IEICE TRANS. FUNDAMENTALS, VOL.E87–A, NO.4 APRIL 2004

PAPER

Selective Block-Wise Reordering Technique for Very Low Bit-Rate Wavelet Video Coding*

Ta-Te LU[†], *Nonmember and* **Pao-Chi CHANG**[†a)], *Member*

SUMMARY In this paper, we present a novel energy compaction method, called the selective block-wise reordering, which is used with SPIHT (SBR-SPIHT) coding for low rate video coding to enhance the coding efficiency for motion-compensated residuals. In the proposed coding system, the motion estimation and motion compensation schemes of H.263 are used to reduce the temporal redundancy. The residuals are then wavelet transformed. The block-mapping reorganization utilizes the wavelet zerotree relationship that jointly presents the wavelet coefficients from the lowest subband to high frequency subbands at the same spatial location, and allocates each wavelet tree with all descendents to form a wavelet block. The selective multi-layer block-wise reordering technique is then applied to those wavelet blocks that have energy higher than a threshold to enhance the energy compaction by rearranging the significant pixels in a block to the upper left corner based on the magnitude of energy. An improved SPIHT coding is then applied to each wavelet block, either re-ordered or not. The high energy compaction resulting from the block reordering can reduce the number of redundant bits in the sorting pass and improve the quantization efficiency in the refinement pass of SPIHT coding. Simulation results demonstrate that SBR-SPIHT outperforms H.263 by 1.28–0.69 dB on average for various video sequences at very low bit-rates, ranging from 48 to 10 kbps.

key words: *block-wise reordering, energy compaction, H.263, SPIHT, wavelet video coding*

1. Introduction

Very low bit-rate video coding is an interesting and challenging research field. Although the commonly used video compression standards, e.g. MPEG-4 [1] and H.263 [2], provide solutions for low bit-rate applications, they may generate annoying artifacts at very low rates. Namely, the discrete cosine transform (DCT), which is the main component in these video compression standards, suffers from the blocking effect and mosquito noise at very low bit-rates [3]–[6].

Embedded zero tree coding techniques such as Said and Pearlman's set partitioning in hierarchical trees (SPIHT) [7] and Shapiro's embedded zerotree wavelet (EZW) coding [8] exhibit the progressive coding capability. The embedding property of progressive coding allows the decoder to stop decoding at any point, without wasting any received information. The SPIHT coding is based on the pyramid structure to exploit the self-similarity across frequency bands, and the wavelet coefficients are organized into spatial orientation trees from the lowest subband to high frequency subbands. In image coding, SPIHT with a set partitioning sorting algorithm and an ordered bit plane transmission typically outperforms EZW and JPEG at a low bit-rate [9]. SPIHT has also been applied to video coding. Kim and Pearlman proposed 3D-SPIHT at the bit-rates of 30–60 kbits/s [10]. They proposed 3D wavelet decompositions and 3D spatial-temporal dependence trees that perform better than MPEG-2 [11]. Khan and Ghanbari proposed virtual SPIHT to reduce the effective number of zerotrees for video coding [12]. Lin and Gray proposed video residual coding based on SPIHT that applied Lagrangian optimization to the SPIHT encoder to achieve better rate-distortion performance than H.263 [13]. Although SPIHT coding is effective for images, however, it is still less efficient in representing block based motion-compensated residuals that have arbitrarily dispersed energy distributions in each prediction frame.

In this paper, SPIHT is applied to video coding with the selective block-wise reordering technique to enhance the coding efficiency for motion-compensated residuals. The motion estimation and motion compensation methods in H.263 are used to reduce temporal redundancy. The motion-compensated residuals are then encoded in the wavelet domain. The block-mapping reorganization technique utilizes the wavelet tree relationship that jointly presents the wavelet coefficients from the lowest subband to high frequency subbands at the same spatial location, and allocates each wavelet tree with all descendents to form a wavelet block. The block-wise reordering based on the threshold scan rearranges the significant blocks in the descending order of the energy. Then, the multi-layer block reordering technique reorders the wavelet sub-blocks recursively, according to the energy of each sub-block, to yield the maximum energy compaction that allows the SPIHT coding to operate efficiently on the motion-compensated residuals. Finally, motion vectors and the SBR-SPIHT coded bit-planes are encoded by the adaptive arithmetic coding [14].

The paper is organized as follows. The introduction to the selective block-wise reordering SPIHT (SBR-SPIHT) video coding algorithm is presented in Sect. 2. Section 3 describes the details of the motion-compensated residual coding. Section 4 shows the simulation results and comparisons of H.263 and SBR-SPIHT. Conclusions are given in the final section.

Manuscript received December 9, 2002.
Manuscript revised September 1, 2003.
Final manuscript received January 13, 2004.

[†]The authors are with the Department of Electrical Engineering, National Central University, Chung-Li, Taiwan.
a) E-mail: pcchang@ee.ncu.edu.tw

*This work was supported by the National Science Council, Taiwan, under Grant NSC-89-2219-E-008-002.

2. Selective Blockwise Reordering SPIHT Video Coding Algorithm

Figure 1 presents the block diagram of SBR-SPIHT encoder. The original SPIHT algorithm is used to encode the first intra-frame because SPIHT coding is one of the best methods to encode still images at a given target bitrate. The inter-frame coding basically includes three major parts—motion estimation, motion compensation, and motion-compensated residual coding. The motion estimation and overlapped block motion compensation (OBMC) methods of H.263 that operate in the pixel domain are used to reduce the temporal redundancy. The motion-compensated residuals are first processed by the discrete wavelet transform (DWT) [15]. The block mapping allocates each wavelet tree with all descendents to form a wavelet block. Then, the multi-layer block reordering relocates wavelet coefficients to yield maximum energy compaction and thus high efficient SPIHT coding. The details of the intra-frame coding and inter-frame coding are described separately as follows.

2.1 Intra-frame Coding

Intra-frames are encoded by the original SPIHT. The original SPIHT algorithm consists of two passes, namely, the sorting pass and the refinement pass. In the sorting pass, SPIHT sets a magnitude threshold, $T = 2^n$, where n is the level of significance. If any wavelet coefficient, $c_{i,j}$, satisfies $T \leq |c_{i,j}| < 2T$, it is set as significant. The sorting pass is used to find the significant pixels that were defined as insignificant before the current threshold scan. The list of insignificant pixels (LIP), the list of significant pixels (LSP), and the list of insignificant sets (LIS) are employed to indicate whether the pixels are significant or not in the sorting pass. Each entry of the LIP produces one bit to describe its significance. If the pixel is significant, then it is moved to the LSP and a sign bit is transmitted. Otherwise, the pixel is insignificant and remains in the LIP. Next, each set in the LIS generates one bit to indicate if the existent descendents are significant or not, and the set is partitioned into subsets when the set has at least one significant descendent. In the refinement pass, all coefficients in the LSP that have been identified as significant in the previous sorting pass are refined bit by bit. Each time when the above procedure is carried out the level of significance n is reduced by one.

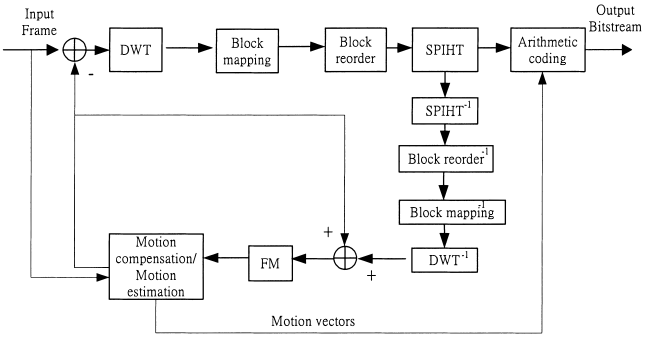


Fig. 1 Encoder of SBR-SPIHT video coding system.

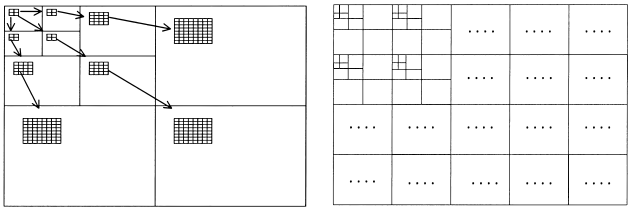
2.2 Inter-frame Coding

(1) Motion Estimation: The motion estimation used in this work from the H.263 standard, including OBMC and the possibility of four motion vectors per macroblock. The original frame is divided into non-overlapping 16×16 macroblocks with each macroblock containing four 8×8 blocks. For each macroblock, the one/four motion vectors are estimated to obtain the minimum distortion from the previous frame. The distortions are determined by the sum of absolute difference (SAD) in each macroblock. The motion estimation algorithm has intra/inter modes. If the motion estimation result is not accurate enough in this macroblock, then the intra mode is activated. Otherwise, this macroblock is set to be the inter mode and the motion vector is estimated for this macroblock.
(2) Motion Compensation: In SBR-SPIHT, OBMC of H.263 is used to reduce the overall energy of the motion-compensated residuals and the blocking effect at low bitrates [3], [4]. However, the blocking effect between inter/intra coded blocks is still serious because of different statistic properties. Therefore, in this work, the mean removing technique is used that subtracts the mean value of the intra block from the pixel values in an intra block. The mean value of an intra block is sent as side information.
(3) Motion-Compensated Residuals Coding: With the mean values removed, pixels in intra blocks and residual values in inter blocks are then combined to form a frame and coded with SBR-SPIHT algorithm in the wavelet domain. To improve the SPIHT coding efficiency, SBR-SPIHT utilizes the block mapping and the multiple-layer block reordering, which are described in details in the next section.

3. Motion-Compensated Residuals Coding

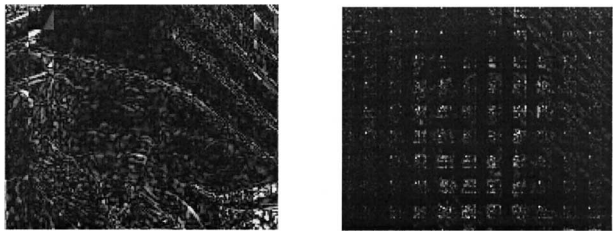
The motion-compensated residuals are first decomposed into multiresolution subbands by L level discrete wavelet transform (DWT) with Daubechies-4 orthogonal filters [15]. The motion-compensated residuals have very different characteristics from natural images. The residuals usually have much more high frequency components that may not be efficiently coded by traditional image coding approaches. In other words, the DWT may decompose these residual components into large wavelet coefficients in high frequency bands without significant coefficients in the corresponding lower frequency bands. This property significantly reduces the coding efficiency of SPIHT because SPIHT coding needs to spend many redundant bits in the LIS and LIP finding those isolated significant coefficients in high frequency bands.

According to the SPIHT bits structure, all bits can be classified into three categories: sorting significance bits,



(a) (b)

Fig. 2 Block mapping. (a) Multiple-layer wavelet structure. (b) Reorgainized wavelet blocks.



(a) (b)

Fig. 3 The 168th motion-compensated residual frame of Foreman (QCIF format). (a) The motion-compensated residual frame. (b) Wavelet blocks after block mapping.

sign bits, and refinement bits. The sorting significance bits are used to indicate the entry and descendents in the LIS and LIP that are significant or not, and these bits in the LIS and LIP are not used directly to reconstruct wavelet coefficients. The LSP is used to transmit the sign bits and the refinement pass is used to refine the wavelet coefficients bit by bit. Sign bits and refinement bits represent the sign and magnitude of wavelet coefficients respectively and they are used to reconstruct the wavelet coefficients by bit planes. The motivation of the proposed method is to enhance the ratio of sign bits and refinement bits in SPIHT bit stream so as to increase the coding efficiency. The multi-layer block reordering technique can reduce the number of bits in the sorting pass by moving high energy sub-blocks to early stages in the scan, and therefore reduces the sorting cost. The block mapping, block reordering, multi-layer block reordering, and sorting pass modification of SPIHT are described as follows.

3.1 Block-Mapping Reorganization

The block mapping reorganizes the wavelet coefficients into wavelet blocks as shown in Fig. 2. It utilizes the wavelet tree relationship that jointly presents the wavelet coefficients from the lowest subband to high frequency subbands at the same spatial location. Each frame is represented by a set of square wavelet blocks $\{B_0, B_1, B_2, \ldots, B_{N_B-1}\}$. The number of wavelet blocks N_B can be calculated by

$$N_B = \frac{W \times H}{2^L \times 2^L \times r \times r} \tag{1}$$

where $W \times H$ is the video frame size, $r \times r$ is the block size in the lowest band, and L is the number of DWT levels. Both W and H are assumed to be multiples of 16. Then the wavelet block size $w \times w$ can be derived directly from the block size in the lowest band and the level of DWT as

$$w = 2^L \times r \tag{2}$$

In this work, r is set to be 2 and L is chosen as 3 for the QCIF (176×144) format. After block mapping, SPIHT is applied to each block. Each wavelet block has different characteristics and gets a different stream length after SPIHT coding. Since most of the low frequency energy is removed from motion compensation, only a part of wavelet blocks have significant wavelet coefficients that need to be encoded. Figure 3 is an example of the 168th motion-compensated residual frame of Foreman (QCIF format). Figure 3(a) presents the motion-compensated residual frame and Fig. 3(b) shows the wavelet blocks with block size of 16×16 after the block-mapping reorganization.

3.2 Block Reordering

In general, most energy of an image is concentrated in the low frequency components that are located at the upper-left corner of the subband pyramid. However, the motion-compensated residuals have much more high energy components that are distributed in high frequency DWT bands than regular images. Therefore, it is inefficient to apply the zerotree wavelet coding, such as SPIHT, to the residuals directly. To increase the coding efficiency, the sub-block reordering is used before the SPIHT is applied. The block reordering procedure for processing each block B_k, $k = 0, 1, \ldots, N_B - 1$, is described in detail as follows.

Step 1. Initialization:

(1.1) Determine the initial significance threshold n for the whole image as

$$n = \lfloor \log_2(\max\{|c_{i,j}|\}) \rfloor \quad for\ all\ i, j, \tag{3}$$

where $c_{i,j}$ is the wavelet coefficient at (i, j), $i = \{0, 1, 2, \ldots, W-1\}$, $j = \{0, 1, 2, \ldots, H-1\}$.

(1.2) Define each wavelet block B_k as insignificant before the first threshold scan.

Step 2. Significance detection: Determine each wavelet block B_k with the block size $w \times w$ as significant or not.

Case 1: If B_k has been defined as significant in the previous scan, then go to step 3 refinement pass.

Case 2: If $\lfloor \log_2 |c^k_{i,j}| \rfloor \geq n$, for any $i \in \{0, 1, 2, \ldots, w-1\}$, $j \in \{0, 1, 2, \ldots, w-1\}$, $c^k_{i,j} \in B_k$, then B_k is set as the significant block and go to step 4 significant pass.

Case 3: Otherwise, B_k is defined as the insignificant block in this threshold scan and go to step 5 significant threshold updated.

Step 3. Refinement pass: The positions of significant sub-blocks after reordering are reserved and go to step 4.2 sub-block reordering. In other words, the reordering in later scans will not affect the already reordered results.

Step 4. Significant pass:

(4.1) Sub-block decomposition: Divide the significant block B_k into $m \times m$ sub-blocks, where m is assumed to be an exact power of 2, $B_k = \{B_{k,0}, B_{k,1}, \ldots, B_{k,b}, \ldots, B_{k,m^2-1}\}$. Each sub-block $B_{k,b}$ consists of $p \times p$ coefficients, i.e. $B_{k,b} = \{c_b^0, c_b^1, \ldots, c_b^{p^2-1}\}$ where $p = \frac{w}{m}$, and c_b^i is the wavelet coefficient in sub-block $B_{k,b}$, $i = \{0, 1, 2, \ldots, p^2 - 1\}$. The number of significant sub-blocks in block B_k is denoted as Ns_k, and $Ns_k = 0$ in initialization.

(4.2) Sub-block reordering: For all wavelet coefficients $c_b^i \in B_{k,b}$ for any $i = \{0, 1, 2, \ldots, p^2 - 1\}$, $b = \{0, 1, 2, \ldots, m^2 - 1\}$.

(4.2.1) Obtain the norm of wavelet coefficients for each significant sub-block.

• For any $i = \{0, 1, 2, \ldots, p^2-1\}$, if $\lfloor \log_2 |c_b^i| \rfloor \geq n$, then the sub-block $B_{k,b}$ is significant and the norm of all significant coefficients is calculated as

$$Norm = \sum_{i=significant} \|c_b^i\| \quad i \in \{0, 1, 2, \ldots, p^2 - 1\} \quad (4)$$

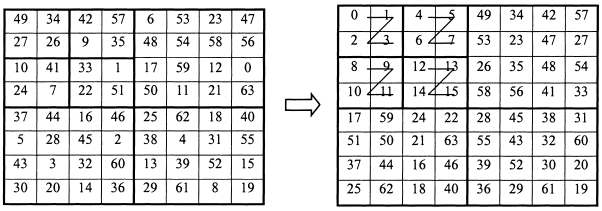
(4.2.2) Reorder the sub-blocks based on the norm. The sub-block with the maximum norm is moved to the top left corner and other significant sub-blocks are placed following the zig-zag scan order. To reduce the overhead in recording the reordering positions, the reordering sub-block operations are restricted in the upper left quadrature of the significant sub-blocks. Let Ns_k be the counter to store the total number of significant sub-blocks including the current and previous scans, one of the following operations is performed depending on Ns_k.

• If $Ns_k \leq (\frac{1}{4} \times m^2)$, check the zig-zag scan order with the sub-block significance order. If they are the same, i.e., the sub-blocks are already located in the descent energy order, then stop the reordering in this scan and go to step 5. Otherwise, perform reordering and record the original locations of significant sub-blocks as side information, then go to step 5 significant threshold updated.

• If $Ns_k > (\frac{1}{4} \times m^2)$, check the zig-zag scan order with the sub-block significance order. If they are the same, then stop and exit the algorithm. If they are different, perform reordering and record original locations of significant sub-blocks as side information, then stop and exit the algorithm.

Step 5. Significant threshold updated: Decrease n by one and go to step 2 significance detection. If n is the least significant bit (LSB), then exit the algorithm.

Figure 4 gives an example that describes the sub-block reordering. A significant block, size 16×16, is divided into 64 sub-blocks, {'0','1','2',...,'63'}, and the sub-block of number '0' represents the maximum energy sub-block while number '63' represents the minimum energy sub-block. Figure 4(a) is the distribution of original sub-blocks, while Fig. 4(b) is the sub-block reordering result. The significant sub-blocks, {'0','1','2',...,'15'}, are moved to the upper left quadrature with the shift record kept, while other wavelet sub-blocks, {'16','17','18',...,'63'}, are only sequentially shifted afterwards with no need to record the original positions.



(a) (b)

Fig. 4 Sub-block reordering of SBR-SPIHT. A block is divided into 64 sub-blocks {'0', '1', '2',...,'63'} with '0' representing the maximum energy sub-block. (a) Original 16×16 wavelet block. (b) Reordering result.

3.3 Multi-Layer Block Reordering

To achieve high energy compaction effect, the proposed block reordering method needs to divide a block into a large number of sub-blocks. However, this will spend too many bits in recording the original locations of significant sub-blocks as the side information. In order to keep low overhead and achieve high energy compaction, we propose a recursive recording algorithm, called the multi-layer block reordering method, which divides the significant sub-blocks recursively and perform reordering in each layer.

The multi-layer block reordering procedure is the same as the block reordering method described in the previous section except Step 4.2.2 reorder the sub-blocks. In this step, after the first time sub-block reordering, all the significant sub-blocks are further divided into four small sub-blocks: the upper left, the upper right, the lower left, and the lower right. These small sub-blocks are reordered again based on the Step 4.2.2 reorder the sub-blocks. The significant sub-block can then be recursively divided into four sub-blocks and the new significant sub-blocks are moved to the upper left quadrature until the positions of the significant sub-blocks are fixed, i.e. the energy in this quadrature is optimally compacted in the sense that all significant sub-blocks are in the descent order of the energy. Then, the multi-layer block reordering for this significant block is finished. The detailed procedure of recursive sub-block reordering is detailed described as follows.

- If $Ns_k^{(l)} \leq (\frac{1}{4} \times m^2)^{(l)}$, check the zig-zag scan order with the sub-block significance order at the l-th layer, where l is the reordering layer number. If they are the same, i.e., the sub-blocks are already located in the descent energy order, then stop the reordering in this scan and go to step 5 significant threshold update. Otherwise, perform reordering and record the original locations

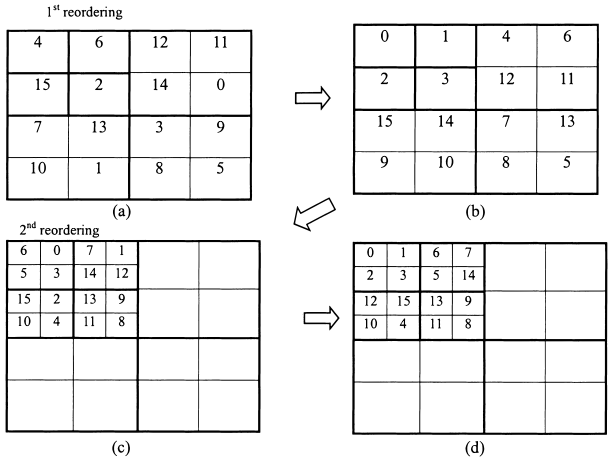


Fig. 5 Multi-layer sub-block reordering for SBR-SPIHT. A block is divided into 16 sub-blocks and 4 sub-blocks reordering recursively. (a) Original 4×4 wavelet block. (b) The first layer sub-block reordering results. (c) The second layer 4×4 sub-blocks. (d) The second layer sub-block reordering result.

of significant sub-blocks as side information, then all significant sub-blocks are divided into four small sub-blocks, $l \rightarrow l+1$, and repeat this step.

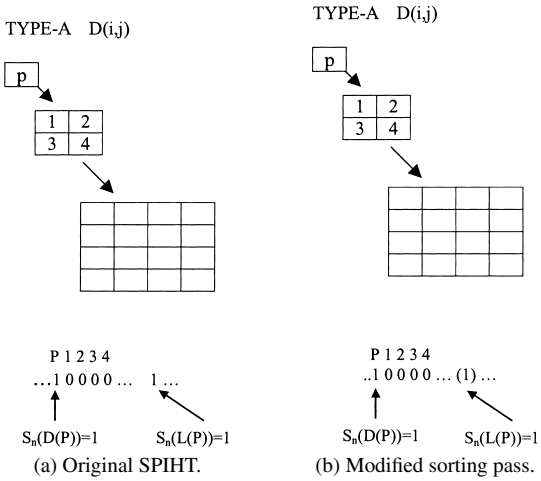
- If $Ns_k^{(l)} > (\frac{1}{4} \times m^2)^{(l)}$, check the zig-zag scan order with the sub-block significance order at the l-th layer, where l is the reordering layer number. If they are the same, then stop and exit the algorithm. Otherwise, perform reordering and record original locations of significant sub-blocks as side information, then all significant sub-blocks are divided into four small sub-blocks, $l \rightarrow l+1$, and repeat this step.

Figure 5 shows an example that describes the procedure of the sub-block reordering. The significant block, size of 16×16, is divided into 4×4 sub-blocks, {'0','1','2',...,'15'}, and the sub-block of number '0' represents the sub-block with the maximum energy. Figure 5(a) is the distribution of original sub-blocks, and Fig. 5(b) is the first layer of the sub-block reordering result. The significant sub-blocks, {'0','1','2','3'}, are moved to the upper left quadrature, while the positions of other wavelet sub-blocks, {'4','5','6',...,'15'}, are only shifted afterwards without being recorded. Then, the significant sub-blocks {'0','1','2','3'}, size of 8×8, are again divided into 4×4 sub-blocks, {'0','1','2',...,'15'}, and the sub-block reordering is performed recursively as shown in Fig. 5(c). Figure 5(d) is the sub-block reordering result of the second layer.

While the reordering method provides high energy compact effect, it does need overhead to record the original positions. At the l-th layer, each sub-block consists of $p_l \times p_l$ coefficients

$$p_l = \left(\frac{1}{2}\right)^l \times \frac{w}{m} \tag{5}$$

In this work, w is set to be 16 and m is chosen as an exact power of 2. The overhead k_i bits for coding the i-th significant block can be calculated as follows,



(a) Original SPIHT. (b) Modified sorting pass.

Fig. 6 Sorting pass modification of SPIHT.

$$K_i = \sum_{l=1}^{j} ((\log_2 m_l^2) \times Ns_l) \tag{6}$$

where Ns_l is the number of wavelet sub-blocks that need reordering in the l-th layer, m_l^2 is number of wavelet sub-blocks in the l-th layer, m_l is chosen as an exact power of 2, j is the total number of reordering levels in a significant wavelet block, and $\log_2 m_l^2$ represents the number of bits needed to identify any subblock. The recorded sequence of recursive sub-block positions is transmitted by their corresponding binary values as side information from the j-th layer to the first layer in the significant block. For example in Fig. 5, the positions are recorded as follows. The significant sub-blocks in the first layer, {'0_1','1_1','2_1','3_1'}†, are moved as shown in Fig. 5(b) and each significant sub-block needs four bits to record the original positions. The sub-block reordering is performed recursively as shown in Fig. 5(c), and the significant sub-blocks in the second layer, {'0_2','1_2','2_2','3_2'}, are moved as shown in Fig. 5(d). Finally, The recorded sequence of original sub-block positions, {'0_2','1_2','2_2','3_2','0_1','1_1', '2_1','3_1'}, is transmitted as side information, i.e., $m_l^2 = 16$, $Ns_l = 4$, and the overhead is 32 bits ($= \sum_{l=1}^{2}((\log_2 16) \times 4)$) for the significant block. Then, the side information is coded by the arithmetic coding to further reduce the overhead. If there are N_b significant wavelet blocks that need to be reordered per frame, then the total overhead K bits for coding each frame can be calculated as follows,

$$K = \sum_{i=1}^{N_b} K_i \quad \text{(bits/frame)} \tag{7}$$

3.4 Sorting Pass Modification of SPIHT Algorithm

Regardless the coding process is for image or video, with

†The subscript that represents the layer number is added in this example to avoid confusion.

reordering or not, the original SPIHT algorithm can be improved by the following modification of the sorting pass without any cost. Figure 6(a) illustrates the sorting pass of original SPIHT algorithm. In the case that the entry p of LIS has at least one significant descendant, a "1" is transmitted ($S_n(D(p)) = 1$). If the next four offspring 1, 2, 3, and 4 of p are not significant but at least one descendant is significant, p will be moved to the end of the LIS as an entry of type B and a significant bit "1" ($S_n(L(p)) = 1$) is transmitted to index some of their descendants as significant. In this case, the sorting algorithm can determine the existence of significant bits in descendants solely from the sequence $S_n(D(p)) = 1$ and succeeding four 0's. Therefore, as in Fig. 6(b), the succeeding "1" bit ($S_n(L(p)) = 1$) becomes redundant and can be ignored to reduce the bit-rate without harming the SPIHT decoding process. For a typical image, the bit-rate reduction obtained from this modification can range from 0.5 to 2%.

4. Simulation Results

Simulations are performed on various video-coding algorithms, including H.263, SPIHT, as well as the proposed SBR-SPIHT methods at very low bit-rates. Class A sequences, e.g. Akiyo, Container, and Mother&Daughter, are sampled at frame rate 5 fps and encoded at bit-rate 10 kbps. Class B sequences, e.g. Foreman, News, and Silent, are sampled at frame rate 7.5 fps and encoded at bit-rate 48 kbps. All video sequences are in the QCIF (176×144) format. The H.263 encoder includes OBMC that operates at QP=10 with TMN5 test model. In SBR-

Table 1 Number of bits in the sorting pass and refinement pass for motion-compensated residuals with SPIHT and SBR-SPIHT methods.

Sequence	Method	Number of bits per frame of SPIHT stream (Luminance)					
		Total	LIS	LIP	LSP	Refinement	Overhead
Akiyo	SPIHT	2571	554 (21.5%)	739 (28.7%)	134 (5.2%)	1144 (44.5%)	
	SBR-SPIHT	2571	336 (13.1%)	423 (16.5%)	180 (7.0%)	1498 (58.3%)	134(5.2%)
Container	SPIHT	2325	559 (24.0%)	748 (32.2%)	102 (4.4%)	916 (39.4%)	
	SBR-SPIHT	2325	415 (17.8%)	490 (21.1%)	174 (7.5%)	1116 (48.0%)	120(5.2%)
Mother & Daughter	SPIHT	2284	522 (22.9%)	690 (30.2%)	126 (5.5%)	946 (41.4%)	
	SBR-SPIHT	2284	268 (11.7%)	391 (17.1%)	166 (7.3%)	1362 (59.6%)	97(4.2%)
Foreman	SPIHT	6313	1285 (20.4%)	1605(25.4%)	721(11.4%)	2702 (42.9%)	
	SBR-SPIHT	6313	1149 (18.2%)	1117 (17.7%)	789(12.5%)	2961 (46.9%)	297(4.7%)
News	SPIHT	7383	1328 (18.0%)	1732 (23.5%)	720 (9.8%)	3602 (48.8%)	
	SBR-SPIHT	7383	982 (13.3%)	1137 (15.4%)	797(10.8%)	4111 (55.7%)	354(4.8%)
Silent	SPIHT	7237	1307 (18.0%)	1623 (22.4%)	786(10.9%)	3521 (48.7%)	
	SBR-SPIHT	7237	1043 (14.4%)	1151 (15.9%)	839(11.6%)	3894 (53.8%)	310(4.3%)
Average	SPIHT	4685	926 (19.8%)	1190 (25.4%)	432 (9.2%)	2139 (45.7%)	
	SBR-SPIHT	4685	699 (14.9%)	784 (16.7%)	491(10.5%)	2490 (53.1%)	219(4.7%)

Table 2 Performance comparison of intra-frame coding at 14 kbits per frame.

Sequence	Bits	Luminance, PSNR [dB]			Cb, PSNR [dB]			Cr, PSNR [dB]		
		H.263	SPIHT	SBR-SPIHT	H.263	SPIHT	SBR-SPIHT	H.263	SPIHT	SBR-SPIHT
Akiyo	14000	33.06	35.32	35.45	35.37	36.55	36.67	37.25	38.33	38.56
Container	14088	28.81	30.23	30.48	36.40	37.43	37.61	35.02	36.10	36.32
Mother& Daughter	13624	33.78	36.22	36.45	39.12	40.52	40.72	39.54	41.12	41.44
Foreman	13976	30.11	32.12	32.31	38.14	38.66	38.94	38.23	38.95	39.23
News	14080	28.60	29.93	30.12	33.16	34.68	34.87	34.48	35.45	35.67
Silent	13688	30.34	31.98	32.16	34.63	35.72	35.93	36.40	37.53	37.65

Table 3 Performance comparison of inter-frame coding for Class-A sequences at 5 fps and 10 kbps.

Sequence	Bit-rate (kbps)	Luminance, PSNR [dB]			Cb, PSNR [dB]			Cr, PSNR [dB]		
		H.263	SPIHT	SBR-SPIHT	H.263	SPIHT	SBR-SPIHT	H.263	SPIHT	SBR-SPIHT
Akiyo	8.61	35.02	34.75	36.56	37.59	38.15	38.95	39.81	41.44	42.21
Container	7.92	30.72	30.28	32.09	37.58	38.34	39.25	36.72	37.52	38.34
Mother& Daughter	7.95	33.31	32.85	34.24	39.42	39.03	40.63	40.08	40.06	41.06

Table 4 Performance comparison of inter-frame coding for Class-B sequences at 7.5 fps and 48 kbps.

Sequence	Bit-rate (kbps)	Luminance, PSNR [dB]			Cb, PSNR [dB]			Cr, PSNR [dB]		
		H.263	SPIHT	SBR-SPIHT	H.263	SPIHT	SBR-SPIHT	H.263	SPIHT	SBR-SPIHT
Foreman	46.48	31.57	30.97	32.12	37.31	36.37	37.57	37.40	36.23	37.82
News	45.78	34.84	34.11	35.57	38.61	37.24	38.87	39.21	38.67	39.81
Silent	46.15	35.17	34.67	35.96	38.38	38.72	39.14	39.54	39.43	40.63

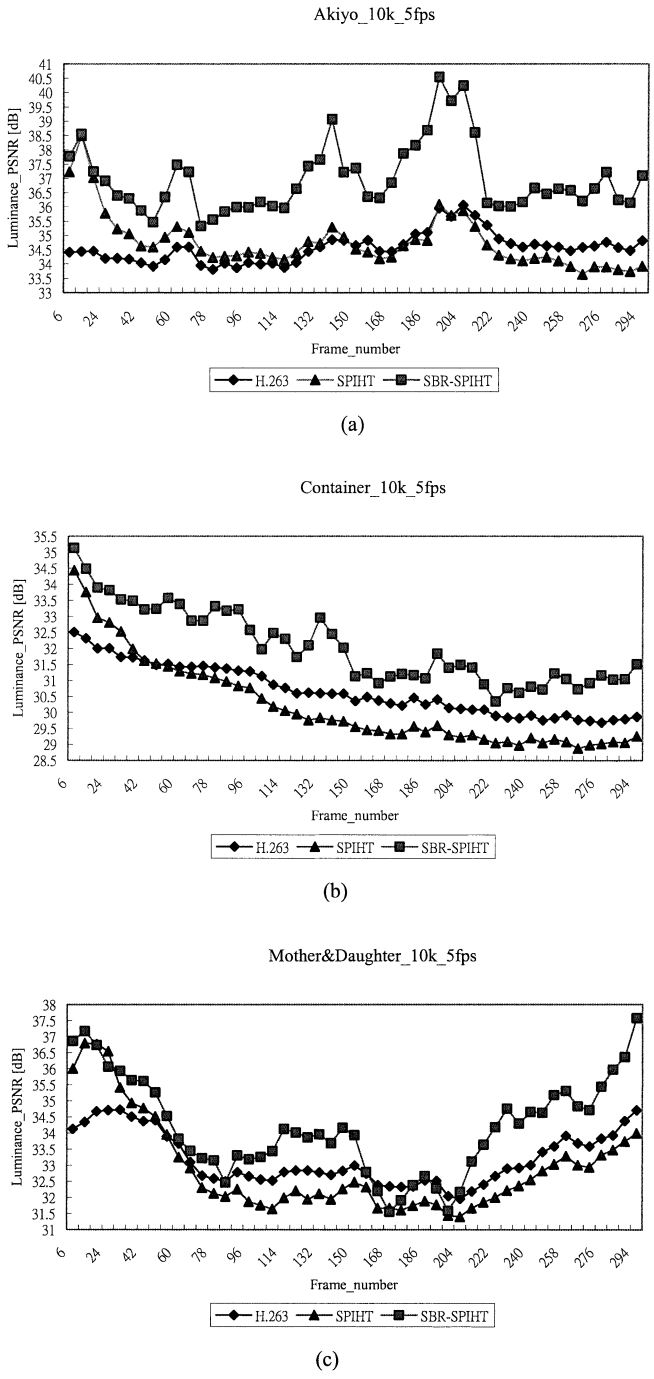


Fig. 7 PSNR performance comparisons of H.263, SPIHT, and SBR-SPIHT for Class A video sequences at 5 fps and 10 kbps.

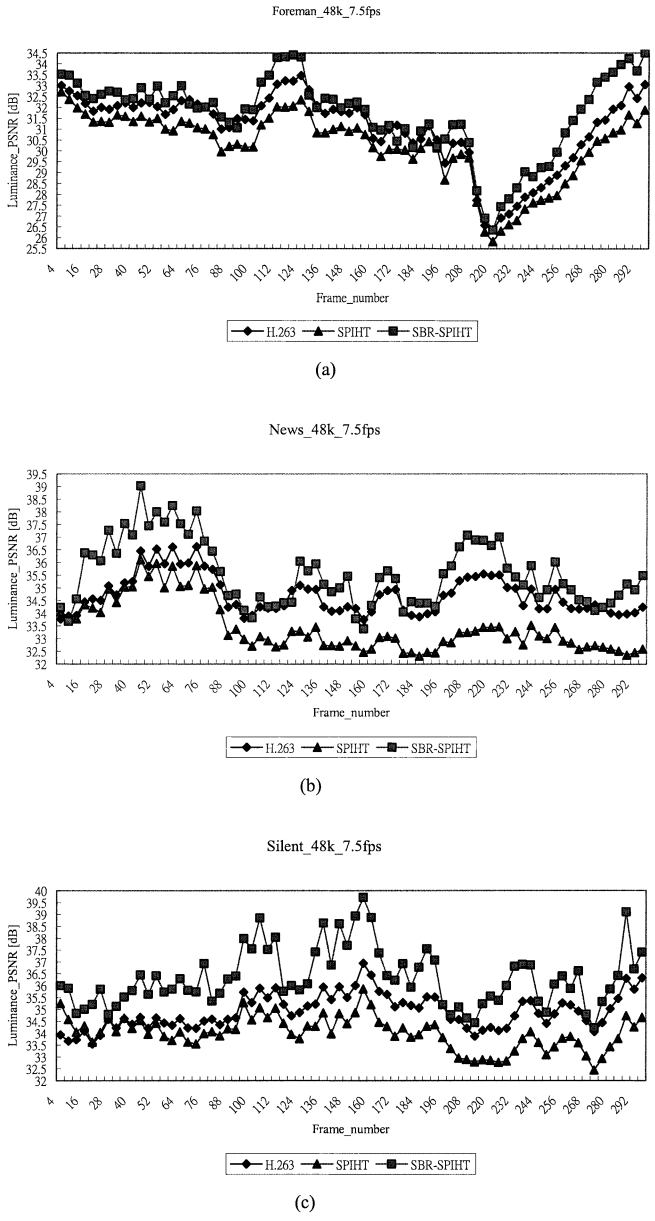


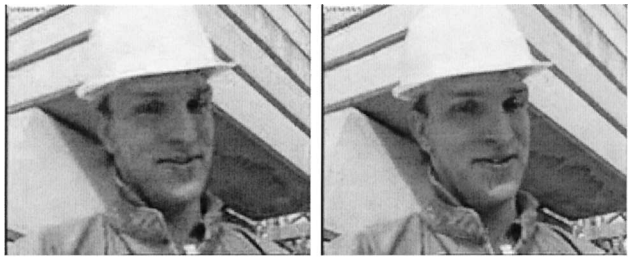
Fig. 8 PSNR performance comparisons of H.263, SPIHT, and SBR-SPIHT for Class B video sequences at 7.5 fps and 48 kbps.

SPIHT, each block is divided into four sub-blocks initially. All simulations are running on Pentium III 800 MHz.

Table 1 compares the number of bits per frame in the sorting pass and the refinement pass with the SPIHT and SBR-SPIHT methods for motion-compensated residuals. The sign bits in the LSP and the refinement bits represent the sign and the magnitude of wavelet coefficients that are truly used to reconstruct the wavelet coefficients bit by bit. Therefore, it is worthy to examine the ratio of the number of bits used in the LSP and the refinement pass because it in some sense represents the coding efficiency. The percentage of the LSP bits and the refinement bits are nearly 9.2% and 45.7%, respectively, on average in SPIHT video coding. In the condition of the same bit-rate including the reordering overhead, the percentage of the sum of the refinement bits and the LSP bits increases from 54.9% to 63.6%, which is a 8.7% bit budget increase, by using SBR-SPIHT method. On the other hand, the ratio of the sum of the LIS and the LIP reduces from 45.2% to 31.6%, which shows a 13.6% bits saving. In addition, the ratio of the bit overhead in recursive sub-block reordering is nearly 4.7%. Nevertheless, SBR-

(a) (b)

Fig. 9 The 192th reconstructed frame of Akiyo (QCIF format) by (a) H.263 (b) SBR-SPIHT at 5 fps and 10 kbps.



(a) (b)

Fig. 10 The 116th reconstructed frame of Foreman (QCIF format) by (a) H.263 (b) SBR-SPIHT at 7.5 fps and 48 kbps.

SPIHT still has more bit budget to reconstruct the decoded video frames, and therefore, achieves higher peak signal to noise ratio (PSNR). Table 2 compares intra-frame coding results for the first frames in Class A and Class B video sequences at 14 kbits per frame (0.55 bpp) with H.263, SPIHT, and SBR-SPIHT methods. For the luminance, the SBR-SPIHT outperforms the DCT-based H.263 by 2.05 dB on average. For the chrominance, the SBR-SPIHT outperforms H.263 by 1.32 dB and 1.33 dB on average for the Cb and Cr components, respectively. On the other hand, the improvement of SBR-SPIHT over SPIHT for the intra-frames is very limited because SPIHT is already able to utilize the zerotree relationship very well for still images. Table 3 and Table 4 compare inter-frame coding results for Class A and Class B video sequences at very low bit-rates with the H.263, SPIHT, and SBR-SPIHT methods. For the luminance component at 10 kbps, on average, SBR-SPIHT outperforms H.263 by 1.28 dB and SPIHT by 1.67 dB for Class A video sequences, and the processing time is about 344 ms/frame for H.263, 342 ms/frame for SPIHT, and 347 ms/frame for SBR-SPIHT, respectively. At 48 kbps, SBR-SPIHT performs better than H.263 by 0.69 dB and SPIHT by 1.3 dB on average for Class B video sequences, and the processing time is about 352 ms/frame for H.263, 351 ms/frame for SPIHT, and 360 ms/frame for SBR-SPIHT, respectively. The simulation results show that only limited time is needed for the block reordering in SBR-SPIHT. Similar improvements are obtained for the chrominance of both the Class A and Class B video sequences. Figure 7 and Fig. 8 compare the PSNRs of the luminance component of each frame of SPIHT, H.263, and SBR-SPIHT in Class A and Class B video sequences, respectively. They are compared under the circumstance that all three methods are operated at the same average bit-rate.

Figure 9 and Fig. 10 show the luminance component of reconstructed frame by using H.263 and SBR-SPIHT in Class A and Class B video sequences, respectively. The blocking artifacts are generated by H.263 at very low bit-rates are shown as in Fig. 9(a). The visual quality of SBR-SPIHT is clearly superior in the edges of reconstructed frames such as Akiyo's shoulder in Fig. 9(b) and Forman's eyes in Fig. 10(b). Since SBR-SPIHT performs wavelet transform on the whole image, the blocking artifact is not serious. On the other hand, the DCT based image coding methods, such as H.263, suffers more serious artifact. These results reveal that the proposed algorithm offers a more efficient way to encode the significant wavelet coefficients than the original SPIHT. The SBR-SPIHT can be efficiently applied to video residuals by concentrating the encoding of randomly dispersed energy in local areas.

5. Conclusions

We have presented SBR-SPIHT coding that uses a novel selective block-wise reordering technique for video coding. The approach is particularly suitable for coding video sequences at very low bit-rates. The proposed method provides the advantage that the energy compaction in the upper left corner indeed improves the SPIHT coding efficiency by reducing the sorting cost and increasing the bit budget for actual reconstruction. The proposed block reordering algorithm with the wavelet transform has shown a solution for the long time problem that the zerotree based wavelet coding does not perform as well for videos as for images.

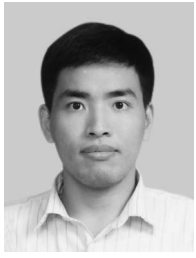
Acknowledgement

The authors would like to thank the reviewers for their valuable comments to enhance the quality of the paper.

References

[1] Streaming video Profile-Final Draft Amendment (FDAM4), MPEG01/N3904.
[2] Draft ITU-T Recommendation H.263, Video coding for low bit rate communication, May 1996.
[3] S.A. Martucci, I. Sodagar, T. Chiang, and Y.Q. Zhang, "A zerotree wavelet video coder," IEEE Trans. Circuits Syst. Video Technol., vol.7, no.1, pp.109–118, Feb. 1997.
[4] J. Vass, B.B. Chai, and X. Zhuang, "Significance-linked connected component analysis for very lower bit-rate wavelet video coding," IEEE Trans. Circuits Syst. Video Technol., vol.9, no.4, pp.630–647, June 1999.
[5] K. Shen and E.J. Delp, "Wavelet based rate scalable video compression," IEEE Trans. Circuits Syst. Video Technol., vol.9, no.1, pp.109–122, Feb. 1999.
[6] G. Xing, J. Li, S. Li, and Y.Q. Zhang, "Arbitrarily shaped video-object coding by wavelet," IEEE Trans. Circuits Syst. Video Technol., vol.11, no.10, pp.1135–1139, Oct. 2001.
[7] A. Said and W.A. Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees," IEEE Trans. Circuits

Syst. Video Technol., vol.6, no.3, pp.243–250, June 1996.
[8] J.M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," IEEE Trans. Signal Process., vol.41, no.12, pp.3445–3462, Dec. 1993.
[9] T.T. Lu, K.W. Wen, and P.C. Chang, "Block reordering wavelet packet SPIHT image coding," Proc. IEEE Pacific-Rim Conference on Multimedia, pp.442–449, Oct. 2001.
[10] B.J. Kim, Z. Xiong, and W.A. Pearlman, "Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees," IEEE Trans. Circuits Syst. Video Technol., vol.10, no.8, pp.1374–1387, Dec. 2000.
[11] B.J. Kim and W.A. Pearlman, "An embedded video wavelet coder using three-dimensional set partitioning in hierarchical tree," Proc. Data Compression Conf., pp.251–260, 1997.
[12] E. Khan and M. Ghanbari, "Very low bit rate video coding using virtual SPIHT," Electron. Lett., vol.37, pp.40–41, Jan. 2001.
[13] K.K. Lin and R.M. Gray, "Video residual coding using SPIHT and dependent optimization," Proc. Data Compression Conf., pp.113–122, 2001.
[14] I.H. Witten, R.M. Neal, and J.G. Cleary, "Arithmetic coding for data compression," Commun. ACM, vol.30, pp.520–540, June 1987.
[15] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," IEEE Trans. Image Process., vol.1, no.2, pp.205–220, April 1992.



Ta-Te Lu was born in Taiwan on March 19, 1966. He received the B.S. degree in Electrical Engineering from Fu Jen University in 1989 and the M.S. degree in Electrical Engineering from Chung Cheng Institute of Technology in 1991. He is currently a Ph.D. candidate in National Central University. His current interests are in the area of image/video coding, and watermarking techniques.

Pao-Chi Chang was born in Taipei, Taiwan, in 1955. He received the B.S. and M.S. degrees from National Chiao Tung University, Taiwan, in 1977 and 1979, respectively, and the Ph.D. degree from Stanford University, California, 1986, all in electrical engineering. From 1986–1993, he was a research staff member of the department of communications at IBM T.J. Watson Research Center, Hawthorne, New York. At Watson, his work centered on high speed switching system, efficient network design algorithms, network management, and multimedia teleconferencing. In 1993, he joined the faculty of National Central University, Taiwan. His current interests are in the area of speech/image coding, watermarking, and video coding over high speed networks and wireless communications.