FACTA UNIVERSITATIS (NIŠ)
Series: Electronics and Energetics vol. 11, No.3 (1998), 321-343

SUBBAND CODING OF IMAGES: NOVEL APPROACHES AND SOME IMPROVEMENTS

Miodrag Popović and Dragutin Šević

Abstract. The novel approaches and some computational improvements of subband coding of images are described in this paper. Special attention is given to: 2×1–D and nonseparable 2–D wavelet decompositions, computationally improved 2–D ELT and fuzzy postprocessing of subband coded reconstructions. A common characteristic of presented case studies is that the problem of image coding is conceptually treated as a two–dimensional one, although separability is largely exploited for efficiency purposes. Some experimental results of simulations of subband coding of images using different algorithms are also presented.

1. Introduction

The basic idea of subband coding (SBC) of images is to split up the frequency band of the signal, and then to code each subband separately using a coder and bit rate appropriate to that subband [41]. Two main criteria for allocating bits among the subbands are the energy they carry and the different sensitivity of the human eye to different spatial frequencies.

A comparative study of various subband coding algorithms could be found in our earlier paper [25]. In this paper, our intention is to describe some of the newer approaches and computational improvements in subband coding of images. Special attention is given to: nonseparable 2–D and 2×1–D wavelet decompositions, computationally improved 2–D ELT and fuzzy postprocessing of subband coded reconstructions. A common characteristic of presented case studies is that the problem of image coding is conceptually treated as a two–dimensional one, although separability is largely exploited for efficiency

Manuscript received April 5, 1998.

A version of this paper was presented at the third Conference Telecommunications in Modern Satellite and Cables Services, TELSIKS'97, Niš, Yugoslavia.

Prof. Dr M. Popović is with Faculty of Electrical Engineering, University of Belgrade, P.O. Box 35-54, 11120 Belgrade, Yugoslavia, E-mail: pop@el.etf.bg.ac.yu. Dr. D. Šević is with Institute of Physics, P.O. Box 68, 11080 Zemun, Belgrade, Yugoslavia, E-mail: sevic@phy.bg.ac.yu.

purposes. Some experimental results for simulations of subband coding of images using different algorithms are also presented. Our aim was to code pictures with low bit rates, between 0.2 to 1.0 bits per pixel (bpp) and to compare objective and subjective results, as well as arithmetic complexity of algorithms. The test picture "Lena" 512×512, with 256 gray levels, is used for presentation of results. Computer simulations were conducted also using other pictures, however, no important differences in results have been found, except in the case of the fuzzy postprocessing of reconstructed images, where experimental results for "Lena", "Peppers" and "Awl" images are presented.

2. Coding of image subbands using vector quantization

A fundamental result of Shannon's rate–distortion theory is that better performance can always be achieved by coding vectors instead of scalars, even if the data source is memoryless. Excellent overviews and tutorials on vector quantization can be found in [10,9]. To exploit orientation of local image content, *orientation adaptive vector quantization* (OAVQ) [32,24,25] is used. The basic idea of orientation adaptive VQ is to use vectors with varying shapes, for example: for vector length of 2, we use blocks with following shapes: 1×2, 2×1; for vector length of 4, blocks have shapes: 2×2, 1×4 and 4×1; and for vector length of 16, blocks have following shapes: 4×4, 2×8, 8×2, 1×16 and 16×1. The shapes are selected depending on Lagrange functional. By varying Lagrange multiplier it is possible to constrain coder with respect to entropy or distortion.

Detailed explanation of lattice and pyramidal lattice vector quantization algorithms can be found in [4,5,6,8]. LVQ algorithms are in theory suboptimal compared to the LBG algorithm [16], however, algorithms for lattice VQ have much more efficient computer implementations. The existence of code table is not necessary condition for the realization of LVQ. Moreover, operation of coder is not dependent on image contents, so coder is not "trained" on some specific source distribution. Because distortion measures are not evaluated for determining of codewords, coding process is much faster when using LVQ compared to the LBG VQ. As a practical consequence of this, it is possible, by varying scaling parameters, to achieve better results with LVQ than with LBG, keeping computing time still lower for LVQ.

In [2,1] the statistics of wavelet coefficient's subimages were modelled by the generalized Gaussian. Laplacian approximation of Gaussian function leads to the use of pyramidal lattice vector quantization for all subbands, except the lowest one. The coding of subbands in our simulations is realized as follows: the lowest subband is lattice vector quantized [4] using D_2 lattice, and the other subbands are coded using pyramidal lattice vector quantization [8,2], using D_2 or D_4 lattice. In order to optimize the trade–off between minimum distortion and bit rate using a lattice quantizer, the lattice has to be truncated and scaled. However, instead of scaling the lattice codebook, it is much better to scale the source vector. We used the encoding algorithm described in [6].

Our encoding strategy is to define scaling factor in some range, and encoding is done successively by gradual increasing of scaling factor. Because lattice VQ and pyramidal lattice VQ algorithms are computationally very efficient, overall timings of lattice and pyramidal lattice VQ are still much shorter than timings of LBG algorithm for similar performances.

The shapes and the scaling factors of lattice quantization are selected depending on Lagrange functional. By varying Lagrange multiplier, it is possible to constrain coder with respect to entropy or distortion. Subbands are coded in blocks of dimension 32 × 32, so the scaling factor and the orientation of vectors are determined for each of these blocks.

The strategy of lattice VQ allows realization of automatic bit allocation, adjustable in a very wide range. Automatic bit allocation among the 32 × 32 blocks is achieved by combination of Lagrange multiplier method and by varying scaling factors for LVQ. Note that 32 × 32 blocks are smaller than subbands, so "busy" and "quiet" blocks are also distinguished in subbands.

For each of 32×32 blocks *cost* is determined and minimized, by varying scaling factor of LVQ and orientation of vectors:

$$cost = D + L_m \cdot E \tag{1}$$

where D is distortion, L_m is Lagrange multiplier, and E is signal entropy. Distortion and entropy are functions of scaling factor of LVQ and vector orientation. Large values of L_m constrain coder entropy, and small ones constrain distortion. Depending on target bit rate, the specific value of L_m is adopted.

3. Image coding using Wavelet transform

For detailed and excellent explanation of image coding using wavelet transform the reader is referred to [1]. For even more studious reading, see [38]. Our simulations were performed using all three filter families given in [1]. Based on our experimental results, for all types of wavelet decompositions, the simulations using filter denoted as no. 2 9-7 were chosen as the best. For nonseparable DWT, the same filter pair is extended to 2–D case using McClellan transformation.

Various extensions of the 1–D wavelet transform to higher dimensions exist. Applying two 1–D wavelet transforms separately is the most often used way to generate 2–D wavelet transform. This produces a multiresolution scale factor of 2 and privileges horizontal and vertical orientations. Such a multiresolution analysis provides, at each resolution level, three wavelet coefficient sub–images having different orientations and one low–resolution sub–image. One stage of filter bank for realization of separable 2–D DWT is shown in Fig. 1. This, pyramid-structured wavelet decomposition of 2–D signals, proposed by Mallat [17], was followed by many authors. Pyramid–structured wavelet representation is schematically presented in Fig. 2.a.

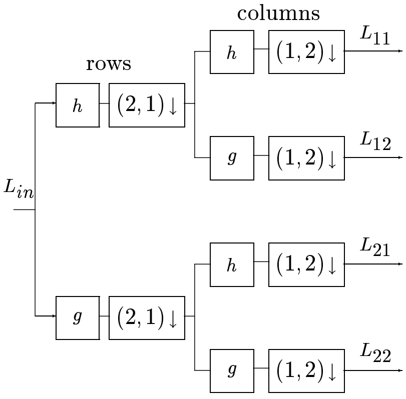

Figure 1. One stage of filter bank for realization of separable 2–D DWT.

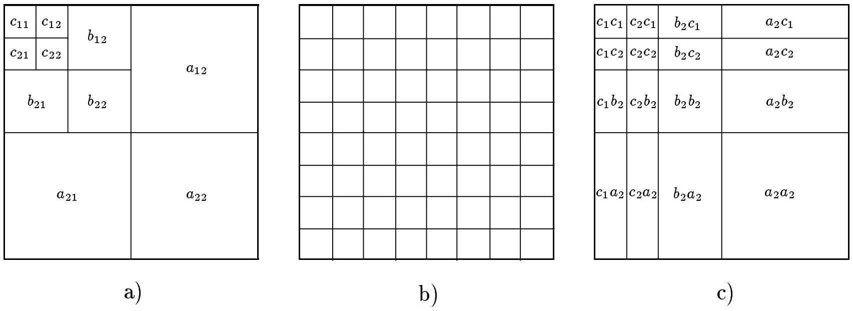
c_{11} c_{12} c_{21} c_{22} b_{12} b_{21} b_{22} a_{12} a_{21} a_{22}
c_1c_1 c_2c_1 b_2c_1 a_2c_1
c_1c_2 c_2c_2 b_2c_2 a_2c_2
c_1b_2 c_2b_2 b_2b_2 a_2b_2
c_1a_2 c_2a_2 b_2a_2 a_2a_2
a)
b)
c)

Figure 2. Three stages of wavelet decompositions.
a) Pyramid–structured wavelet representation.
b) Tree–structured wavelet representation (uniform decomposition).
c) 2×1–D wavelet representation.

3.1 Image coding using tree–structured filter bank based on wavelet prototype

Another type of wavelet transform, suitable for texture analysis, was proposed in [3]. To overcome limitations of hierarchical transforms (based on 2–band splitting) in image coding applications, we also used full tree structured wavelet transform]3], decomposing the image in the same manner as with the Lapped transforms. An example of this tree–structured wavelet transform representation is shown in Fig. 2.b. Decomposition is uniform, because we used full tree structure. We used the same wavelet prototype as for the 2–band splitting [1]. This kind of decomposition is obviously computationally less efficient than decompositions with usual hierarchical wavelet transform. Moreover, it is also less efficient than decompositions with lapped transforms. Therefore, it is included only for the sake of comparison.

3.2 Image coding using discrete Wavelet transform with direction dependent resolution

The examples of wavelet decompositions, shown in Fig. 2.a. and b., could be viewed as special cases of a general approach to wavelet decomposition, known as *Wavelet packets* (WP) [28]. The common feature of all these algorithms is that the separability is exploited at each stage of decomposition, so each stage of decomposition includes both row and column transforms. Because of alternating rows/columns processing, it is not possible to realize any kind of merging of consecutive stages of 1–D transforms. In [21,26] we proposed another type of 2–D representation, suboptimal compared to WP, interesting because of the efficiency of its realization.

The basic idea of the proposed algorithm is very simple: instead of separable rows/columns computation at each stage, all octaves in each dimension are computed at once, so complete 2–D wavelet transform is implemented as separable computation of two complete 1–D transforms (actually one repeatedly applied on all N rows, and another on all N columns). Because of that, the new algorithm will be denoted 2×1–D DWT. 2×1–D discrete wavelet representation is shown in Fig. 2.c. Number of octaves for this example is $J = 3$. The evaluation of complete 1–D DWT is not the only way to achieve decomposition shown in Fig. 2.c, although it is most efficient way to do that. The decomposition shown in Fig. 2.c. can be obtained following decomposition from Fig. 2.a. with only row (column) decompositions of certain subbands. Perfect reconstruction property of this type of transform is not hard to prove. It is easy to see in Fig. 2.c. that horizontal and vertical resolution for all subbands are not the same. Hence, this scheme enables more flexible bit allocation in image coding applications compared to the wavelet representation in Fig. 2.a. For example, most of information about horizontal and vertical edges is packed in border subbands a_2c_1 and c_1a_2.

Because length-$N/2$, length-$N/4$, etc., DWT's are applied (only once) to all rows and columns, the computational complexity of the representation in the Fig. 2.c. is higher than for the representation in Fig 2.a. However, merging of successive octaves is easy to implement in this type of analysis/synthesis, so it is possible to use various improvements for computation of 1–DWT, proposed in [22,37,29]. Merging of successive 1–D octaves is not possible to achieve for decomposition shown in Fig. 2.a. If decomposition in Fig. 2.b. is obtained by recursive application of 2–band filter bank, decomposition shown in Fig. 2.c. has lower computational complexity.

Compared to the decomposition strategy using wavelet packets [28], the possibility of different resolutions in different directions is included in the new algorithm. Furthermore, the new algorithm offers simpler (because it is nonadaptive) realization, so good compromise between simplicity and performance is achieved. Yet, primarily because for coding of subbands, in our simulations we used pyramidal lattice vector quantization [2], better experimental results than in [28] were obtained.

It should be emphasized that the concept of different resolutions in different directions, obtained by only 1–D decomposition of certain subbands (row decomposition without column decomposition, or vice versa), although so far not used, could be also implemented in 2–D wavelet packets decomposition. In this way, wavelet packet tree could be better adapted to the specific orientation of image. However, this doesn't necessarily mean the further increase of the computational complexity. Namely, further decomposition of rows of a subband could be viewed as a saving on column decomposition.

3.3 Image coding using nonseparable Wavelet transform

In our investigation, we also included the decomposition of image by using two–dimensional nonseparable Wavelet transform [33]. The subband signals are subsampled using so–called "quincunx" lattice. This algorithm uses nonseparable and nonoriented filters. It decomposes the original image with a multiresolution scale factor of $\sqrt{2}$. One stage of filter bank for realization of nonseparable 2–D DWT is shown in Fig. 3. The quincunx lattice is shown in Fig. 4. Note that the quincunx lattice in wavelet theory is equivalent to D_2 lattice in the information theory.

Our approach is very similar to the realization proposed in [2], with the main differences being: the number of octaves in our realization of discrete wavelet transform is six (two more than in [2], instead of DPCM we used lattice VQ for lowest subband, and finally, our vector quantization algorithm is orientation adaptive.

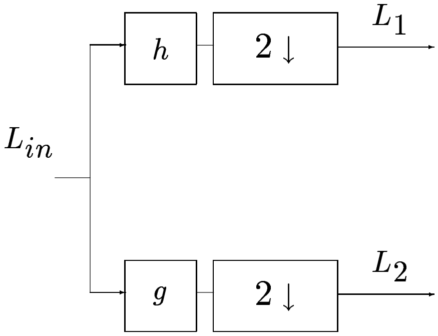

Figure 3. One stage of filter bank for realization of nonseparable 2–D DWT.

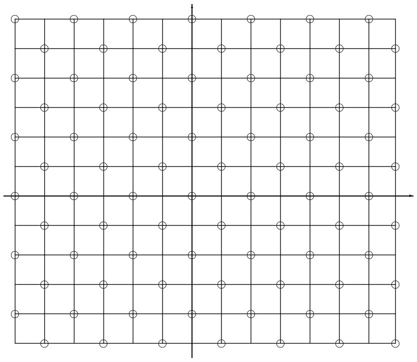

Figure 4. Quincunx lattice.

Nonseparable DWT decomposition of "Lena" image using quincunx subsampling is shown in Fig 5. Odd numbered levels of decomposition are, because of scale factor of $\sqrt{2}$, presented as rotated by 45 degrees.

In order to achieve compact use and access to memory, subbands corresponding to odd numbered levels of decomposition are packed with odd numbered rows shifted. "Packed" subband decomposition is shown in Fig. 6. It should be noted that this method of storing wavelet coefficients, because of different indexing, requires different subroutines for odd and even numbered levels of decomposition.

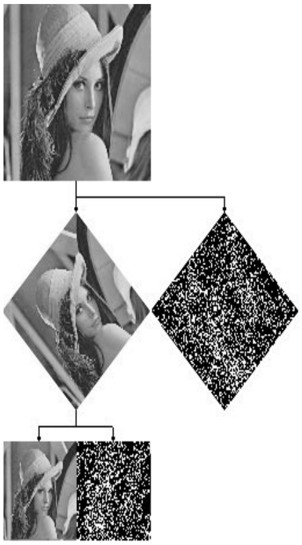

Figure 5. Nonseparable DWT decomposition of "Lena" image using quincunx subsampling.

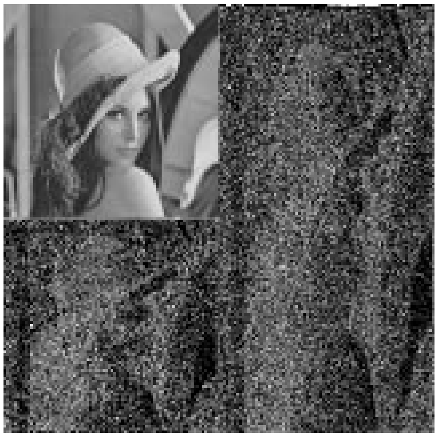

Figure 6. Nonseparable DWT decomposition of "Lena" image "packed" for compact use of memory.

Simulations based on nonseparable DWT were performed using filters given in Table II in [2]. Those filters are derived from filters no. 2 9-7 [1] applying the transformation:

$$\cos w \rightarrow \frac{1}{2}(\cos \omega_x + \cos \omega_y) \tag{2}$$

to the 1–D trigonometric polynomials. This transformation ensures that all properties of the 1–D filters are satisfied by the 2–D filters. Note that 2–D filters, derived in this way, are nonoriented, i.e. their filtering characteristics are not direction dependent. This is the main distinction between nonseparable DWT decomposition used in this paper for image coding simulations and the other separable decompositions presented here.

4. Image coding using 2–D Extended Lapped transforms

The Extended lapped transforms (ELT) [19,20] are generalization of the MLT with arbitrarily long basis functions. They offer better subband separation than MLT.

For the sake of simplicity and to achieve computational savings, 2–D transforms are often implemented as separable operators, in two steps: first, all rows in a block are transformed with a 1–D transform, and then all columns in the transformed block are transformed using the same 1–D transform (or vice versa, result is the same). In [34] we proposed a further computational optimization of 2–D separable extended lapped transform (ELT), based on fully optimized 1–D ELT, proposed by Malvar [20]. The new fast algorithms for the ELT are based on the use of FFT algorithm for computation of the DCT–IV operator, firstly proposed by Duhamel *et al.* in [7]. Duhamel's algorithm includes input and output rotations, with butterfly matrices very similar to window butterfly matrices of ELT's.

Also, contrary to results presented in [19], results of our numerous image coding simulations fulfilled the theoretical expectations: we have obtained improvement in coding results when overlapping factor K of ELT is increased, or when ELT with $K > 1$ instead of MLT or LOT [18] is used. However, for coding simulations, instead of 256 × 256 images as in [19], we used 512 × 512 images. To avoid border effects, we have used the periodic extension of image.

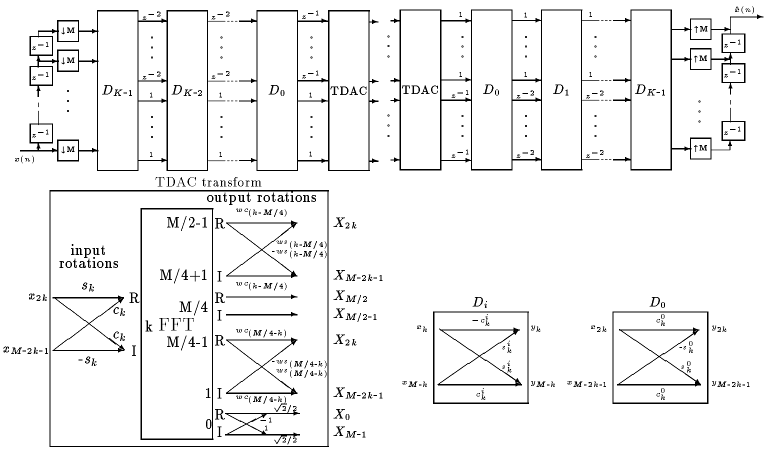

Figure 7. The structure of the fast implementation of the ELT analysis/synthesis filter bank, and the elements of ELT structure: TDAC transform and butterflies. $s_k = \sin\theta_k$, $c_k = \cos\theta_k$, $\theta_k = (4k+1)\pi/(4M)$, $ws_k = \sin(k\pi/M)$, $wc_k = \cos(k\pi/M)$. *Note that the butterflies* D_i*'s and* D_0 *are different.*

4.1 The FFT based implementation of the ELT

The Extended Lapped Transform, first proposed in [20], and further explained in [19], is usually implemented using the DCT–IV transform, although the FFT is used for the efficient computation of DCT–IV [19]. Our implementation of ELT is also based on the use of FFT but it is derived in the following way: using the approach of Duhamel *et al.* [7], one can suppose that the window has been applied to the signal, and concentrate to the central part of ELT, the transform itself. Since this transform was named by Duhamel as the "TDAC transform", the same name will be used in this paper. Malvar's approach in [20] and [19] was different: he has concentrated on the efficient computation of the DCT–IV operator. Since the derivation of the TDAC transform is the straight repetition of the work performed in [7] and [31], it will not be presented here.

The implementation of TDAC transform is shown by a flowgraph in Fig. 7. It should be emphasized that the output rotation by angle $\theta = 0$ uses no real operations, and the rotation by $\theta = \pi/4$ uses only 2 real multiplications and 2 real additions. The FFT is optimally implemented using split-radix algorithm. For the number of bands $M \leq 16$, the FFT could be optimally implemented with a single stage radix-2, 4, or 8 algorithms, which means no indexing. Duhamel *et al.* [7] showed that the TDAC is self–inverse transform, so there is no need to derive the inverse TDAC transform.

The structures of the ELT analysis filter bank and the ELT synthesis filter bank are shown in Fig. 7. Because of the orthogonality of ELT, the synthesis filter bank is the transpose of the analysis filter bank.

Using the angle values from Table D.3 [19], this filter bank has the same frequency response as ELT from [19] or [20] (disregarding some irrelevant channel multiplications by -1). The angle values θ_k^i can be read directly from Table D.3 in [19]. However, the values for θ_k^0 should be obtained by following relations:

$$\theta_k^0 = \hat{\theta}_{2k}^0, \qquad k = 0, 1, ..., M/4 - 1 \tag{4}$$

$$\theta_k^0 = \pi/2 - \hat{\theta}_{M-2k-1}^0, \qquad k = M/4, ..., M/2 - 1 \tag{5}$$

where $\hat{\theta}_k^0$ are the angles from Table D.3 [19]. This permutation is essential to achieve necessary reordering of elements for TDAC transform. On the other hand, in programs for ELT proposed by Malvar [19], data unshuffling steps were moved outside of recursive modules. In our approach, however, it is easier to recognize and to use full possibilities of scaling and inverse scaling of butterfly matrices.

4.2 Scaling of butterfly matrices

As proposed by Malvar [19], all the coefficients in the cascade of window butterflies could be scaled, so that diagonal entries would be equal to 1 or -1, and necessary inverse scaling would be applied to the last butterfly in cascade (D_0). Computational complexities of ELT's in Table D.3 [19] correspond to this way of scaling butterflies. However, looking at the Fig. 7, it is easily perceived that the inverse scaling could be applied to the input rotations of FFT based DCT–IV realization, for all possible numbers of bands, M. This first step in optimization procedure, saves one multiplication per sample. If rotations are realized using 3 real multiplications and 3 real additions (denoted by 3/3), then saving is equal to 0.5 multiplications and 0.5 additions per sample. It should be noted here that there is a subtle computational difference between the MLT [18], which uses the sine window, and ELT with overlapping factor $K = 1$, which uses butterfly angles given in [19]. Because of similar frequency responses, MLT is usually considered equivalent to ELT with $K = 1$.

An efficient MLT implementation was proposed by Duhamel *et al.* in [7], and synthesis filter bank algorithm for this MLT implementation was completed by Šević and Popović [31]. If butterflies in MLT are merged, as proposed by Duhamel *et al.* [7], this part of MLT algorithm requires 2 multiplications and 3 additions per sample. However, this kind of merging is not possible for ELT with $K = 1$, where computational savings are achieved by scaling and inverse scaling of butterfly coefficients, so this part of ELT algorithm requires 3 multiplications and 2 additions per sample (if rotations are realized as 3/3, this part of ELT algorithm requires 2.5 multiplications and 2.5 additions per sample).

4.3 2–D extension of ELT

The basic structure of the 2–D separable ELT analysis filter bank, as proposed by Malvar [19], and based on the use of FFT for DCT–IV realization, is shown in Fig. 8. In the simplest form of implementation, whole rows or columns are fetched from image matrix, and, after processing with 1–D ELT's, returned to matrix. Inverse scaling is applied to input rotations. Although row/column calculations are easy to implement using **for** loops, it is not easy to correctly represent them in the flowgraph.

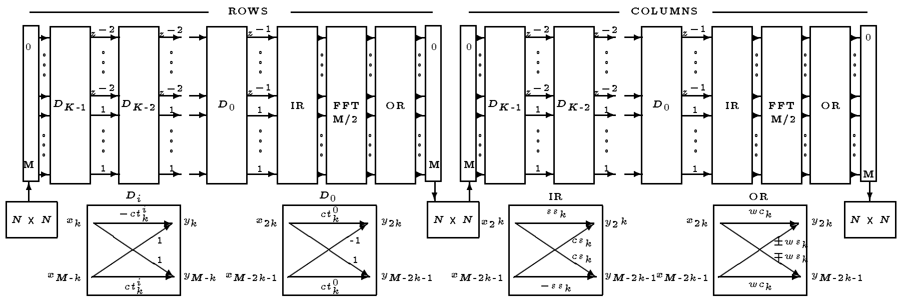

Figure 8. The basic structure of the 2–D separable ELT analysis filter bank. $N \times N$ denotes image matrix, IR and OR are input and output rotations, respectively.

It is possible to reorder some of row and column computations, which are independent, without affecting the filter bank output. The reordered flowgraph, shown in Fig. 9 resembles the "true" 2–D implementation. 2–D input window is computed first (row/column), after that 2–D input rotations (column/row, to save on number of row/column fetchings), after that 2–D FFT (row/column), and finally, 2-D output rotations (column/row). To save on the number of accesses to matrix, it is possible to reorder input window and input rotations row/column computations, as shown in flowgraph in Fig. 10. Implementations shown in Figs. 9 and 10 are equivalent.

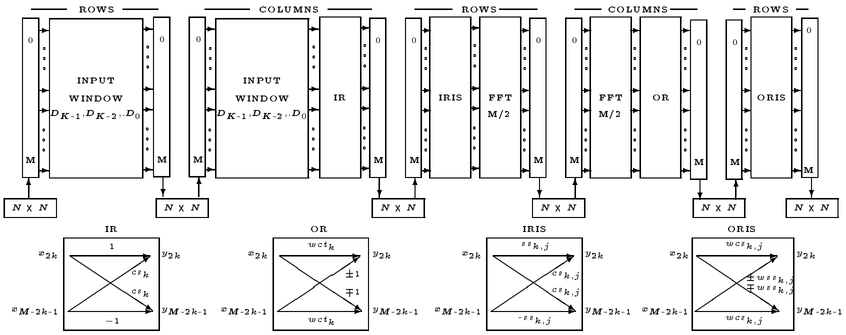

Figure 9. The structure from Fig. 8 with reordered rows/columns computation. The cascade of butterflies and delays is substituted by the INPUT WINDOW block. D_i are D_0 are the same as in Fig. 8. IR and OR - scaled input and output rotations. IRIS and ORIS - inverse scaled input and output rotations.

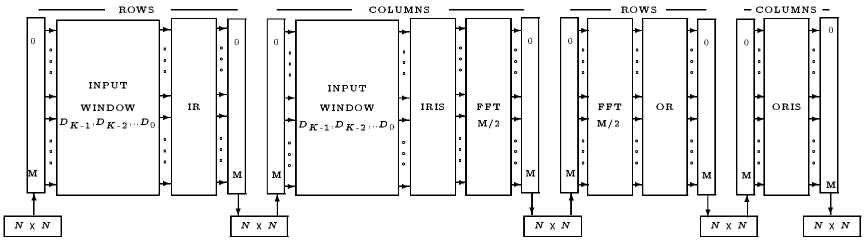

Figure 10. The structure from Fig. 9 with reordered input window and input rotations rows/columns computation. Butterflies are same as in Fig. 9.

Reordering, shown in Figs. 9 and 10, makes possible computational savings based on scaling of all butterflies (left or right from FFT computations), and inverse scaling of the last one in the cascade, as shown in Figs. 8 and 9. However, inverse scaling in this last butterfly should compensate for scaling performed both in row and column computations. Inverse scaling in the same dimension is well explained for the 1–D case [19]. Inverse scaling for scaling performed in another dimension is easy to accomplish by using the following rule: if k'th element in row computation is scaled by factor $S(k)$, then subsequent computations in k'th column are to be inverse scaled by the factor $S(k)$. Because of that, inverse scaled butterfly coefficients in Figs. 9 and 10 have both row and column indices, k and j. After scaling and inverse scaling, row and column calculations before and after FFT computation are not independent any more, so it is not possible to put scaled operators back in order of Fig. 8. It should be stressed that the new implementations shown in Figs. 9 and 10 have the same frequency response as implementation in [19].

The numbers of operations per sample for the new implementation and for the implementation based on usual rows/columns computations, using 1–D ELT [19] are given in Table 1. (rotations are counted as 3/3). Number of operations per sample for 2–D ELT proposed in [19] are determined from Table 5.1 [19] by multiplying items by 2 (to account for row/column calculations) and dividing by M, number of input/output elements. LT denotes the size of look–up table necessary for realization of fast algorithm. For rotations counted as 4/2, the size of look–up table would be about 30% less. Savings (Mul + Add) per sample are dependent on the number of bands M and vary between (0.5 + 0.5) for $M = 2$ and (1.9375 + 1.9375) for $M = 32$.

The increase of size of the look–up table for butterfly coefficients is the price to be paid for implementation of the new algorithm. However, memory requirements for look–up table are still negligible compared to memory requirements for whole image processing. Namely, memory requirements for look–up table for original 2–D ELT algorithm [19] are roughly proportional to $M/2$. In the new algorithm, memory requirements for scaled butterflies are roughly proportional to $M/2$, for inverse scaled input rotation butterflies proportional to $(M/2)^2$, and for inverse scaled output butterflies proportional to $(M/4)^2$, because inverse scaling should account both for row and column indices. However, in all practical cases, the number of bands M should be small compared to the number of pixels in a row (or column) of the image, N.

In the simplest form of implementation, whole rows and columns are fetched from and retrieved to image matrix. However, looking at Figs. 8 and 9, it is seen that the new algorithm requires more frequent accesses to image matrix. If realized in this way, the new implementation, with reduced number of real operations, and increased number of real data transfers, wouldn't be much faster compared to the implementation from Fig. 8 on most computers. To take advantage of reduced computational complexity of the new implementation, matrix elements should be directly accessed and processed, using pointers.

Because of orthogonality of the ELT, the synthesis filter bank is the transpose of the analysis filter bank, and has the same number of operations as the analysis filter bank.

Table 1. Computational complexity of old [24], and new implementations of the 2–D ELT. M – number of bands in one dimension, K - overlapping factor, LT -size of look-up table.

		$K = 1$			$K = 2$		
	M	Mul/s	Add/s	LT	Mul/s	Add/s	LT
[19]	2	5.0000	5.0000	4	7.0000	7.0000	5
[19]	4	7.0000	9.0000	8	9.0000	11.0000	10
[19]	8	8.0000	12.0000	18	10.0000	14.0000	22
[19]	16	9.0000	15.0000	44	11.0000	17.0000	52
[19]	32	10.0000	18.0000	90	12.0000	20.0000	106
new	2	4.5000	4.5000	4	6.5000	6.5000	5
new	4	5.5000	7.5000	10	7.5000	9.5000	12
new	8	6.2500	10.2500	40	8.2500	12.2500	44
new	16	7.1250	13.1250	163	9.1250	15.1250	171
new	32	8.0625	16.0625	631	10.0625	18.0625	647

		$K = 3$			$K = 4$		
	M	Mul/s	Add/s	LT	Mul/s	Add/s	LT
[19]	2	9.0000	9.0000	6	11.0000	11.0000	7
[19]	4	11.0000	13.0000	12	13.0000	15.0000	14
[19]	8	12.0000	16.0000	26	14.0000	18.0000	30
[19]	16	13.0000	19.0000	60	15.0000	21.0000	68
[19]	32	14.0000	22.0000	122	16.0000	24.0000	138
new	2	8.5000	8.5000	6	10.5000	10.5000	7
new	4	9.5000	11.5000	14	11.5000	13.5000	16
new	8	10.2500	14.2500	48	12.2500	16.2500	52
new	16	11.1250	17.1250	179	13.1250	19.1250	187
new	32	12.0625	20.0625	663	14.0625	22.0625	679

5. Comparisons

The test picture "Lena" 512 × 512, with 256 gray levels, is used for presentation of results, see Fig. 11.

Computer simulations were conducted also using other pictures, however, no important differences in results have been found.

Nonseparable decompositions are computationally more expensive than separable ones. Average number of real operations per stage (multiplications and additions) for the separable 2–D DWT implementation using 1-D DWT (with 9-7 filter pair) per pixel is $M+A = 2\times[(9+7)/2+(8+6)/2] = 16+14$. Multiplying by 2 takes into account row/column calculations. Dividing by 2 averages between 9-tap and 7-tap filters. For nonseparable DWT, using filter pair derived from 9-7 filters, average number of real operations per pixel is $M + A = (9+7)/2 + (40+24)/2 = 8 + 32$.

Figure 11. 512 × 512 "Lena" original.

To compare results with the M-band decompositions, results obtained with 2–D ELT are also included. The length of input block is $2KM$, where M is number of subbands. We used ELT with $K = 4$ and $M = 8$. Values of butterfly angles (parameter θ_k) for window rotations are taken from table D.3 [19]. This kind of 2–D ELT decomposition of image, realized as proposed in [34] requires (14+18) mults+adds per pixel.

The objective results of simulations using 2–D separable and nonseparable DWT and pyramidal lattice VQ are shown in Fig. 12. One of the reconstructed images, subjectively very similar to the original, is shown in Fig. 13.

Nonseparable DWT, used with the well developed coding techniques for separable decompositions, achieves slightly inferior results compared to the separable decompositions. As predicted in [23,27] M-band filter banks (2–D ELT and full tree–structured wavelet decomposition) have advantages compared to the hierarchical (DWT) transforms. It is interesting to note that computationally more efficient 2–D ELT has better results than full tree–structured wavelet decomposition.

Only the best results are presented in Fig. 12. Our other results show that orientation adaptive VQ has small advantages when used with nonoriented nonseparable DWT, so it seems that nonoriented filters are more disadvantage than advantage in image coding.

Our results also suggest why the simulations using nonseparable DWT are so rarely published, or why even the authors having large theoretical contribution to the field of the nonseparable wavelet transform [38,13] present only simulation examples obtained by using modulated filter banks.

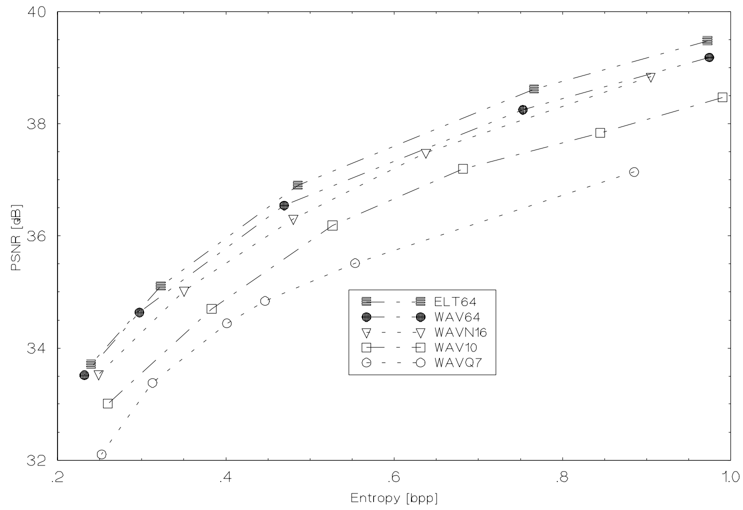

Figure 12. PSNR versus bit rate for simulations using 2–D DWT and ELT. WAVQ7 denotes quincunx DWT with 7 subbands, WAV10 denotes separable DWT with 10 subbands, WAVN16 denotes 2×1–D DWT with 16 subbands, ELT64 denotes 2–D ELT with 64 subbands. The size of the lowest subband is the same for all decompositions.

It is interesting to compare our results with other results of DWT coders recently published in literature for the same test image "Lena". For example, some results for the resolution 512×512 are:
$R = 0.37$ bpp, PSNR $= 30.85$ dB, [1]
$R = 0.174$ bpp, PSNR $= 30.3$ dB, [2]

Our simulations also imply that results in [2] are better than in [1] because in [2] pyramidal lattice VQ instead of LBG VQ is used for coding of subbands, and not because of nonseparable wavelet decomposition. In these examples the visual criterions are satisfied. However, the printed reconstructions in [2,1] have large white areas on parts where distortions are usually present, probably masking them. Our results are slightly better than results in [2,1] mainly because we used wavelet decompositions with larger number of octaves, both in separable and nonseparable realizations.

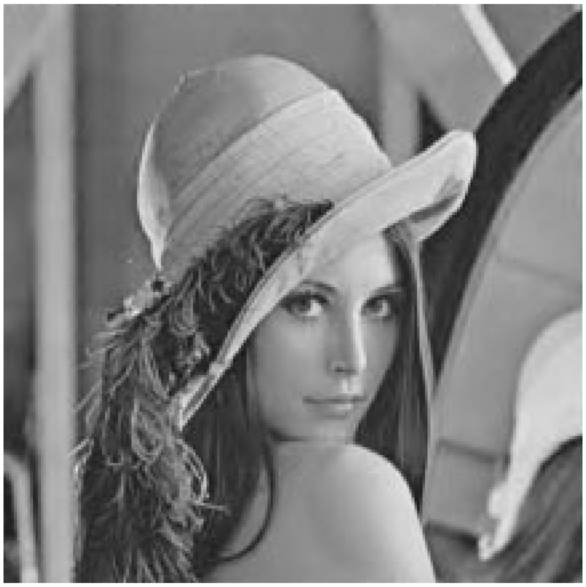

Figure 13. The reconstructed "Lena" image, coded with 0.35 bpp using 2×1–D DWT.

6. A new postprocessing technique based on fuzzy median filter

Various modifications of the basic median filter are proposed recently [39,15,36,30]. Novel postprocessing techniques for block effect elimination in DCT based coding systems are also proposed recently [12,11].

Although all currently used subband coding schemes use perfect reconstruction filter banks, the quantization of image subbands results in non–perfect reconstruction. Very detailed analysis of errors caused by quantization effects could be found in [40,14].

In [35] we proposed a new postprocessing technique based on fuzzy median filter to improve quality of reconstruction of subband coded images. Operation of proposed median filter is controlled by fuzzy rules which are adapted to purpose of elimination of certain types of errors caused by non–perfect reconstruction of subband coded image.

We consider the median value of some of the elements (possibly all), which satisfy certain condition, inside a 7 × 7 square window centered at the (y, x)th pixel. In order to remove errors caused by non–perfect reconstruction, the median filter uses following fuzzy rules:

1) IF $(L(y)(x) - L(y+i)(x+j)) < \alpha$, $i, j \in (-3, -2, -1, 1, 2, 3)$,
THEN $c = c + L(y+i)(x+j)$, $d = d + 1$.
2) IF $(L(y)(x) - c/d) > \beta$, THEN $L_s(y)(x) = c/d$, ELSE $L_s(y)(x) = L(y)(x)$.
where:
$L(y)(x)$ - luminance of the (y, x)th pixel,
$L_s(y)(x)$ - output of the fuzzy median filter for the (y, x)th pixel,
c - the median value of some of the elements obeying the fuzzy rule 1),
d - counter,
α, β - parameters of the fuzzy median filter.

For each calculation of $L_s(y)(x)$, c and d are initialized as $L(y)(x)$ and 0, respectively. Parameter α obviously takes only integer values, while β is not restricted to integer values.

For best results, parameters α and β should be adapted to image contents. However, as a rule of thumb, for images with 255 grey levels, very good results are obtained by choosing $\alpha = 15$, $\beta = 1$. Because the filter needs not to be adaptive, its realization is simple and computationally efficient.

For simulation experiments we used several 512 × 512 pictures with 255 grey levels. The most interesting experimental results, corresponding to images "Lena", "Peppers" and "Awl", are summarized in [35]. Objective improvements are the worst for the "Awl", which subjectively needs no postprocessing. Subjective improvements are significant for the other pictures. Very good results are obtained even with the nonadaptive version of the filter, so its realization is simple and computationally efficient. Signal to noise improvement is, for used test images, between 0.059 and 0.576 dB.

7. Conclusion

In this paper, some novel approaches and improvements in subband coding of images are presented. A new separable extension of one–dimensional wavelet transform to two–dimensional case is described. Compared to the previous proposals, where resolutions for each subband were the same in horizontal and vertical directions, the new extension is characterized by different resolutions in different directions. Our proposal enables more flexible bit allocation in image coding applications and easy merging of octaves, resulting in improved efficiency of computation of 2–D wavelet coefficients.

The proposed solution can be straightforwardly extended to higher dimensional cases. The wavelet packets concept could be also generalized with different resolutions for different directions.

A new implementation of the 2–D Extended Lapped Transform is also described. Compared to the separable solution [19], the new realization of 2–D ELT has reduced arithmetic complexity. Computational savings are achieved because scaling and inverse scaling of butterfly matrices, suggested by Malvar for 1–D case, are, after some modifications of the basic algorithm, extended to 2–D case. The new implementation has the same frequency response as Malvar's.

The experimental results obtained by simulations of subband coding of images using nonseparable and separable DWT algorithms and 2–D ELT's were presented. The reconstructed images were compared, showing that for well developed coding algorithms usually used in coding simulations, nonseparable DWT has no advantages. One of immediate directions for further research is investigation of coding techniques better suited for utilizing indisputable advantages of nonseparable decomposition. Another direction of further research is connected with the fact that better filtering characteristics of nonseparable DWT could be utilized for various kinds of image analysis. Also, supported is the conclusion that for subband image coding M-band filter banks are better suited than hierarchical 2–band filter banks. From this point of view, the improved algorithm for computation of 2–D ELT is important contribution to the subband coding of images.

In this paper we also describe a new postprocessing technique based on fuzzy median filter to improve quality of reconstruction of subband coded images. Very good results are obtained even with the nonadaptive version of the filter, so its realization is simple and computationally efficient.

Acknowledgment

This work was supported by the Ministry of Science and Technology of Republic of Serbia.

REFERENCES

1. M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies: *Image coding using wavelet transform.* IEEE Trans. Image Processing, vol. 1, no. 2, pp. 205–220, April 1992.

2. M. Barlaud, P. Sole, T. Gaidon, M. Antonini, and P. Mathieu: *Pyramidal lattice vector quantization for multiscale image coding.* IEEE Trans. Image Processing, vol. 3, no. 4, pp. 367–381, July 1994.

3. T. Chang and C.-C. J. Kuo: *Texture analysis and clasification with tree-structured wavelet transform.* IEEE Trans. Image Processing, vol. 2, no. 4, pp. 429–441, Oct. 1993.

4. J.H. Conway and N.J.A. Sloane: *Voronoi regions of lattices, second moments of polytopes, and quantization.* IEEE Trans. Inform. Theory, vol. IT–28, no. 2, pp. 211–226, Mar. 1982.

5. J.H. Conway and N.J.A. Sloane: *Fast quantizing and decoding algorithms for lattice quantizers and codes.* IEEE Trans. Inform. Theory, vol. IT–28, no. 2, pp. 227–232, Mar. 1982.

6. J.H. Conway and N.J.A. Sloane: *A fast encoding method for lattice codes and quantizers.* IEEE Trans. Inform. Theory, vol. IT–29, no. 6, pp. 820–824, Nov. 1983.

7. P. Duhamel, Y. Mahieux, and J.P. Petit: *A fast algorithm for the implementation of filter banks based on Time domain aliasing cancellation.* In: Proc. IEEE Int. Conf. on Acoust., Speech, Signal Processing, Toronto, Canada, May 1991, pp. 2209–2212.

8. T.R. Fischer: *A pyramid vector quantizer.* IEEE Trans. Inform. Theory, vol. 32, no. 4, pp. 820–824, July 1986.

9. A. Gersho and R.M. Gray: *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, USA, 1993.

10. R.M. Gray: *Vector quantization.* IEEE ASSP Magazine, vol. 1, no. 2, pp. 4–29, Apr. 1984.

11. S.-W. Hong, Y.-H. Chan and W.-C. Siu: *A practical real-time post-processing technique for block effect elimination.* In: Proc. IEEE Int. Conf. on Image Proc., Lausanne, Switzerland, Sept. 1996, Vol. II, pp. 21–24.

12. H.C. Kim and H.W. Park: *Signal adaptive postprocessing for blocking effects reduction in JPEG images.* In: Proc. IEEE Int. Conf. on Image Proc., Lausanne, Switzerland, Sept. 1996, Vol. II, pp. 41–44.

13. J. Kovačević: *Filter Banks and Wavelets: Extensions and Applications.* PhD thesis, Columbia Univ., New York, USA, 1992.

14. J. Kovačević: *Subband coding systems incorporating quantizer models.* IEEE Trans. on Image Processing, Vol. 4, no 5, pp. 543–553, May 1995.

15. A. Kundu and J. Zhou: *Combination median filter..* IEEE Trans. on Image Processing, Vol. 1, no 3, pp. 422–429, July 1992.

16. Y. Linde, A. Buzo, and R.M. Gray: *An algorithm for vector quantizer design..* IEEE Trans. Communications, vol. 28, no. 1, pp. 84–95, Jan. 1980.

17. S. Mallat: *A theory for multiresolution signal decomposition: The wavelet representation..* IEEE Trans. Patt. Anal. Machine Intell., vol. 11, no. 7, pp. 674–693, July. 1989.

18. H.S. Malvar: *Lapped transforms for efficient transform/subband coding..* IEEE Trans. Acoust. Speech, Signal Processing, vol. 38, no. 6, pp. 969–978, June 1990.

19. H.S. Malvar: *Signal Processing with Lapped Transforms.*, Artech House, Norwood, MA, 1992.

20. H.S. Malvar: *Extended lapped transforms: Properties, applications, and fast algorithms.*. IEEE Trans. Signal Processing, vol. 40, no. 11, pp. 2703–2714, Nov. 1992.

21. A. Mojsilović, M. Popović, and D. Šević: *Classification of the ultrasound liver images with the $2N \times 1D$ wavelet transform.*. In: Proc. IEEE Int. Conf. on Image Proc., Lausanne, Switzerland, Sept. 1996, Vol. I, pp. 367–370.

22. Z.-J. Mou and P. Duhamel: *Short–length FIR filters and their use in fast nonrecursive filtering.*. IEEE Trans. Signal Processing, vol. 39, no. 6, pp. 1322–1332, Mar. 1994.

23. S. Nanda and W.A. Pearlman: *Tree coding of image subbands.*. IEEE Trans. Image Processing, vol. 1, no. 2, pp. 133–147, April 1992.

24. M. Popović and D. Šević: *Adaptive oriented image coding using wavelet transform.*. In: Proc. 2nd Conf. TELSIKS'95, Niš, Yugoslavia, Oct. 1995, pp. 198–201.

25. M. Popović and D. Šević: *A comparative study of algorithms for subband coding of images.*. In: Proc. 40th ETRAN Conf., Budva, Yugoslavia, June 1996, Vol. 1, pp. 249–254.

26. M. Popović, D. Šević and A. Mojsilović: *Image coding using discrete wavelet transform with direction dependent resolution.*. In: Proc. 41st ETRAN Conf., Zlatibor, Yugoslavia, June 1997, Vol. 1, pp. 261–263.

27. R.L. de Queiroz and H.S. Malvar: *On the asymptotic performance of hierarchical transforms.*. IEEE Trans. Signal Processing, vol. 40, no. 11, pp. 2703–2715, Oct. 1992.

28. K. Ramchandran and M. Vetterli: *Best wavelet packet bases in a rate-distortion sense.*. IEEE Trans. Image Processing, vol. 2, no. 2, pp. 160–175, April 1993.

29. O. Rioul and P. Duhamel: *Fast algorithms for discrete and continuous wavelet transforms.*. IEEE Trans. Information Theory, vol. 38, no. 2, pp. 569–586, Mar. 1992.

30. F. Russo and G. Ramponi: *Removal of impulse noise using a FIRE filter.*. In: Proc. IEEE Int. Conf. on Image Proc., Lausanne, Switzerland, Sept. 1996, Vol. II, pp. 975–978.

31. D. Šević, and M. Popović: *A new efficient implementation of the oddly stacked Princen-Bradley filter bank.*. IEEE Signal Processing Letters, vol. 1, no, 11, pp. 166–168, Nov. 1994.

32. D. Šević and M. Popović: *Adaptive oriented image coding using extended lapped transform.*, In: Proc. 39th ETRAN Conf., Zlatibor, Yugoslavia, June 1995, Vol. 1, pp. 242–245.

33. D. Šević and M. Popović: *Image coding using nonseparable wavelet transform and lattice vector quantization.*. In: Proc. 4th Symp. TELFOR'96, Belgrade, Yugoslavia, Nov. 1996, pp. 547–550.

34. D. Šević and M. Popović: *On computing the 2–D extended lapped transforms.*. In: Proc. IEEE Int. Conf. on Acoust., Speech, Signal Processing, Munich, Germany, Apr. 1997, Vol. 3, pp. 1977–1980.

35. D. Šević and M. Popović: *Fuzzy median filter adapted to subband coding schemes.*. In: Proc. 3rd Int. Conf. TELSIKS'97, Niš, Yugoslavia, Oct. 1997, Vol. 1, pp. 162–163.

36. A. Taguchi: *A design method of fuzzy weighted median filters.*. In: Proc. IEEE Int. Conf. on Image Proc., Lausanne, Switzerland, Sept. 1996, Vol. I, pp. 423–426.

37. M. Vetterli and C. Herley: *Wavelets and filter banks: Relationships and new results.*. In: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, Albuquerque, USA, Apr. 1990, pp. 1723–1726.

38. M. Vetterli and J. Kovačević: *Wavelets and Subband Coding.*, Prentice Hall, Englewood Cliffs, NJ, USA, 1996.

39. H. Wang: *Adaptive multistage median filter.*. IEEE Trans. Signal Processing, Vol. 40, no 4, pp. 1015–1017, April 1992.

40. P. Westerink, J. Biemond, and D. Boekee: *Scalar quantization error analysis for image subband coding using QMF's.*. IEEE Trans. on Signal Processing, Vol. 40, no 2, pp. 421–428, Feb. 1992.

41. J. Woods and S. O'Neil: *Subband coding of images.*. IEEE Trans. Acoust. Speech, Signal Processing, vol. 34, no. 5, pp. 1278–1288, Oct. 1986.