

LABORATOIRE



INFORMATIQUE, SIGNAUX ET SYSTÈMES  
DE SOPHIA ANTIPOLIS  
UMR 6070

# A SCALABLE VIDEO CODER WITH SCAN-BASED LIFTED MCWT AND MODEL-BASED BIT-RATE ALLOCATION

*Thomas André, Marco Cagnazzo, Marc Antonini, Michel Barlaud*

*Projet CREATIVE*

Rapport de recherche  
ISRN I3S/RR-2004-42-FR

Décembre 2004

---

RÉSUMÉ :

Nous présentons un codeur vidéo scalable à base de transformées en ondelettes. Ce codeur gère la scalabilité spatiale, temporelle et en qualité à la demande, et est entièrement compatible avec le standard industriel de compression d'images fixes JPEG-2000. Il utilise une nouvelle classe de transformées en ondelettes par schéma lifting au fil de l'eau et compensées en mouvement, les transformées (N,0), spécialement adaptées au codage vidéo. Les sous-bandes temporelles résultantes sont traitées par un codeur EBCOT, qui fournit une représentation scalable et efficace du signal au sein d'un flux binaire compatible avec JPEG-2000. Un algorithme basé modèle efficace est proposé pour allouer les ressources disponibles de façon optimale entre les sous-bandes temporelles. L'information sur le mouvement est également encodée avec EBCOT, de sorte que la plupart du décodage peut être effectué par n'importe quel décodeur standard. Nous montrons que le codeur proposé, fondé sur des techniques émergentes, atteint d'ores et déjà un bon niveau de performances, tout en étant totalement scalable.

MOTS CLÉS :

Codage vidéo, transformée en ondelettes, schéma lifting, compensation en mouvement, allocation binaire basée modèle, scalabilité, JPEG-2000

---

ABSTRACT:

We present a new scalable wavelet-based video coder. This coder is able to provide on-demand spatial, temporal and SNR scalability, and is fully compatible with the industrial standard for still image compression JPEG2000. It features a new class of scan-based motion-compensated temporal filters, called (N,0) lifting scheme, which are specifically developed for video coding. The resulting temporal subbands are processed by an EBCOT coder which builds an efficient and fully scalable representation of the signal within a JPEG2000-compatible bitstream. An efficient model-based algorithm is proposed in order to optimally allocate the coding resources between the temporal subbands. The motion information is encoded using EBCOT as well, so that most of the decoding process can be performed by any standard-compliant decoder. We show that the proposed coder, based on emerging techniques, already reaches a good level of performances while being fully scalable.

KEY WORDS :

Video Coding, Wavelet Transform, Lifting Scheme, Motion Compensation, Model-Based Bit Allocation, Resource Allocation, Smooth Scalability, JPEG-2000

# A scalable video coder with scan-based lifted MCWT and model-based bit-rate allocation

Thomas André\*, Marco Cagnazzo, Marc Antonini, Michel Barlaud *Fellow, IEEE*,

## Abstract

We present a new scalable wavelet-based video coder. This coder is able to provide on-demand spatial, temporal and SNR scalability, and is fully compatible with the industrial standard for still image compression JPEG2000. It features a new class of scan-based motion-compensated temporal filters, called  $(N,0)$  lifting scheme, which are specifically developed for video coding. The resulting temporal subbands are processed by an EBCOT coder which builds an efficient and fully scalable representation of the signal within a JPEG2000-compatible bitstream. An efficient model-based algorithm is proposed in order to optimally allocate the coding resources between the temporal subbands. The motion information is encoded using EBCOT as well, so that most of the decoding process can be performed by any standard-compliant decoder. We show that the proposed coder, based on emerging techniques, already reaches a good level of performances while being fully scalable.

## EDICS Category: 1-VIDC

### I. WAVELET-BASED VIDEO CODING

The recent years have seen the impressive performance growth of video compression algorithms. The latest developing standard, known as MPEG-4 part 10 or H.264 [1], [2], is by now capable of 60% bit-rate saving for the same quality with respect to the standard MPEG-2.

Nevertheless, some problems are still not completely solved. The next generation of video coders will have to deal with higher resolution and higher frame-rate sequences. Even though the current video

T. André, M. Antonini and M. Barlaud are with I3S Laboratory, UMR 6070 of CNRS, Université de Nice-Sophia Antipolis - Bât. Algorithmes/Euclide B, 2000 route des Lucioles - BP121 - 06903 Sophia-Antipolis Cedex, France - Tel. +33 4.92.94.27.49 - Fax. +33 4.92.94.28.98 - emails: {andret,am,barlaud}@i3s.unice.fr

M. Cagnazzo is with Dipartimento di Ingegneria Elettronica e delle Telecomunicazioni, Università Federico II di Napoli, via Claudio 21 - 80125 Napoli, Italy - Tel. +39 81.7683154 - Fax. +39 81.7683149 - email: cagnazzo@unina.it

coding standards are very good at compressing today's sequences in QCIF, CIF or even SD formats, some emerging techniques may prove to be much more efficient for encoding high-definition television (HDTV) or digital cinema (DC) sequences. Furthermore, the generalization of these new, large formats will inevitably create new needs, such as scalability. A *scalable* bit-stream is composed by embedded subsets, which are efficient compression of original data, but at a different resolution (both spatially or temporally) or quality. In other words, the user should be able to extract from a part of the full-rate, full-resolution bit-stream (for example DC) a degraded version of the original data, i.e. with a reduced resolution or an increased distortion (for example adapted to HDTV or even to internet streaming) and with no transcoding.

The recent standards already offer a certain degree of scalability, such as the Fine Grain Scalability (FGS) in the MPEG-4 standard [3], which is well-suited for internet streaming applications. However, the quality of a video sequence built from subsets of a scalably encoded bitstream is usually poorer than the one of the same sequence separately encoded at the same bit-rate, but with no scalability support.

Moreover, these new standards do not provide any convergence with any emerging still-images compression algorithm such as JPEG2000. Thus, they are not able to exploit the widespread diffusion of hardware and software JPEG2000 codecs which is expected for the next years. However, a video coder should take advantage of a fast JPEG2000 core encoding algorithm that combines good compression performances, especially for large images, and a full scalability support.

All these considerations have lead many video compression research works towards *Wavelet Transform* (WT). WT has been used for many years in still image coding, proving to offer superior performances with respect to *Discrete Cosine Transform* (DCT) and a natural and full support of scalability due to its multiresolution property [4]–[6]. For these reasons, WT is used in the new JPEG2000 standard, but the first attempts to use WT in video coding date back to late 80s [7]. It was soon recognized that one of the main problems was how to perform *Motion Compensation* (MC) in the WT framework [8], [9]. This problem has been solved with Motion-Compensated Lifting Scheme [10]. With this approach, WT-based video encoders begin to compare to the last generation of DCT-based coders in terms of performances [11]–[14].

The objective of the *Motion-Compensated Wavelet-Transform* (MCWT) -based video coder presented in this paper is to obtain good performances, comparable to state-of-the-art algorithms, with a full and flexible support to scalability, and with a large compatibility with JPEG2000. The main problems in developing this video coder are the choice of the temporal filters and their implementation, and the

resource allocation between the *subbands* (SBs) produced by the temporal filter(s).

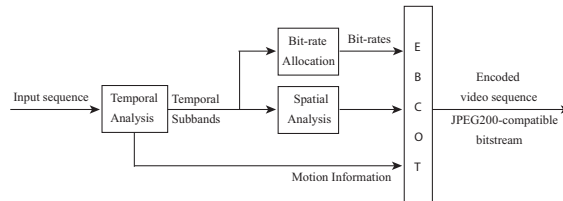


Fig. 1. General structure of the proposed encoder.

The paper is organized as follows: it goes through the different parts of the proposed coder, of which the structure is described in Fig. 1. The input video data firstly undergoes a temporal analysis based on the motion-compensated lifted wavelet transform presented in the section II. This temporal analysis outputs motion vectors and temporal wavelet subbands (section II). The motion information is encoded losslessly using an EBCOT coder, while the remaining available bit-rate is then distributed among the temporal subbands by an optimal bit-rate allocation algorithm presented in the section III. This algorithm is eventually run several times according to the scalability needs, as shown in section IV. The temporal subbands are then processed by a JPEG2000 coder, at the bit-rates specified by the rate allocation algorithm. In section IV, we also review the performances of the proposed coder, in terms of scalability capabilities and in terms of PSNR.

## II. SCAN-BASED MOTION-COMPENSATED LIFTED WAVELET TRANSFORM

### A. Motion-compensated wavelet transform

The wavelet transforms proved to be very powerful tools for still image coding. WT-based encoders achieve better performances than those based on Direct Cosine Transforms (DCT) in terms of compression rate. WT can also be easily used for a multiresolution analysis, which is the key to scalability features.

In spite of its success in the field of still image coding, early attempts to adapt WT to video coding have been quite deceiving, for two main reasons.

Firstly, as soon as there is some motion, the video signal is far from being as correlated in the temporal domain as in the spatial domain. In fact, WT cannot be efficiently used for the temporal filtering, unless some motion compensation is applied. However, the motion compensation operation is a non-linear process, which suppresses the perfect reconstruction property of the motion-compensated WT. The lifting

scheme [15] is an efficient and elegant solution to this problem. Each regular transversal WT can be implemented as an equivalent lifting scheme [16], whose prediction and update steps can be modified to take motion compensation into account without altering the perfect reconstruction property. Motion-compensated lifting schemes, mostly based on the 5/3 wavelet kernel, obtain much better performances than uncompensated temporal WTs.

Secondly, the performances of the hybrid video coders have been steadily improving. Even though the currently best performing wavelet-based video coders easily outperform the old standard MPEG-2, they still don't compare to the latest standards such as MPEG-4 SVC and H.264, based on DCT and temporal prediction. Obviously, the weakness of the wavelet-based video coders is the temporal analysis, since the 2D wavelet coders outperform the DCT-based coders. More precisely, whereas the temporal analysis of H.264 requires one Motion Vector Field (MVF) per frame and is very flexible, the 5/3 temporal wavelet analysis requires up to 4 MVFs per frame in average, depending of the number of decomposition levels that are usually chosen. The motion bit-rate overhead is too heavy a penalty, and thus, it must be reduced.

To this end, the (2,0) lifting scheme presented in [17], [18] appears to be a very simple and yet interesting alternative to the (2,2) lifting scheme that corresponds to the 5/3 transversal WT. The (2,0) lifting scheme is obtained from the (2,2) by suppressing the update step; the low-pass filtering is then reduced to a simple temporal sub-sampling of the original sequence. The expression of the motion-compensated temporal high-pass and low-pass filters of the (2,0) lifting scheme is thus the following:

$$\begin{cases} h_k[\mathbf{m}] = x_{2k+1}[\mathbf{m}] - \frac{1}{2}(x_{2k}[\mathbf{m} + \mathbf{v}_{2k+1 \rightarrow 2k}(\mathbf{m})] \\ \quad + x_{2k+2}[\mathbf{m} + \mathbf{v}_{2k+1 \rightarrow 2k+2}(\mathbf{m})]), \\ l_k[\mathbf{m}] = x_{2k}[\mathbf{m}] \end{cases} \quad (1)$$

where  $x_k$ ,  $h_k$  and  $l_k$  are respectively the  $k^{th}$  input frame, high-frequency coefficient and low-frequency coefficient, and  $\mathbf{v}_{i \rightarrow j}$  is the motion vector that displaces the pixel  $\mathbf{m}$  of the image  $x_i$  to the corresponding pixel in the image  $x_j$ .

Even though the produced low-pass subband (SB) suffers from aliasing, its effects are not very visible on a well motion-compensated video sequence. Besides this aliasing problem, the (2,0) filtering is advantageous, since it requires twice as few MVFs than the (2,2).

Additionally, Konrad has shown in [19] that the Motion-Compensated Lifting Scheme does not exactly implement its equivalent motion-compensated WT, unless the MVFs satisfy certain conditions of

invertibility and additivity. In general, these conditions are not satisfied. As shown in [18], the  $(2, 0)$  lifting scheme does not require any condition on the motion vectors, and does not propagate any motion compensation error due to the non-additivity of the motion vectors.

For these reasons, we chose to implement the  $(2, 0)$  motion-compensated lifting scheme as temporal analysis for our coder. In the next section, we describe in details the implementation of the temporal wavelet filtering.

### *B. Scan-based filtering*

Unlike the temporal analysis of H.264 which can be performed independently on the successive frames of the input video sequence, the wavelet filtering is a continuous process. In order to compute the wavelet coefficients corresponding to a given sample of the input sequence, the neighboring samples are needed. The first and the last sample of the sequence require a specific process, since they have only one neighbor.

The wavelet filtering of a  $1D$  signal or, to some extent, of a  $2D$  signal with a small-enough support, could be eventually performed all at the same time. But it is impossible to filter a full  $2D + t$  signal (i.e. a video sequence of several dozens of frames) along the temporal axis without splitting it into several smaller parts. Indeed, the memory requirements would be too heavy. For this reason, the temporal WT is generally computed independently on consecutive groups of frames (temporal blocks). The first and the last frames of each temporal block require a specific process.

However, this partitioning of the input data into temporal blocks is not without consequences. One of them is that each group of wavelet coefficients suffer from border effects (Fig. 3), which alter the decorrelation efficiency and thus the final performances of the coder. More precisely, these border effects are due to the specific process applied on the first and the last frames of each block. These frames don't have enough neighbors to be filtered by the full wavelet. Thus, the missing frames are usually replaced by their symmetrical frame compared to the frame being filtered (Fig. 2a). As a result, the border frames are not filtered by the original wavelet, but undergo another transform which is less efficient.

For example, in the case of the  $(2, 2)$  or the  $(2, 0)$  transform, the last high frequency subband coefficient is actually a simple prediction of the last frame from its neighbor. For longer filters, there are even more border frames that would be concerned by the problem; similarly, the number of badly-filtered frames increases with the number of temporal decomposition levels. To sum up, the temporal groups of frames have to be large enough to limit the influence of the border effects, but small enough to fit into the

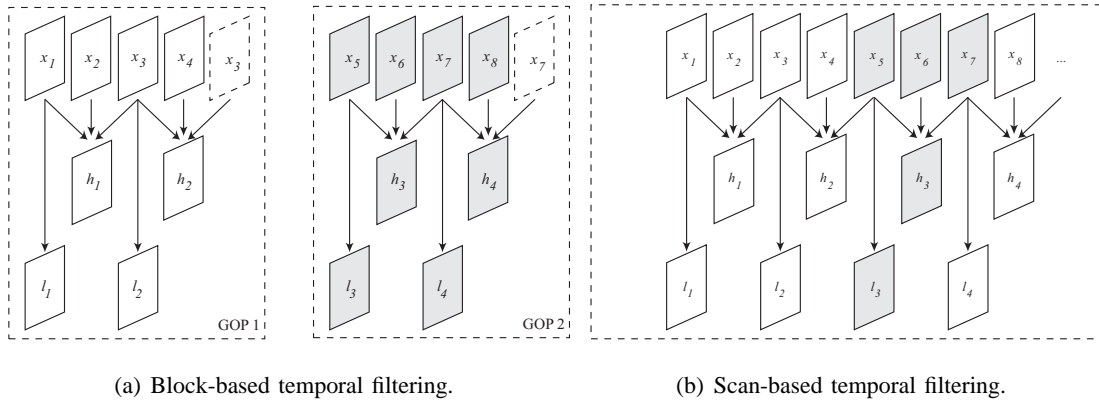


Fig. 2. Comparison between block-based and scan-based temporal filtering (1 decomposition level), on the first 8 frames of a sequence  $x_k$ , producing low-pass  $l_k$  and high-pass  $h_k$  coefficients. The block-based method requires certain frames (dashed) to be symmetrized, which causes the transform to be modified. The scan-based method computes directly the correct transform without any symmetrization. It also requires less frames to be loaded in the memory at the same time (grayed frames).

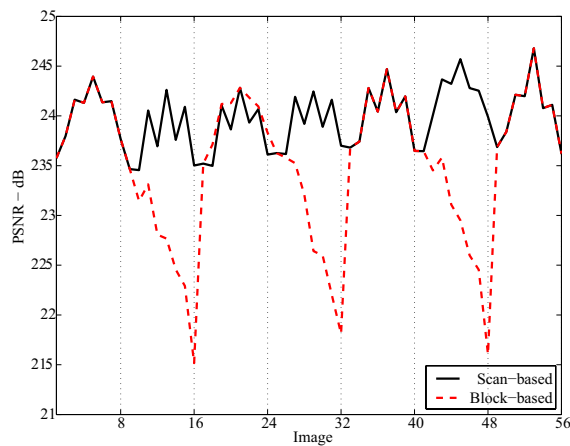


Fig. 3. Comparison between block-based (dashed curve) and scan-based (solid curve) temporal filtering: example on the first 56 images of the sequence "Mobile & Calendar", with 3 temporal decomposition levels. In both cases, the rate allocation is performed on groups of 8 "temporally coherent" coefficients (see section II-C), hence the 8-periodic oscillations of the curves. In the block-based case, the temporal transform is computed on groups of 16 images; the necessary symmetrizations (Fig. 2a) cause the observed PSNR gaps. In the scan-based case, there is no symmetrization, and thus no PSNR gap causing flickering effects.

memory; the number of temporal decomposition levels should be limited, and the wavelet filters should be small enough to reduce the number of border frames affected by the border effects. All these conditions are not only very constraining, they also greatly affect the performances of the WT.

The scan-based WT has thus been introduced as an alternative to this block-based “wavelet transform”. The scan-based filtering was first proposed to remove the blocking artifacts caused by 2D wavelet transforms of large satellite images that had to be split into blocks [20]. The technique has been successfully adapted to WTs [21].

The scan-based temporal WT does not require the entire input sequence to be split into temporal blocks or to be loaded all at the same time into the memory. The wavelet coefficients are computed progressively as the input sequence is acquired by the encoder. Thanks to this sliding-window filtering, only the few frames required to compute a given wavelet coefficient are loaded into the memory at the same time. The border effects discussed above concern only the first and the last frames of the sequence; in the case of the  $(2, 0)$  lifting scheme, only the last frame is involved (Fig. 2b).

The advantages of the scan-based WT over the block-based WT for the temporal filtering are twofold. First, the memory requirements of the scan-based WT are much lower. And second, the whole sequence is filtered by the true wavelet without any border effect, excepted at the beginning and at the end of the sequence. This last point causes the scan-based WT to be much more efficient than the block-based WT. Additionally, larger filters and more temporal decomposition levels can be used to further improve the performances.

### *C. Bit-rate allocation and scan-based filtering*

Although the implementation of the scan-based WT itself is straightforward, the encoding of the wavelet subbands has to be carried out very carefully. After the motion-compensated temporal transform stage, the available bit-rate has to be distributed between the resulting wavelet coefficients. This operation must be adapted to the way the coefficients are computed.

Depending on the number of temporal decomposition levels, the coefficients of the highest resolution require a certain number of input frames to be processed first. For example, in the case of the  $(2,0)$  filters, 3 input images are needed in order to produce one pair of coefficients (HF and LF) of the first level (Fig. 2b). This pair of coefficients is temporally aligned with the two first input images from which they have been computed. We call these coefficients “temporally coherent”. Similarly, for the second decomposition

level, 3 of the previous LF coefficients (obtained from 7 input frames) are required to compute one pair of  $2^{nd}$ -level coefficients. These two  $2^{nd}$ -level coefficients, along with the first 4  $1^{st}$ -level coefficients from which they derive, are called “temporally coherent” (Fig. 4). In a general way, for a given filter and  $N$  temporal decomposition levels, we call “temporally coherent” the set of  $2^N$  subband coefficients resulting from the same  $2^N$  input images.

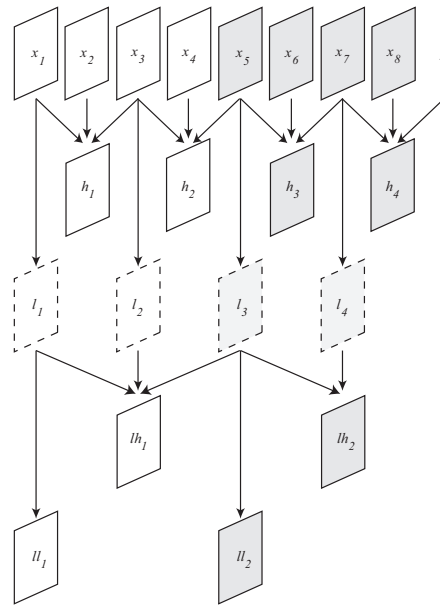


Fig. 4. Temporal coherency of wavelet subbands: example with 2 levels of decomposition. The  $1^{st}$ -level low-pass subband  $(l_i)_i$  (dashed frames) is used to compute the second-level coefficients  $(lh_i)_i$  and  $(ll_i)_i$ . The coefficients  $l_i$ ,  $lh_i$  and  $ll_i$  corresponding to the first 4 frames  $x_i$  are temporally coherent (white frames) and should be analyzed altogether. Similarly, the next 4 coefficients (grayed frames) are temporally coherent, because they correspond to the same 4 input images. Thus, they should not be processed separately.

The bit-rate allocation algorithm described in the next section performs a rate-distortion analysis on the subbands. For this operation to be efficient and accurate, temporally coherent coefficients have to be processed at the same time. Consequently, depending on the available memory, the rate-distortion analysis has to be run on groups of  $K \cdot 2^N$  coefficients, from which  $K \cdot 2^{N-n}$  belong to the  $n^{\text{th}}$  resolution, where  $K$  is an integer constant.

The bit-rate allocation process is described in more details in the next section.

### III. SPATIAL ANALYSIS AND RESOURCE ALLOCATION

In the spatial analysis stage, the temporal subbands (SBs) produced by the temporal analysis undergo a spatial WT, resulting in a global three-dimensional separable WT. Then, these WT coefficients have to be encoded.

Each temporal subband can be seen as a set of images. Thus, JPEG2000-like algorithms are well-suited for encoding them. A simple encoding of each SB using an EBCOT coder provides a full compatibility with the JPEG2000 standard and assures very good performances. Indeed, the lowest temporal subband is composed by nothing but images, for which JPEG2000 is very efficient; concerning the encoding of the higher frequency SB, the context-based arithmetic coder used by EBCOT proved to have competitive performances as well.

Once the encoding technique has been chosen, the main problem is the resource allocation. In other words, we have to decide how much bit-rate should be assigned to each temporal SB, in order to achieve the best possible performances. This problem is challenging as it requires to model accurately the encoder, to find an analytical description of the problem, to solve it and to find a feasible implementation of the solving algorithm. These topics are discussed in the following.

#### A. Stating the problem

In this section, we consider the problem of allocating the coding resources between temporal subbands. This operation presents two kind of difficulties: the first one is related to the definition and the modelling of the general framework, and the second one is related to the computational complexity of the possible solutions.

In a very general way, this problem can be described as follows. Let us consider  $M$  sets of data to encode, using a given encoding technique. We can consider the temporal SBs resulting from a MCWT as these sets of data. The resource allocation problem consists in finding a rate-allocation vector,  $\mathbf{R}^* = \{R_i^*\}_{i=1}^M$  such that, when the  $i$ -th set is encoded using the given encoding technique at the bit-rate  $R_i^*$  for each  $i \in \{1, 2, \dots, M\}$ , then a suitable cost function is minimized under certain constraints. This allocation is then optimal for the chosen encoding technique.

In our case, as previously noted, the sets of data are the SBs. The distortion of the decoded sequence can be chosen as a cost function; the constraint is then imposed on the total bit-rate, which must be lower than a given threshold (Rate Allocation Problem). Note that another possibility would be to choose the total bit-rate as a cost function, which would have to be minimized under a constraint on the total distortion (Distortion Allocation Problem). These two problems show a deep symmetry, and, as we will

see later, can be solved in a very similar way using the approach proposed below; in the following, we will focus on the Rate Allocation problem.

Thus, we aim to define a framework for optimal resource allocation in the context of Motion-Compensated WT-based video coding. In order to approach this problem, several tools are needed, as we have to model the global distortion, to solve analytically the allocation problem, and to find an algorithm capable of extracting the optimal solution. Once the resolving algorithm has been found, another problem is to find an efficient yet simple implementation.

In the next subsection, we review briefly some existing resource allocation methods presented in the literature.

### *B. Existing Solutions to the Resource Allocation Problem*

The allocation of coding resources has been early recognized as a key issue in transform coding and in subband coding problems. A first analytical approach is due to Huang and Schultheiss, who stated the theoretical optimal bit-rate allocation for generic transform coding in the high-resolution hypothesis [22]. They derived a formula which defines the optimal bit-rate to be allocated to each set of data, depending on their variances. Unfortunately, this simple and elegant solution only holds when a high rate is available for encoding. Shoham and Gersho proposed in [23] an optimal algorithm with no restriction on the target bit-rate; however, it requires the computation of the RD characteristics for each possible quantization step, and thus its computational complexity is high. Ramchandran and Vetterli presented in [24] an RD approach to encode adaptive trees using generalized multiresolution wavelet packets.

For the general case, an analytical expression of optimal rates has not been found, and different approaches have been applied. The most successful and widespread ones consist in modelling the relationship between the RD characteristics of the data sets and the global RD characteristics. The target is to find an optimal allocation condition on the *data sets rates*, which assures the minimization of distortion for a given maximal rate of the *reconstructed data*.

For example, the well known SPIHT algorithm models the relationship between the quantization of wavelet coefficients and the distortion of the reconstructed image. The most recent still image compression standard JPEG2000 divides the wavelet coefficients into code-blocks, and then defines an optimality condition on their RD curves which assures the minimum distortion of reconstructed image. For the case of orthogonal subband coding, Gersho and Gray showed in [25] that the global distortion can be

expressed as a sum of the subbands distortions:

$$D(\mathbf{R}) = \sum_{i=1}^M D_i(R_i) \quad (2)$$

Usevitch extended this approach to the case of biorthogonal WT [26], bringing in some examples for the common Daubechies filters. In this case, (2) should be modified by using some weights which account for the non-orthogonality of the filters:

$$D(\mathbf{R}) = \sum_{i=1}^M w_i D_i(R_i) \quad (3)$$

A crucial step of the rate allocation algorithm is thus the estimation of each subband's RD curve. A first and simple approach consists of evaluating each curve at many points: each subband must be encoded and decoded several times at different rates, and the resulting distortions computed and stored; we call such an approach "brute force approach". Unfortunately, in order to obtain accurate estimates of each curve in the whole range of possible rate allocation values, many test points are required. So this approach is extremely complex. More sophisticated approaches have been proposed in the literature. For example, scalar-quantized coefficients can be represented using generalized gaussian models [27].

In the following, we first present our approach for the resource-allocation problem in the framework of WT-based video coding. We then introduce our contribution, which mainly consists of three parts: extending the existing distortion models to the case of  $(N, 0)$  temporal filtering; developing an analytical solution to both rate allocation and distortion allocation problems; defining a model for subbands RD curves, in order to lower the computational complexity of the algorithm while improving its accuracy.

### C. Rate allocation problem

In this subsection, we approach the resource allocation problem. We start by recalling some existing results, and we extend them to our specific case.

We assume that a SB encoding technique (in our case, we chose EBCOT/JPEG2000) has been designated. Let  $D_i(R_i)$  be the Distortion-Rate curve for the  $i$ -th SB and for the given encoding technique. We take the Mean Square Error (MSE) between the original and the quantized subband as a distortion measure, while the rate  $R_i$  is expressed in bit per pixel (bpp).

Let us first explicit the cost function to be minimized. In the rate allocation problem, the cost function is the distortion  $D(\mathbf{R})$  of the reconstructed sequence, which must be expressed as a function of the rate allocation vector  $\mathbf{R} = \{R_i\}_{i=1}^M$ .

We extend the results obtained by Usevitch in [26] for the case of WT decomposition by Daubechies 9/7 or 5/3 filters, to the case of  $(N, 0)$  Lifting Schemes. It is worth noting that, while Daubechies' 9/7 filters (and, to some extent, also 5/3 filters) are very close to be orthogonal as their weights are close to 1,  $(N, 0)$  filters are far from being orthogonal. Without a correct weighting, the distortion model won't be valid at all, and thus, the allocation won't be optimal. As an example, we computed these weights  $w_i$  for the biorthogonal 5/3 and Daubechies 9/7 filters, and for the (2,0) Lifting Scheme, i.e. the 1/3 filter. We considered four levels of temporal decomposition. This table highlights how far  $(N, 0)$  filters are from orthogonality, and thus how important the weighting is in this case.

Filter	Subband				
	H	LH	LLH	LLLH	LLLL
9/7	1.040435	1.022700	1.005267	0.988131	0.933540
5/3	1.4375	1.07813	0.808594	0.606445	0.316406
1/3	2	1.5	1.125	0.84375	0.316406

TABLE I

TEMPORAL SUBBANDS WEIGHTS (4 DECOMPOSITION LEVELS) FOR SOME BIORTHOGONAL FILTERS

Once the correct weights have been obtained for each temporal subband, the expression of the global distortion described in the equation (3) can be used safely.

Let us now focus on the constraint imposed on the total bit-rate of the subbands  $R_{SB}$ . This total bitrate should be smaller than a target value  $R_{MAX}$ . The relationship between the total bit-rate  $R_{SB}$  and the SBs bit-rates  $R_i$  (all of them expressed in bit per pixel) is:

$$R_{SB} = \sum_{i=1}^M a_i R_i \quad (4)$$

where  $a_i$  is the fraction of total pixels in the  $i$ -th subbands. Namely, if  $P_i$  is the number of pixels in the  $i$ -th SB,

$$a_i = \frac{P_i}{\sum_{i=1}^M P_i} \quad (5)$$

Thus, the constraint to be imposed can be written as follows:

$$\sum_{i=1}^M a_i R_i \leq R_{MAX} \quad (6)$$

In conclusion, the rate allocation problem consists in finding  $\mathbf{R}$  which minimize the cost function (3) under the constraint (6).

This problem can be easily solved using a Lagrangian approach. We introduce the Lagrangian functional  $J(\mathbf{R}, \lambda)$ :

$$J(\mathbf{R}, \lambda) = \sum_{i=1}^M w_i D_i(R_i) - \lambda \left( \sum_{i=1}^M a_i R_i - R_{\text{MAX}} \right) \quad (7)$$

By imposing the zero-gradient condition, we find that the resulting optimal rate allocation vector  $\mathbf{R}^* = \{R_i^*\}_{i=1}^M$  verifies the following set of equations:

$$\frac{w_i}{a_i} \frac{\partial D_i}{\partial R_i}(R_i^*) = \lambda \quad \forall i \in \{1, \dots, M\} \quad (8)$$

where  $\lambda$  is the Lagrange multiplier. We can read (8) this way: the optimal rates correspond to points having the same slope on the “weighted” curves  $(R_i, \frac{w_i}{a_i} D_i)$ . Note that  $\lambda < 0$  since the RD curve are strictly decreasing.

A simple dichotomic search algorithm is proposed to find the optimal rate allocation vector. Let us introduce the set of functions  $R_i(\lambda)$ , defined implicitly by the following equation:

$$\frac{w_i}{a_i} \frac{\partial D_i}{\partial R_i}(R_i) \Big|_{R_i=R_i(\lambda)} = \lambda \quad (9)$$

In other words, the value of  $R_i(\lambda)$  is the rate of the  $i$ -th subband which corresponds to a slope  $\lambda$  on the weighed RD curve of this SB. The rate allocation problem consists in finding the slope value  $\lambda^*$  so that the equation (10) is verified:

$$\sum_{i=1}^M a_i R_i(\lambda^*) = R_{\text{MAX}} \quad (10)$$

The solution is found by an iterative algorithm. Let  $\varepsilon$  be an acceptable tolerance. If  $j$  represents the number of successive attempts, it is sufficient to find the first value  $\lambda^{(j)}$  so that

$$\left| \sum_{i=1}^M a_i R_i(\lambda^{(j)}) - R_{\text{MAX}} \right| < \varepsilon \quad (11)$$

A guess interval for slope values, say  $[\lambda_{\min}^{(0)}, \lambda_{\max}^{(0)}]$ , is firstly chosen. Then, we initialise  $j = 0$ , and we set  $\lambda^{(0)} = \frac{1}{2}(\lambda_{\min}^{(0)} + \lambda_{\max}^{(0)})$ . Now, as long as the condition (11) is not satisfied, the values of  $\lambda$  are updated as follows:

$$\begin{cases} \lambda_{min}^{(j)} = \lambda^{(j-1)} \\ \lambda_{max}^{(j)} = \lambda_{max}^{(j-1)} \end{cases} \text{ if } \sum_i R_i(\lambda^{(j-1)}) < R_{MAX} \quad (12)$$

$$\begin{cases} \lambda_{min}^{(j)} = \lambda_{min}^{(j-1)} \\ \lambda_{max}^{(j)} = \lambda^{(j-1)} \end{cases} \text{ otherwise} \quad (13)$$

Finally, the new guess for the slope value is:

$$\lambda^{(j)} = \frac{1}{2} \left( \lambda_{min}^{(j)} + \lambda_{max}^{(j)} \right) \quad (14)$$

Once the condition (11) is verified, each subband  $i$  can be encoded at the bit-rate  $R_i$ . The global distortion of the reconstructed sequence is then minimized for the given target bitrate  $R_{MAX}$ .

#### D. Distortion allocation problem

The dual problem, the Distortion Allocation problem, can also be addressed with a similar approach to the one presented above. In this case, the cost function is the SBs total rate  $R_{SB}$ :

$$R_{SB}(\mathbf{R}) = \sum_{i=1}^M a_i R_i \quad (15)$$

which should be minimized under a constraint on the global distortion:

$$D(\mathbf{R}) = \sum_{i=1}^M w_i D_i(R_i) \leq D_{MAX} \quad (16)$$

We obtain the following Lagrangian functional:

$$J(\mathbf{R}, \lambda) = \sum_{i=1}^M a_i R_i - \lambda \left( \sum_{i=1}^M w_i D_i(R_i) - D_{MAX} \right) \quad (17)$$

and, by imposing again the zero-gradient condition, we obtain:

$$\frac{w_i}{a_i} \frac{\partial D_i}{\partial R_i}(R_i^*) = \frac{1}{\lambda} \quad \forall i \in \{1, \dots, M\} \quad (18)$$

This means, once again, that the optimality condition is the uniform slope on the weighted curves  $(R_i, \frac{w_i}{a_i} D_i)$ . The algorithm proposed to find the best allocation vector is then very similar to the previous one. Indeed, it is sufficient to change the termination condition, which is now:

$$\left| \sum_{i=1}^M w_i D_i(R_i(\lambda^{(j)})) - D_{MAX} \right| < \varepsilon \quad (19)$$

while the values of  $\lambda$  must be updated as follows:

$$\begin{cases} \lambda_{min}^{(j)} = \lambda^{(j-1)} \\ \lambda_{max}^{(j)} = \lambda_{max}^{(j-1)} \end{cases} \text{ if } \sum_i w_i D_i(\lambda^{(j-1)}) > D_{MAX} \quad (20)$$

$$\begin{cases} \lambda_{min}^{(j)} = \lambda_{min}^{(j-1)} \\ \lambda_{max}^{(j)} = \lambda_{max}^{(j-1)} \end{cases} \text{ otherwise} \quad (21)$$

### E. Model-Based RD curves estimation

In order to use one of the algorithms presented in III-C and in III-D, we should be able to compute, for each  $\lambda$  and for each subband, the function  $R_i(\lambda)$  (or  $D_i(\lambda)$ ). In other words, the RD curves of each subband are needed.

The “brute force approach” referred to in section III-B is not only extremely heavy, it is also quite imprecise. Indeed, the presented allocation algorithms work on the derivatives of the RD curves as well. Unfortunately, even if numerous points are computed on each curve, the RD curves obtained with the brute force method are still discrete. Thus, the derivatives computed from these discrete curves are far from being regular and even present some discontinuities, especially at low bitrates (Fig. 5b). Consequently, there is little reason to consider them as monotonic and smooth, and the rate allocation algorithm can’t be robust nor precise when used with those curves.

The method presented below was first proposed in [28]. It addresses the problem of robustness and precision while its complexity is negligible compared to the one of the “brute force approach”.

We introduce a parametric model based on splines, which describes accurately the RD curves with only a few parameters. Each curve is determined by a limited number of points computed experimentally, and by an interpolation method, which can be based on interpolation splines, or smoothing splines [29]. These two models include some intrinsic regularity constraints on the curves. In the case of the smoothing splines, an additional parameter must be introduced, which is a smoothing factor. In both cases, the resulting curve model describes regular curves by their analytical equations. Thus, it is straightforward to obtain the analytical expression of the first derivatives of the RD curves, which is what we finally need in order to apply the proposed rate allocation algorithms. We note explicitly that spline description of the RD curves and their derivatives is very compact and can be obtained from a few sample points with a very little computational effort.

Many experiments were carried out in order to verify the efficiency of the model. In all our experiments, spline proved to provide a very good fit to any RD curve, even if different SBs have quite different curves.

For example, the lowest frequency SBs have a very steep curves for the lower range of rate and much more flat curves for higher rates. On the contrary, high frequency SBs have more regular RD curves. Nevertheless, the proposed approach is able to well represent any RD curve, usually with as few as 7 to 10 points designated empirically.

An example is shown in Fig.5a, where we report as a reference the “experimental” RD curve for the highest frequency SB computed after 4 levels of motion-compensated temporal decomposition on the first 16 frames of the “foreman” sequence (solid line). This curve has been obtained by encoding and decoding the SB at 200 different rates. On the same graph, we reported the parametric representations of this curve as well (dotted lines). These curves have been obtained by using just 7 points, namely those highlighted with a circle. We used both interpolation and smoothing splines, and the results in both cases appear to be satisfactory, as the original curve and its parametric representations are almost indistinguishable.

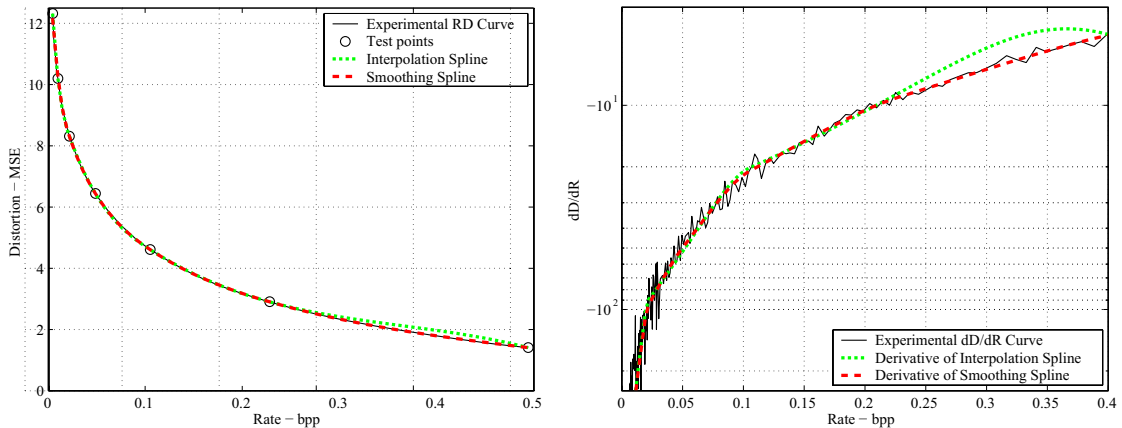
In Fig. 5b, we reported the first derivatives of the same “experimental” curve and of the spline curves. The “experimental” curve must be approximated from the 200 experimental points, whereas the computation of the spline derivatives can be easily accomplished analytically. The resulting spline curves do not show the irregularities which characterize the experimental data. It means that when the allocation algorithm looks for points with the same derivative, we have more robust results, especially at low bit rates.

In conclusion, the model-based representation of RD curves is computationally very efficient (as it mainly requires the computation of a few sample points of the RD curves), and also very reliable. It appears to be a very useful tool for applying the proposed optimal bit-rate allocation algorithm.

#### IV. PERFORMANCES

In the previous sections, we described the different parts of the proposed video coder. The temporal analysis part described in the section II computes temporal subbands and motion vectors. The motion information is losslessly encoded with an EBCOT coder; the remaining bitrate is distributed between the temporal subbands using the optimal allocation algorithm described in the section III. The EBCOT coder is then used to encode these subbands.

As shown in the following, the resulting video coder presents interesting performances. We first review its scalability features, and then compare its global level of performances on several different test sequences.



(a) Spline approximations of an “experimental” RD curve (b) Derivatives of the RD curves presented above. The (solid curve) composed by 200 points computed experimentally. The interpolation-spline curve (dotted curve) and the smoothing-spline curve (dashed curve) have been obtained by interpolating the 7 marked points.

Fig. 5. The smoothing-spline curve seems to match perfectly the “experimental” curve (a). Moreover, its derivative fits better to the real data than the interpolation-spline curve (b). The obtained RD curve and its derivative are smooth and continuous.

In all the experiments, we used a simplified motion description, based on  $16 \times 16$  blocks at half-pixel precision. The motion vectors were computed using an iterative diamond-search algorithm which minimizes a color criterion [30]. As test sequences, we chose several videos which present different characteristics: “Garden & flowers” which is a very regular sequence, “Foreman” in which the motion is much more chaotic, “Waterfall” which features a regular zooming camera motion.

#### A. Scalability

In a general way, a scalable bitstream has no better performance than what can be reached by encoding directly the sequence at the desired resolution, frame-rate and bit-rate. So we call *scalability cost* the difference between the quality (expressed in terms of PSNR) of the scalable bitstream decoded at a different resolution, frame-rate or bit-rate from the original, and the quality that could have been achieved by directly encoding the original sequence with the desired parameters.

Moreover, the introduction of scalability increases the complexity of the encoding algorithm. A second kind of scalability cost is thus the complexity increase of the encoding algorithm.

A *smoothly scalable* encoder should have a null or very little scalability cost, i.e. the same (or almost the same) performances of its non-scalable version, with the same (or almost the same) complexity.

Building such a video coder is not an easy task, as scalability imposes an additional constraint on the encoding algorithm, which can easily affect the global performances. In [31], Li deeply investigated this problem, in the general case and more specifically for MPEG-4 Fine Grain Scalability (FGS). He showed that the hybrid video coders are usually strongly affected by the scalability cost. For example, a gap of several dB of PSNR separates MPEG-4 FGS from its non-scalable version (in particular for time scalability). Concerning the complexity, it is not easy to make general considerations; however, scalability usually imposes tight constraints on the encoder's architecture, which can seriously affect its complexity.

WT-based encoders have a much easier job with scalability, thanks to the multiresolution analysis. Nevertheless, some problems remain to be solved, mainly related to resource allocation and low-pass filtering effects. In the following, we adapt the presented video coder in order to obtain a *smooth scalability*. The proposed method, firstly presented in [32], preserves the bit-rate allocation optimality regardless of the chosen bit-rate, frame-rate or resolution. The resulting coder is capable of achieving almost the same performances as the non-scalable version, and at almost the same computational cost.

We will use the following notations. Let  $R^{(0)}$  be the bit-rate budget available for the SBs. The non-scalable encoder must distribute these resources between the  $M$  SBs, finding the optimal rates vector  $\mathbf{R}^{(0)} = \{R_i^{(0)}\}_{i=1}^M$ , under the constraint  $\sum_{i=1}^M a_i R_i^{(0)} = R^{(0)}$ .

1) *Rate scalability*: The rate scalability should allow to decode the bitstream at a set of predefined bit-rates  $R^{(N)} < \dots < R^{(1)}$  different from the encoding bit-rate  $R^{(0)}$ . Since the  $i^{th}$  spatiotemporal SB is scalably encoded using EBCOT, we could truncate its bitstream at any arbitrary rate  $R_i^{(j)}$ , provided that  $\sum_{i=1}^M a_i R_i^{(j)} = R^{(j)}$ . However, with such a simple strategy, if the sequence is decoded at the  $j$ -th bit-rate, we lose the optimal bit-rate allocation property.

To overcome this problem, we perform in advance the bit-rate allocation for each target bit-rate  $R^{(j)}$ , which computes the optimal vector  $\mathbf{R}^{(j)} = \{R_i^{(j)}\}_{i=1}^M$ . The allocation must be repeated for each one of the  $n$  target bit-rates, until  $n$  optimal rate vectors are obtained for each SB. Then, as shown in Fig. 6, we can encode the  $i$ -th subband with the  $n$  quality layers corresponding to the bit-rates  $R_i^{(j)}$  (for  $j = 1, \dots, n$ ). Finally, we regroup all the layers corresponding to the same level. Thus, in order to decode the sequence at the given bit-rate  $R^{(j)}$ , we simply decode each SB at the quality level  $j$ .

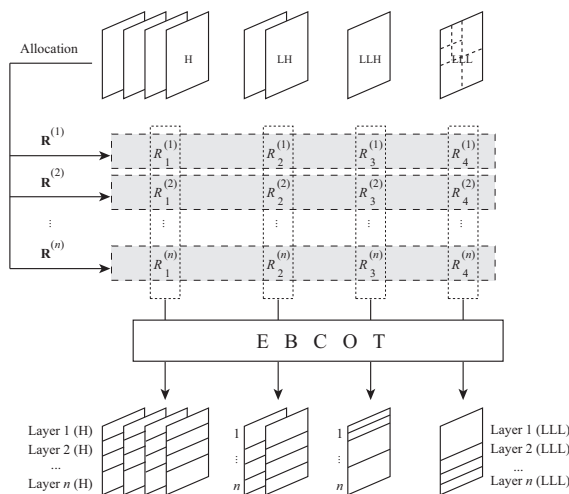


Fig. 6. Optimal bit-rate allocation for quality scalability. Example with 3 temporal decomposition levels (4 subbands, H, LH, LLH and LLL) and  $n$  quality layers. The bit-rate allocation algorithm is repeated for each target bit-rate  $R^{(i)}$  corresponding to a quality layer  $i$  (dashed parts). Thus,  $n$  sets of optimal bit-rates are computed for each subband (dotted parts).

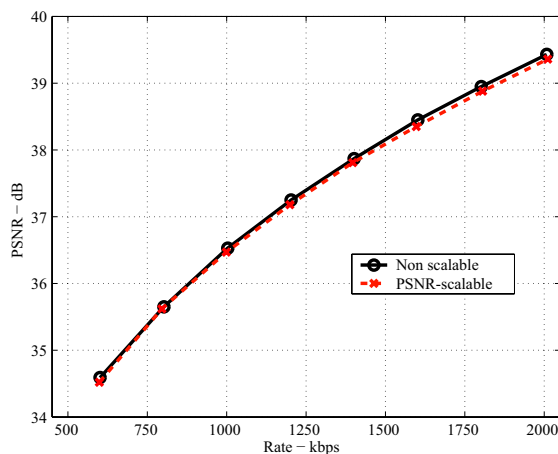


Fig. 7. Comparison of the PSNR curves for the sequence “foreman” (first 64 images), encoded with the quality scalability feature (dashed curve) and independently for each bit-rate (solid curve). 3 temporal decomposition levels were used. The block matching was performed using  $16 \times 16$ -blocks and a half pixel precision.

In order to evaluate the cost of this scalability method, we compared the PSNR of the test sequences encoded and decoded at the same bit-rates, with the following two methods: the first one consists in encoding each sequence separately for each target bit-rate; the second consists in producing only one scalable bitstream for each sequence, and then decoding it for each bit-rate. It appears that, regardless of the demanded bit-rate, the scalable bitstream is almost identical to the non-scalable bitstream, since

the SBs allocation is optimal in both cases. The only difference is the additional headers required for the quality layers. As an example, experimental results for the sequence “foreman” are reported in Fig. 7 and in Table II. In all test configurations, we noted that the proposed method assures a very little and practically negligible performance degradation, never greater than 0.1 dB.

Rate kbps	PSNR [dB]	PSNR [dB]	Scalability
	non-scalable	scalable	Cost [dB]
600	34.59	34.52	<b>0.07</b>
800	35.65	35.62	<b>0.03</b>
1000	36.53	36.47	<b>0.06</b>
1200	37.25	37.18	<b>0.07</b>
1400	37.87	37.81	<b>0.06</b>
1600	38.45	38.36	<b>0.09</b>
1800	38.95	38.88	<b>0.07</b>
2000	39.43	39.36	<b>0.07</b>

TABLE II

PSNR COST OF THE QUALITY SCALABILITY, FOR THE SEQUENCE “FOREMAN” (FIRST 64 IMAGES). 3 TEMPORAL DECOMPOSITION LEVELS WERE USED. THE BLOCK MATCHING WAS PERFORMED USING  $16 \times 16$ -BLOCKS AND A HALF PIXEL PRECISION.

We note that the MV information is not affected by the bit-rate scalability, as we still need the same vectors than for the non-scalable case.

We also stress that the proposed method only requires the allocation algorithm to run  $N$  times instead of once, if  $N$  quality layers are needed. The complexity of the allocation algorithm is much lower than those of the motion estimation part, or the WT part, as shown in section III-C. The overall complexity increase is thus negligible.

2) *Temporal scalability*: The proposed video coder involves a temporal wavelet-based multiresolution analysis. Thus, it is straightforward to obtain a temporal subsampled version of the compressed sequence from the encoded bitstream, by decoding selectively the lower temporal SBs.

However, when generic temporal filters are used, such as the  $5/3$  filters, reconstructing the sequence without the higher temporal SBs is equivalent to reconstructing a subsampled *and filtered* version of input sequence. This temporal filtering causes ghosting and shadowing artifacts. On the contrary, when

$(N, 0)$  filters are employed, the temporal low pass filtering is a pure subsampling. Thus, reversing the WT of a sequence without using the higher temporal SBs is equivalent to reversing the WT of its temporal subsampled version.

Moreover, the  $(N, 0)$  filters allow the optimal rate allocation between the SBs to be preserved by the temporal subsampling. Indeed, recalling the optimality condition (8) or (18), we note that discarding subbands only decreases  $M$ , but does not compromise the optimality condition for the remaining SBs.

The only problem to deal with is the following. If we simply discard the higher temporal SBs, we loose control on the final total bit-rate. The solution is once again to run the allocation algorithm only for the desired number of temporal SBs, with the suitable target bit-rate. This will generate a new set of quality layers (Fig. 8). A simple signaling convention can be established for the decoder to choose correctly the quality layers according to the desired level of temporal (and possibly quality) scalability.

We point out that MVs can be easily organized in different streams for each temporal scalability layer. Indeed, they can be encoded separately according to the temporal decomposition level, and each temporal scalability layers needs MVs from a single temporal decomposition level.

We remark that, in this case as well, the complexity increase is only due to the fact that the allocation algorithm has to be run a few more times. But, as mentioned before, its computational cost is negligible with respect to other part of encoder.

Experiments were made in order to assess the cost of the temporal scalability. An example is shown in Fig. 9. We encoded each test sequence at full frame-rate, and we decoded it at half the frame rate (dashed curve). Then we compared the results with those obtained by encoding directly the temporal subsampled sequence (solid curve). The results presented in Table III show a small scalability cost, not greater than 0.17 dB, as expected from theoretical considerations, due to the quality layers overhead.

It is worth noting that if other filters than  $(N, 0)$  had been used, a much greater performance cost would have been observed, due to the temporal filtering. This objective quality impairment would correspond to annoying subjective effects such as shadowing and ghosting artifacts.

3) *Spatial scalability*: Subband coding provides an easy way to obtain spatial scalability as well. Indeed, it is sufficient, once again, to discard high frequency SBs, in this case *spatial* frequencies. The spatial scalability cost should be also comparable to those of other scalabilities discussed above. The only additional problem is linked to the motion vectors which, in our coder, are not spatially scalable: in our experiments, we simply subsampled the original motion vectors, i.e. we used the full-resolution motion vectors with half the block-size and half their original values. In order to achieve a smooth spatial

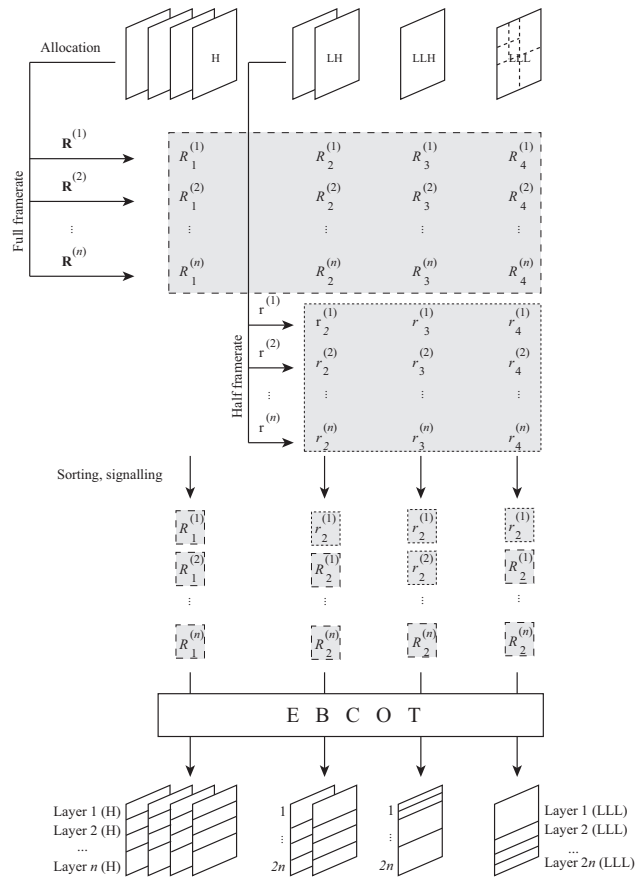


Fig. 8. Optimal bit-rate allocation for temporal and quality scalability. Example with 3 temporal decomposition levels (4 subbands, H, LH, LLH and LLL), 2 available framerates, and  $n$  quality layers. The allocation process presented in Fig. 6 (dashed part), which computes  $n$  bit-rates  $(R_i)_{i=1..n}$ , is repeated for all but the high-pass subbands (dotted part). A new set of  $n$  optimal bit-rates  $(r_i)_{i=1..n}$  is then determined. For each subband (excepted for the high-pass subband),  $2n$  bit-rates are obtained and sorted, defining  $2n$  optimal framerate-and-quality layers.

scalability, we would need a spatially progressive representation of the motion vectors as well.

Nevertheless, fairly assessing the spatial scalability cost is more difficult. Indeed, the choice of the reference reduced-resolution sequence is not straightforward. We can use the corresponding QCIF sequence as a reference for a CIF input video, but a more general solution is needed. A filtered and subsampled version of input data can be considered, but, in this case, the performances would become dependent from the low-pass filter used in the spatial analysis. In our codec, we used the classical 9/7 wavelet filters [4].

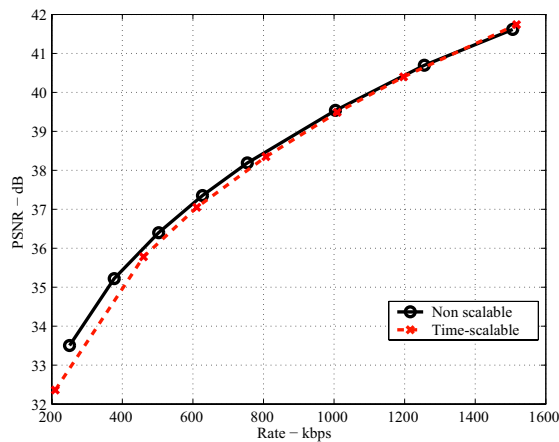


Fig. 9. Comparison of the PSNR curves of the sequence “foreman” (first 64 images), encoded with the temporal scalability feature (dashed curve) and independently for each frame-rate (solid curve). 3 temporal decomposition levels were used. The block matching was performed using  $16 \times 16$ -blocks and a half pixel precision.

Rate kbps	PSNR [dB] non-scalable	PSNR [dB] scalable	Scalability Cost [dB]
375	35.22	35.05	<b>0.17</b>
500	36.40	36.27	<b>0.13</b>
625	37.35	37.22	<b>0.13</b>
750	38.19	38.07	<b>0.12</b>
1000	39.54	39.50	<b>0.04</b>
1250	40.70	40.67	<b>0.03</b>

TABLE III

PSNR COST OF THE TEMPORAL SCALABILITY, FOR THE SEQUENCE “FOREMAN” (FIRST 64 IMAGES). 3 TEMPORAL DECOMPOSITION LEVELS WERE USED. THE BLOCK MATCHING WAS PERFORMED USING  $16 \times 16$ -BLOCKS AND A HALF PIXEL PRECISION.

### B. Compression performances

The presented coder, used with 4 temporal decomposition levels, was opposed to a constrained implementation of the standard H.264. Indeed, we forced the motion parameters to match those of our coder (fixed-size  $16 \times 16$  blocks with half-pixel precision), in order to obtain a fair performance assessment.

Depending on the input sequence, we found that our coder achieves performances close to or better than

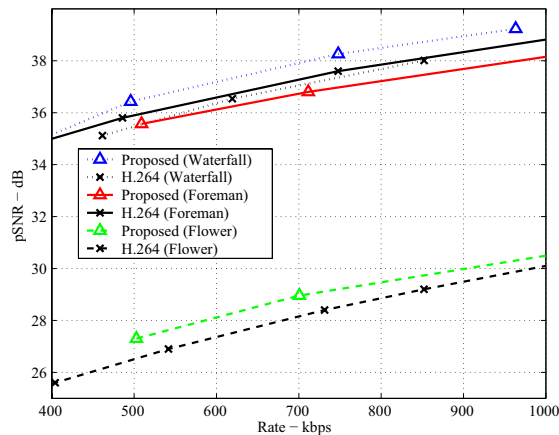


Fig. 10. Performance comparison (average PSNR) between the proposed coder (triangles) and a restrained H.264 (crosses), on sequences “Waterfall” (dotted), “Foreman” (solid) and “Garden & Flowers” (dashed) - first 64 images. 4 temporal decomposition levels were used. The block matching was performed using  $16 \times 16$ -blocks and a half-pixel precision.

those obtained by the restrained H.264 (Fig. 10). However, whereas H.264 is not natively scalable, our coder produces a highly scalable bitstream compatible with JPEG2000. Note that the sequences obtained by our coder and analyzed in the presented curves have been extracted from a single bitstream.

## V. CONCLUSION

We have presented in this paper a highly scalable video coder with a good level of performances. A specific  $(2, 0)$  scan-based temporal filtering, an efficient, robust and optimal bit-rate allocation algorithm, and a JPEG2000-compatible bitstream are the main components of this flexible coder which provides rate, temporal and spatial scalability. Although most of the presented techniques are still emerging, the performances of the proposed coder are good in a large range of bitrates, which make it suitable for high-quality video coding as well as for a use over heterogeneous networks.

Nevertheless, the offered perspectives are even more interesting. Of course, the coder would certainly benefit from a more sophisticated motion estimation algorithm (variable-size block matching, scalable motion vector representation, etc.), which would improve the temporal analysis efficiency and the spatial scalability performances. But most of all, even though we only tested our coder on CIF and QCIF sequences, we expect it to be even more efficient on larger-resolution and higher-framerate sequences, such as HDTV videos, for three main reasons. First, the presented model-based bit-rate allocation algorithm is more accurate with larger sets of data, because the statistics of the signal can be better evaluated. Second,

so is JPEG2000, because the border effects are less preponderant, and because more decomposition levels can be used. And third, a higher framerate results in a smoother motion, and thus leads to a more accurate motion estimation and a more efficient temporal decorrelation.

For all these reasons, we intend to further develop our coder based on the framework proposed in this paper. Future works will mostly be directed towards improvements of the motion representation and coding, and will include tests on high-definition video sequences.

#### REFERENCES

- [1] R. Schäfer, T. Wiegand, and H. Schwarz, "The emerging H.264/AVC standard," *EBU Technical Review*, Jan. 2003.
- [2] *Joint Committee Draft, JVT-C167*, Joint Video Team of ISO/IEC MPEG and ITU-T VCEG, May 2002.
- [3] *Information Technology - Coding of Audio Visual Objects - Part 2: Visual AMENDMENT 4: Streaming Video Profile*, MPEG 2000/N3518, July 2000.
- [4] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transforms," *IEEE Trans. on Image Processing*, vol. 1, no. 2, pp. 205–220, Apr. 1992.
- [5] J. Shapiro, "Embedded image coding using zerotrees of wavelets coefficients," *IEEE Trans. Signal Processing*, vol. 41, pp. 3445–3462, Dec. 1993.
- [6] A. Said and W. Pearlman, "A new, fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circ. Syst. for Video Tech.*, vol. 6, no. 3, pp. 243–250, June 1996.
- [7] G. Karlsson and M. Vetterli, "Three-dimensional subband coding of video," in *Proc. of IEEE Intern. Conf. on Acoustics, Speech and Signal Processing*, vol. 2, New York, USA, Apr. 1988, pp. 1100–1103.
- [8] J.-R. Ohm, "Three dimensional subband coding with motion compensation," *IEEE Trans. on Image Processing*, vol. 3, no. 5, pp. 559–571, Sept. 1994.
- [9] S. Choi and J. Woods, "Motion-compensated 3-D subband coding of video," *IEEE Trans. on Image Processing*, vol. 8, no. 2, pp. 155–167, Feb. 1999.
- [10] A. Secker and D. Taubman, "Motion-compensated highly scalable video compression using an adaptive 3D wavelet transform based on lifting," in *Proc. of IEEE Intern. Conf. on Image Processing*, Thessaloniki, Greece, Oct. 2001, pp. 1029–1032.
- [11] D. Taubman and A. Secker, "Highly scalable video compression with scalable motion coding," in *Proc. of IEEE Intern. Conf. on Image Processing*, vol. 3, Barcelona, Spain, Oct. 2003, pp. 273–276.
- [12] M. Flierl and B. Girod, "Investigation of motion-compensated lifted wavelet transforms," in *Proceedings of Picture Coding Symposium*, Saint-Malo, France, Apr. 2003, pp. 59–62.
- [13] G. Pau, C. Tillier, B. Pesquet-Popescu, and H. Heijmans, "Motion compensation and scalability in lifting-based video coding," *EURASIP Signal Processing: Image Communications, special issue on Wavelet Video Coding*, pp. 577–600, Aug. 2004.
- [14] M. Cagnazzo, T. André, M. Antonini, and M. Barlaud, "A model-based motion compensated video coder with jpeg2000 compatibility," in *Proc. IEEE International Workshop on Multimedia Signal Processing*, Singapore, Oct. 2004.

- [15] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelets," *Appl. Comput. Harmon. Anal.*, vol. 3, no. 2, pp. 186–200, 1996.
- [16] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 245–267, 1998.
- [17] L. Luo, J. Li, S. Li, Z. Zhuang, and Y.-Q. Zhang, "Motion compensated lifting wavelet and its application in video coding," in *Proc. of IEEE Intern. Conf. on Multimedia and Expo*, Tokyo, Aug. 2001, pp. 481–484.
- [18] T. André, M. Cagnazzo, M. Antonini, M. Barlaud, N. Božinović, and J. Konrad, "(N,0) motion-compensated lifting-based wavelet transform," in *Proc. IEEE Intern. Conf. on Acoustics, Speech and Signal Processing*, Montreal, Canada, May 2004.
- [19] J. Konrad, "Transversal versus lifting approach to motion-compensated temporal discrete wavelet transform of image sequences: equivalence and tradeoffs," *Proc. SPIE Visual Communications and Image Process.*, 2004.
- [20] C. Parisot, M. Antonini, M. Barlaud, C. Lambert-Nebout, C. Latry, and G. Moury, "On board stripe-based wavelet image coding for future space remote sensing missions," in *Proc. of IEEE International Geoscience and Remote Sensing Symposium*, Honolulu, USA, July 2000.
- [21] C. Chrysafis and A. Ortega, "Line based, reduced memory, wavelet image compression," *IEEE Transactions on Image Processing*, vol. 9, no. 3, pp. 378–389, Mar. 2000.
- [22] J. Y. Huang and P. M. Schultheiss, "Block quantization of correlated gaussian random variables," *IEEE Trans. Commun.*, vol. 11, pp. 289–296, Sept. 1963.
- [23] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 36, pp. 1445–1453, 1988.
- [24] K. Ramchandran and M. Vetterli, "Line based, reduced memory, wavelet image compression," *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 160–175, Apr. 1993.
- [25] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic, Jan. 1988.
- [26] B. Usevitch, "Optimal bit allocation for biorthogonal wavelet coding," in *Proc. of Data Compression Conf.*, Mar. 1996, pp. 387–395.
- [27] C. Parisot, M. Antonini, and M. Barlaud, "3D scan based wavelet transform and quality control for video coding," *EURASIP Journal on Applied Signal Processing*, Jan. 2003.
- [28] M. Cagnazzo, T. André, M. Antonini, and M. Barlaud, "A model-based motion compensated video coder with JPEG2000 compatibility," in *Proc. of IEEE International Conference on Image Processing*, Singapore, Oct. 2004, pp. 2255–2258.
- [29] M. Unser, "Splines: A perfect fit for signal and image processing," *IEEE Signal Processing Magazine*, vol. 16, no. 6, pp. 22–38, Nov. 1999.
- [30] T. André, B. Pesquet-Popescu, M. Gastaud, M. Antonini, and M. Barlaud, "Motion estimation using chrominance for wavelet-based video coding," in *Proc. IEEE Picture Coding Symposium*, San Francisco, USA, Dec. 2004.
- [31] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," vol. 11, no. 3, pp. 301 – 317, Mar. 2001.
- [32] M. Cagnazzo, T. André, M. Antonini, and M. Barlaud, "A smoothly scalable and fully JPEG2000-compatible video coder," in *Proc. of Intern. Workshop on Multimedia Signal Processing*, Sept. 2004, pp. 91–94.