# A NEW BITPLANE CODER FOR SCALABLE TRANSFORM AUDIO CODING

*Thi Minh Nguyet Hoang*[1], *Stéphane Ragot*[1], *Marie Oger*[*], *and Marc Antonini*[2]

[1]Orange Labs/TECH/SSTP, Av. Pierre Marzin, 22307 Lannion Cedex
[2]Lab. I3S-UMR 6070 CNRS and Univ. of Nice Sophia Antipolis, rte des Lucioles, 06903 Sophia Antipolis
E-mail: {thiminhnguyet.hoang, stephane.ragot}@orange-ftgroup.com, am@i3s.unice.fr

**ABSTRACT**

This paper proposes a new bit plane coding method for signed integer sequences. This method consists in mapping successive bit planes onto quinary symbols (+, -, 0, 1, $EoP$), where the symbol "$EoP$" stands for "End of Plane", and applying arithmetic coding. Sign bits are efficiently coded in combination with the corresponding most significant bit of non-zero integers. Moreover, bit planes are scanned and coded in a non-sequential manner to exploit the correlation between successive planes. Results for conversational transform coding of wideband speech and audio signals – sampled at 16 kHz – show that the performance/complexity of the proposed bitplane coder is near equivalent to non-embedded coding (stack-run coding), while offering additional flexibility (bitstream scalability).

***Index Terms***— Audio coding, speech codecs, entropy codes.

## 1. INTRODUCTION

Recently several scalable (or embedded) speech coders have been standardized in ITU-T: G.729.1 [1], G.711.1 [2], and G.718 [3]. These coders are all based on a *layered bitstream format*, to extend in bitrate and quality existing standards – respectively G.729, G.711, and G.722.2 (AMR-WB) at 12.65 kbit/s for one mode of G.718. *Bitstream scalability* is indeed a very attractive feature to develop new, improved coders in a smooth way by keeping interoperability with existing coding formats used in existing equipments [4].

In this work we study bit plane coding, which is one possible technique for achieving bitstream scalability. Bit plane coding consists in decomposing integer sequences in binary, from most significant bits (MSB) to least significant bits (LSB), and entropy coding each successive bit plane. The resulting bitstream is scalable by nature, since least significant bit planes may be skipped if needed. This technique has been extensively studied in audio coding (e.g. MPEG-4 BSAC [5], MPEG-4 SLS [6], proprietary audio coders [7, 8]), image coding (e.g. JPEG2000 [9]) and video coding (e.g. MPEG-4 FGS [10]). We focus here on speech and audio coding for conversational applications. Previous results in this context were reported in [11], with a coding approach based on a probability model of the signal spectrum. We revisit and extend this work by proposing a new, flexible bit plane coding method, with better efficiency and significantly lower complexity than the method described in [11].

This paper is organized as follows. In Section 2, we present the principle of bit plane coding and review a non-embedded coding technique known as stack-run coding. The new bitplane coding method is presented in Section 3. An application to transform coding of speech and audio signals is described in Section 4, including a performance/complexity analysis, before concluding on the proposed approach in Section 5.

This work was supported in part by the European Union under Grant FP6-2002-IST-C 020023-2 FlexCode.

[*]Marie Oger (marie.oger.sa@gmail.com) was with Orange Labs when this work was done.

## 2. ENTROPY CODING OF INTEGER SEQUENCES

We address the general problem of entropy coding a signed integer sequence $\mathbf{Y} = [y_1, \ldots, y_N]$, e.g. resulting from scalar quantization of transform coefficients. In the following we review the principle of bit plane coding, after presenting stack-run coding which serves in this work as a reference method with respect to coding efficiency.

### 2.1. Stack-run coding

Stack-run coding [12] is a lossless coding method designed initially for wavelet image coding. An example for the integer sequence $\mathbf{Y}$ is provided for the next discussion:

$$\mathbf{Y} = [0, 0, 0, +35, +4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -11] \qquad (1)$$

As shown in Fig. 1 (a), the integer sequence $\mathbf{Y}$ is mapped onto a quaternary sequence (+, -, 0, 1) and then coded by context-based arithmetic coding.

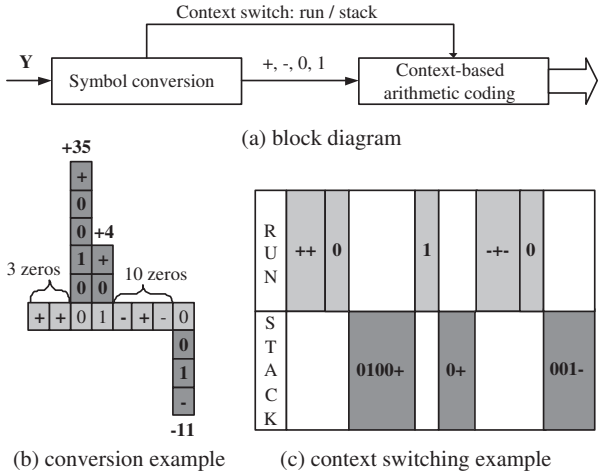


**Fig. 1**. Stack-run coding: principle and example.

The integer sequence $\mathbf{Y}$ is partitioned into two alternating contexts ("run" and "stack"): a *run* is a run length of zeros, while a *stack* is a non-zero integer value. The exact mapping rules onto the

quaternary alphabet are described in [12]. In short, run lengths are expressed in binary using "-" and "+" instead of usual "0" and "1". The symbols "0" and "1" are associated with the binary decomposition of a stack; the MSB of each stack (always equal to 1) is replaced by the associated sign ("+" or "-").

Fig. 1 (b) shows the conversion of sequence $\mathbf{Y}$. The three zeros at the beginning give a run length of 3 ("11" in binary) converted to "++". Then, based on [12], the stack value "35" is incremented by one to "36", giving "001001" in binary from LSB to MSB. Replacing the MSB by the "+" sign, the stack representation of "+35" then becomes "00100+". The rest of $\mathbf{Y}$ is converted in a similar way. Overall, the 16 integer elements of $\mathbf{Y}$ are mapped onto 19 quaternary symbols "++00100+10+-+-0001-", which are coded as shown in Fig. 1 (c) by alternating contexts.

An application of stack-run coding to transform coding of wideband signals (sampled at 16 kHz) is described in [13].

### 2.2. Bit plane coding

In bit plane coding, the integer sequence $\mathbf{Y}$ is first decomposed in binary with a sign-magnitude representation, to form a $(K+1)\times N$ binary matrix:

$$\mathbf{P}(\mathbf{Y}) = \left[\begin{array}{ccc} s_1 & \dots & s_N \\ \hline b_{K-1}(y_1) & \dots & b_{K-1}(y_N) \\ \vdots & & \vdots \\ b_0(y_1) & \dots & b_0(y_N) \end{array}\right] = \left[\begin{array}{c} \mathbf{S}(\mathbf{Y}) \\ \hline \mathbf{P}_{K-1}(\mathbf{Y}) \\ \vdots \\ \mathbf{P}_0(\mathbf{Y}) \end{array}\right] \tag{2}$$

where $K$ is the number of bit planes for magnitude, $s_i$ is the sign bit of $y_i$ and $b_k(y_i)$ is the $k^{th}$ bit in the binary decomposition of $|y_i|$. Then, bit planes $\mathbf{P}_k(\mathbf{Y})$ $(0 \le k < K)$, are coded successively from MSB to LSB by entropy coding. In general, the number of bit planes $K$ has to be coded. Moreover, the bit $s_i$ in the sign plane $\mathbf{S}(\mathbf{Y})$ is transmitted only if $|y_i| \neq 0$; in such a case the sign bit $s_i$ is said to be *significant*. To allow decoding of partially received coded data, $s_i$ is transmitted as soon as one of the coded bits $b_k(y_i)$ is equal to one.

The bit plane decomposition of the sequence given in Eq. 1 is:

$$\mathbf{P}(\mathbf{Y}) = \left[\begin{array}{c} 0000000000000001 \\ \hline 0001000000000000 \\ 0000000000000000 \\ 0000000000000001 \\ 0001000000000000 \\ 0001000000000001 \\ 0001000000000001 \end{array}\right] = \left[\begin{array}{c} \mathbf{S}(\mathbf{Y}) \\ \hline \mathbf{P}_5(\mathbf{Y}) \\ \mathbf{P}_4(\mathbf{Y}) \\ \mathbf{P}_3(\mathbf{Y}) \\ \mathbf{P}_2(\mathbf{Y}) \\ \mathbf{P}_1(\mathbf{Y}) \\ \mathbf{P}_0(\mathbf{Y}) \end{array}\right] \tag{3}$$

There are $K = 6$ bit planes. The 16 integer elements of $\mathbf{Y}$ are represented by 99 bits in total ($6 \times 16$ bits for magnitude + 3 significant sign bits). At first sight the bit plane representation appears to be very redundant. The coding method proposed hereafter aims at compacting this representation to reach an efficiency close to stack-run coding.

## 3. NEW BITPLANE CODER

The notations introduced in Section 2 are reused in this section. The bit plane coding method proposed in this paper is based on several observations:

- The most straightforward approach to code $\mathbf{P}(\mathbf{Y})$ consists in scanning each $\mathbf{P}_k(\mathbf{Y})$ sequentially (from $b_k(y_1)$ to $b_k(y_N)$). In general successive planes exhibit some correlations, which ought to be exploited.
- In particular, runs of zeros in $\mathbf{Y}$ produce columns of zero bits in $\mathbf{P}(\mathbf{Y})$. While stack run coding represents such runs by a form of run length coding with "+" or "-", bit plane coding typically repeats the encoding of zero runs for each bit plane.
- If the integer sequence $\mathbf{Y}$ represents transform coefficients of an audio signal, in many cases the spectrum is tilted in such a way that there is more energy in low frequency than in high frequency. In other words for most significant bit planes ($k$ close to $K-1$) the probability $P[b_k(y_j) = 0]$ with $j$ close to $N$ is quite high.

To cope with runs of zeros, run lengths can be coded using "+" or "-" symbols as in stack-run coding. To exploit correlation between bit planes, a two-pass coding approach is developed. To address the last point, a symbol "$EoP$" (End of Plane) is introduced.

### 3.1. Mapping onto 5 symbols and one-pass coding of MSB

The proposed method starts with converting bit planes to 5 symbols: "+", "-", "1", "0" and "$EoP$". For each bit plane $\mathbf{P}_k(\mathbf{Y})$ of $nb$ bits where $nb = N$ in case of MSB and $nb < N$ for remaining bit planes, the position $p_k$ of $EoP$ is defined as follows:

$$p_k = \arg \min_{1 \le i \le nb} \{\forall j \in [i, nb] \;\; b_k(y_j) = 0\} \tag{4}$$

The symbol "$EoP$" indicates that all bits $b_k(y_j)$ in $\mathbf{P}_k(\mathbf{Y})$ are zeros for $j \ge p_k$. This allows finishing the encoding of $\mathbf{P}_k(\mathbf{Y})$ at the position $p_k$ of "$EoP$".

In the MSB $\mathbf{P}_{K-1}(\mathbf{Y})$, except for the last run which is replaced by "$EoP$", all runs of zeros are converted to symbols "+" and "-". Run lengths are computed and written in binary and the bit values 0 and 1 are replaced by "-" and "+", respectively, as in stack-run coding [12]. If the run length is $2^n - 1$ ($n$ is integer), the MSB of the associated binary decomposition (always "+") is omitted. On the other hand, every time $b_k(y_j) = 1$ in the MSB, $b_k(y_j)$ is replaced by the sign bit $s_i$ (either "0" or "1").

For example, the MSB in the bit plane decomposition given in Eq. 3 is: $\mathbf{P}_5(\mathbf{Y}) = \left[\begin{array}{c} 0001000000000000 \end{array}\right]$. This bit plane is converted to the quinary sequence $[+ + 0 \;\; EoP]$, where "++" represent the first three zeros (3 verifies $2^2 - 1$); "0" represents the sign associated with the MSB value of 1 (+34 is positive) and the last symbol "EoP" represents the series of zeros at the end.

### 3.2. Two-pass coding of other bit planes

To exploit the correlation between the successive bit planes, the natural, sequential order $i = 1$ to $N$ is not used. Instead bit planes $\mathbf{P}_k(\mathbf{Y})$ with $k < K-1$ are coded with two passes which are conditional on previous bit planes $\mathbf{P}_m(\mathbf{Y})$, $m > k$.

In the first pass, only bits in $\mathbf{P}_k(\mathbf{Y})$ located at *significant positions* are coded. The remaining bits, which are not coded in this first pass, are coded in the second pass. To do so, we define the indicator $significant_k(i)$, $i = 1, \dots, N$, as:

$$significant_k(i) = \left\lfloor \frac{|y_i|}{2^{k+1}} \right\rfloor = \sum_{m=k+1}^{K-1} b_m(y_i) 2^m \tag{5}$$

where $\lfloor . \rfloor$ corresponds to the rounding of inferior integer. Based on the values $significant_k(i)$ each bit plane is partitioned in two sub-bitplanes: the first part contains only the bits at positions verifying $significant_k(i) \neq 0$, the second part contains the bits at positions

where $significant_k(i) = 0$. In other words, the *significant part* of $\mathbf{P}_k(\mathbf{Y})$ is defined as:

$$\mathbf{P}_k^{sig}(\mathbf{Y}) = \{b_k(y_i), i \leq 1 \leq N | significant_k(i) \neq 0\} \quad (6)$$

and the *non-significant part* is defined as:

$$\mathbf{P}_k^{nonsig}(\mathbf{Y}) = \{b_k(y_i), i \leq 1 \leq N | significant_k(i) = 0\} \quad (7)$$

In the first pass, $\mathbf{P}_k^{sig}(\mathbf{Y})$ is arithmetically coded directly in binary with 2 symbols ("0" and "1") using one context. In the second pass, $\mathbf{P}_k^{nonsig}(\mathbf{Y})$ is arithmetically coded with 5 symbols ("+", "-", "1", "0" and "$EoP$") using another context; the mapping onto 5 symbols is the same as for the MSB (see Section 3.1).

| $k$ | $\mathbf{P}_k^{sig}(\mathbf{Y})$ | $\mathbf{P}_k^{nonsig}(\mathbf{Y})$ | First pass | Second pass |
|---|---|---|---|---|
| 5 | 0001000000000000 | | $++0\ EoP$ | |
| 4 | 0 | 000000000000000 | 0 | $EoP$ |
| 3 | 0 | 000000000000001 | 0 | $-++1$ |
| 2 | 00 | 00010000000000 | 00 | $++0\ EoP$ |
| 1 | 101 | 0000000000000 | 101 | $EoP$ |
| 0 | 101 | 0000000000000 | 101 | $EoP$ |

A coding example is shown above for the bit plane decomposition given in Eq. 3. The MSB is coded as $[++0\ EoP]$ as explained in Section 3.1. The next bit planes are coded by the two-pass approach. In the first pass, significant positions are located. In this example, only the fourth position is significant in the plane below MSB. The associated bit value is "0", therefore $\mathbf{P}_4^{sig} = [0]$ and only symbol "0" is coded in the first pass. The remaining bits in $\mathbf{P}_4^{nonsig} = [000000000000000]$ are coded in the second pass, using one symbol $EoP$ in the quinary representation. For the next bit plane, again only the fourth bit with value "0" is coded in the first pass, so $\mathbf{P}_3^{sig} = [0]$. In the second pass, the remaining bits give $\mathbf{P}_3^{nonsig} = [000000000000001]$. After mapping onto 5 symbols, the sequence of 14 zeros is represented by "-+++"; the last bit value 1 is replaced by the sign "1" associated to $y_i = -11$. The symbol EoP is not used in this plane because the plane ends with "1". Subsequent bit planes are coded in a similar way. Consequently, 24 symbols are used to represent the input integer sequence $\mathbf{Y}$.

## 4. APPLICATION EXAMPLE

The proposed technique is evaluated using a predictive transform coding framework shown in Fig. 2. The input (mono) signal is sampled at 16 kHz and divided in blocks of 20 ms (320 samples). The encoder employs a linear-predictive weighting filter followed by MDCT coding. An elliptic high-pass filter (HPF) is applied to the input signal $x(n)$ in order to remove the frequency component under 50 Hz. Then, an 18th order LPC analysis and quantization described in [13] is applied. The signal is perceptually weighted in time domain by $W(z)$. The resulting signal $x_w(n)$ is transformed in frequency domain and further weighted (by pre-shaping). The transform coefficients $X_w(k)$ are divided by a step size $q$ and scalar quantized. The resulting integer sequence $\mathbf{Y}(k)$ is entropy-coded by bit plane coding or stack-run coding. A rate control is used to optimize the step size $q$ to match a constraint of fixed bit rate. The bit allocation to parameters is identical to [11] except that no bits are reserved for noise injection: 43 bits for LPC parameters, 7 bits for the quantization step, the remaining bits being allocated to entropy coding.

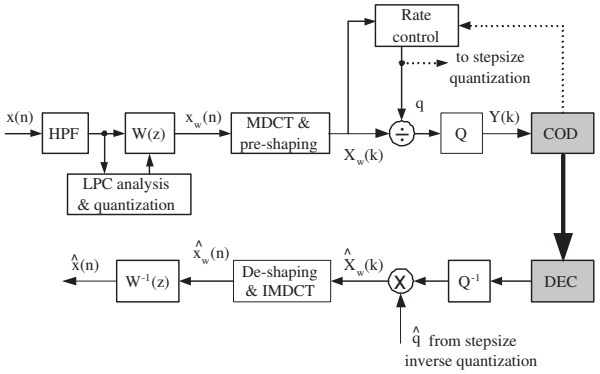


**Fig. 2**. Predictive transform coding framework.

### 4.1. Objective quality results

WB-PESQ [14] was used to evaluate objectively the quality of the proposed coder and compare it with reference coders. Only clean speech samples were used to compute the average MOS-LQO (Mean Opinion Score – Listening Quality Objective) scores at various bitrate. The bit rate varied from 16 to 40 kbit/s. Fig. 3 shows the results obtained for the reference coders (stack-run coder [13], model-based bitplane coder [11], and ITU-T G.722.1 [15]) and the proposed bit plane coder. Note that, except for G.722.1, all tested coders share the same coding framework illustrated in Fig. 2; the difference lies in the entropy-coding method, which impacts the selected quantization step $q$ for each coded MDCT frame. To evaluate the intrinsic entropy-coding performance, enhancements such as pre-echo reduction or noise injection/substitution were not employed. The G.722.1 scores are given for information, because WB-PESQ is not recommended for comparing different coding models.

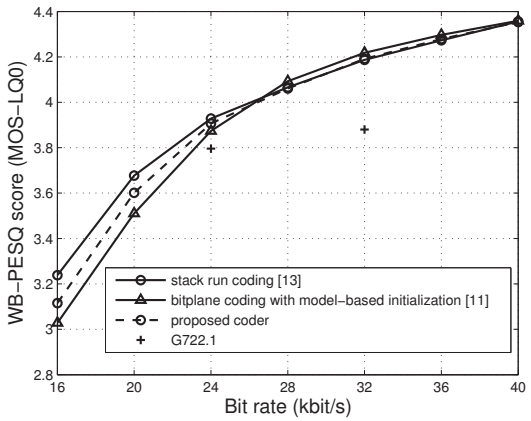


**Fig. 3**. Average WB-PESQ score when encoding and decoding at the same bitrate (*without* noise injection).

Fig. 3 confirms the results of [11] where it was shown that stack-run coding and model-based bit plane coding have very close performance, except at low bit rates (e.g. 16 kbit/s) where there is a gap around 0.2 MOS-LQO at low bit rates (e.g. 16 kbit/s). The new method proposed in this paper reduces further this gap to 0.1 MOS-LQO at low bit rates. The small performance difference between stack-run coding and the proposed method could be predicted from

the examples given previously in Sections 2 and 3, where the sequence Y in Eq.1 is represented by 19 and 24 symbols respectively.

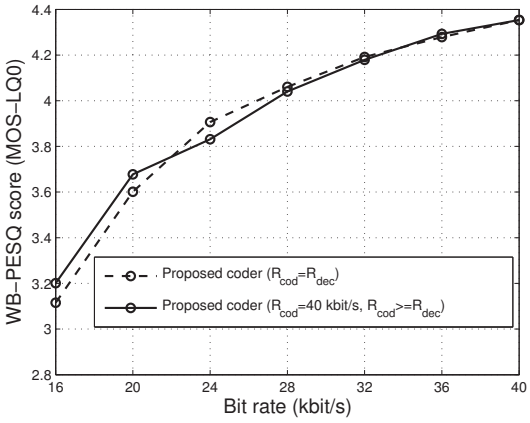


**Fig. 4**. Average WB-PESQ score with partial decoding (*without* noise injection).

To show the benefit of bitstream scalability, Fig. 4 presents the WB-PESQ results obtained for the proposed bit plane coder in the case of partial decoding. In one case (solid line), the encoder and decoder bit rates, $R_{cod}$ and $R_{dec}$, are forced to be identical (as in Fig. 3). In the other case (dashed line), the encoder bit rate is fixed ($R_{cod}$ = 40 kbits/s), the decoder operates at bit rates $R_{dec} \leq R_{cod}$. The latter case makes use of partial bit plane decoding. Fig. 4 shows that the partially decoded case performs slightly better than non hierarchical coder case above 20 kbit/s. The marginal difference at low bit rates between the two cases can be explained by the fact that the step size $q$ and the reconstruction of the integer sequence $\mathbf{Y}$ are optimized for each bit rate $R_{cod}$ in the former case, while they are optimized only once at the maximum of decoding bitrate 40 kbit/s in the latter case.

### 4.2. Subjective quality results

An informal subjective test for speech and music signals coded at 24, 32 and 40 kbit/s was carried out with 4 expert listeners. Four mono test signals sampled at 16 kHz were used: two speech samples (one male, one female) and two music samples (piano, trumpet). MUSHRA-like results showed that quality using the proposed method or stack-run coding is equivalent at 32 and 40 kbit/s, which is not the case at 24 kbit/s, where slightly more artefacts were noted for the bit plane coding version. Further work will be needed to address important issues, such as pre-echo reduction and noise injection/substitution, so as to minimize musical noise and pre-echo at 16 and 24 kbit/s.

### 4.3. Memory requirements and computational complexity

The proposed bit plane method requires no storage of coding tables (i.e. table ROM), like stack-run coding [13] and model-based bit plane coding [11]. Furthermore, the method proposed here has a computational complexity equivalent to that of stack-run coding. This is a differentiating factor compared with the model-based bit plane coding in [11], which involves estimating probability model parameters and calculating integrals; the computation load in [11] can be quite high and depends on the number of bit planes to be coded. The computational complexity of the proposed coder may be reduced by exploiting the embedded nature characteristic of bitplane coding (setting fixed step size and using partially encoding bit planes).

## 5. CONCLUSION

In this paper we proposed a new method of bit plane coding which makes use of two-pass arithmetic coding with 5 symbols (+, -, 0, 1, $EoP$) including a special "End of Plane" symbol. Using a predictive transform coding framework for signals sampled at 16 kHz, the proposed coder was compared with stack-run coding [13], and model-based bitplane coding [11]. It was shown that the resulting objective performance from 16 to 40 kbit/s is very close to the non-embedded reference method (stack-run coding), despites the additional feature of bitstream scalability. Subjective quality results confirmed these findings at 32 and 40 kbit/s. However, at 24 kbit/s the actual quality difference appeared to be more pronounced, which requires further work on important aspects such as noise injection/substitution or pre-echo reduction. This paper focused mainly on entropy coding efficiency.

Note that the proposed method may be applied to image or video coding, though some parts such as the "End of Plane" symbol are tuned for transform coefficients of audio signals.

## REFERENCES


[1] S. Ragot and al., "ITU-T G.729.1: An 8-32 kbit/s scalable coder interoperable with G.729 for wideband telephony and Voice over IP," in *Proc. ICASSP*, April 2007.

[2] Y. Hiwasaki and al., "G.711.1: a wideband extension to ITU-T G.711," in *EUSIPCO*, Aug. 2008.

[3] T. Vaillancourt and al., "ITU-T EV-VBR: a robust 8-32 kbit/s scalable coder for error prone telecommunications channels," in *EUSIPCO*, Aug. 2008.

[4] B. Geiser, S. Ragot, and H. Taddei, "Embedded Speech Coding: From G.711 to G.729.1," in *Advances in Digital Speech Transmission (R. Martin, U. Heute, C. Antweiler, eds.)*, chapter 8, pp. 201–248. Wiley, Jan. 2008.

[5] S.H. Park and al., "Multi-layer bit-sliced bitrate scalable audio coding," in *AES 103rd Convention*, Aug 1997.

[6] R. Yu and al., "MPEG-4 scalable to lossless audio coding," in *AES 117th convention*, 2004.

[7] C. Dunn, "Efficient audio coding with fine-grain scalability," in *AES 111th convention*, Sept. 2001.

[8] J. Li, "Embedded audio coding (EAC) with implicit auditory masking," *ACM Multimedia 2002*, Dec 2002.

[9] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Trans. on Image Proc.*, vol. 9, pp. 1158–1170, July 2000.

[10] H. Radha, M. v.d. Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Trans. Multimedia*, vol. 3, Mar. 2001.

[11] T.M.N. Hoang, M. Oger, S. Ragot, and M. Antonini, "Embeddedd transfrom coding of audio signals by model-based bit plane coding," in *Proc. ICASSP*, Apr 2008, pp. 4013–4016.

[12] M.J. Tsai and al., "Stack-run image coding," *IEEE Trans. Circuits Syst. Video Techno.*, vol. 6, pp. 519–521, October 1996.

[13] M. Oger, S. Ragot, and M. Antonini, "Transform audio coding with arithmetic-coded scalar quantization and model-based bit allocation," in *Proc. ICASSP*, 2007, vol. 4, pp. 545–548.

[14] ITU-T Rec P.862.2, *Wideband extension to Recommendation P.862 for the assessment of wideband telephone networks and speech codecs*, Nov 2005.

[15] ITU-T G.722.1, *Coding at 24 kbit/s and 32 kbit/s for Hand-free Operations in Systems with Low Frame Loss*, 1999.