This document is published in:

2013 IEEE International Conference on Image Processing (ICIP), 15-18 Sept. 2013, Melbourne (Australia), pp. 1948-1952

DOI: 10.1109/ICIP.2013.6738401

FILTER OPTIMIZATION AND COMPLEXITY REDUCTION FOR VIDEO CODING USING GRAPH-BASED TRANSFORMS

E. Martinez-Enriquez, F. Diaz, J. Cid-Sueiro

Dept. of Signal Theory and Communications
Universidad Carlos III
Madrid, Spain

Antonio Ortega

Department of Electrical Engineering
University of Southern California
Los Angeles, California, USA

ABSTRACT

The basis functions of lifting transform on graphs are completely determined by finding a bipartition of the graph and defining the prediction and update filters to be used. In this work we consider the design of prediction filters that minimize the quadratic prediction error and therefore the energy of the detail coefficients, which will give rise to higher energy compaction. Then, to determine the graph bipartition, we propose a distributed *maximum-cut* algorithm that significantly reduces the computational cost with respect to the centralized version used in our previous work. The proposed techniques show improvements in coding performance and computational cost as compared to our previous work.

Index Terms— Wavelet transforms, Video coding, MCTF, Lifting, Graphs

1. INTRODUCTION

When using typical coding approaches based on standard separable wavelet transforms, overall bit-rates can be reduced for a given image or video if the signal is compacted into a smaller number of large coefficients. Encoding images or videos with large luminance discontinuities (e.g., complex contours) may give rise to many large high-pass coefficients near these discontinuities, which can be costly in terms of rate. Directional transforms are able to adapt their basis functions in order to filter along high-correlation paths (e.g., directions of low variation in pixel intensity in an image or video sequence), avoiding filtering across large discontinuities, resulting in smaller high frequency coefficients in those locations. For image coding, several works have been proposed in this area [1], [2], [3], [4]. For video coding, some approaches have been developed in which the filtering directions follow motion trajectories (motion-compensated temporal filtering (MCTF)) [5],[6]. The lifting scheme [7], which allows to construct multiresolution signal representations in a simple way, has been used in some of these works [5],[6], [2].

In [8] we presented a new lifting-based wavelet transform for video signals that allows performing the filtering operations following arbitrary 3-dimensional (spatio-temporal) directions. Our transform is invertible, critically sampled and "non-separable", in the sense that spatio-temporal filtering operations are used, and can be considered as a generalization of wavelet-based video encoders. The transform is based on the extension of lifting wavelet transforms on arbitrary graphs data [9] to J levels of decomposition. The work in [8] showed improvements in performance as compared to the LIMAT method [5] and to a motion compensated DCT video encoder in terms of non-linear approximation (PSNR as a function of non-zero coefficients). In [10] the transform was used to obtain a complete video encoder. This work proposed a new coefficient reordering technique and a low complexity version of the transform that operates on subgraphs of the original graph, reducing the overall complexity. The proposed scheme showed encouraging results as compared to a simplified DCT based encoder. Nevertheless, our previous work in [10] had two major drawbacks. First, edge weights in the graph, which were used to define the prediction and update filters and the reordering of the coefficients, were experimentally set to $w_t = 10$ and $w_s = 2$ for temporal and spatial edges respectively, based on the assumption that temporal prediction is usually more accurate that the spatial one. These weights were fixed and could not change with the video content. Second, in the low complexity version of the transform, which performs the bipartition of the graph working in subgraphs, the final encoder complexity depended on the motion of the sequence, so that there would be no complexity reduction for relatively high motion sequences such as Carphone or Football.

The problem of optimizing prediction filters in lifting transforms has been considered by several authors, typically using optimization criteria to minimize the expected energy of the detail coefficients. In this way, [11] obtains the optimal predictors of an arbitrary lifting scheme and applies them to lossless image compression. [12] minimizes the energy of the detail coefficients through an additional prediction step, improving the compression performance. [13] designs a prediction step that minimizes the expected energy of the detail signal in a generalized lifting scheme, and [14] proposes to jointly find the forward and backward motion vectors that minimize the energy of detail coefficients in a motion compensated 5/3 transform.

The main contribution of this paper is to extend these prediction filter optimization techniques to a graph-based lifting transform in order to minimize the quadratic prediction error (the energy of the detail coefficients). Specifically, we consider an undirected graph (that represents the video information) in which any node can have an arbitrary number of spatial and temporal neighbors, and we obtain the optimal spatial and temporal prediction weights.

We also discuss here a new low complexity approach for determining the graph bipartition that operates in a distributed manner, reducing the complexity of the *maximum-cut* greedy algorithm used in [10] independently of the video content. We achieve average gains of around 0.4 dB as compared to [10], while reducing the complexity of the graph partition process by a factor of over 200. Note that the proposed low complexity approach deals with the graph partition process, which is one of the most complex processes of the system, incurring a negligible loss of performance.

The rest of the paper is organized as follows. Section 2 summa-

This work was supported in part by NSF under grant CCF-1018977

rizes the graph-based transform for video coding proposed in [10]. Section 3 introduces the proposed weighting of the graph. Our low complexity approach is presented in Section 4. Experimental results are provided in Section 5 and conclusions in Section 6.

2. GRAPH-BASED TRANSFORM FOR VIDEO CODING

2.1. Graph Representation of the Video Sequence

The first step to perform the transform is to obtain a graph representation of the video content in which every node represents one pixel. In the graph, any pixel can be linked to multiple spatial or temporal neighbors. To obtain an efficient transform it is essential that linked pixels have similar luminance values. Therefore, to exploit spatial correlation, a pixel is linked to any pixel in its 8-connected neighborhood if no contour[1] exists between them. Regarding the temporal correlation, we link those pixels that are connected to each other by a motion vector transmitted as side information. Finally, note that links between pixels in the graph are weighted as a function of the expected correlation between the pixel intensities. This weighting will influence the design of the basis functions, as well as the reordering of the coefficients generated by the transform. In [10], temporal connections were weighted with a value of $w_t = 10$ and spatial connections with $w_s = 2$.

2.2. Lifting Transforms on Graphs

Once we have the graph, we want to obtain a multi-resolution representation of this graph using wavelets constructed by means of the lifting approach [7]. In lifting, transform invertibility can be guaranteed if the input data at each specific level of decomposition j is split into prediction ($\mathcal{P}_j$) and update ($\mathcal{U}_j$) disjoint sets; we refer to this as the $\mathcal{U}/\mathcal{P}$ assignment process. Then, prediction ($\mathbf{p}_{m,j}(m \in \mathcal{P}_j)$) and update ($\mathbf{u}_{n,j}(n \in \mathcal{U}_j)$) filters may be defined so that the m-th detail coefficient at level j, $d_{m,j}$, can be computed from $h \in \mathcal{U}_j$ update neighbors, and the n-th smooth coefficient $s_{n,j}$ can be computed from $l \in \mathcal{P}_j$ prediction neighbors as:

$$\begin{aligned} d_{m,j} &= s_{m,j-1} + \sum_{h \in \mathcal{U}_j} \mathbf{p}_{m,j}(h) s_{h,j-1} \\ s_{n,j} &= s_{n,j-1} + \sum_{l \in \mathcal{P}_j} \mathbf{u}_{n,j}(l) d_{l,j}. \end{aligned} \quad (1)$$

Note that, using this notation, the original raw data (the luminance value of the pixels) will be $x^{(k)} = s_{n,1}$, where k is the pixel index. To carry out a multi-resolution analysis, we apply this process iteratively. First, we construct the graph at decomposition level $j+1$ using information from the graph at level j, by connecting those update nodes that were either directly linked or two hops away from each other in the graph at level j. Then, we perform the $\mathcal{U}_{j+1}/\mathcal{P}_{j+1}$ assignment process. Finally, the smooth coefficients at (j)-th decomposition level of the transform are projected onto the approximation and detail subspaces, giving rise, respectively, to the smooth-low pass ($s_{n,j+1}$) and detail-high pass ($d_{m,j+1}$) coefficients at the $(j+1)$-th decomposition level. Figure 1 illustrates the complete process and the subband decomposition obtained after two levels of the transform. Refer to [8] for details.

[1] To avoid confusion we call image "contours" edges that appear in the image, between sets of pixels of different intensities, while we reserve the term "edge", for the links between vertices in the graph.

2.3. $\mathcal{U}/\mathcal{P}$ assignment and Filter Design

Given that the transform operates on a bi-partite graph (i.e., we discard the edges between nodes of the same color), the $\mathcal{U}/\mathcal{P}$ assignment is essentially a 2-color graph coloring problem [9]. The criterion used in [10] to perform the $\mathcal{U}/\mathcal{P}$ assignment was to maximize the total weight of the edges between the $\mathcal{P}$ and the $\mathcal{U}$ sets, which is equivalent to the formulation of the *weighted maximum cut problem*. We employed the greedy solution given in [15]. The prediction and update filters were choosen to take into account the expected correlation between nodes as described in [8].

3. OPTIMAL FILTER DESIGN

In our previous work the weights on the graph were experimentally fixed, with values $w_t = 10$ and $w_s = 2$ for temporal and spatial correlation, respectively. However, the local correlation between temporal and spatial neighbors changes with the video content, and thus the value of optimal prediction weights would change as well. We now find the prediction filter weights that minimize the quadratic prediction error, achieving small detail coefficients and thus high energy compaction.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph, where $\mathcal{V} = \{1, \ldots, N\}$ is a set of nodes and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ a set of edges. Let $\mathcal{S}, \mathcal{T}$ the set of spatial and temporal edges respectively, with $\mathcal{S} \cup \mathcal{T} = \mathcal{E}$. Denote one-hop spatial neighborhood of k, $\mathcal{N}_s^k = \{u \in \mathcal{V} : ku \in \mathcal{S}\}$ for all nodes $k \in \mathcal{V}$. Thus, the mean value of the spatial neighbors of a node k is defined as

$$\bar{x}_s^{(k)} = \frac{1}{|\mathcal{N}_s^k|} \sum_{i \in \mathcal{N}_s^k}^{|\mathcal{N}_s^k|} x^{(i)}, \quad (2)$$

where $\left|\mathcal{N}_s^k\right|$ is the number of spatial neighbors of k. Similarly for the temporal neighbors. Assuming that every node k is linearly predicted from its spatial and temporal neighbors as $\hat{x}^{(k)} = w_s \bar{x}_s^{(k)} + w_t \bar{x}_t^{(k)}$, we want to find the weights w_s and w_t that minimize the quadratic prediction error over all the nodes k:

$$\min_{w_s, w_t} \sum_k \left(x^{(k)} - w_s \bar{x}_s^{(k)} - w_t \bar{x}_t^{(k)} \right)^2. \quad (3)$$

Differentiating with respect to w_s and w_t we obtain the solution:

$$\mathbf{w}^* = (w_s^*, w_t^*) = \mathbf{R}^{-1}\mathbf{r} \quad (4)$$

where

$$\mathbf{R} = \begin{bmatrix} \sum_k \bar{x}_s^{(k)} \bar{x}_s^{(k)} & \sum_k \bar{x}_s^{(k)} \bar{x}_t^{(k)} \\ \sum_k \bar{x}_t^{(k)} \bar{x}_s^{(k)} & \sum_k \bar{x}_t^{(k)} \bar{x}_t^{(k)} \end{bmatrix}$$

and

$$\mathbf{r} = \sum_k x^{(k)} \begin{bmatrix} \bar{x}_s^{(k)} \\ \bar{x}_t^{(k)} \end{bmatrix}$$

are the correlation matrices.

Usually, the graph topology is defined by means of its adjacency matrix. Let $\mathbf{A_s} = \left[a_{s_{i,j}}\right]$ and $\mathbf{A_t} = \left[a_{t_{i,j}}\right]$ be the adjacency matrices of the subgraphs containing only the spatial and temporal edges, respectively, where each column is normalized (i.e., $a_{s_{i,j}} = 1/\left|\mathcal{N}_s^j\right|$ if $ij \in \mathcal{S}$; $a_{s_{i,j}} = 0$ if $ij \notin \mathcal{S}$). "Vectorizing" the sequence into a $1 \times MNF$ row vector $\mathbf{x}$, where MN is the frame size and F the number of frames considered, we can write:

$$\mathbf{w}^* = \begin{bmatrix} \mathbf{x}\mathbf{A_s}\mathbf{A_s^T}\mathbf{x^T} & \mathbf{x}\mathbf{A_s}\mathbf{A_t^T}\mathbf{x^T} \\ \mathbf{x}\mathbf{A_s}\mathbf{A_t^T}\mathbf{x^T} & \mathbf{x}\mathbf{A_t}\mathbf{A_t^T}\mathbf{x^T} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \mathbf{x}\mathbf{A_s}\mathbf{x^T} \\ \mathbf{x}\mathbf{A_t}\mathbf{x^T} \end{bmatrix}.$$

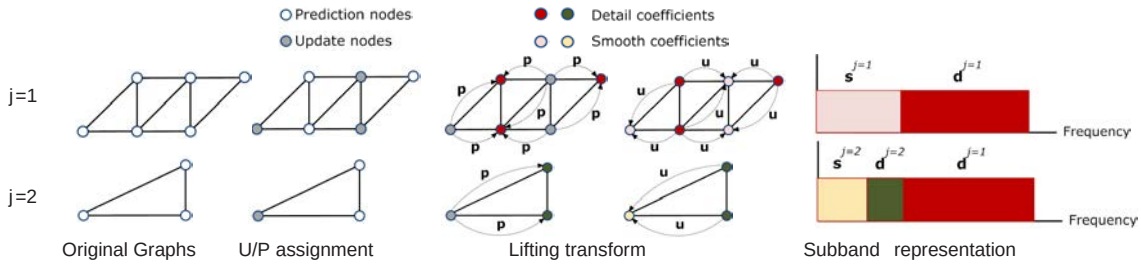


Fig. 1. Lifting transforms on graphs. Two levels of decomposition.

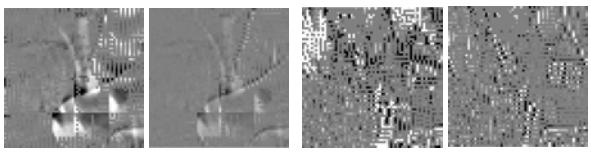
Table 1. Comparison between different prediction filters. Detail coefficients energy per coefficient in $j = 1$: $E_{d_{j=1}}$

	Carphone	**Mobile**	**Airshow (scene cut)**	**Football (fast motion)**
$E_{d_{j=1}}$ **[10]**	14	44	34	408
$E_{d_{j=1}}$ **Proposed ($\mathbf{w}^*$)**	12	37	17	240

These coefficients can be computed for any subgraph $\mathcal{H} \subset \mathcal{G}$ (e.g., block-by-block or frame-by-frame) and at any level of decomposition j. Given that the weights should be sent to the decoder as side information, a trade-off exists between accuracy in the weights selection (lower subgraphs sizes) and side information to be sent.

Table 1 shows examples of detail coefficient energy normalized by the number of prediction nodes in the first level of the transform $j = 1$ $\left(E_{d_{j=1}} = \frac{1}{|\mathcal{P}_{j=1}|} \sum_{m \in \mathcal{P}_{j=1}}^{|\mathcal{P}_{j=1}|} d^2_{m,j=1}\right)$ obtained coding 20 frames using the proposed weights (calculated in a frame-by-frame basis) and using the weights of [10]. Note that the $E_{d_{j=1}}$ is lower with the proposed method for all the considered cases.

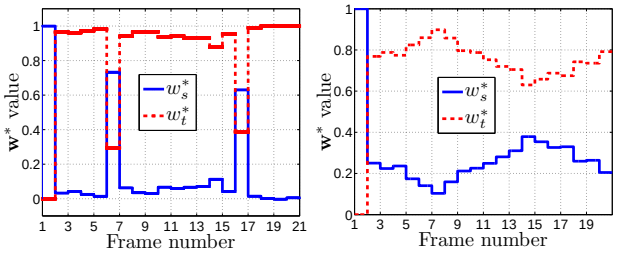
Fig. 2 shows the detail coefficient values obtained using the propposed approach (right part of each subfigure) and the fixed weights of [10] (left part of each subfigure). The example corresponds to a region of a specific frame of "Airshow" (scene cut) and "Football" (fast motion). Darker colors indicate higher negative coefficient values, while brighter colors mean higher positive coefficient values. Grey indicates coefficients close to zero. It can be seen that the absolute value of the detail coefficients is lower in the proposed approach. Specifically, in the scene cut of "Airshow", $(w_s^*, w_t^*) = (0.7, 0.3)$, and thus the filtering mainly follows the spatial directions, giving rise to better predictions and lower detail coefficients energy. The evolution of the (w_s^*, w_t^*) values is shown in Fig. 3. Observe that, in "Airshow", w_t^* is close to one (actually "Airshow" is a very static sequence) except in the scene cuts (frames number 6 and 16), where w_s^* becomes larger. Also note that the first frame does not have any temporal forward neighbor, and therefore $w_s^* = 1$ and $w_t^* = 0$.



(a) Airshow (b) Football

Fig. 2. Detail coeffients value.

The $\mathcal{U}/\mathcal{P}$ assignment and the optimal weighting problems are coupled, so that to obtain the *weighted maximum-cut* solution we need to know the graph weights and, on the other hand, to obtain the optimal prediction weights we need to know which nodes are



(a) Airshow (b) Football

Fig. 3. $\mathbf{w}^*$ evolution.

prediction nodes (i.e., the $\mathcal{U}/\mathcal{P}$ assignment solution). In this way, given an $\mathcal{U}/\mathcal{P}$ assignment, the optimal weights should be obtained summing just over the nodes $k \in \mathcal{P}$ in the minimization problem in (3). The problem could be solved iteratively, but the computational cost would be significant. We experimentally observed that the weight values do not significantly change if they are calculated before (over all the nodes of the graph) or after the $\mathcal{U}/\mathcal{P}$ assignment (over the prediction nodes), and therefore we consider our solution a good approximation.

Once we have defined the prediction filters, for each update node we design an update filter that is orthogonal to the prediction filters of its neighboring prediction neighbors, as proposed in [16]. In spite of the fact that this do not imply that the update filters are orthogonal to all the prediction filters, this solution reduces the impact of the "worse-case" coherence, because the prediction filters centered in prediction nodes that are not neighbors have little or no common support with the given update filter. Other approaches for the update filters design can be found in the literature [17], [18].

So far we have completely defined the weighting on the graph and the prediction and update filters. In the next section we explain how to perform the $\mathcal{U}/\mathcal{P}$ assignment.

4. DISTRIBUTED $\mathcal{U}/\mathcal{P}$ ASSIGNMENT

In [10] we proposed a transform that operated on subgraphs of the original graph in order to reduce the $\mathcal{U}/\mathcal{P}$ assignment complexity, which increases rapidly with the number of nodes N of the graph. Nevertheless, this approach has the drawback that the final complexity depends on the motion of the video sequence (faster motion sequences tend to lead to larger subgraphs and thus to higher computational cost), leading to practically no complexity reduction in sequences such as Football or Carphone.

We now propose an approach for performing the $\mathcal{U}/\mathcal{P}$ assignment that works in a distributed manner, leading to a computational complexity almost independent of the video content. This method reduces the complexity of the *weighted maximum-cut* greedy algorithm used, from the $O\left(N^3 \cdot \log N\right)$ worse-case complexity of [15], to $O\left(\frac{N}{B} B^3 \cdot \log B\right)$, where B is the block size used in the algorithm. Note that for a fixed B, the complexity increases linearly

with N in the proposed approach.

The idea consists in calculating the *weighted maximum-cut* solution in blocks of size B, making local $\mathcal{U}/\mathcal{P}$ decisions, and "transferring" this information to neighboring blocks. This is achieved by operating with overlapping blocks. Note that there exists a complexity-precision trade-off in the selection of B. The larger the block size B, the more complex and accurate solution.

The proposed greedy solution is described in Algorithm 1, where $\mathcal{U}_j$ and $\mathcal{P}_j$ form a bipartition of the node set $\mathcal{U}_{j-1}$, $\mathcal{F}_i$ and $\mathcal{G}_i$ form a bipartition of $\mathcal{B}_i$, and we consider *Gain* of a node to be the sum of weights of all its incident edges. The algorithm requires $\mathcal{N}_\mathcal{B}$ blocks of size B so that $\bigcup_{i \in \mathcal{N}_\mathcal{B}} \mathcal{B}_i = \mathcal{V}$, covering all the nodes of the graph. Every block must "see" the decisions taken in neighboring blocks, which in the algorithm means that $\mathcal{B}_i \cap \mathcal{B}_j \neq \varnothing$, where i is the block to be processed and j is each one of the already processed neighboring blocks. The intersection is the information that they share, and must include the nodes in $\mathcal{B}_j$ that have edges that go from block j to block i. Fig. 4 illustrates two iterations of the algorithm. In the first iteration (left part of the figure), a local *weighted maximum-cut* solution is found in block B_1. Then, in the second iteration, block B_2 includes the nodes of B_1 that have edges that go from B_1 to B_2 (boundary nodes). Therefore, the local *weighted maximum-cut* in B_2 is influenced by the already known colors of these boundary nodes, which means that the solution for block B_1 affects the solution for block B_2. With this simple approach we get the speed-up benefits of operating with blocks, while guaranteeing a consistent solution across blocks.

Algorithm 1 Distributed Weighted Maximum Cut Algorithm

Require: $\mathcal{U}_j = \{\varnothing\}, \mathcal{P}_j = \{\mathcal{U}_{j-1}\}, \mathcal{N}_\mathcal{B}$ blocks of size B
for $i = 1$ **to** $\mathcal{N}_\mathcal{B}$ **do**
  $\mathcal{F}_i = \{\varnothing\}$ and $\mathcal{G}_i = \mathcal{B}_i$
  $\mathcal{F}_i \leftarrow \mathcal{B}_i \cap \mathcal{U}_j$ and $\mathcal{G}_i \leftarrow \mathcal{G}_i \backslash \mathcal{F}_i$
  Change the sign of the incident edge weights to every node $f \subset \mathcal{F}_i$
  Calculate the $Gain$ of the nodes $\subset \mathcal{B}_i$
  Select the node a with largest $Gain$, $a = \max(Gain)$
  while $Gain > 0$ **do**
    Let $\mathcal{F}_i \leftarrow \mathcal{F}_i \cup \{a\}$
    Let $\mathcal{G}_i \leftarrow \mathcal{G}_i \backslash \{a\}$
    Change the sign of the incident edge weights to node a
    Update $Gains$ of adjacent nodes
    Select the node a with largest $Gain$, $a = \max(Gain)$
  end while
  $\mathcal{U}_j \leftarrow \mathcal{U}_j \cup \mathcal{F}_i$
  $\mathcal{P}_j \leftarrow \mathcal{P}_j \backslash \mathcal{F}_i$
end for
return $\mathcal{U}_j$ and $\mathcal{P}_j$

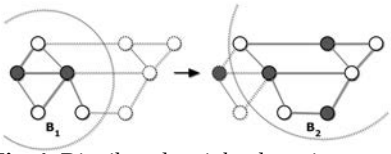
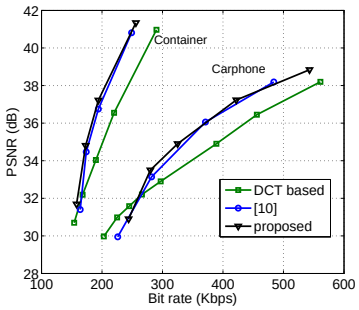


Fig. 4. Distributed *weighted maximum-cut*.

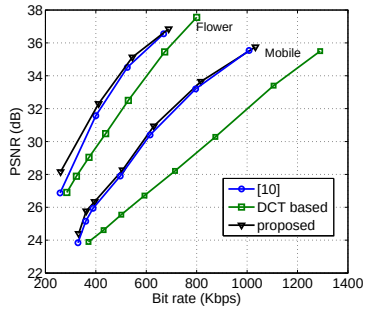
The experimental results for the complexity reduction (CR), calculated as the ratio of encoding times when coding 20 frames using the centralized and the distributed approaches ($CR = \frac{time_{cent}}{time_{dist}}$), show the efficiency of the proposed method. Using a block size in the algorithm of $B = 512$, we obtain $CR = 228$ in Carphone, $CR = 203$ in Mobile and $CR = 197$ in Container, keeping the cut of the graph and the number of $\mathcal{U}$ and $\mathcal{P}$ nodes selected very similar to those chosen in the centralized approach, and thus causing a negligible loss in performance.

5. EXPERIMENTAL RESULTS

To evaluate the coding performance of the proposed encoder, we compare it against the work in [10] and against a motion-compensated DCT video encoder in terms of Rate-Distortion values for different test sequences. The test conditions are similar to those used in [10]. The coefficients are quantized using a uniform dead-zone quantizer in the DCT, and a subband dependent quantization in [10] and in our proposed work. These quantized coefficients are scanned as is explained in [10], and in the tradicional zigzag scanning order in the DCT. Then, a run-length encoding is performed in the encoders, obtaining the symbols to be entropy coded. Finally, the bitstream is obtained coding the symbols using an adaptive arithmetic coder. Regarding the side information, our encoder will have an extra overhead because it should send the weight prediction values every frame. In the experiments, five levels of decomposition of the proposed transform are performed. Block sizes of 16×16 and one reference frame are assumed in the motion estimation process. In the DCT encoder, we use 8×8 DCT. The block size used in the low cost approach is set to $B = 512$.



(a) Container (QCIF) and Carphone (QCIF)



(b) Flower (QCIF) and Mobile (QCIF)

Fig. 5. PSNR versus Bit Rate.

Fig. 5 shows the Rate-Distortion curves for four different $QCIF$ sequences, $Mobile$, $Carphone$, $Flower$ and $Container$. In general, our proposed method outperforms our previous work and the DCT based approach. The gain is about 0.3-0.6 dB over [10] and about 2-4 dB over the DCT based approach in the test sequences. Besides, the complexity of the $\mathcal{U}/\mathcal{P}$ is greatly reduced.

6. CONCLUSIONS

We have developed a way to select the optimal spatio-temporal prediction weights in a video encoder based on lifting transforms on graphs. This is equivalent to choose the filtering directions as a function of the correlation between pixels, leading to a higher energy compaction. A low complexity approach of the $\mathcal{U}/\mathcal{P}$ assignment has been proposed. This provides improved performance over our previous work.

7. REFERENCES

[1] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P. L. Dragotti, "Directionlets: anisotropic multidirectional representation with separable filtering," *Image Processing, IEEE Transactions on*, vol. 15, no. 7, pp. 1916 –1933, July 2006.

[2] G. Shen and A. Ortega, "Compact image representation using wavelet lifting along arbitrary trees," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, October 2008, pp. 2808 –2811.

[3] E. Le Pennec and S. Mallat, "Sparse geometric image representations with bandelets," *Image Processing, IEEE Transactions on*, vol. 14, no. 4, pp. 423 –438, April 2005.

[4] Raanan Fattal, "Edge-avoiding wavelets and their applications," in *SIGGRAPH '09: ACM SIGGRAPH 2009 papers*, New York, NY, USA, 2009, pp. 1–10, ACM.

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

[6] G. Pau, C. Tillier, B. Pesquet-Popescu, and H. Heijmans, "Motion compensation and scalability in lifting-based video coding," *Signal Processing: Image Communication*, vol. 19, no. 7, pp. 577 – 600, 2004, Special Issue on Subband/Wavelet Interframe Video Coding.

[7] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," Tech. report 1995:6, Industrial Math. Initiative, Dept. of Math., University of South Carolina, 1995.

[8] E. Martínez-Enríquez and A. Ortega, "Lifting transforms on graphs for video coding," in *Data Compression Conference (DCC), 2011*, march 2011, pp. 73 –82.

[9] S. K. Narang and A. Ortega, "Lifting based wavelet transforms on graphs," in *APSIPA ASC 2009: Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference*, October 2009.

[10] E. Martínez-Enríquez, F. Díaz-de María, and A. Ortega, "Video encoder based on lifting transforms on graphs," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, sept. 2011, pp. 3509 –3512.

[11] N.V. Boulgouris, D. Tzovaras, and M.G. Strintzis, "Lossless image compression based on optimal prediction, adaptive lifting, and conditional arithmetic coding," *Image Processing, IEEE Transactions on*, vol. 10, no. 1, pp. 1 –14, jan 2001.

[12] A.T. Deever and S.S. Hemami, "Lossless image compression with projection-based and adaptive reversible integer wavelet transforms," *Image Processing, IEEE Transactions on*, vol. 12, no. 5, pp. 489 – 499, may 2003.

[13] J. Sole and P. Salembier, "Generalized lifting prediction optimization applied to lossless image compression," *Signal Processing Letters, IEEE*, vol. 14, no. 10, pp. 695 –698, oct. 2007.

[14] Gregoire Pau Christophe, Christophe Tillier, and Batrice Pesquet-popescu, "Optimization of the predict operator in lifting-based motion compensated temporal filtering," in *in Proc. of Visual Communications and Image Processing*, 2004.

[15] C.-P. Hsu, "Minimum-via topological routing," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 2, no. 4, pp. 235 – 246, 1983.

[16] G. Shen and A. Ortega, "Tree-based wavelets for image coding: Orthogonalization and tree selection," in *Picture Coding Symposium, 2009. PCS 2009*, May 2009, pp. 1 –4.

[17] B. Girod and S. Han, "Optimum update for motion-compensated lifting," *Signal Processing Letters, IEEE*, vol. 12, no. 2, pp. 150 – 153, feb. 2005.

[18] C. Tillier, B. Pesquet-Popescu, and M. van der Schaar, "Improved update operators for lifting-based motion-compensated temporal filtering," *Signal Processing Letters, IEEE*, vol. 12, no. 2, pp. 146 – 149, feb. 2005.