SCALABLE CONTEXT-BASED MOTION VECTOR CODING FOR VIDEO COMPRESSION

Valéry VALENTIN, Marco CAGNAZZO, Marc ANTONINI, Michel BARLAUD*

I3S Laboratory, UMR 6070 of CNRS, University of Nice-Sophia Antipolis
Bât. Algorithmes/Euclide, 2000 route des Lucioles - BP 121 - 06903 Sophia-Antipolis Cedex, France
Phone: 33(0)4.92.94.27.21
Fax: 33(0)4.92.94.28.98
Email: {vvalenti,am,barlaud}@i3s.unice.fr

*Dipartimento di Ingegneria Elettronica e delle Telecomunicazioni
Universitá Federico II di Napoli
Via Claudio, 21 - 80125 Napoli, Italy
Email: cagnazzo@unina.it

ABSTRACT

State of the art video compression algorithms use motion compensation. Using such method gives better results by increasing the temporal correlation between pixels from consecutive frames. Most of the effort in increasing the performance of existing algorithms concentrates on the improvement of motion vector uses and residual coding.
In this paper we propose a motion vectors computation and coding scheme for video compression which combines constrained block matching and scalable coding based on a three-step search-like decomposition.

1. INTRODUCTION

Video coding is an important challenge in multimedia applications, and promising algorithms have been published over the past years. Ermegent standards, such as MPEG4 [1] or H26L [2], use advanced motion compensation to match pixels between frames. The advantages of matching pixels between frames are numerous, but the most usefull one is probably the ability to track camera and objects motion, resulting in an increase of the decorrelation transform performances. However, these algorithms use a Discret Cosinus Transform (DCT), and then introduce block artifacts in the decoded video, especially at low bitrates.

To achieve better compression than DCT based algorithms, 3D Wavelet coding has been introduced [3–11] as the new temporal and spatial transform. It provides encouraging results compared to MPEG by increasing the decorellation performance in both temporal and spatial dimension, and by removing block artifacts resulting from the use of a DCT transform. Another advantage of the Discret Wavelet Transform (DWT) in comparison with the DCT is its scalability properties. However, the main drawback of the 3D DWT is that it does not take motion compensation into account, and therefore cannot take advantage of it. To bypass this problem, lifting implementation of the 3D DWT has been developed [5, 9, 11–16]. Lifting can be viewed as an efficient DWT implementation that allows usage of motion compensation methods, and can be optimized to lower the memory requirements [17], while preserving the scalability properties of the DWT.

Generally, existing state of the art methods don't consider scalability in the motion vectors computation and coding steps, usefull for transmission on a variable bitrate environment. They simply transmit the vectors using prediction and entropy coding. Moreover, for each new frames to be coded, these methods need two motion vector estimations, one for the forward direction and the other for the backward direction. Since they need to be fully transmitted in complement to the residual information, their size can occupy more than 30% of the global rate. To take these two problems into account, we propose a vector computation and coding scheme that generates a scalable bitstream which can be progressively decoded by gradually adding vectors and residual information.

This paper first describes the Motion Compensated Lifting (MCLift) schema used in our method. We then present the computation and coding algorithms for the motion vectors we used in MCLift. We then propose a bitstream structure that allows progressive decoding and visualization. We finally present our experimental results.

2. MOTION COMPENSATED TEMPORAL TRANSFORM

The proposed video coder is based on a motion compensated 3D wavelet transform. We assume that temporal transform is separable, i.e. that it can be performed independently from spatial transform. The temporal transform generates many temporal subbands, according to the decomposition tree, and each of them is composed of several frames, on which the 2D spatial transform will be carried out.

2.1. The (2,2) lifting scheme

In this section, we will not consider the problem of motion compensation, focusing instead on the application of lifting scheme in video coding. More precisely, here we review how to use lifting scheme in order to perform wavelet transform along temporal dimension.

The lifting scheme is an efficient implementation of wavelet transform. As shown in [18], a wavelet filter bank (both for analysis and synthesis) can be implemented by a lifting scheme.

Basically, the lifting scheme implementation of wavelet transform consists in dividing the input signal in odd and even samples (i.e. samples from odd and even frames in the case of temporal analysis of video), on which linear operators are recursively applied. Let us see, for example, how the biorthogonal 5/3 filter can be implemented by lifting.

Let I_t be the frame number t, h_t (resp. l_t) the t^{th} high (resp. low) subband obtained after a wavelet transformation. Considering the biorthogonal 5/3 wavelet kernel applied on the temporal axis, we can write this filter into the following lifting scheme:

$$\begin{cases} h_t(x,y) = & I_{2t+1}(x,y) \quad -\frac{1}{2}[I_{2t}(x,y) - I_{2t+2}(x,y)] \\ l_t(x,y) = & I_{2t}(x,y) \quad +\frac{1}{4}[h_{t-1}(x,y) + h_t(x,y)] \end{cases} \tag{1}$$

The filter is then implemented by adding to the current frame samples the output of two linear operators (both of length two) which in turn depends on previous and next frames samples. A lifting scheme is often named after the length of these operators, so the 5/3 filter is also referred to as (2,2) lifting scheme. This lifting scheme is often used for temporal analysis as for example in [13, 15, 19].

2.2. Motion Compensation in (2,2) lifting scheme

In a typical video sequence, there is movement due to both camera panning and zooming, and objects' displacement and deformation. Then, if we perform wavelet transform along the temporal dimension without taking into account this movement, we end up to appling this transform to a signal characterized by many sudden changes. This means that we will obtain a high frequency band with significant energy, and a low frequency band with many artifacts as result of temporal low-pass filtering on moving objects. Then, we have a low coding gain and, moreover, temporal scalability is compromised, as the visual quality of the low temporal subband could be not satisfactory. In order to overcome these problems, motion compensation is introduced in the temporal analysis stage, as described in [15] and [19].

The basic idea is to carry out the temporal transform along motion direction. To better explain this concept, let us start with the simple case of constant uniform motion (this happens e.g. with a camera panning on a static scene). Let $(\Delta x, \Delta y)$ be the global motion vector. This means that an object (or, rather, a pixel) that is in position (x, y) in the frame t will be in position $(x + T\Delta x, y + T\Delta y)$ [1] in the frame $t + T$. Then, if we want to perform the wavelet transform along motion direction – or, in other words, if we want to perform motion compensated wavelet transform – we have to "follow the pixel" in its motion from a frame to another. This is possible as we know its position in each frame. So in the equations 1 we must replace all the next and previous frames' pixels with those at the correct locations:

$$\begin{cases} h_t(x,y) = & I_{2t+1}(x,y) - \frac{1}{2}[I_{2t}(x-\Delta x, y-\Delta y) \\ & -I_{2t+2}(x+\Delta x, y+\Delta y)] \\ l_t(x,y) = & I_{2t}(x,y) + \frac{1}{4}[h_{t-1}(x-\Delta x, y-\Delta y) \\ & +h_t(x+\Delta x, y+\Delta y)] \end{cases} \tag{2}$$

Now it is not difficult to generalize these formulas to generic motion. At this purpose, we use backward and forward estimated Motion Vector Fields (MVFs). We define $BW_t(x, y)$ (respectively $FW_t(x, y)$) as the (estimated) position that the pixel (x, y) in the frame t occupies in the frame $t - 1$ (respectively $t + 1$). Then in the case of constant uniform motion we have:

$$\begin{cases} BW_t(x,y) = & (x-\Delta x, y-\Delta y) \\ FW_t(x,y) = & (x+\Delta x, y+\Delta y) \end{cases} \tag{3}$$

In the general case, we can then modify equation 1 as follows:

$$\begin{cases} h_t(x,y) = & I_{2t+1}(x,y) - \frac{1}{2}[I_{2t}(BW_{2t+1}(x,y)) \\ & -I_{2t+2}(FW_{2t+1}(x,y))] \\ l_t(x,y) = & I_{2t}(x,y) + \frac{1}{4}[h_{t-1}(BW_{2t}(x,y)) \\ & +h_t(FW_{2t}(x,y))] \end{cases} \tag{4}$$

The resulting motion compensated (2,2) lifting scheme, for a one-level decomposition, is represented in Fig.1.

If motion estimation is accurate, motion compensated wavelet transform will generate a high frequency band with

[1] We are neglecting what happens near frame's boundaries.

low energy, as it take into account only objects' luminance and/or color variation and not their movement or deformation, and a low frequency band in which objects' position is precise and their shape is clear. So, thanks to motion compensation, we are able to preserve both high coding gain and scalability.

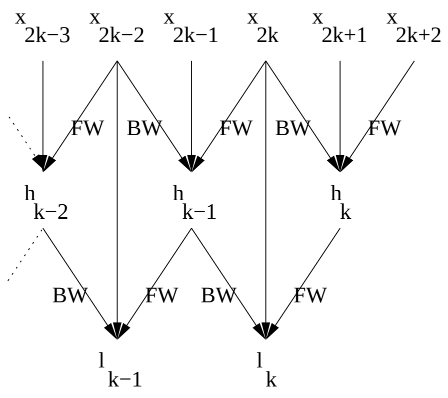

Fig. 1. Illustration of (2,2) Motion Compensated Lifting Scheme

3. MOTION VECTORS

Motion vectors estimation is an important step in a video coding algorithm, as it conditions the performance of the temporal decorrelation, and then the quality of the compression.

3.1. Exhaustive search

We consider block-based motion estimation techniques, so we introduce here some notation for block handling. Let $B_t^{(\mathbf{p})}$ be a block of size [2] $(n \times n)$ in the frame t, centred on pixel $\mathbf{p} = (x_p, y_p)$. We consider even values for n. Set $\mathcal{B}_\mathbf{0}^n = \{-n/2, \ldots, n/2-1\}^2$, $B_t^{(\mathbf{p})}(x,y) = I_t(x_p+x, y_p+y)\ \forall(x,y) \in \mathcal{B}_\mathbf{0}^n$.

To compute our motion vectors, we first need a good metric between blocks, $d(B_1, B_2)$. The usual Mean Square Error (MSE) criterion provides good results as long as the mean intensity in both frames is the same. But in real video shots, intensity may vary between two consecutive frames, and therefore MSE criterion can produce inaccurate motion estimations. To overcome this problem, we use the Zero-mean Normalized Sum of Squared Differences (ZNSSD) criterion, which is robust to affine intensity transformations while remaining easy to compute. For two given blocks $B_t^{(\mathbf{p})}$ and $B_t^{(\mathbf{q})}$ the ZNSSD formula is given by:

$$\begin{aligned}
&\text{ZNSSD}(B_t^{(\mathbf{p})}, B_t^{(\mathbf{q})}) = \\
&\textstyle\sum_{(x,y)\in\mathcal{B}_0^n} \left[(B_t^{(\mathbf{p})}(x,y) - \overline{B_t^{(\mathbf{p})}}) - (B_t^{(\mathbf{q})}(x,y) - \overline{B_t^{(\mathbf{q})}})\right]^2 \\
&\div \left[\textstyle\sum_{(x,y)\in\mathcal{B}_0^n} (B_t^{(\mathbf{p})}(x,y) - \overline{B_t^{(\mathbf{p})}})^2 \cdot \right. \\
&\left. \textstyle\sum_{(x,y)\in\mathcal{B}_0^n} (B_t^{(\mathbf{q})}(x,y) - \overline{B_t^{(\mathbf{q})}})^2 \right]^{1/2}
\end{aligned} \tag{5}$$

where overlining stands for the mean value of a block.

For a given block $B_t^{(\mathbf{p})}$ in frame t, we look for the best corresponding block $B_{t+1}^{(\mathbf{p}+\mathbf{v})}$ in frame $t+1$ with a maximal allowed distance of d, by minimizing the ZNSSD criterion:

$$\mathbf{v}^* = \min_{\mathbf{v}\in W} \left[\text{ZNSSD}\left(B_t^{(\mathbf{p})}, B_{t+1}^{(\mathbf{p}+\mathbf{v})}\right)\right] \tag{6}$$

where $W = \{-d, \ldots, d\} \times \{-d, \ldots, d\}$ is the allowed search window for block matching.

This exhaustive search provides a $\mathbf{v}^*$ vector which guarantees the best performances for the subsequent temporal analysis. The estimated FW_t is:

$$FW_t(\mathbf{q}) = \mathbf{q} + \mathbf{v}^* \qquad \forall \mathbf{q} \in \mathcal{B}_\mathbf{p}^m \tag{7}$$

where

$$\begin{aligned}
\mathcal{B}_\mathbf{p}^m &= \{x_p - m/2, \ldots, x_p + m/2 - 1\} \\
&\times \{y_p - m/2, \ldots, y_p + m/2 - 1\}
\end{aligned} \tag{8}$$

and $m \leq n$. For $m = n$, we have the usual non-overlapped block-matching criterion, while if we set $\mathcal{B}_\mathbf{p}^m = \mathbf{p}$ we compute a different motion vector for each pixel. The advantage of overlapped block-matching ($m < n$) with respect to non-overlapped block-matching is that, at the cost of a slightly increased computational burden, we achieve a smoother (less sensitive to noise) MVF.

3.2. Constrained Motion Vectors

In order to allow the receiver to correctly decode the video sequence, we need to send the two motion vector fields for each frame. This side information can grow up to constitute a remarkable share of the total bit-rate. Here we have a resource allocation problem: we can use our bit budget to encode the MVFs, but we can also try to have a less accurate description of them, using the spared bits to better encode the WT coefficients. This problem calls for an optimal rate-distortion solution, but it seems quite difficult to take into account MVF encoding and residual encoding at the same time, as the second term depends on the first one.

So we look for some suboptimal solution. A promising one is what we call Constrained Motion Vectors. We impose a constrain on MVFs that, on one hand, allows us to consistently reduce the rate needed for MVF's encoding,

[2]Square blocks are not necessary. They are used here just for sake of simplicity.

but, on the other hand, prevents us to achieve the best motion estimation. The imposed constraint is very simple: we want the backward MVF to be the opposite of the forward one. Then we search for a displacement vector, which under this constrain minimizes a quantity depending on both the forward and the backward error. In this work we chose the sum of them:

$$\mathbf{v}^* = \min_{\mathbf{v}\in W} \left[d\left(B_t^{(\mathbf{p})}, B_{t-1}^{(\mathbf{p}-\mathbf{v})}\right) + d\left(B_t^{(\mathbf{p})}, B_{t+1}^{(\mathbf{p}+\mathbf{v})}\right)\right] \quad (9)$$

Where $d(B_1, B_2)$ is a suitable metric, as the ZNSSD proposed in the previous section. The advantage of constrained search is that we obtain symmetrical MVF, so we can send just every second MVF, using the spared bit budget to better encode wavelet coefficients.

On the other hand, constrained search does not allow us to get the best motion estimation, except for some specific motion configuration, as the constant motion (i.e. zero acceleration) case. In order to better understand the trade-off between accurate MVF description and improved wavelet coefficient encoding, some experiments were performed.

In the first one we compared the codec performances when the MVF where estimated with the unconstrained and with the constrained search. The experiments were carried out on the sequence “Foreman”, with a block size of 16 pixel and an overlap of four. The results are shown in figure 2. Note that the constrained search method requires about half the rate for MVF with respect to the unconstrained method. Anyway, in this figure we did not take into account the rate needed for MVF encoding, as we want just to understand how much the worse motion estimation affects motion compensated wavelet transform. The graph shows that the loss is quite small, as the MSE increase varies from 1% to 9%

In order to perform a whole comparison between the two methods, in figure 3 we considered also the cost of MVFs encoding (assessed by computing their entropy). Moreover, the performances of the codec without motion compensation were also added. The graph shows that the proposed method is the best at low and medium rates and is roughly equivalent to unconstrained search method at high rates. If motion compensation is not carried out, performances are competitive only at very low rates, where it is better to use the meagre bit budget to encode WT coefficient rather than the MVFs.

3.3. Proposed Scalable Coding Scheme

3.3.1. Motion Vectors Scalability

Existing Motion Compensated Lifting schemes [15,19] suffer from their lack of scalability in motion vectors representation. Our method introduces scalability property in motion vectors by iteratively refining their estimation and therefore allowing a progressive decoding of the coded stream. We first present our method on a practical example, before extending it to the general case.

For a given block (i, j), we consider a search window of size 15 for our motion vectors estimation, with pixel accuracy. The criteria presented in previous part is applied on all the positions in the search window defined above, and error values are stored in a 2 dimension table $T_{(i,j)}$ where $T_{(i,j)}(x, y)$ gives the error value for motion vector (x, y). Let $(X_{i,j}, Y_{i,j})$ be the best motion vector given by the lowest error value in the table for the considered block. The aim of our method is to decompose this vector so that it can be progressively decoded. We base our decomposition method on the 3 Steps Search (3SS) algorithm presented in [20].

The 3SS algorithm uses an iterative process to progressively refine the motion vector. In a first step, it seeks the best motion vector among nine general vectors which both

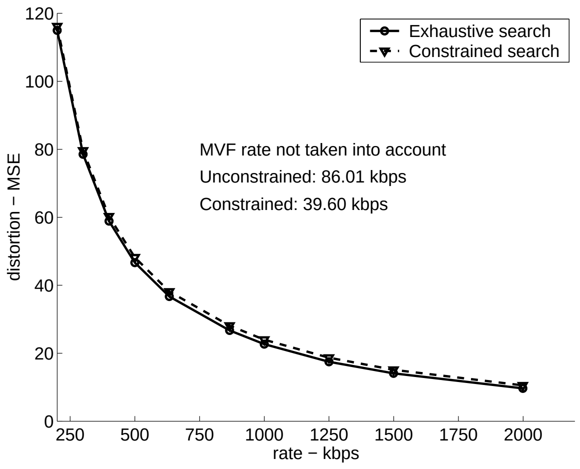

Fig. 2. Comparison between ME methods

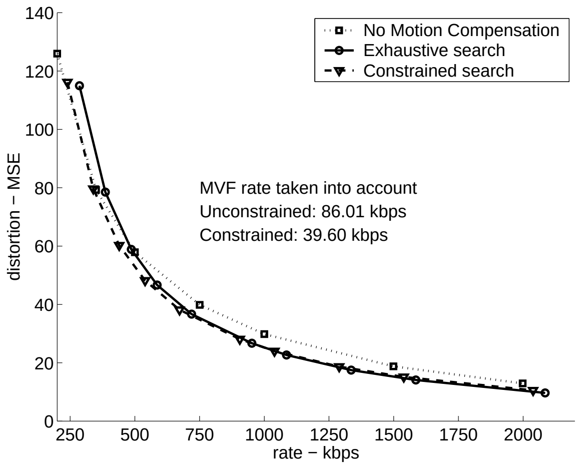

Fig. 3. Impact of different ME methods on codec performances

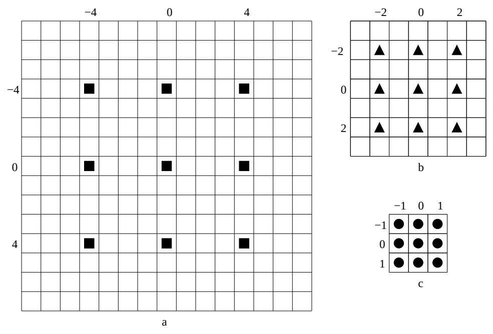

Fig. 4. 3SS Refinement steps

coordinates $X_{i,j}(1)$ and $Y_{i,j}(1)$ can independantly take their value in the set $\{-4,0,4\}$ (see Fig.4a). Once the first approximation is found, the method consists of refining the vectors by seeking a new estimation around the selected position, again among nine possibilities, but with a closer search area. Each coordinate $X_{i,j}(2)$ and $Y_{i,j}(2)$ can now take value in the set $\{-2,0,2\}$ (see Fig.4b). The final step is then done by refining a last time the motion vector with a pixel accuracy search area around the selected position. $X_{i,j}(3)$ and $Y_{i,j}(3)$ takes their value in the set $\{-1,0,1\}$ (see Fig.4c). The total motion vector $(X_{i,j},Y_{i,j})$ is then deduced using the Eq.10.

$$(X_{i,j},Y_{i,j})=\sum_{k=1}^{3}(X_{i,j}(k),Y_{i,j}(k)) \qquad (10)$$

This method of decomposition introduces more decomposition paths than the number of possible motion vectors. In our case, we see there are 9^3 distinct decompositions for only 15^2 available motion vectors. This results in a multiple paths possibility for a given vector. For exmple, Fig.5 represents the 4 possible decompositions (see Eq.11) for the motion vector $(5,5)$.

$$\begin{cases} \text{case a} & : \quad (4,4)+(0,2)+(1,-1) \\ \text{case b} & : \quad (4,4)+(2,0)+(-1,1) \\ \text{case c} & : \quad (4,4)+(2,2)+(-1,-1) \\ \text{case d} & : \quad (4,4)+(0,0)+(1,1) \end{cases} \qquad (11)$$

In the first iteration, only one motion vector is allowed $(4,4)$, while in step 2, four different decompsitions are possible $(0,2)$, $(2,0)$, $(2,2)$, $(0,0)$, resulting in four distinct decomposition paths. The third step is determined by the previous one. To select one decomposition among the different ones available, we choose at each step the path that gives the lowest criteria error value.

A closer look to the decomposition algorithm shows us that in the kth step, motion vector coordinates are in the set $(-\frac{8}{2^k},0,\frac{8}{2^k})$. We can then write Eq.10 into the form of

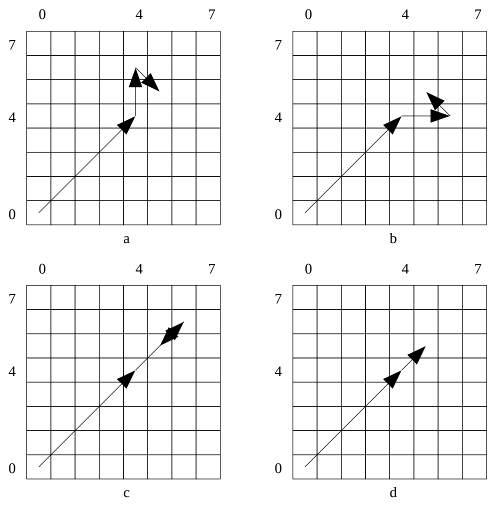

Fig. 5. Example of the multiple decompositions problem

Eq.12, with $mvd^x_{i,j}(k)$ and $mvd^y_{i,j}(k)$ in $\{-1,0,1\}$.

$$(X_{i,j},Y_{i,j})=\sum_{k=1}^{3}((mvd^x_{i,j}(k),mvd^y_{i,j}(k))*\frac{8}{2^k}) \qquad (12)$$

For a given decomposition step k, we can construct two bitplanes $P_x(k)$ and $P_y(k)$ filled with the $mvd^x_{i,j}(k)$ and $mvd^y_{i,j}(k)$ values of all the blocks (i,j). Those bitplanes are then coded using a context-based coder. The context is estimated using information around the current bit to code for spatial correlation, and with previous frame information for temporal correlation.

4. CONCLUSIONS AND FUTURE WORK

In this work, a new approach was presented, to estimate and encode motion vector fields in video compression.

The new technique of motion estimation, called Constrained Motion Vectors, allows good estimation with respect to the usual unconstrained search and a reduced rate requirement, ending up with better overall performances. Variations of this technique are under investigation, in which the constrained search minimizes the maximum of backward and backward errors, instead of their sum as in (9).

The proposed method of MVF encoding introduces scalability in motion vectors representation. This property can be exploited in order to improve the scalability of the encoded video stream. In fact, we are currently studying a new layered bitstream structure, in which, thanks to this property, we can have, right from the base layer, a rough description of MVFs, and we can progressively refine this information in successive layers. This structure allows a better resource allocation between different layers, as we are no longer obliged to send motion information all at the same time, and then we can expect better performances than schemes that do not have MVF scalability, especially for first layers.

5. REFERENCES

[1] *ISO MPEG4 Standard*.

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

[3] G. Karlsson and M. Vetterli, “Three-dimensionnal subband coding of video,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, New York, USA, Apr. 1988, pp. 1100–1103.

[4] C. Podilchuck, N. Jayant, and N. Farvardin, “Three-dimensionnal subband coding of video,” *IEEE Transactions on Image Processing*, 1995.

[5] S. Choi and J. Woods, “Motion-compensated 3-D subband coding of video,” *IEEE Transactions on Image Processing*, vol. 8, no. 2, Feb. 1999.

[6] D. Taubman and A. Zakhor, “Multirate 3-D Subband Coding of Video,” *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 572–588, Sept. 1994.

[7] B.-J. Kim and W. Pearlman, “An embedded wavelet video coder using three-dimensionnal set partitionning in hierarchical trees (SPIHT),” in *Proceedings of Data Compression Conference*, Snowbird, USA, Mar. 1997, pp. 251–260.

[8] B. Felts and B. Pesquet-Popescu, “Efficient context modeling in scalable 3D wavelet-based video compression,” in *Proceedings of IEEE International Conference on Image Processing*, Vancouver, Canada, Sept. 2000.

[9] A. Wang, Z. Xiong, P. Chou, and S. Mehrotra, “Three-dimensional wavelet coding of video with global motion compensation,” in *Proceedings of Data Compression Conference*, Snowbird, USA, Mar. 1999.

[10] J. Xu, S. Li, Y.-Q. Zhang, and Z. Xiong, “A wavelet video coder using three-dimensional embedded subband coding with optimized truncation (3-D ESCOT),” in *Proceedings of IEEE Pacific-Rim Conference on Multimedia*, Sydney, Australia, Dec. 2000.

[11] J.-R. Ohm, “Three dimensional subband coding with motion compensation,” *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 559–571, Sept. 1994.

[12] ——, “Motion-compensated wavelet lifting filters with flexible adaptation,” in *Proceedings of Tyrrhenian International Workshop on Digital Communications*, Capri, Italy, Sept. 2002.

[13] J. Viéron, C. Guillemot, and S. Pateux, “Motion compensated 2D+t wavelet analysis for low rate fgs video compression,” in *Proceedings of Tyrrhenian International Workshop on Digital Communications*, Capri, Italy, Sept. 2002.

[14] T. Wiegand and B. Girod, *Multiframe Motion-compensated Prediction for Video Transmission*, 2001.

[15] 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*, Tokyo, Aug. 2001, pp. 481–484.

[16] C. Parisot, M. Antonini, and M. Barlaud, “Motion-compensated scan based wavelet transform for video coding,” in *Proceedings of Tyrrhenian International Workshop on Digital Communications*, Capri, Italy, Sept. 2002.

[17] C. Parisot, “Allocations basées modèles et transformées en ondelettes au fil de l’eau pour le codage des images et des vidéos,” Ph.D. dissertation, Universite de Nice Sophia-Antipolis, France, 2003.

[18] I. Daubechies and W. Sweldens, “Factoring wavelet transforms into lifting steps,” *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 245–267, 1998.

[19] 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*, Thessaloniki, Greece, Oct. 2001, pp. 1029–1032.

[20] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion-compensated interframe coding for video conferencing,” in *Proceedings NTC (IEEE)*, 1981.