions are mid-level representations that aim at localizing transients in an audio signal. These are generally subsampled, and ideally have peaks located at transients. These functions are obviously useful for onset detection, the onsets are simply detected by peak-picking the detection function (see [11] for a review on onset detection algorithms). They are also useful for beat tracking (see e.g. [5]–[8]), the basic principle is to look for periodically related peaks in the onset detection function, these particular onsets are called “beats”.

In the following, we propose several onset detection functions that are computed in the transform-domain and based on the MP3, AAC and 8xMDCT codecs. The reference time-domain onset detection function that we will use as comparison is the complex spectral difference onset detection function first proposed in [29] and used in the beat tracking system of [8]. The reference onset detection function used in our experiments is based on a Hanning analysis window with length 2048 samples and a hop size of 1024 samples, which gives a time resolution of 23.2 ms at 44.1 kHz; the function is then interpolated by a factor of two in order to have one sample every 11.6 ms at 44.1 kHz.

1) MP3/AAC transform-domain onset detection functions: We propose a detection function similar to the spectral flux (i.e. spectral difference [29]). The proposed onset detection function is the same for the MP3 and AAC codecs, it is defined as

$$\Gamma(q) = \sum_{k=1}^{K^{\text{long}}} |S_k(q) - S_k(q-1)|^{1/2} \tag{1}$$

where $S_k(q)$ is a “pseudo-spectrogram” at frame q and frequency k. It is defined for a “long-window frame” as

$$S_k(q) = |X_k^{\text{long}}(q)|^2 \tag{2}$$

For a “short-window frame”, it is defined as the interleaved coefficients of the P^{short} short windows in one frame

$$S_k(q) = |X_{a,b}^{\text{short}}(q)|^2 \tag{3}$$

where a and b are respectively the rest and the quotient of the Euclidean division of k by P^{short} ($k = P^{\text{short}}b + a$). The time resolution is here determined by the frame length and is thus equal to 576 samples for MP3 (13ms at 44.1 kHz) and 1024 samples for AAC (23.2ms at 44.1 kHz). To get the same sample rate as the reference approach, the AAC detection function is interpolated by a factor of two, resulting in one sample every 11.6 ms at 44.1 kHz.

2) 8xMDCT transform-domain onset detection function: The signal representation used in the 8xMDCT codec is based on a union of 8 MDCT bases with analysis window sizes from 128 to 16384 samples. Since high amplitude components with small window sizes (128 and 256) are often located around attacks, we can build a very simple onset detection function by sorting the decomposition such that we keep only small window sizes components, and then sum the absolute value of the corresponding coefficients in temporal bins to construct a downsampled signal with peaks located at attacks. The length of one bin is defined such that the corresponding time resolution is the same as in the reference detection function which is 11.6 ms and it is equivalent to 512 samples at 44.1 kHz sampling rate. The function $\Gamma(q)$ at frame q is then defined as

$$\Gamma(q) = \sum_{m,p,k} |X_{m,p,k}| \tag{4}$$

where we sum only the atoms satisfying the following two conditions: the window size is 128 or 256 samples; the center of the analysis window is in the temporal support of the q-th bin.

Fig. 5 shows the four onset detection functions obtained with a 5-second signal of rock music. The reference function is computed on the original PCM signal; and the transform-domain functions are computed on coded versions of this signal with a bitrate of 64 kbps. In this example, the onset detection functions have peaks that correspond to the drum strokes.

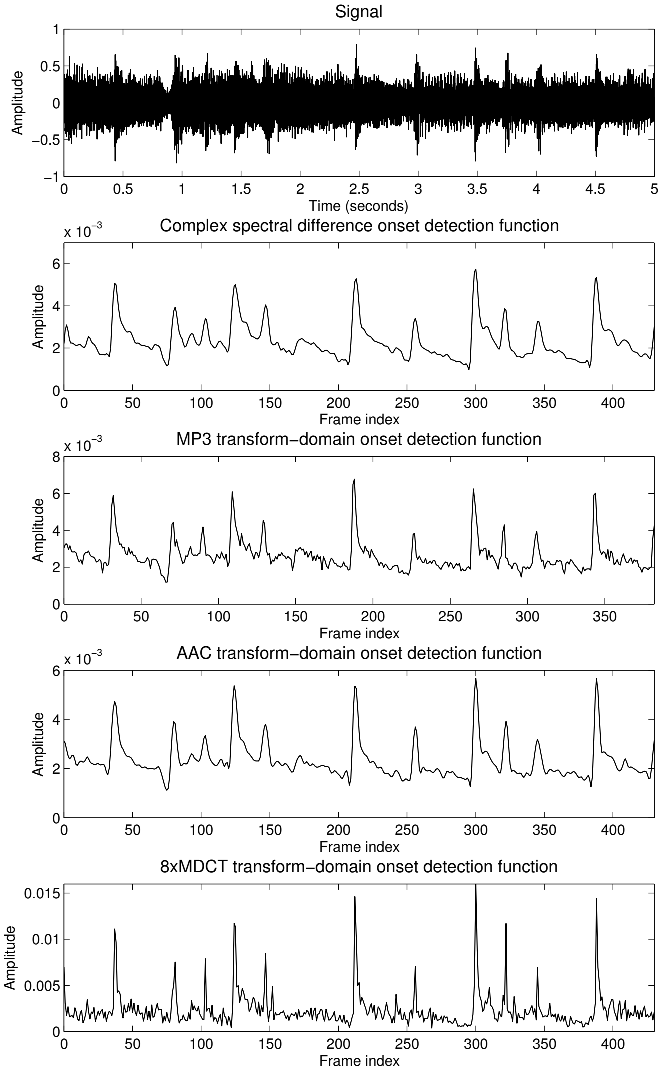

Fig. 5. *A 5 seconds signal of rock music; the complex spectral difference onset detection function; the MP3 transform-domain onset detection function; the AAC transform-domain onset detection function; the 8xMDCT transform-domain onset detection function.*

B. Chromagram

A chromagram or Pitch Class Profile (PCP) [9] traditionally consists of a 12-dimensional vector, with each dimension corresponding to the intensity of a semitone class (chroma). The procedure collapses pure tones of the same pitch class, independent of octave, on the same chromagram bin; for complex tones, the harmonics also fall into particular related bins. Though the simplest way is to use a 12-bin chromagram, better modeling is obtained by using more bins (24 or 36), in order to obtain better resolution and compensate for possible mis-tuning. These features find obvious interests in tonality-related applications, such as key estimation [30], [31] and chord recognition [9]–[12].

The reference chromagram that we will use as comparison is based on a constant-Q transform applied on a downsampled signal (see [11] for complete reference). The reference chromagram used in our experiments downsamples the input signal at 11.025 kHz and applies a constant-Q transform with a lowest frequency resolution of 1.3 Hz and a hop size of 2048 samples; the resulting chromagram has 36 bins and a time resolution of 185.8 ms.

The proposed algorithm for the calculation of a transform-domain chromagram is the same for the three codecs MP3, AAC and 8xMDCT. The first step is to keep only MDCT components with best frequency resolution, these components correspond to the analysis windows with largest size: 36 sample windows for the MP3 codec (frequency resolution: 38.3 Hz); 2048 sample windows for the AAC codec (frequency resolution: 21.5 Hz); 8192 and 16384 sample windows for the 8xMDCT codec (frequency resolution: 5.4 and 2.7 Hz). The second step is to keep only MDCT components with low center frequency and to map the center frequency of these components to chroma-related bin. Given a component with center frequency f (in Hz), it is mapped to the chroma bin b such that

$$\text{mod}\left(\text{round}\left(B\ \text{log2}\left(\frac{f}{f_{min}}\right)\right), B\right) = b \tag{5}$$

and

$$f_{min} \leq f \leq f_{max} \tag{6}$$

with B the number of bins per octave, f_{min} the minimum frequency, and f_{max} the maximum frequency. The final step is to sum the absolute value of the corresponding MDCT coefficients in time/chroma bins. The time bins have equal size of 185.8 ms allowing same time resolution as the reference chromagram. A given MDCT coefficient is mapped to the time bin whose temporal support includes the center of the corresponding analysis window, and to the chroma bin given by the previous formula.

As we have seen, MP3 and AAC have limited frequency resolution as compared to the reference approach. This limitation prevents the calculation of an efficient chromagram, because the frequency analysis cannot distinguish neighboring notes, and this is particularly true at low frequencies. We will see later that this limitation prevents a good performance in practical applications such as chord recognition.

Fig. 6 shows the chromagrams obtained with a 50-second signal of rock music. The reference chromagram is computed on the original PCM signal; and the transform-domain chromagrams are computed on coded versions of this signal with a bitrate of 64 kbps.

C. Mel-Frequency Cepstral Coefficients

Mel-Frequency Cepstral Coefficients (MFCC) aims at providing a compact representation of the spectral envelope of an audio signal. These features were originally developed for speech recognition [32], as they model the vocal tract transfer function. They are now widely used in musical applications, as they appear to be a good description of the timbre. They find useful applications in e.g. musical genre classification [13] and music similarity [33]. More recently, MFCC are used in baseline systems for evaluating audio classification systems (e.g. [15], [34]). We first detail the implementation we have chosen for the reference MFCC that are computed in the time-domain, then we propose simple algorithms for the computation of transform-domain MFCC based on the three considered audio codecs.

1) Reference time-domain MFCC: The computation of a set of C MFCC coefficients is described as follows. A small frame $x(n), n = 0,..,N-1$ is first extracted from the signal. In our implementation, the frames are non-overlapping and have a length of 23.2 ms i.e. $N = 1024$ samples at 44.1 kHz. Then, the magnitude of the Discrete Fourier Transform is computed

$$X(k) = |\sum_{n=0}^{N-1} x(n)w(n)e^{-ikn/N}| \tag{7}$$

with $k = 0,..,N/2-1$, and $w(n), n = 0,..,N-1$ the analysis window. In our implementation, it is a Hamming window. Then, the resulting spectrum $X(k)$ is mapped onto the Mel scale using L triangular overlapping windows that are equally spaced on the Mel scale. In our implementation, we use $L = 40$ triangular windows whose frequency bounds range from 20 Hz to 16000 Hz, and we use the following formula for the Mel-scale:

$$m = 1127.01048 \log(1 + f/700) \tag{8}$$

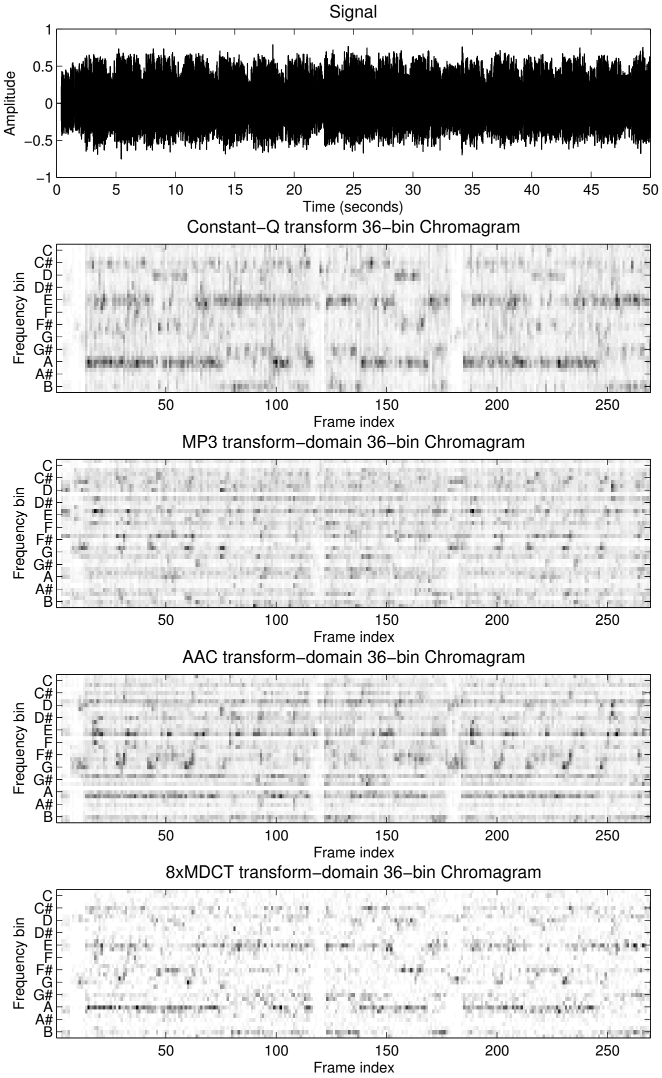

Fig. 6. *A 50 seconds signal of rock music; the reference chromagram; the MP3 transform-domain chromagram; the AAC transform-domain chromagram; the 8xMDCT transform-domain chromagram.*

where m is the frequency in mel and f is the frequency in Hz. The mapped spectrum $Y(l), l = 0,..,L-1$ is then defined as

$$Y(l) = \sum_{k=0}^{N/2-1} X(k)W_l(k) \tag{9}$$

with $l = 0,..,L-1$, and $W_l(k), k = 0,...,N/2-1$ is the l-th triangular window. Finally, the mapped spectrum is log-scaled and transformed with a Discrete Cosine Transform. The final MFCC coefficients are defined as

$$MF(c) = \sum_{l=0}^{L-1} \log_{10}(Y(l) + \varepsilon)\text{dct}_L(c,l) \tag{10}$$

with

$$\text{dct}_L(c,l) = \frac{\Lambda(c)}{L/2} \cos\left(\frac{\pi}{L}\left(l + \frac{1}{2}\right)c\right) \tag{11}$$

and $c = 0,...,C-1$. $\Lambda(c) = \sqrt{2}/2$ if $c = 0$ and $\Lambda(c) = 1$ otherwise. The constant $\varepsilon = 1e-16$ avoids log of zero. In our implementation, we keep $C = 13$ coefficients.

2) MP3/AAC transform-domain MFCC: We propose here a simple algorithm for the computation of a set of MFCC in the transform-domain of MP3/AAC audio files. The basic principle is to use the absolute value of the MDCT coefficients instead of the magnitude of the DFT in the MFCC computation described previously. The rest of the algorithm is exactly the same. It is important to note that the MFCC are computed on long and symmetric windows only. We do this for two reasons, firstly the frequency resolution of small-window blocks is too low, secondly we want to have comparable feature vectors in order to estimate long-term statistics.

3) 8xMDCT transform-domain MFCC: We propose here an algorithm to compute MFCC-like features from the transform-domain representation of the 8xMDCT codec. These features are computed on a frame-by-frame basis, where a vector of features is computed for each frame of 8192 samples. In each frame, a scale-frequency representation is computed, where the frequency axis is on the same Mel-scale as in the reference MFCC computation, and the scale axis corresponds to the window size. This representation can be seen as scale-dependant MFCC. This scale-frequency representation is simply a weighted histogram where the amplitude of the atoms are summed in scale-frequency bins. The scale-frequency representation $Y(l,m)$ is defined as

$$Y(l,m) = \sum_{p,k} |c_{m,p,k}| W_{m,l}(k) \tag{12}$$

with $l = 0,..,L-1$, $m = 0,..,M-1$, and $W_{m,l}(k)$ is the l-th window of the scale m. This scale-frequency representation is then log-scaled and transformed with a 2D-DCT. The final MFCC coefficients are defined as

$$MF(i,j) = \sum_{l=0}^{L-1} \sum_{m=0}^{M-1} \log_{10}(Y(l,m) + \varepsilon) \mathrm{dct2}_{L,M}(i,j,l,m) \tag{13}$$

with

$$\mathrm{dct2}_{L,M}(i,j,l,m) = \mathrm{dct}_L(i,l) \mathrm{dct}_M(j,m) \tag{14}$$

and $i = 0,..,C_j - 1$, $j = 0,..,J-1$. The total number of MFCC coefficients is then equal to $C = \sum_{j=0}^{J-1} C_j$. In our implementation, we choose $J = 4$, and $C_0 = 7$ coefficients on the first scale axis, $C_1 = 3$ on the second scale axis, $C_2 = 2$ on the third scale axis and $C_3 = 1$ on the fourth scale axis. This results in an identical total number of coefficients $C = 13$.

4) Texture window: MFCC are computed on segments of length 23.2 ms for the reference implementation, 13.0 ms for MP3, 23.2 ms for AAC, and 185.8 ms for 8xMDCT coding. As proposed in [13]–[15], the MFCC are grouped in longer frames, also called texture windows. In our implementation, we take the mean and the variance of the MFCC on non-overlapping texture windows of length 3 seconds. This results in a vector of 26 features for each 3 seconds of an audio signal. Fig. 7 shows a 30-second signal of rock music and the different MFCC implementations.

IV. Evaluation

In this section, we evaluate the proposed transform-domain mid-level representations which are integrated in state-of-the-art audio indexing systems. We first describe the experimental setup: implementation of the coding/decoding, a brief description of the audio indexing systems, and the evaluation databases and metrics. Then, we give results and discuss the performance and computation times of the final systems.

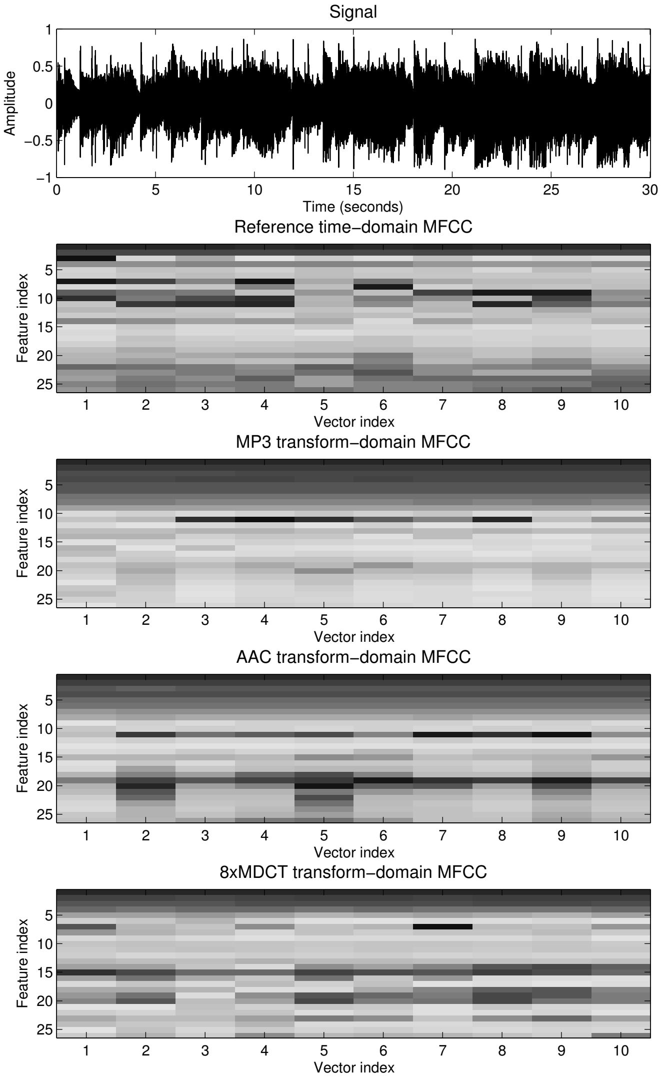

Fig. 7. *A 30-second signal of rock music; the reference time-domain; the MP3 transform-domain MFCC; the AAC transform-domain MFCC; the 8xMDCT transform-domain MFCC (mean and variance for each texture window).*

A. Experimental setup

1) Coding/decoding: We detail here the implementation and configuration of the coding/decoding process. For reproducing results, the source code is open-source and freely available[2].

We use the following coders: LAME [26], a well-known open-source and high-quality MPEG-1 Layer 3 encoder; Nero AAC Codec, a freely available and high-quality MPEG-4 AAC LC encoder [27]; the 8xMDCT coder described in [25], it is open source and freely available. Coders are setup with

[2] Source code can be downloaded at the following address: http://www.emmanuel-ravelli.com/downloads

default parameters, constant bitrate (i.e. equal bit budget per frame) and no downsampling at low bitrate (input and output signals are sampled at 44.1 kHz). The 8xMDCT coder uses the standard Matching Pursuit algorithm (without pre-echo control) with a target SNR of 60 dB, and the simple bitplane encoding algorithm (without psychoacoustic model). Each sound file is encoded at 3 bitrates for the MP3 and AAC coders (32kbps, 64kbps, and 128kbps). For the 8xMDCT coder, each sound file is encoded at a unique bitrate of 128kbps, lower bitrates are simply obtained by truncating the bitstream of each frame.

We use the following decoders: libMAD [35], an open-source library for MPEG-1 audio decoding; FAAD [36], an open-source library for MPEG-2/4 AAC decoding; the 8xMDCT decoder described in [25], it is open source and freely available. For each decoder, we have implemented Matlab MEX functions in the C/C++ language that are able to : decode the PCM signal, decode the MDCT coefficients, and compute the 3 transform-domain mid-level representations described in the previous section. We have also implemented the reference mid-level representations in the C++ language in order to have a fair comparison of the computation times.

2) Audio indexing systems: The proposed transform-domain audio indexing systems are based on reference state-of-the-art audio indexing systems; only the mid-level representations are different, the rest of the systems are exactly the same. We briefly describe the 3 reference systems in the following.

The reference beat tracking system [8] first post-processed the onset detection function 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. It is worth noting that this system obtained the 2nd place at the MIREX 2006 audio beat tracking contest.

The reference chord recognition system [11] first circularly shifts the 36-bin chromagram according to the estimated tuning of the piece, low-pass filter it, and map it 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. The system recognizes 24 chords only (C major, C minor, C# major, C# minor...). A slightly updated version of this system obtained the 1st place at the MIREX 2008 audio chord detection contest.

The reference musical genre classification system is a simple system based on SVM. MFCC are first computed on temporal segments, then the mean and the variance of the coefficients are computed on longer frames called texture windows (see previous section). The length of the texture window is 3 seconds, and the number of MFCC is 13. There are then 26 features for each 3 seconds of audio signal. As an example, for a 30-second signal, there are 10 vector of 26 features. A SVM classifier is used to classify each vector in a genre class. We use libSVM, a high-performance and easy to use open source library. Finally, each vector votes for a genre class, and the class with the maximum number of votes is attributed to the whole song. This system obtains results competitive with those obtained in most recent work on musical genre classification [15].

3) Evaluation databases and metrics: The audio indexing systems are evaluated with same databases and metrics as used in recent work. We detail them in the following.

The beat tracking database is the same as used in [8], which was originally provided by S. Hainsworth [37]. There are 222 files of several music genres. The files are mono, sampled at 44.1 kHz and have a length of approximately 60 seconds. The database was annotated by a trained musician, by recordings taps in time to the audio recordings. The annotations were then corrected and refined using synthesized beat sounds over each track (see [37] for details). Evaluating beat tracking systems is not a straightforward problem. Several evaluation metrics have been proposed and discussed in [7] and used also in [8]. We have chosen the metric “accept d/h” which was proposed in [7] as the best single metric for evaluating beat tracking systems. This metric is defined as the length of the longest continuous segment of well recognized beats (with an acceptance window of 17.5% the beat period) normalized by the total number of beats (the metric is expressed in %). Moreover this metrics allows cases where tapping occurs at twice or half the annotated rate: the metric is calculated for the three annotations (annotated beats, twice and half) and the higher result is chosen.

The chord recognition database is the same as used in [11]. It consists of 2 albums of the Beatles: Please Please Me (14 songs) and Beatles for Sale (14 songs). Audio signals are mono and sampled at 44.1 kHz. The database has been annotated by C. Harte et al [38]. As some chords in the database do not belong to the set of 24 recognized chords, these complex chords are mapped to their root triad as explained in [11]. We use a simple evaluation metric as proposed in [11], it is the percentage of well recognized frames.

The musical genre classification database is one of those used in several recent publications including [15]. It is a database originally provided by G. Tzanetakis [13]. It is composed by 1000 tracks classified in 10 genres (blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, rock), where each genre class contains 100 tracks. The tracks are mono, 30-second length and sampled at 22.05 kHz. As the coders need input signals sampled at 44.1 kHz , the tracks have been resampled at 44.1 kHz. To evaluate the systems, the standard 5-fold cross-validation procedure is used (as in [15]). The dataset is first randomly partitioned into 5 equal-size subsets. Then, 4 subsets are chosen to train the SVM model, and the remaining subset is evaluated using the train model. The classification accuracy is then the percentage of good classifications. To avoid biased results due to the random partitioning, the cross-validation procedure is repeated 100

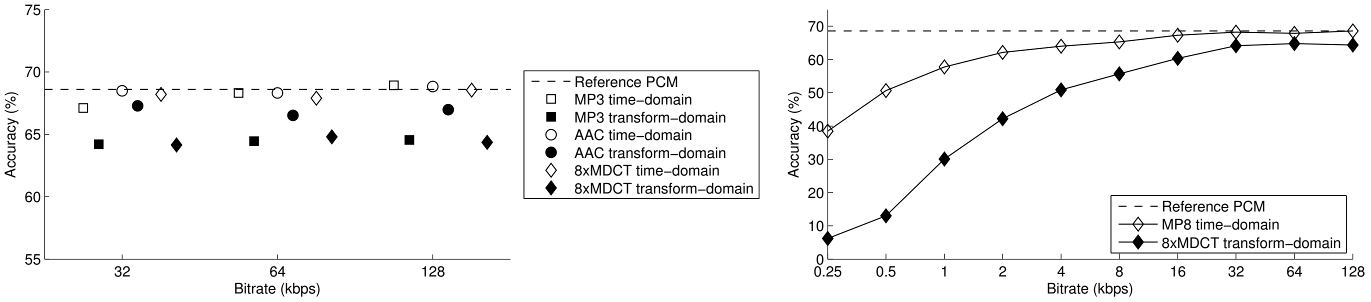

Fig. 8. Mean of the beat tracking accuracy. Left: performance of the 3 time-domain and transform-domain systems based on MP3, AAC and 8xMDCT for 3 bitrates 32, 64 and 128 kbps. Right: performance of the time-domain and transform-domain systems based on 8xMDCT for a wider range of bitrates. The dashed line corresponds to the performance of the reference system on the original PCM signal.

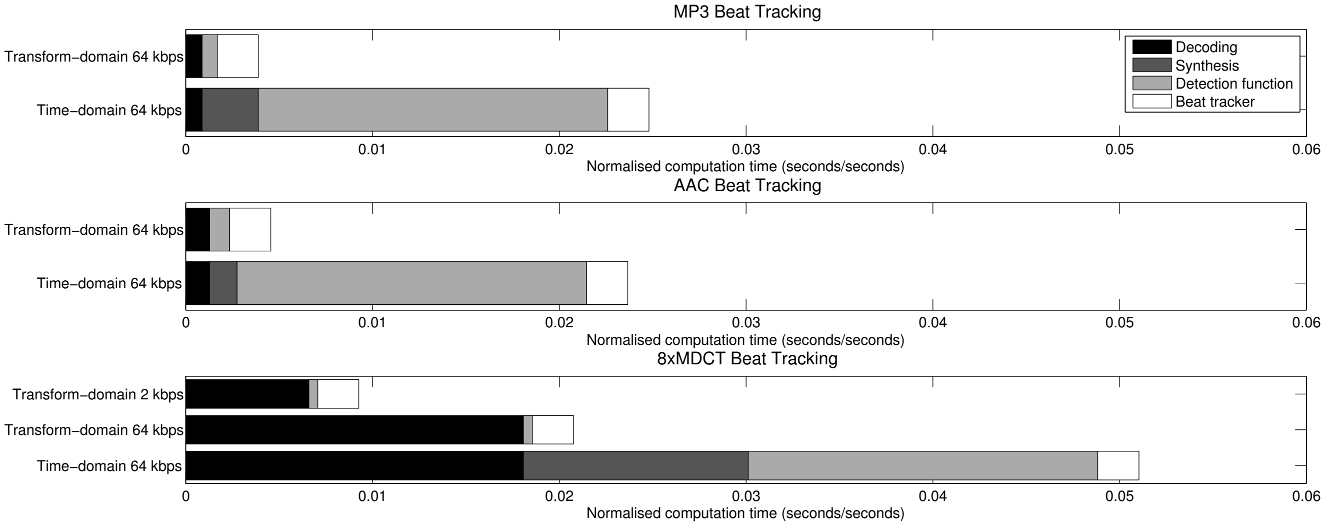

Fig. 9. Computation times of 3 time-domain and transform-domain beat tracking systems based on MP3, AAC and 8xMDCT.

times and the final result is the mean of the 100 classification accuracies.

B. Results and discussion

1) Performance: Figures 8, 10 and 12 show performance of respectively the beat tracking, chord recognition and musical genre classification systems. For each codec and each application, we give results for both time-domain (decoding+reference system) and transform-domain systems, we also compare these systems with the reference system on original PCM audio.

The results of the beat tracking systems (Fig. 8) show that all transform-domain systems obtain very good performance. At 32, 64 and 128 kbps, transform-domain systems obtain performance close to the corresponding time-domain systems. For the standard audio codecs MP3 and AAC, these results confirm that efficient transform-domain beat tracking is possible [23]. For the new 8xMDCT audio codec, these results show that efficient transform-domain beat tracking is possible too. Moreover, the performance of the 8xMDCT system decreases slowly with the bitrate, it is thus quite robust against the bitrate.

The results of the chord recognition systems (Fig. 10) show that, contrary to the beat tracking case, only the 8xMDCT transform-domain system obtains good performance. The transform-domain systems based on MP3 and AAC obtain bad performance, the reason is that their MDCT representation has too low frequency resolution (as explained in the previous section), this limitation prevents the calculation of an efficient chromagram, and thus prevent good performance in chord recognition. The transform-domain system based on 8xMDCT obtains performance close to the corresponding time-domain system. Moreover, its performance is robust against the bitrate, it decreases with the bitrate less slowly than the beat tracking system. This could be explained by the fact that the Matching Pursuit algorithm extract first components with highest energy which are in most cases components with long windows; at low bitrates, only components with highest energy are encoded; and consequently, at low bitrates, there are in great majority long-window components which are useful for chord recognition but not for beat tracking.

The results of the musical genre classification systems (Fig. 8) show that all transform-domain systems obtain very good performance (with the only exception of the systems based on MP3 at 32kbps, which is the lowest possible bitrate of this

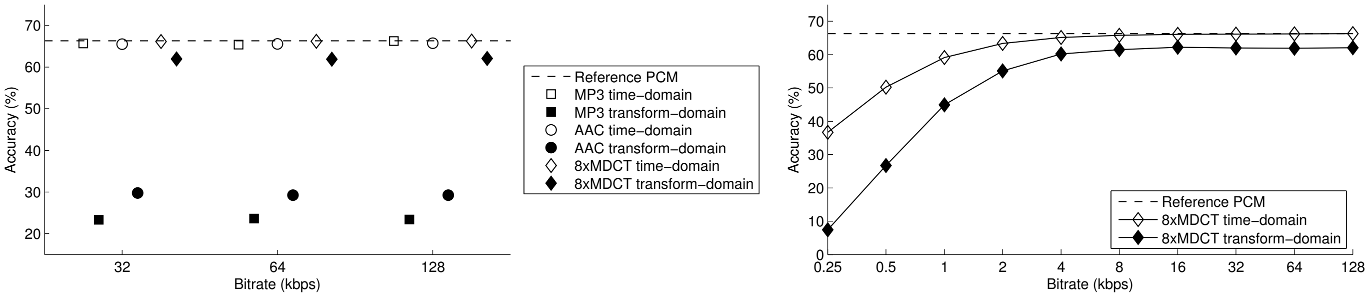

Fig. 10. Mean of the chord recognition accuracy. Left: performance of the 3 time-domain and transform-domain systems based on MP3, AAC and 8xMDCT for 3 bitrates 32, 64 and 128 kbps. Right: performance of the time-domain and transform-domain systems based on 8xMDCT for a wider range of bitrates. The dashed line corresponds to the performance of the reference system on the original PCM signal.

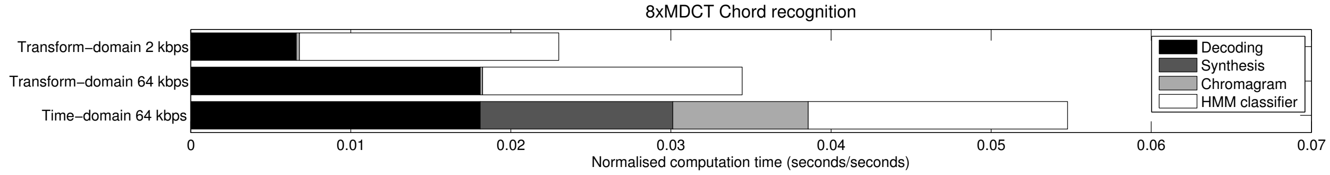

Fig. 11. Computation times of the time-domain and transform-domain chord recognition systems based on 8xMDCT.

codec and the quality of the decoded sound is very bad in this case). The transform-domain systems based on the standard coders MP3 and AAC obtain performances close to the corresponding time-domain systems. For the new 8xMDCT codec, the performance of the transform-domain system is even higher than the corresponding time-domain system. The transform-domain system based on 8xMDCT is highly robust against the bitrate with an accuracy above 50% at 1kbps (it is interesting to note that the listening quality of the decoded sound at this very low bitrate is extremely bad). Moreover, at high bitrate (above 128kbps), the performance is even higher than the reference system on the original PCM audio. This results shows that the proposed scale-dependant MFCC-like features are better features than the standard MFCC features for musical genre classification, being able to grab some of the temporal structures.

2) Computation times: Figures 9, 11 and 13 show computation times of respectively the beat tracking, chord recognition and musical genre classification systems. For each codec and each application, we give computations times for both time-domain (decoding+reference system) and transform-domain systems. Computation times were evaluated on a Laptop with a Core2duo 2.0Ghz processor, 2GB of memory, a Linux 64 bits system and Matlab 7.6 as front-end. Computation times are calculated on a whole database, then normalized by the total duration of the tracks.

These results first show that the transform-domain systems are always faster than the corresponding time-domain systems. This is due to two reasons: firstly, the transform-domain systems avoid the computation of the signal synthesis; secondly, the calculation of the mid-level representation is faster for the transform-domain case than for the time-domain case. The computation time of the transform-domain systems thus depends mainly on the two other operations, which are exactly in both cases: the decoding of the MDCT coefficients, and the machine learning stage.

In the case of the MP3 and AAC codecs, the decoding stage is very fast, much faster than the decoding stage of the 8xMDCT codec. Consequently, the systems based on MP3 and AAC are faster than the corresponding systems based on the 8xMDCT codec. However, it is interesting to note that the 8xMDCT codec is scalable, and thus the complexity of the decoding stage is also scalable. As an example, decoding at 2kbps is more than twice faster than decoding at 64kbps. This property allows user to balance complexity and performance of the 8xMDCT-based systems: decreasing the bitrate decreases the complexity but also decreases the performance, reversibly increasing the bitrate increases the performance but also increases complexity.

The computations times depend not only on the decoding stage, but also on the machine learning stage. In the case of the beat tracking and musical genre classification systems, the machine learning stage is very fast and thus it does not really influence the total times. However, in the case of the chord recognition systems, the machine learning stage is costly, it represents a large proportion of the total time, and thus the computation time ratios between the transform- and time-domain systems is lower than for the two other applications.

Finally, it is worth noting that the calculation of the mid-level representations in the transform-domain are very fast, with the only exception of the MFCC-like features of the 8xMDCT codec. In this case, the computation time is relatively high, this is due to the many calculations involved during the triangular-filtering stage. However, the total computation time of the transform-domain system still remains lower than the corresponding time-domain system; moreover, as seen previously, the transform-domain system obtains better performance than the time-domain system.

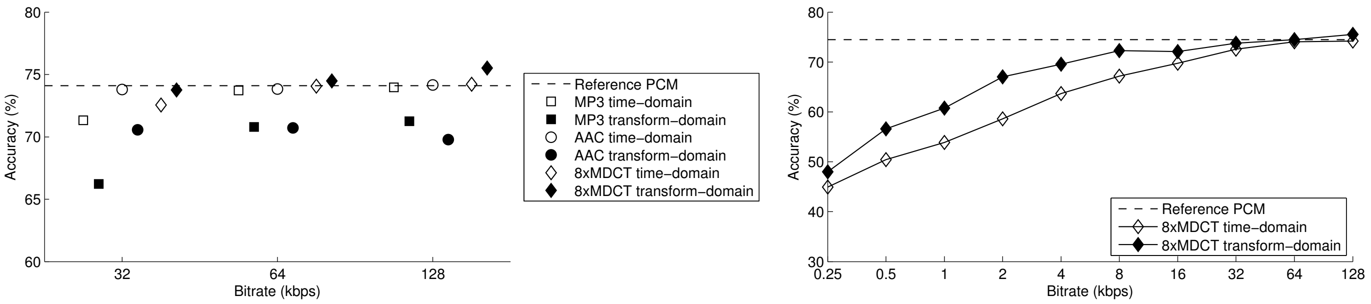

Fig. 12. Mean of the musical genre classification accuracy. Left: performance of the 3 time-domain and transform-domain systems based on MP3, AAC and 8xMDCT for 3 bitrates 32, 64 and 128 kbps. Right: performance of the time-domain and transform-domain systems based on 8xMDCT for a wider range of bitrates. The dashed line corresponds to the performance of the reference system on the original PCM signal.

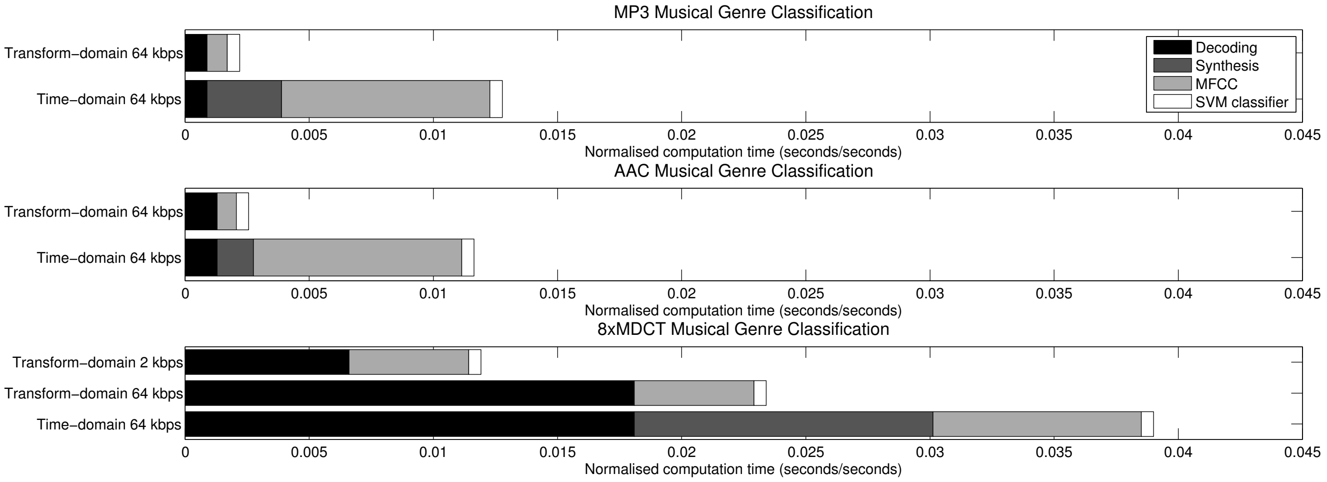

Fig. 13. Computation times of 3 time-domain and transform-domain musical genre classification systems based on MP3, AAC and 8xMDCT.

V. Conclusion

The main purpose of this paper was to investigate alternative audio signal representations for transform-domain audio indexing. Most existing work on transform-domain audio indexing deals with standard audio codecs such as MP3 and AAC. However, the internal audio signal representations used in these codecs have limitations such as limited frequency resolution, which prevent efficient transform-domain audio indexing for tonality-related applications such as chord recognition. We have thus investigated a new audio codec that is based on a sparse signal representation which does not have these limitations. We have shown that this new audio codec is able to give very good performance for several different transform-domain audio indexing applications, including beat tracking, chord recognition and musical genre classification. We have also shown that this new audio signal representation allows the calculation of MFCC-like features that give better performance than the standard MFCC features for musical genre classification. Finally, due to its scalability, the 8xMDCT codec has the advantage to allow a user to choose how to balance performance and complexity. Given a sound file encoded at high bitrate, the audio indexing system can decode any portion of the bitstream : decoding only the first few bits is very fast but decreases the performance, decoding more bits is slower but increases the performance.

This study opens new ways for designing audio signal representations. To be useful for both audio coding and audio indexing, an audio signal representation must provide not only a compact representation, but also must deliver an explicit information on the sound content. Future research will consider other techniques of audio signal representations, such as representations based on complex transforms (e.g. MCLT), or object-based audio signal representations (e.g. sinusoidal modeling or so-called ”molecular” representations).

Acknowledgment

The authors would like to thank Matthew Davies from QMUL and Juan P. Bello from NYU for providing their Matlab code. They also acknowledge Marcela Morvidone for useful discussions on scale-dependent MFCC coefficients.

References

[1] 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.

[2] ——, “Information technology - generic coding of moving pictures and associated audio information - part 3: Audio,” IS13818-3 1998.

[3] ——, “Information technology - coding of audio-visual objects - part 3: Audio,” IS14496-3 2001.

[4] ——, “Report on the MPEG-2 AAC stereo verification tests,” Feb. 1998, tech. report N2006.

[5] E. D. Scheirer, “Tempo and beat analysis of acoustic musical signals,” *J. Acoust. Soc. Am.*, vol. 103, no. 1, pp. 588–601, 1998.

[6] S. Dixon, “Automatic extraction of tempo and beat from expressive performances.” *J. New Music Research*, vol. 30, no. 1, pp. 39–58, 2001.

[7] A. P. Klapuri, A. J. Eronen, and J. T. Astola, “Analysis of the meter of acoustic musical signals,” *IEEE Trans. on Audio, Speech and Lang. Proc.*, vol. 14, no. 1, pp. 342–355, 2006.

[8] M. E. P. Davies and M. D. Plumbley, “Context-dependant beat tracking of musical audio,” *IEEE Trans. on Audio, Speech and Lang. Proc.*, vol. 15, no. 3, pp. 1009–1020, 2007.

[9] T. Fujishima, “Realtime chord recognition of musical sound: A system using common lisp music,” in *Proc. Int. Comput. Music Conf.*, 1999, pp. 464–467.

[10] A. Sheh and D. P. W. Ellis, “Chord segmentation and recognition using EM-trained hidden Markov models,” in *Proc. Int. Conf. Music Inf. Retrieval*, 2003, pp. 185–191.

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

[12] K. Lee and M. Slaney, “Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio,” *IEEE Trans. on Audio, Speech and Lang. Proc.*, vol. 16, no. 2, pp. 291–301, 2008.

[13] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. 10, no. 5, pp. 293–302, 2002.

[14] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl, “Aggregate features and ADABOOST for music classification,” *Machine Learning*, vol. 65, no. 2-3, pp. 473 – 484, 2006.

[15] A. Holzapfel and Y. Stylianou, “Musical genre classification using nonnegative matrix factorization-based features,” *IEEE Trans. on Audio, Speech and Lang. Proc.*, vol. 16, no. 2, pp. 424–434, 2008.

[16] N. Patel and I. Sethi, “Audio characterization for video indexing,” in *Proc. SPIE*, 1996, pp. 373–384.

[17] L. Yapp and G. Zick, “Speech recognition on MPEG/Audio encoded files,” in *Proc. IEEE Int. Conf. on Multimedia Computing and Systems*, 1997, pp. 624–625.

[18] Y. Nakajima, Y. Lu, M. Sugano, A. Yoneyama, H. Yamagihara, and A. Kurematsu, “A fast audio classification from MPEG coded data,” in *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, vol. 6, 1999, pp. 3005–3008.

[19] G. Tzanetakis and P. Cook, “Sound analysis using MPEG compressed audio,” in *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, vol. 2, 2000, pp. 761–764.

[20] Y. Wang and M. Vilermo, “A compressed domain beat detector using MP3 audio bitstreams,” in *ACM Multimedia*, 2001, pp. 194–202.

[21] X. Shao, C. Xu, Y. Wang, and M. Kankanhalli, “Automatic music summarization in compressed domain,” in *Proc. IEEE Int. Conf. Acoustics, Speech and Sig. Proc.*, vol. 4, 2004, pp. 261–264.

[22] S. Kiranyaz, A. F. Qureshi, and M. Gabbouj, “A generic audio classification and segmentation approach for multimedia indexing and retrieval,” *IEEE Trans. on Audio, Speech and Lang. Proc.*, vol. 14, no. 3, pp. 1062–1081, 2006.

[23] J. Zhu and Y. Wang, “Complexity-scalable beat detection with MP3 audio bitstreams,” *Computer Music Journal*, vol. 32, no. 1, pp. 71–87, 2008.

[24] S. Pfeiffer and T. Vincent, “Formalisation of MPEG- 1 compressed domain audio features,” Technical Report 01 / 196, CSIRO Mathematical and Information Sciences, Australia, Tech. Rep., 2001.

[25] E. Ravelli, G. Richard, and L. Daudet, “Union of MDCT bases for audio coding,” *IEEE Trans. on Audio, Speech and Lang. Proc.*, vol. 16, no. 8, pp. 1361–1372, Nov. 2008.

[26] LAME, “LAME mp3 encoder webpage,” 2008, http://lame.sourceforge.net.

[27] Nero, “Nero aac codec webpage,” 2008.

[28] iTunes, “Apple iTunes 7 webpage,” 2008, http://www.apple.com/fr/itunes/download/.

[29] J. Bello, C. Duxbury, M. Davies, and M. Sandler, “On the use of phase and energy for musical onset detection in the complex domain,” *IEEE Sig. Proc. Letters*, vol. 11, no. 6, pp. 553–556, 2004.

[30] S. Pauws, “Musical key extraction from audio,” in *Proc. of the 5th ISMIR*, 2004, pp. 96–99.

[31] E. Gomez and P. Herrera., “Estimating the tonality of polyphonic audio files: Cognitive versus machine learning modelling strategies.” in *Proc. of the 5th ISMIR*, 2004, pp. 92–95.

[32] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. 28, no. 4, pp. 357–366, 1980.

[33] F. Pachet and J. J. Aucouturier, “Improving timbre similarity: How high is the sky?” *J. Negative Results Speech Audio Sci.*, vol. 1, no. 1, 2004.

[34] P. Leveau, E. Vincent, G. Richard, and L. Daudet, “Instrument-specific harmonic atoms for mid-level music representation,” *IEEE Trans. on Audio, Speech and Lang. Proc.*, vol. 16, no. 1, pp. 116–128, 2008.

[35] libMAD, “libMAD mpeg audio decoder webpage,” 2008, http://www.underbit.com/products/mad/.

[36] FAAC, “FAAC and FAAD webpage,” 2008, http://sourceforge.net/projects/faac/.

[37] S. Hainsworth, “Techniques for the automated analysis of musical audio,” Ph.D. dissertation, Dept. Eng., Cambridge University, 2004.

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