Hindawi Publishing Corporation
EURASIP Journal on Image and Video Processing
Volume 2007, Article ID 30852, 11 pages
doi:10.1155/2007/30852

Research Article

JPEG2000-Compatible Scalable Scheme for Wavelet-Based Video Coding

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

I3S Laboratory, UMR 6070/CNRS, Université de Nice-Sophia Antipolis, Bâtiment Algorithmes/Euclide B, 2000 route des Lucioles, BP121, 06903 Sophia-Antipolis Cedex, France

Received 14 August 2006; Revised 5 December 2006; Accepted 16 January 2007

Recommended by James E. Fowler

We present a simple yet efficient scalable scheme for wavelet-based video coders, able to provide on-demand spatial, temporal, and SNR scalability, and fully compatible with the still-image coding standard JPEG2000. Whereas hybrid video coders must undergo significant changes in order to support scalability, our coder only requires a specific wavelet filter for temporal analysis, as well as an adapted bit allocation procedure based on models of rate-distortion curves. Our study shows that scalably encoded sequences have the same or almost the same quality than nonscalably encoded ones, without a significant increase in complexity. A full compatibility with Motion JPEG2000, which tends to be a serious candidate for the compression of high-definition video sequences, is ensured.

1. INTRODUCTION

The current video coding standards, such as MPEG-4 part 10 or H.264 [1–3], are very good at compressing today's video sequences at relatively low resolution (QCIF, CIF, or even SD formats). However, video coders based on wavelet transforms (WT) may prove to be much more efficient for encoding high-definition television (HDTV) or digital cinema (DC) sequences. For example, Motion JPEG2000, which extends JPEG2000 to video coding applications, proved to be as efficient as H.264/AVC in intramode for high-resolution sequences encoded at high-bit rate [4] and might be adopted as a future standard for digital cinema and high-definition television.

Furthermore, the generalization of these new, large formats will inevitably create new needs, such as scalability. A scalable bitstream is composed by embedded subsets, which are efficient compression of original data, but at a different resolution (both spatially or temporally) or rate. In other words, the user should be able to extract from a part of the full-rate, full-resolution bitstream (e.g., DC) a degraded version of the original data, that is, with a reduced resolution or an increased distortion (e.g., adapted to HDTV or even to Internet streaming) and with no additional computation. The recent standards already offer a certain degree of scalability, like the fine grain scalability (FGS) in the MPEG-4 standard [5]. However, in this case, scalability is obtained by substantially modifying the encoding algorithm, and this results in an increase in complexity and a decrease of quality for a given bit rate [6]. A more natural solution to the scalability problem comes from wavelet-based encoders, which can offer superior performances in terms of scalability cost in the case of video, as they already did for images [7–9]. However, the quality of temporally scaled videos can be impaired due to the lowpass wavelet filtering in the temporal domain [10]. Moreover, rate-scaled videos may loose rate-distortion optimality.

In this work, we describe a wavelet-based video coder and we discuss its spatial, temporal, and rate scalability. The main characteristics of this coder have been briefly presented in [11, 12]. In addition to a more detailed description of this coder, we provide here extended experimental results which better illustrate all the scalability properties. In particular, we show that our simple structure allows SNR, temporal, and spatial scalability, thanks to a specific temporal filtering and a careful bit allocation. We also provide an algorithm for modeling rate-distortion curves using the most appropriate smoothing spline. As a consequence, the scalability comes without impairing objective as well as subjective quality of the decoded sequence, neither increasing significantly the

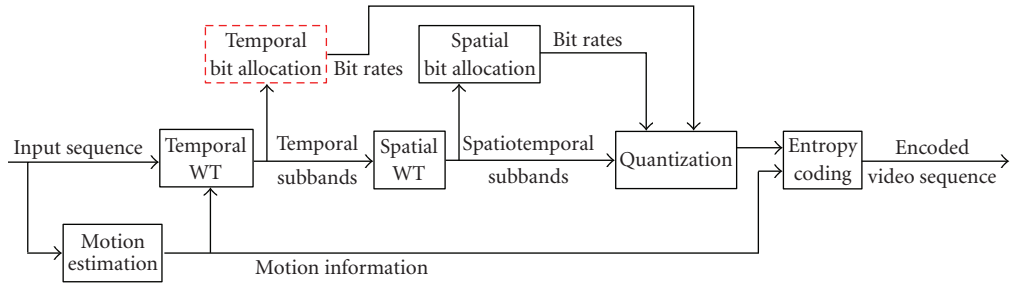

Figure 1: General structure of the proposed encoder.

encoding algorithm complexity. The result is a high scalability, which is transparent in terms of quality and complexity—that is what we call *smooth scalability*.

Let us first describe briefly the principles of wavelet-based video coding through the example of the coder presented in Figure 1. Wavelet transforms proved to be very powerful tools for still-image coding. WT-based encoders achieve better performances than those based on discrete 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. For these reasons, much attention has been devoted to WT-based video coding. In the proposed scheme, the input video data firstly undergo a temporal analysis based on the motion-compensated lifted wavelet transform. The motion information is encoded losslessly using EBCOT [9] and the remaining available bit budget is distributed among the temporal subbands (SB) using a model-based optimal bit allocation algorithm. In the temporal domain, motion-compensated lifting schemes [13–16], mostly based on the 5/3 wavelet kernels (also called $(2,2)$ lifting scheme), obtain better performances than uncompensated temporal WT. However, whereas the temporal analysis of hybrid coders requires one motion vector field (MVF) per frame and is very flexible, the 5/3 temporal wavelet analysis requires $4(1-2^{-L})$ MVFs per frame in average, when the number of decomposition levels is L. This number halves if symmetrical MVF are used (usually with a negligible loss in motion compensation accuracy), but it is still a high penalty. An alternative is to change the temporal filter, using a lifting scheme without update step, from now on indicated as $(2,0)$ lifting scheme. This filter has been presented in [10, 17] and its possible adoption into the standard JVT-SVC [18] is under study [19]. The expression of the motion-compensated temporal highpass and lowpass filters of the $(2,0)$ lifting scheme is the following:

$$\begin{aligned} h_k[\mathbf{m}] &= x_{2k+1}[\mathbf{m}] - \frac{1}{2}\left(x_{2k}[\mathbf{m}+\mathbf{v}_{2k+1\to 2k}(\mathbf{m})]\right. \\ &\qquad \left. + x_{2k+2}[\mathbf{m}+\mathbf{v}_{2k+1\to 2k+2}(\mathbf{m})]\right), \\ l_k[\mathbf{m}] &= x_{2k}[\mathbf{m}], \end{aligned} \tag{1}$$

where x_k, h_k, and l_k are, respectively, the kth input frame, high-frequency coefficient and low-frequency coefficient, and $\mathbf{v}_{i\to j}(\mathbf{m})$ is the motion vector that displaces the pixel $\mathbf{m}$ of the image x_i to the corresponding pixel in the image x_j. The lowpass filtering is then reduced to a simple temporal subsampling of the original sequence. These filters reduce the number of required motion vectors, and leave the lowpass subband unaltered in case of unprecise motion compensation. We observe that the $(2,0)$ lifting scheme is related to the unconstrained motion-compensated temporal filter (UMCTF) framework [20], which is characterized by adaptive choice of lowpass filter between a so-called delta filter and a more traditional averaging filter (in [20] it is Haar). The delta-filter in the UMCTF framework is a pure subsampling (like in our scheme) and the choice between the delta-filter and the averaging filter depends on the video motion content. This adaptation allows a better representation of fast motion in the case of very low frame rate. In our scheme the lowpass filter is not adaptively chosen, since we assume that when temporal scalability is requested, the reference video sequence is the pure subsampling of the original one. In this case, a pure subsampling temporal filter minimizes the scalability cost, as we will show later on.

As far as spatial stage is concerned, we use a JPEG2000-compatible algorithm handling the spatial analysis as well as the encoding process. As a consequence, the available resources are automatically allocated among the different spatial subbands. On the other hand, the bit allocation between the temporal subbands remains to be done. To do so, the knowledge of the rate-distortion (RD) curve of each temporal subband is required, and the estimation of these curves is computationally heavy. Furthermore, a high precision is required in order to obtain regular, differentiable curves. For these reasons, model-based algorithms are desirable as long as they can combine computational efficiency with robustness and accuracy. According to these ideas, the model-based algorithm described in Sections 2 and 3 performs an optimal bit allocation using spline models of RD curves. This algorithm only needs the computation of a few points for each RD curve, and interpolates them by a cubic spline. The spline modeling allows a concise yet accurate representation of the RD curves and has a very low complexity.

Once the temporal analysis and the bit allocation have been performed, the low-frequency (LF) subbands undergo a spatial WT using the 9/7 filters [21]. If spatial scalability is required, the high-frequency (HF) subbands undergo the same

spatial transform. The MQ coder of EBCOT is then used to encode all subbands as well as motion vectors. The full JPEG2000 compatibility of the whole coder is thus ensured.

At first sight, it seems that the video coder described above is already completely scalable. Indeed, spatial and temporal scalability are natively supported thanks to the use of spatiotemporal wavelet transforms, and rate scalability is a feature of EBCOT. However, we show in Section 4 that some specific operations are needed in order to limit, or possibly to cancel out, the performance losses due to scalability.

The remaining of the paper is organized as follows. In Section 2, we introduce the problem of temporal bit allocation and review some existing approaches. We also present optimal algorithms for rate and quality allocation based on RD curves. In Section 3, we present an improvement to the previous algorithms by introducing a model for RD curves based on splines. In Section 4, we investigate the possibilities of the proposed video coder in terms of scalability. Finally, Section 5 concludes the paper.

2. RESOURCE ALLOCATION

The temporal analysis produces several types of temporal subbands, according to the wavelet transform used and the number of decomposition levels. We will consider a dyadic one-dimensional decomposition on N levels resulting in $M = N + 1$ subbands: N high-frequency (HF) and 1 low-frequency (LF). The problem arises of assigning the coding resources to the subbands, so that either the distortion is minimized for a given target bit-rate, or the bit-rate is minimized for a given target quality.

Analytic solutions have been proposed in the literature in the hypothesis of high bit-rate, but in the general case, this problem is not trivial. On the other hand, methods based on empirical RD curves analysis do not require any assumption on the target bit-rate, and thus have been widely used. Shoham and Gersho proposed in [22] an optimal algorithm with no restriction on bit-rate, at the expense of a high computational cost since it requires the computation of the RD characteristics for each possible quantization step. Ramchandran and Vetterli presented in [23] an RD approach to encode adaptive trees using generalized multiresolution wavelet packets. The most recent still image compression standard JPEG2000 is based on the EBCOT algorithm, which 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.

In the following, we recall a general bit allocation algorithm based on analytical RD curves, and we provide a method to obtain these curves from experimental data. Both the rate allocation and the distortion allocation points of view are considered.

2.1. The rate allocation problem

Let us first suppose that the user wants to optimize the reconstruction quality for a given target bit-rate. The problem is to find a suitable set of bit-rates $\mathbf{R} = \{R_i\}_{i=1}^{M}$ (where R_i is the bit-rate assigned to the ith subband) so that the resulting distortion $D(\mathbf{R})$ of the reconstructed sequence is minimized. Of course there is a constraint on the total available bit-rate.

In the case of orthogonal subband coding, Gersho and Gray showed [24] that the global distortion can be expressed as a sum of the subbands distortions:

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

where $D_i(R_i)$ is the RD curve for the ith subband, and it has to be computed or estimated in some way. We notice that we do not use orthogonal filters, but the previous formula can be extended [25] by using filter weights w_i which account for the nonorthogonality:

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

The minimization of the distortion is subject to a constraint on the total bit-rate of the subbands, R_{SB}, which should be equal to or smaller than a target value R_{MAX}.

The total bit-rate is a weighted sum of subband rates, $R_{\mathrm{SB}} = \sum_{i=1}^{M} a_i R_i$, where the coefficient a_i is simply the fraction of total pixels in the ith subband. Thus, the rate allocation problem consists in finding $\mathbf{R}$ which minimizes the cost function (3) under the constraint $\sum_{i=1}^{M} a_i R_i \leq R_{\mathrm{MAX}}$.

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

$$J(\mathbf{R}, \lambda) = \sum_{i=1}^{M} w_i D_i(R_i) - \lambda \left(\sum_{i=1}^{M} a_i R_i - R_{\mathrm{MAX}} \right). \tag{4}$$

In the hypothesis of differentiability, 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:

$$\frac{w_i}{a_i} \frac{\partial D_i}{\partial R_i}(R_i^*) = \lambda \quad \forall i \in \{1, \ldots, M\}, \tag{5}$$

where λ is the Lagrange multiplier. Equation (5) states that the optimal rates correspond to points having the same slope on the "weighted" curves $(R_i, (w_i/a_i)D_i)$. Note that $\lambda \leq 0$ since the RD curves are decreasing.

Let us introduce the set of functions $R_i(\lambda)$, defined implicitly by the following equation:

$$\frac{w_i}{a_i} \frac{\partial D_i}{\partial R_i}(R_i) \bigg|_{R_i = R_i(\lambda)} = \lambda. \tag{6}$$

The value of $R_i(\lambda)$ is the rate of the ith subband which corresponds to a slope λ on its weighted RD curve. The rate allocation problem consists in finding the slope value λ^* so that

$$\sum_{i=1}^{M} a_i R_i(\lambda^*) = R_{\mathrm{MAX}}. \tag{7}$$

Simple algorithm exists which allows to find λ^*, among which we can mention the bisection method, the Newton method, the Golden Section method, the Secant method. These algorithms usually converge after 3 to 6 iterations, and their complexity is negligible if compared to the other parts of video coder such as motion estimation and compensation.

Note that this algorithm converges to the optimal solution if and only if the curves $D_i(R_i)$ are both differentiable and convex.

2.2. The quality allocation problem

So far, only the problem of rate allocation has been considered. However, for some applications requiring for example a minimum level of quality, the constraint must be applied on the distortion instead of the bit-rate. This problem turns out to be very similar to the rate allocation problem and can be solved in a very similar way.

Indeed, the cost function to be minimized is now the total bit-rate allocated to the subbands $R_{\mathrm{SB}} = \sum_{i=1}^{M} a_i R_i$, under a constraint on the global distortion:

$$D(\mathbf{R}) = \sum_{i=1}^{M} w_i D_i(R_i) \leq D_{\mathrm{MAX}}. \tag{8}$$

We write the following Lagrangian functional:

$$J(\mathbf{R}, \lambda) = \sum_{i=1}^{M} a_i R_i - \lambda \left(\sum_{i=1}^{M} w_i D_i(R_i) - D_{\mathrm{MAX}} \right) \tag{9}$$

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

$$\frac{w_i}{a_i} \frac{\partial D_i}{\partial R_i}(R_i^*) = \frac{1}{\lambda} \quad \forall i \in \{1, \ldots, M\}. \tag{10}$$

This means, once again, that the optimality condition is the uniform slope on the weighted curves $(R_i, (w_i/a_i)D_i)$. The optimal bit-rates R_i^* are then determined using the algorithms presented in the previous section.

2.3. Obtaining the rate-distortion functions

The algorithms presented in the previous sections not only require the knowledge of the RD curve of each subband, but also suppose that these curves are differentiable, convex, and accurate enough. A crucial step of the bit-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. Unfortunately, in order to obtain accurate estimates of each curve in the whole range of possible bit-allocation values, many test points are required. So this approach is extremely complex. Furthermore, such experimental RD curves are found to be much irregular, especially at low bit-rates, and consequently they can easily result not convex nor differentiable, and the allocation algorithms lack robustness.

To circumvent this difficulty, some approaches have been proposed which do not require RD curves to be estimated. A first analytical approach is due to Huang and Schultheiss, who stated the theoretical optimal bit allocation for generic transform coding in the high-resolution hypothesis [26]. They derived a formula which defines the optimal bit-rate to be allocated to each set of data, depending on their variances. Unfortunately, this solution only holds when a high rate is available for encoding. Later, Parisot et al. proposed in [27] a model of scalar-quantized coefficients using generalized Gaussian models. Using these different models leads to a complexity reduction of the allocation algorithms, and improves their robustness. Unfortunately, these solutions only hold under strong hypotheses, for example, on the total bit-rate, or the quantizer being used. The hypotheses drawn have a limited domain of validity which causes the allocation to be quite imprecise at low bit-rate.

In the following, we propose a model for RD curves which improves the tradeoff between robustness, accurateness, and complexity, and remains valid for the most general case.

3. MODEL-BASED BIT ALLOCATION USING SPLINES

In this section, we propose an analytical model for RD curves which allows the implementation of a data-driven and model-based allocation algorithm. In this way, we try to combine the precision and accuracy of techniques based on experimental data, with the robustness, computational efficiency, and flexibility of model-based methods, while guaranteeing the convexity and differentiability of the obtained RD curves.

Splines are particularly well-suited for this purpose, because they are designed to allow a smooth switching between continuous and discrete representations of a signal. Since their first introduction by Schoenberg [28, 29], they have been successfully used in many problems of applied mathematics and signal processing [30].

A spline of degree n is a piecewise polynomial function of degree n, which is continuous together with its first $n-1$ derivatives. Splines, and in particular cubic splines, proved to be very effective in solving the interpolation problem. In other words, given a set $\mathcal{S}_N$ of N points $\{(x_k, y_k)\}_{k=1,\ldots,N}$, it is possible to find the spline passing through them with a very low complexity. Moreover, the resulting spline can be analytically described with as few as N parameters and it has a pleasantly smooth aspect. In particular, the cubic interpolating spline minimizes the curvature of the resulting function.

However, some adjustment is needed in order to use spline to efficiently interpolate RD curves in the general case. Indeed, the set of RD points obtained experimentally is usually quite irregular, especially at low bit-rates. Interpolating those points directly could result in a nonmonotonic, nonconvex curve, which would cause the algorithms proposed in Section 2 to fail.

In order to solve this problem, smoothing splines can be used instead of interpolation splines. Considering the set of points $\mathtt{a}\mathcal{S}_N$, the solution of the interpolation problem

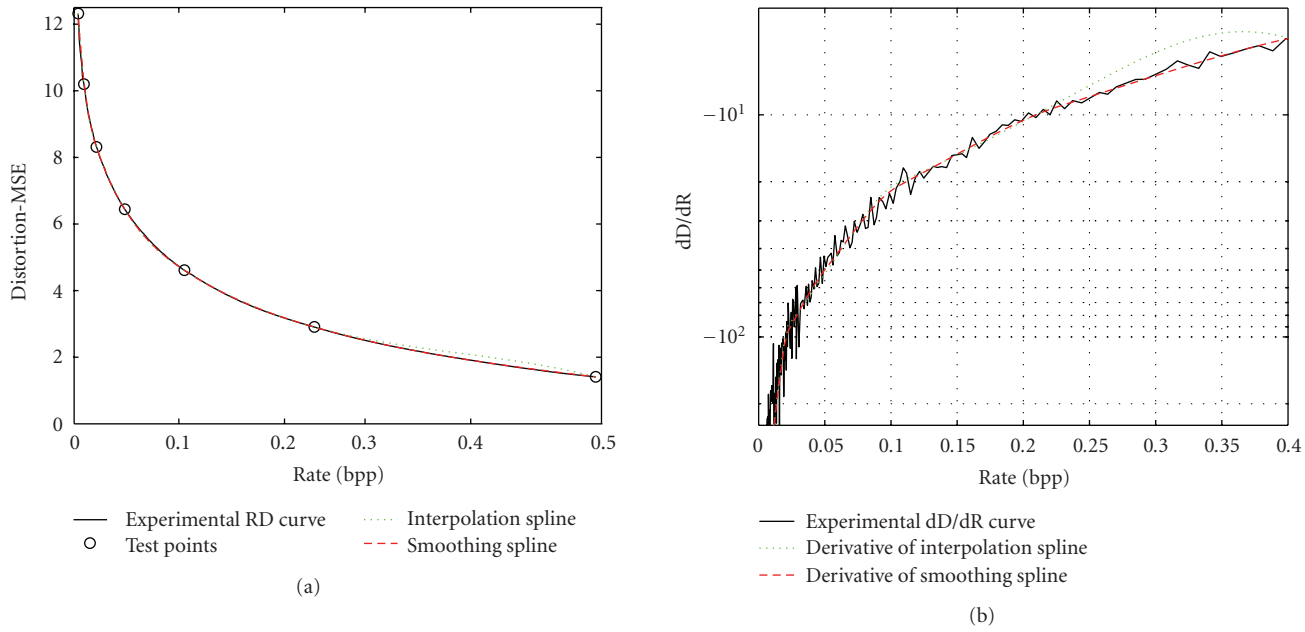

FIGURE 2: (a) The smoothing-spline curve seems to match perfectly the "experimental" curve: spline approximations of an "experimental" RD curve (solid curve) composed by 200 points computed experimentally. The interpolation-spline curve (dotted curve) and the smoothing-spline curve (dashed curve) have been obtained by interpolating the 7 marked points. Moreover, (b) its derivative fits better to the real data than the interpolation-spline curve: derivatives of the RD curves presented above. The derivatives of the spline curves (dotted and dashed) have been computed analytically from the expression of the original curves. The obtained RD curve and its derivative are smooth and continuous.

is the spline function $s(x)$ which sets at zeros the quantity $\sum_{k=1}^{N}(y_k - s(x_k))^2$. If the sample points are affected by error or noise, a tight interpolation of them easily results in an irregular (i.e., nonmonotonic or nonconvex) spline. If we relax the interpolation condition and impose a regularity constraint, much better results can be obtained. Let us consider the following criterion to minimize:

$$J(s(\cdot),\lambda) = \sum_{k=1}^{N}\left(y_k - s(x_k)\right)^2 + \lambda \int_{-\infty}^{+\infty}\left[s^{(2)}(x)\right]^2 dx. \tag{11}$$

In this criterion, there is a first term which imposes that the solution should pass close to the experimental point, and a second one which is (with very good approximation) close to the function curvature. Minimizing this criterion means finding a function passing close to the test points but which is regular. The parameter λ controls the balance between the two constraints. The greater λ, the greater the penalty on the energy of the second derivative, and the smoother the final curve result.

It has been shown [31] that the solution of the minimization problem (11) is a cubic spline. This kind of spline is called "smoothing spline," and fast calculation techniques exist [32] which efficiently find the smoothing spline for an assigned set $\mathcal{S}_N$ and a value of λ.

At this point, only a suitable value for λ remains to be found, so that the obtained spline curve is the convex, monotonic, and as close as possible to the sample points. The algorithm we propose starts by computing the spline interpolating $\mathcal{S}_N$, that is, a smoothing spline with $\lambda = 0$. If it is already regular (i.e., monotonic and convex), it is retained as parametric representation of the RD curve. Otherwise, we set λ to some small value and look for the smoothing spline minimizing J. The algorithm continues iteratively: if the obtained spline is regular, it exits; otherwise λ is incremented and a new smoothing spline is computed. It is worth noting that the algorithm usually converges after a few iterations (less than 10), and that in any case its complexity remains small[1] compared to the global complexity of the encoder.

Many experiments were carried out in order to verify the efficiency of the model, and in all of them spline proved to provide a very good fit to any RD curve. This was not obvious because, for example, the lowest frequency SB has usually a very steep RD curve for the lower range of rate and much more flat curves for higher rates, while high-frequency SBs generally have regular RD curves. Nevertheless, the proposed approach is able to represent any RD curve accurately, usually using as few as 7 to 10 points.

An example is shown in Figure 2(a), where we report as a reference the "experimental" RD curve for the highest frequency SB computed on the first 16 frames of the *foreman* sequence (solid line), obtained by the (2, 2) temporal filter. This curve has been obtained by encoding and decoding the

[1] This is because in any case the number of points in $\mathcal{S}_N$ is very small, for example, with respect to the number of samples of the corresponding subband.

SB at 200 different rates. On the same graph, we reported the spline 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. One can notice that the smoothing spline curve is convex, whereas the interpolation spline is not.

In Figure 2(b), we reported the first derivatives of the same experimental curve and of the splines. The experimental derivative must be approximated from the 200 experimental points, whereas the computation of the spline derivatives can be easily accomplished analytically. The resulting spline curves have not 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.

To conclude this section, we stress that the proposed algorithm was validated by using it in order to model the RD curves of the *spatial* SBs of the WT of natural images. We found that it is able to provide smooth and regular curves in this case as well, even though the statistics of spatial SBs are usually quite different to those of temporal SBs. This is an additional confirmation of the robustness of our algorithm.

4. SCALABILITY

In a general way, a scalable bitstream has lower 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. A smoothly scalable encoder should have a null or very little scalability cost, that is, the same (or almost the same) performances of its nonscalable version. Moreover, we have also to take into account that introducing scalability into a video encoder means increasing its complexity. The smoothly scalable encoder should on the contrary have a complexity comparable to its nonscalable counterpart.

In [6], 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 nonscalable version (in particular for temporal scalability). WT-based encoders have a much easier job with scalability, thanks to the multiresolution analysis properties. Nevertheless, some problems remain to be solved, mainly related to bit allocation and lowpass filtering effects.

In the following, we show that the video coder presented above is *smoothly scalable* provided that the bit-allocation algorithm is slightly modified. The resulting coder is capable of achieving almost the same performances as the nonscalable version, and at almost the same computational cost. In all the experiments, we used a simple motion description, based on 16×16 blocks at quarter-pixel precision.

We will use the following notations. Let $R^{(0)}$ be the bit budget available for the subbands. The nonscalable 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 $\sum_{i=1}^{M} a_i R_i^{(0)} = R^{(0)}$, where $R_i^{(0)}$ is the rate allocated to the ith subband when the total available rate is $R^{(0)}$.

4.1. Rate scalability

The rate scalability should allow to decode the bitstream at a set of predefined rates $R^{(n)} < \cdots < R^{(1)}$ different from the encoding rate $R^{(0)}$. Since the ith 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 jth rate, we lose optimality of the bit allocation.

To overcome this problem, we perform in advance the bit allocation for each target 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 rates, until n optimal rate vectors are obtained for each SB. Then, as shown in Figure 3, we can encode the ith subband with the n quality layers corresponding to the rates $R_i^{(j)}$ (for $j = 1, \ldots, n$). Finally, we regroup all the layers corresponding to the same level. Thus, in order to decode the sequence at the given rate $R^{(j)}$, we simply decode each SB at the quality level j.

In order to evaluate the cost of this scalability method, we compared the PSNR of the test sequences encoded and decoded at the same rates, with the following two methods: the first one consists in encoding each sequence separately for each target rate; the second consists in producing only one scalable bitstream for each sequence, and then decoding it for each rate. It appears that, regardless of the demanded rate, the scalable compressed video is almost identical to the nonscalable one, 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 several test sequences are reported in Table 1. For several target bit-rates, this table shows the PSNR achieved by the proposed coder with no rate scalability, as well as the PSNR loss observed for the same bit-rate when the quality scalability is enabled. In all test configurations, we noted that the proposed method assures a very little and practically negligible performance degradation, always inferior to 0.1 dB, increasing with the decoded bit-rate.

We note that the motion information is not affected by the rate scalability, as we still need the same vectors than for the nonscalable 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. If the bit-allocation algorithm is model-based, its complexity is negligible, much lower than the one of the motion estimation or the wavelet transform.

In conclusion, introducing rate scalabilty does not affect reconstructed sequence quality, neither requires a significant increase in complexity in the proposed encoder.

Table 1: PSNR (dB) achieved by the nonscalable version of the coder, and cost (dB, in bold) of the rate scalability, for several CIF sequences. A 3-level $(2,0)$ temporal wavelet transform was used. The block matching was performed using 16×16 blocks and a quarter-pixel precision.

Rate (kbps)	300	500	750	1000	1200
Flower	22.71 (**0.00**)	25.76 (**0.03**)	28.00 (**0.05**)	29.67 (**0.06**)	30.75 (**0.07**)
Foreman	29.75 (**0.00**)	33.45 (**0.03**)	35.61 (**0.03**)	37.06 (**0.05**)	37.97 (**0.05**)
Mobile	23.06 (**0.01**)	25.79 (**0.03**)	27.95 (**0.03**)	29.53 (**0.05**)	30.49 (**0.06**)
Waterfall	31.81 (**0.00**)	34.54 (**0.04**)	36.86 (**0.03**)	38.36 (**0.03**)	39.20 (**0.03**)

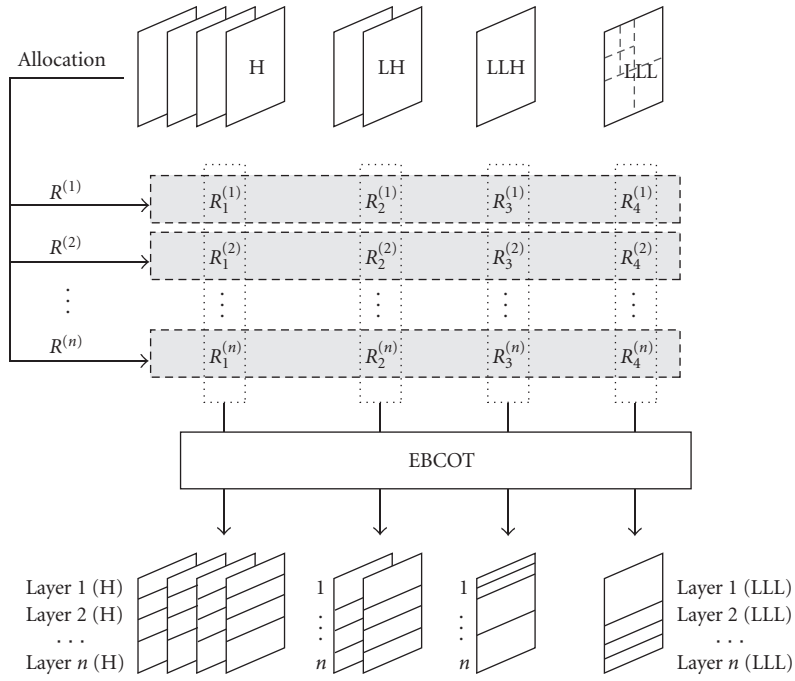

Figure 3: Optimal bit allocation for rate scalability. Example with 3 temporal decomposition levels (4 subbands, H, LH, LLH, and LLL) and n quality layers. The bit-allocation algorithm is repeated for each target rate $R^{(i)}$ corresponding to a quality layer i (dashed parts). Thus, n sets of optimal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

4.2. Temporal scalability

The proposed video coder makes use of 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 bit allocation between the SBs to be preserved by the temporal subsampling, since we entirely discard high frequency subbands, with the residual rate still optimally allocated among surviving bands.

The only problem to deal with is the following. If we simply discard the higher temporal SBs, we loose control on the final total rate. The solution is once again to run the allocation algorithm only for the desired number of temporal SBs, with the suitable target rate. This will generate a new set of quality layers (Figure 4). 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 motion vectors 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 layer needs motion vectors 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

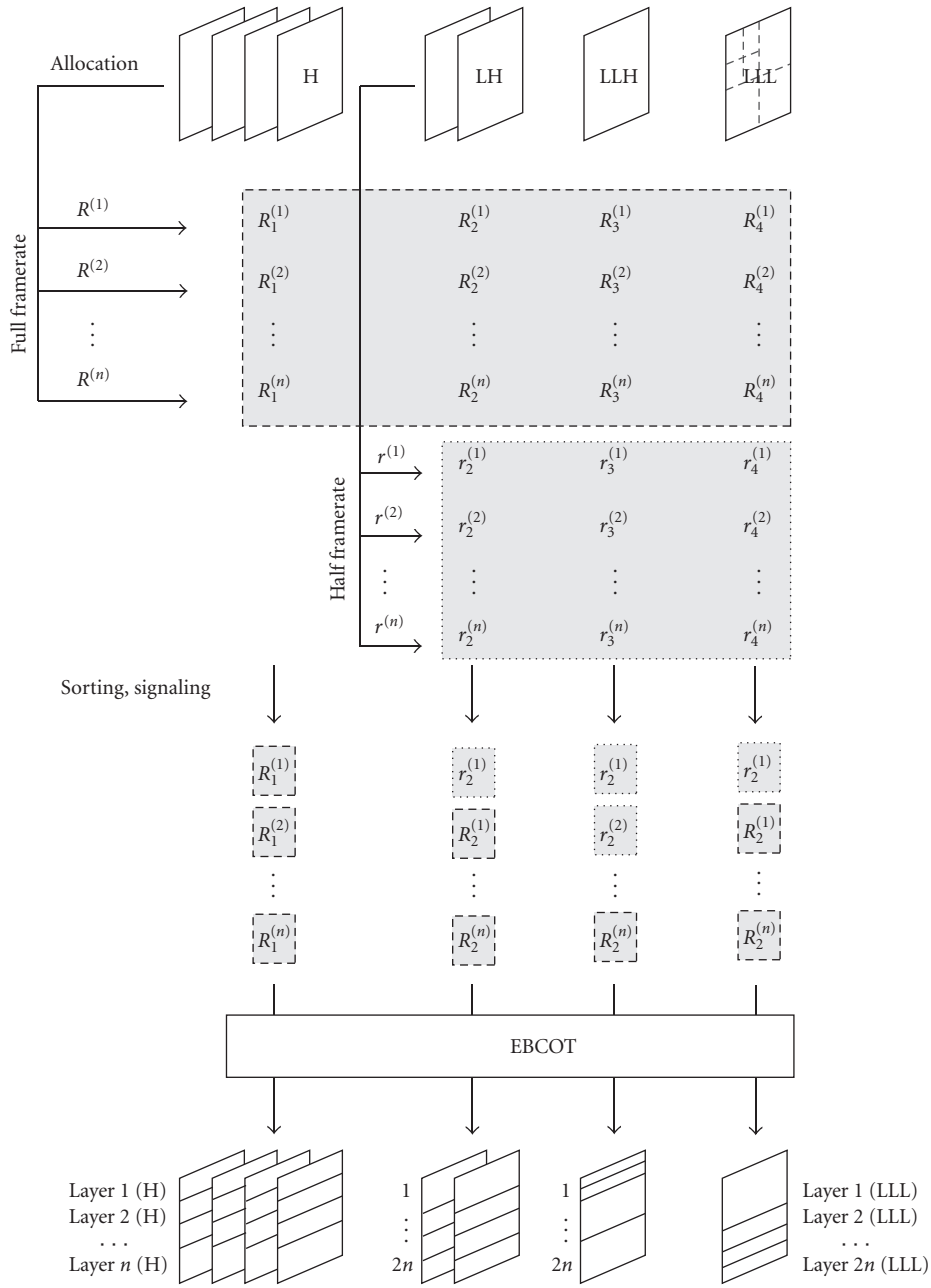

Figure 4: Optimal bit allocation for temporal and quality scalability. Example with 3 temporal decomposition levels (4 subbands, H, LH, LLH, and LLL), 2 available framerates, and n quality layers. The allocation process presented in Figure 3 (dashed part), which computes n rates $(R_i)_{i=1,\ldots,n}$,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

computational cost is negligible with respect to other parts of encoder.

Experiments were made in order to assess the cost of the temporal scalability. We encoded each test sequence at full frame rate, and we decoded it at half the frame rate. Then we compared the results with those obtained by encoding directly the temporal subsampled sequence. The results presented in Table 2 show a small scalability cost, not greater than 0.07 dB, as expected from our theoretical considerations. We also underline that the base layer of the temporal hierarchy is actually the JPEG2000 encoding of the temporal subsampled input sequence. This means that a user can obtain an overview of the encoded sequence with as a simple tool as a JPEG2000 decoder (together with a trivial bitstream parser). This is possible because we use a temporal filter without the update stage.

Table 2: Temporal scalability cost (ΔPSNR, dB) for several CIF sequences, $(2,0)$ lifting scheme.

Rate (kbps)	300	500	750	1000	1200
Flower	0.02	0.01	0.03	0.00	0.01
Foreman	0.07	0.07	0.06	0.05	0.05
Mobile	0.01	0.01	0.02	0.01	0.01
Waterfall	0.01	0.04	0.01	0.01	0.01

Table 3: Temporal scalability cost (ΔPSNR, dB) for several CIF sequences, $(2,2)$ lifting scheme.

Rate (kbps)	300	500	750	1000	1200
Flower	0.19	0.26	0.50	0.70	0.93
Foreman	0.29	0.35	0.46	0.71	0.85
Mobile	0.17	0.21	0.47	0.61	0.79
Waterfall	0.19	0.28	0.48	0.62	0.81

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. We present in Table 3, as an example, the results of an experiment similar to the previous one, but performed with the common $(2,2)$ lifting scheme. We notice a quite high PSNR impairment in this case, up to almost one dB.

4.3. Spatial scalability

Subband coding provides an easy way to obtain spatial scalability as well: it is sufficient to discard high-frequency SBs (in this case *spatial* high frequencies) to obtain reduced-resolution version of the original sequence. The only additional problem is linked to the motion vectors which, in our coder, are not spatially scalable: in our experiments, we simply used the full-resolution motion vectors with half the block-size and half their original values. In order to achieve a smooth spatial scalability, we would need a spatially progressive representation of the motion vectors as well.

However, a fair assessment of the spatial scalability cost is more difficult than the previous cases, because the choice of the reference low-resolution sequence is not straightforward. A raw subsampling, effective in the temporal case, would produce a reference sequence strongly affected by spatial aliasing, and this sequence would be of course a quite poor reference, because of its degraded subjective quality. Therefore, a filtering stage before subsampling seems necessary. However, in this case, the performances would become dependent from choice of the lowpass filter. A reasonable choice is then to use the same filter used in the spatial analysis stage of the encoder, which in our case is the well-known 9/7 wavelet filter. This filter produces a pleasantly smooth low-resolution version of the original image, so we can use the sequence of first-level LL bands as reference low-resolution sequence.

With this settings, we run similar experiments to those presented for temporal and quality scalability. We decoded the sequence at a lower resolution and we compared the resulting performance to those obtained by directly encoding the reduced-resolution sequence. We found, in this case as well, a very small scalability cost, usually less than 0.1 dB all over the range of encoding bit-rates. This cost does not take into account the increase in MVFs representation cost, as it becomes zero as far as a spatial scalable representation of them is used.

4.4. Note on the complexity

Apart from estimating the RD curves, the bit-allocation algorithm used here consists in finding the optimal rate on each RD curve, for each one of the demanded scalability settings. Thanks to the spline modeling of these curves, this operation is extremely fast, and the iterative algorithm usually converges after 5 iterations or less. Thus, even though this step must be repeated several times according to the temporal scalability needs, its complexity is negligible. This process takes a significant, but not overwhelming, amount of computation time within the complete encoding process. Of course, the complexity of the decoder remains totally unaffected by this algorithm.

5. CONCLUSION

We have presented in this paper a simple yet efficient scalability scheme for wavelet-based video coder, able to provide on-demand spatial, temporal and SNR scalability, together with compatibility with the JPEG2000 standard. In addition to a specific temporal wavelet filter, the use of a careful, model-based bit allocation guarantees good performances and optimality in the sense of rate distortion. This is confirmed by tests where we run the *nonscalable* H.264 encoder [33] with a motion model similar to the one used in our encoder.

In Table 4, we show the PSNR values achieved by H.264, together with the performance gain of the proposed scheme, that is, the difference between the PSNR of our encoder (the values reported in Table 1) and that of H.264. We observe that the performances are quite close. Our encoder is only penalized when both the cases of low bit-rates and complex motion occur. In this situation, the current, nonscalable representation of motion vectors adsorbs too much coding resources. We think that with a scalable representation of motion vectors, our coder would benefit from a better tradeoff among MVs and WT coefficients bit-rate. In the other cases, the performances are comparable to those of H.264.

We reported some results for Motion JPEG2000 as well in Table 5. This technique does not use motion compensation, and so it has far worse performances than our coder, which however remains compatible with this standard, since either temporal subbands and motion vectors are encoded with EBCOT.

Of course, the coder would certainly benefit from a more sophisticated motion model (variable-size block matching, scalable motion vector representation, etc.), which would

Table 4: PSNR (dB) achieved by H.264, and (in bold) performance gain Δ PSNR (dB) of the proposed scheme.

Rate (kbps)	300	500	750	1000	1200
Flower	23.92 (**−1.16**)	26.28 (**−0.62**)	27.75 (**0.25**)	29.51 (**0.16**)	30.66 (**0.09**)
Foreman	32.90 (**−3.15**)	35.05 (**−1.60**)	36.88 (**−1.27**)	38.13 (**−1.07**)	38.95 (**−0.98**)
Mobile	22.91 (**0.15**)	25.67 (**0.12**)	27.10 (**0.85**)	29.03 (**0.50**)	30.13 (**0.36**)
Waterfall	31.11 (**0.70**)	33.52 (**0.98**)	35.76 (**1.10**)	37.31 (**1.05**)	38.21 (**0.99**)

Table 5: PSNR (dB) achieved by MJPEG2000, and (in bold) performance gain ΔPSNR (dB) of the proposed scheme.

Rate (kbps)	300	500	750	1000	1200
Flower	19.25 (**3.46**)	20.33 (**5.43**)	21.49 (**6.51**)	22.58 (**7.09**)	23.40 (**7.35**)
Foreman	26.62 (**3.13**)	28.48 (**4.97**)	30.03 (**5.58**)	31.20 (**5.86**)	32.05 (**5.92**)
Mobile	18.14 (**4.92**)	19.28 (**6.51**)	20.23 (**7.72**)	21.11 (**8.42**)	21.77 (**8.72**)
Waterfall	25.65 (**6.16**)	26.82 (**7.72**)	27.95 (**8.91**)	28.91 (**9.45**)	29.50 (**9.70**)

improve the temporal analysis efficiency and the spatial scalability performances. Further studies are under way to obtain an efficient and scalable representation of motion vectors, to find the best rate allocation among vectors and wavelet coefficients, and to optimize the motion estimation with respect to the motion-compensated temporal WT. These new tools together with an adequate motion model could further improve RD performance of the proposed scheme, making it an interesting solution for the scalable video coding problem.

REFERENCES

[1] G. J. Sullivan and T. Wiegand, "Video compression-from concepts to the H.264/AVC standard," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 18–31, 2005.

[2] D. Marpe, T. Wiegand, and G. J. Sullivan, "The H.264/MPEG4 advanced video coding standard and its applications," *IEEE Communications Magazine*, vol. 44, no. 8, pp. 134–143, 2006.

[3] *Joint Committee Draft, JVT-C167*, Joint Video Team of ISO/IEC MPEG and ITU-T VCEG, May 2002.

[4] D. Marpe, V. George, H. L. Cycon, and K. U. Barthel, "Performance evaluation of motion-JPEG2000 in comparison with H.264/AVC operated in pure intra coding mode," in *Wavelet Applications in Industrial Processing*, vol. 5266 of *Proceedings of SPIE*, pp. 129–137, Providence, RI, USA, October 2004.

[5] *Information Technology—Coding of Audio Visual Objects—Part 2: Visual AMENDMENT 4: Streaming Video Profile*, MPEG 2000/N3518, July 2000.

[6] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 301–317, 2001.

[7] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, 1993.

[8] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243–250, 1996.

[9] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Transactions on Image Processing*, vol. 9, no. 7, pp. 1158–1170, 2000.

[10] T. André, M. Cagnazzo, M. Antonini, M. Barlaud, N. Božinović, and J. Konrad, "(N,0) motion-compensated lifting-based wavelet transform," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '04)*, vol. 3, pp. 121–124, Montreal, Quebec, Canada, May 2004.

[11] M. Cagnazzo, T. André, M. Antonini, and M. Barlaud, "A smoothly scalable and fully JPEG2000-compatible video coder," in *Proceedings of the 6th IEEE Workshop on Multimedia Signal Processing (MMSP '04)*, pp. 91–94, Siena, Italy, September 2004.

[12] M. Cagnazzo, T. André, M. Antonini, and M. Barlaud, "A model-based motion compensated video coder with JPEG2000 compatibility," in *Proceedings of IEEE International Conference on Image Processing (ICIP '04)*, vol. 4, pp. 2255–2258, Singapore, October 2004.

[13] W. Sweldens, "The lifting scheme: a custom-design construction of biorthogonal wavelets," *Applied and Computational Harmonic Analysis*, vol. 3, no. 2, pp. 186–200, 1996.

[14] B. Pesquet-Popescu and V. Bottreau, "Three-dimensional lifting schemes for motion compensated video compression," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '01)*, vol. 3, pp. 1793–1796, Salt Lake, Utah, USA, May 2001.

[15] A. Secker and D. Taubman, "Motion-compensated highly scalable video compression using an adaptive 3D wavelet transform based on lifting," in *Proceedings of IEEE International Conference on Image Processing (ICIP '01)*, vol. 2, pp. 1029–1032, Thessaloniki, Greece, October 2001.

[16] A. Secker and D. Taubman, "Lifting-based invertible motion adaptive transform (LIMAT) framework for highly scalable video compression," *IEEE Transactions on Image Processing*, vol. 12, no. 12, pp. 1530–1542, 2003.

[17] L. Luo, J. Li, S. Li, Z. Zhuang, and Y.-Q. Zhang, "Motion compensated lifting wavelet and its application in video coding," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '01)*, pp. 365–368, Tokyo, Japan, August 2001.

[18] J. Reichel, H. Schwarz, and M. Wien, "Scalable Video Coding—Working Draft 2," Joint Video Team (JVT), Busan (KR), April 2005, Doc. JVT-O201.

[19] A. Tabatabai, Z. Visharam, and T. Suzuki, "Study of effect of update step in MCTF," in *Proceedings of the 17th Meeting of*

Joint Video Team (JVT '05), Nice, France, October 2005, Doc. JVT-Q026.

[20] D. S. Turaga and M. van der Schaar, "Content-adaptive filtering in the UMCTF framework," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '03)*, vol. 3, pp. 621–624, Hong Kong, April 2003.

[21] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Transactions of Image Processing*, vol. 1, no. 2, pp. 205–220, 1992.

[22] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 9, pp. 1445–1453, 1988.

[23] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 160–175, 1993.

[24] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic, Norwell, Mass, USA, 1988.

[25] B. Usevitch, "Optimal bit allocation for biorthogonal wavelet coding," in *Proceedings of the Data Compression Conference (DCC '96)*, pp. 387–395, Snowbird, Utah, USA, March-April 1996.

[26] J. Y. Huang and P. M. Schultheiss, "Block quantization of correlated Gaussian random variables," *IEEE Transactions on Communications*, vol. 11, no. 3, pp. 289–296, 1963.

[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*, vol. 2003, no. 1, pp. 56–65, 2003.

[28] I. J. Schoenberg, "Contributions to the problem of approximation of equidistant data by analytic functions—part A: on the problem of smoothing or graduation. A first class of analytic approximation formulae," *Quarterly of Applied Mathematics*, vol. 4, no. 1, pp. 45–99, 1946.

[29] I. J. Schoenberg, "Contributions to the problem of approximation of equidistant data by analytic functions—part B: on the problem of osculatory interpolation. A second class of analytic approximation formulae," *Quarterly of Applied Mathematics*, vol. 4, no. 2, pp. 112–141, 1946.

[30] M. Unser, "Splines: a perfect fit for signal and image processing," *IEEE Signal Processing Magazine*, vol. 16, no. 6, pp. 22–38, 1999.

[31] C. H. Reinsh, "Smoothing by spline functions," *Numerische Mathematik*, vol. 10, no. 3, pp. 177–183, 1967.

[32] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing—part II: efficient design and applications," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834–848, 1993.

[33] *JM 11.0 H.264/AVC reference software*, Joint Video Team (JVT). http://iphome.hhi.de/suehring/tml/.