Available online at www.sciencedirect.com
SCIENCE DIRECT®
Signal Processing: *Image Communication* 18 (2003) 769–785
SIGNAL PROCESSING: IMAGE COMMUNICATION
www.elsevier.com/locate/image

Significance map pruning and other enhancements to SPIHT image coding algorithm

Ulug Bayazit*

Department of Electronics Engineering, IŞIK University, Buyukdere Cad., Maslak, Istanbul 34398, Turkey

Received 25 September 2002; received in revised form 21 February 2003; accepted 5 April 2003

Abstract

This paper proposes several enhancements to the Set Partitioning in Hierarchical Trees (SPIHT) image coding algorithm without changing the original algorithm's general skeleton. First and foremost, a method for significance map pruning based on a rate-distortion criterion is introduced. Specifically, the (Type A) sets of wavelet coefficients with small ratios of estimated distortion reduction to estimated rate contribution are deemed insignificant and effectively pruned. Even though determining such sets requires the computational complexity of the encoder to increase considerably with respect to the original SPIHT encoder, the original SPIHT decoder may still be used to decode the generated bitstream with a low computational complexity. The paper also proposes three low complexity enhancements by more sophisticated use of the adaptive arithmetic coder. Simulation results demonstrate that all these enhancements yield modest compression gains at moderate to high rates.

Keywords: Image coding; Wavelet; SPIHT; Pruning; Arithmetic coding

1. Introduction

Set partitioning in hierarchical trees (SPIHT) algorithm of [15,16] is regarded as a benchmark for lossy and lossless embedded wavelet-based image coding in recent years due to its salient features such as resolution and SNR scalability and low computational complexity as well as a rate-distortion performance matched by only a few other algorithms. SPIHT was an advance over the innovative embedded zerotree wavelet (EZW) image coding method of [17] which employed a tree representation of zeroes of wavelet coefficients for the coding of these coefficients. EZW, itself, yielded several dB's of signal-to-noise ratio improvement for most real world still images at low bit-rates over the discrete cosine transform (DCT) based older JPEG still image coding standard. A variant of EZW coding called zero tree entropy (ZTE, [12,19]) coding has been introduced into the recent MPEG-4 Standard as a texture representation and coding scheme.

*Tel.: +90-212-286-2960 x1803; fax: +90-212-285-9453.
E-mail address: bayazit@isikun.edu.tr (U. Bayazit).

0923-5965/03/$ - see front matter
doi:10.1016/S0923-5965(03)00049-3

In recent years, research on SPIHT has been extended to the coding of DCT coefficients [22], coding and representation of blocks of wavelet coefficients [13], 3-D video [8] and 1-D ECG signals [10]. SPIHT is less efficient in coding performance than embedded block coding with optimized truncation (EBCOT [20]), the main wavelet-based image coding algorithm of the new JPEG-2000 standard, but has dramatically lower computational complexity, since it does not employ any explicit rate-distortion optimization. Wavelet based image compression methods based on Trellis coded quantization (TCQ, [7,1,23]) that are competitive with SPIHT on a performance scale have significantly higher complexity than SPIHT. The quadtree and binary classification schemes of [1,23] that determine which wavelet coefficients should be quantized by TCQ somewhat reduce this complexity. The rate-distortion optimized embedding (RDE) method of [9] is reported to achieve better coding performance than SPIHT on certain test images by allocating the available bits first to the coefficients with the steepest R-D slopes. The two-layer wavelet based algorithm by Marpe et al. [11] employs precoding in its first layer with zerotree coding of insignificant coefficients and context based conditional arithmetic coding, and codes the residual image with a wavelet packet basis in its second layer. The reported several tenths of dB gain over SPIHT comes at the expense of additional complexity due to the second layer.

The low complexity image coding option of JPEG2000 is called SBHP ([4,14]). SBHP is based on SPECK [6], a variant of SPIHT that partitions sets of coefficients on a quadtree structure and does not employ arithmetic coding. Embedded image coding method using ZeroBlocks of wavelet/subband coefficients and Context modelling (EZBC) of [18] incorporates context-based arithmetic coding to SPECK and is competitive with EBCOT on an objective quality scale.

In this work, our first challenge was to rate-distortion optimize the *significance map* (set of significance decisions ordered by the spatial orientation tree, SOT) generated by the SPIHT algorithm. The significance decisions for sets of coefficients made by the SPIHT encoding algorithm are not necessarily rate-distortion optimal since they are based on the l_∞ norm of the set of coefficient magnitudes. The l_2 norm should be the norm to consider when the distortion performance is assessed in terms of mean squared error (MSE). In [13], the significance decisions for 2×2 blocks of coefficients are based on the l_2 norm, but the rate-distortion advantage realized is negligible. With reasonable complexity increase on the encoder side, we propose a set significance decision based on the l_∞ norm to be reversed (node corresponding to the set in the significance map to be "pruned") if the marginal return (ratio of distortion reduction to rate contribution) of coding the set is less than a threshold. The method is similar to that of [3] which successively prunes the nodes with the smallest marginal return from the tree structured codebook, one at a time.

The significance map pruning method requires the original SPIHT encoder to be run multiple times. The marginal returns for significant sets can be computed after estimating the rate increases and distortion reductions in an initial run. After the prunings are performed, a second run generates the actual bitstream, respecting the significance decision reversals. Computational complexity of the proposed encoder is well over that of the original SPIHT encoder due to the multiple runs performed.

The three other enhancements discussed in this paper improve the original SPIHT's coding performance without increasing the algorithmic and computational simplicity of the encoder or the decoder. These enhancements do not alter the process of significance decision-making or the uniform quantization of the significant wavelet coefficients and the resulting quantization values. They only alter the way the sets of quantization levels are represented in the bitstream and take advantage of the arithmetic coding tool [21] to exploit trends in the probability models. In the sorting pass of SPIHT, where the significance decisions for sets of coefficients are made, we propose that the coefficient signs and significance decisions are jointly arithmetic coded rather than separately coded. We also propose the conditioning of the adaptive arithmetic coding of the Type B set (descendants of a coefficient excluding its children) significance decisions on the level of the SOT at which the Type B set is rooted. In the refinement pass of SPIHT, judicious adaptive arithmetic coding of coefficient magnitude refinement decisions is proposed.

In Section 2 the original SPIHT algorithm of [15] is reviewed. Section 3 presents the enhancements in detail. Simulation results demonstrating the effectiveness of the enhancements are presented in Section 4 for the coding of several still images at various rates. Section 5 presents the conclusions.

2. Original SPIHT algorithm

As mentioned in [15], the image is wavelet transformed, and in pass $p \in \{0, 1, \ldots, P\}$, the coefficients with magnitudes exceeding the magnitude threshold

$$T_p = \frac{c_{\max}}{2^{p+1}} \tag{1}$$

are encoded and transmitted, where

$$c_{\max} = 2^{\left\lceil \log_2 \max_{(i,j)} |c(i,j)| \right\rceil},$$

and $c(i,j)$ is the coefficient value at position (i,j). The significance decision for a set of coefficients $\Re$ in pass p is indicated as

$$S_p(\Re) = \begin{cases} 1 & \max_{c(i,j)\in\Re} \{|c(i,j)|\} \geqslant T_p, \\ 0 & \text{otherwise.} \end{cases}$$

The coefficients are ordered in hierarchies, called *spatial orientation trees*. Roots of SOT in the lowest frequency subband branch successively into higher frequency subbands at the same spatial orientation. The set of offspring, $O(i,j)$, of an SOT node corresponding to a wavelet coefficient at coordinates (i,j), consists of the wavelet coefficients of the same spatial orientation in the next (finer resolution) level of the pyramid. The nodes at the lowest level have no offspring. The set of all descendants of a node corresponding to a coefficient at coordinates (i,j), is termed a Type A set and is denoted by $D(i,j)$. A Type A set excluding the offspring of a node corresponding to a coefficient at coordinates (i,j) is termed a Type B set and is denoted by $L(i,j), L(i,j) = D(i,j) - O(i,j)$.

Initially, 2×2 blocks of wavelet coefficients in the lowest frequency subband designate four single coefficient sets and three Type A sets of three of the four coefficients except the upper left one which does not have any descendants.

In the sorting phase of each pass a $D(i,j)$ is partitioned into an $L(i,j)$ and four single coefficient sets $(k,l) \in O(i,j)$ whenever $S_p(D(i,j)) = 1$, and an $L(i,j)$ is partitioned into four sets $\{D(k,l) : (k,l) \in O(i,j)\}$ whenever $S_p(L(i,j)) = 1$. Consequently, when $S_p(L(i,j)) = 1$, Type A set $D(i,j)$ is broken down into four single coefficient sets $(k,l) \in O(i,j)$ and four smaller Type A sets $D(k,l)$ with $(k,l) \in O(i,j)$. The significance decisions imposed on an SOT of coefficients by this partitioning rule and the smallest magnitude threshold will be called *the significance map* which can be conceived as another tree where all nodes correspond to significant Type A or significant single coefficient sets.

Significance information is stored in three ordered lists, called list of insignificant sets (LIS), list of insignificant pixels (LIP), and list of significant pixels (LSP). During the sorting pass, the significances (with respect to T_p) of the coefficients in the LIP are coded and those that become significant are transferred to the end of the LSP after their signs are coded. Similarly, the significance of the sets in the LIS are coded and those that become significant are partitioned and removed from LIS. The significances of the resulting four single coefficient sets are also coded. The significant ones are added to the end of LSP after their signs are coded. The insignificant ones are appended to LIP. Newly formed Type A and Type B sets are appended to LIS and evaluated in the same sorting pass.

In the sorting pass, each significant coefficient is reconstructed at the encoder and the decoder as $\pm 1.5T_p$ depending on its sign. In each refinement pass, the values of the coefficients in the LSP, except the ones included in the last sorting pass, are refined. Specifically, the previous reconstruction errors are quantized by a two level quantizer and values of $\pm 0.5T_p$ are added to the previous reconstruction values of the coefficients depending on the sign of the quantization levels.

Optionally, the significance decisions for the previously insignificant coefficients of a block of 2×2 coefficients or the previously insignificant Type A sets rooted in a block of 2×2 coefficients may be represented by a single symbol which is arithmetic coded.

3. Enhancements to the SPIHT image coding algorithm

3.1. Significance map pruning based on rate-distortion thresholding

The sorting pass in the SPIHT algorithm can be regarded as a greedy tree (significance map) growing method where the nodes of the tree correspond to significant Type A sets and significant single coefficient sets. Nodes are added to this tree as they become significant with respect to the tested magnitude threshold. This greedy tree growth method enables the earlier bits in the bitstream to be expended on the largest magnitude coefficients to yield the largest reductions in distortion. However, there is no guarantee that *the distortion reduction per expended bit* decreases with the number of coded bits. To see this, let us consider a previously insignificant Type A set with only four coefficients (smallest Type A set possible). When the Type A set is found to be significant, four bits are expended to transmit the significance decisions for these coefficients assuming that symbols are fixed length coded. If all four coefficients are indeed significant (which is a rare case) then one bit is spent to indicate each significant coefficient. On the other extreme, if only a single coefficient is significant, then four bits are spent to indicate it. We can extend this example to the case where the Type A set contains more than four coefficients (i.e. 16, 64, …) and likewise compare the extreme case of only a single or a few coefficients being significant against the other extreme case of most or all coefficients being significant. The difference between the number of bits required to indicate each significant coefficient in these two cases is even larger.

It is also straightforward to see that, for each coefficient that becomes significant in one pass, the distortion reduction obtained by indicating a reconstruction value by coding significance and sign information in that pass is strictly positive. The statistical average of this distortion reduction decreases with each pass since the magnitude thresholds are halved. However, *the average distortion reduction per expended bit* for coefficients that become significant is not guaranteed to decrease with each pass, since, as exemplified above, each coefficient that becomes significant may be indicated by coding a different number of bits. Therefore, it may be advantageous to expend the earlier bits in the bitstream to code those coefficients with large expected rate-distortion advantages even though they may become significant with respect to the magnitude threshold in late passes.

One possible means to achieve such a goal is to declare a set of coefficients insignificant if the rate-distortion advantage of coding that set is smaller than a certain rate-distortion threshold λ, even when the largest coefficient magnitude inside the set is larger than the magnitude threshold. Specifically, after a first run of the SPIHT encoder, we compute the marginal return, $\Delta D/\Delta R$, for each (single coefficient set and Type A) set that becomes significant at some point during the run where ΔD and ΔR are the distortion reduction and rate increase, respectively, as a result of coding that set. The sets with $(\Delta D/\Delta R) < \lambda$ are deemed insignificant (pruned from the significance map) even though the largest coefficient magnitudes in these sets may exceed the smallest magnitude threshold, T_P of the first run. During a second run of the SPIHT algorithm, the actual bitstream is generated by observing these (pruning) decisions.

The distortion reduction, ΔD, due to a single coefficient set that becomes significant and gets coded by sending sign and refinement bits, is strictly positive since the largest distortion is incurred when a single coefficient set is deemed insignificant and not coded. The distortion reduction, ΔD, due to a Type A set that becomes significant and gets coded, is the sum of the distortion reductions for the significant subsets of the Type A set. Since the smallest subsets are the single coefficient subsets for which the distortion reductions are positive, the distortion reduction for the Type A set is also positive.

On the other hand, ΔR, the change in rate due to a set (single coefficient or Type A) that becomes significant and gets coded, could be either positive, negative or zero. To see this, we decompose ΔR into two: The change in rate due to the subsets of the set that become significant and the change in rate due to the coding of the significance decisions for the set and its neighbors (as shown in Fig. 1, we call two sets neighbors if they are the children of the same Type A parent set). The latter quantity could be positive, negative, or zero depending on the rank of the set becoming significant among its neighbors. We can see this by considering the memoryless and fixed length coding of the significance decisions. First, suppose that a Type A set is the first one to become significant in a given pass among the four Type A sets rooted in a block of 2×2 coefficients. A rate increase will accompany since the Type B set that encompasses the four Type A sets is split and the significance decisions for the newly formed four Type A sets (including the one that is significant) are coded by additional bits in this and future passes. Next suppose that a Type A set is not the first one to become significant in a given pass. A rate decrease will accompany since coding the Type A set significant in this pass implies that the significance decisions for that Type A set will not be coded in future passes.

Since ΔR depends on the rank of the set becoming significant among its neighbors, so does the marginal return of coding that set. Conversely, the marginal return of pruning a set from a fully grown tree depends on the rank of the set getting pruned among its neighbors. This also means that with each pruned set the

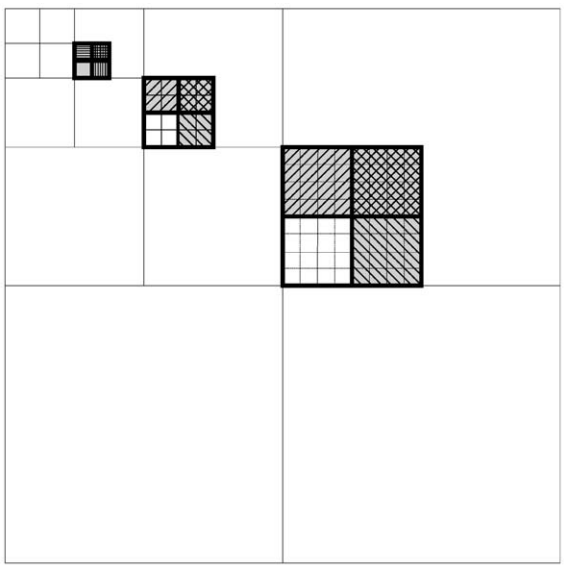


Fig. 1. A Type A set rooted at the third level from the bottom is shown as the unshaded region in the first and second levels from the bottom that is partitioned (by thin solid lines) into 4 single coefficient subsets and 4 smaller Type A subsets (each of which is partitioned by thin dotted lines into 4 single coefficients subsets). The Type A set has 3 Type A set neighbors indicated by diagonally hatched and diagonally cross-hatched shaded regions to the right, top and top-right. It also has 4 single coefficient neighbors indicated by vertically, horizontally and cross-hatched and unhatched shaded regions at the third level from the bottom.

marginal return of pruning each significant neighbor set as well as each ancestor Type A set changes. Therefore, when we prune a single coefficient set or a Type A set we should update the marginal returns of all significant neighbor sets as well as all ancestor Type A sets.

In binary coding mode of SPIHT, the significance of each (single coefficient or Type A) set as well as the sign of each significant coefficient is represented by coding one bit. It is straightforward to get an exact estimate of ΔR for each significant set by simply counting the number of set significance decisions and number of coefficient signs coded under that set. However, in adaptive arithmetic coding of mode of SPIHT, the probability models evolve with coded symbols. When the coded symbols are altered as a consequence of pruning, the probability models are altered as well. This makes it difficult to trace the effect of each pruning on the ΔR of sets. Therefore, in adaptive arithmetic coding mode, we estimate ΔR for each set as if the significance symbols are binary (memoryless, fixed-length) coded.

Below, the iterative significance map pruning method, that embodies the above ideas, is concisely presented in pseudo code for a rate-distortion threshold of λ and P passes:

for each spatial orientation tree root coordinate pair (p,q) {
  do {
$\lambda_{\min} \leftarrow \infty$
for each significant Type A set $D(i,j) : (i,j) \in D(p,q)$ {

$$\text{compute } \Delta D_{D(i,j)} = \sum_{(k,l)\in O(i,j)} \Delta D_{D(k,l)} + \sum_{(k,l)\in O(i,j)} \Delta D_{c(k,l)} \tag{2}$$

//Sum of distortion reductions of its subsets

$$\text{compute } \Delta R_{D(i,j)} = \sum_{(k,l)\in O(i,j)} \Delta R_{D(k,l)} + \sum_{(k,l)\in O(i,j)} \Delta R_{c(k,l)} + \tilde{R}_{D(i,j)} \tag{3}$$

//Sum of rate contributions of its subsets and rate to code the significance decisions for its subsets

$$\text{compute } \lambda_{D(i,j)} = \frac{\Delta D_{D(i,j)}}{\Delta R_{D(i,j)}}.$$

if $(\lambda_{D(i,j)} < \lambda_{\min})$ {
  $\lambda_{\min} = \lambda_{D(i,j)}$//save marginal return of the best set candidate
  $(i^*, j^*) = (i,j)$//save coordinates of the best set candidate
  $B = D(i,j)$//Save type (Type A) of best set candidate
 }
}
for each significant coefficient $c(i,j) : (i,j) \in D(p,q)$ {

$$\text{compute } \lambda_{c(i,j)} = \frac{\Delta D_{c(i,j)}}{\Delta R_{c(i,j)}}.$$

if$(\lambda_{c(i,j)} < \lambda_{\min})$ {
  $\lambda_{\min} = \lambda_{c(i,j)}$ //Save marginal return of the best set candidate
  $(i^*, j^*) = (i, j)$ //Save coordinates of the best set candidate
  $B = c(i,j)$ //Save type (single coefficient) of best set candidate
 }
}

if ($\lambda < \lambda_{\min}$) break
$S_{\lambda,P}(B) \leftarrow 0$//Prune best set by setting significance to zero

$$\Delta D_B \leftarrow 0 \text{//Pruned set does not contribute to distortion reduction} \tag{4}$$

$$\Delta R_B \leftarrow 0 \text{//Pruned set does not contribute to rate increase} \tag{5}$$

$$\tilde{R}_{D(m,n)} \leftarrow \tilde{R}_{D(m,n)} + \Delta\tilde{R}_B \text{ where } (m,n) : (i^*,j^*) \in \mathrm{O}(m,n) \tag{6}$$

//Update rate for significance decisions of neighbors
for each coefficient $\{c(i,j) : (i,j) \in \mathrm{O}(m,n), S_{\lambda,P}(c(i,j)) = 1\}$ {
//Precompute rate updates for significance decisions of neighbors
compute $\Delta\tilde{R}_{c(i,j)}$ //for the prospective pruning of each neighbor single coefficient set
}
for each Type A set $\{D(i,j) : (i,j) \in \mathrm{O}(m,n), S_{\lambda,P}(D(i,j)) = 1\}$ {
//Precompute rate updates for significance decisions of neighbors
compute $\Delta\tilde{R}_{D(i,j)}$ //for the prospective pruning of each neighbor Type A set
}
} while ($\lambda_{\min} < \lambda$) //Continue set prunings until all sets with marginal returns less than λ are exhausted
}

In the above, $\Delta D_{c(i,j)} = c(i,j)^2 - (c(i,j) - \hat{c}(i,j))^2$ is the increase in squared error distortion as a result of pruning the coefficient $c(i,j)$ where $\hat{c}(i,j)$ is the reconstruction of $c(i,j)$ at the end of the first run. The corresponding decrease in the number of expended bits is denoted by $\Delta R_{c(i,j)}$. The quantities $\Delta D_{D(i,j)}$ and $\Delta R_{D(i,j)}$ are similarly defined for the pruning of Type A set $D(i,j)$. The quantity $\tilde{R}_{D(i,j)}$ represents the number of bits expended to code the significance decisions for the sets whose immediate ancestor (parent) is $D(i,j)$. Note that the computations performed by Eqs. (2) and (3) to yield $\Delta D_{D(i,j)}$ and $\Delta R_{D(i,j)}$, respectively, are recursive.

At each iteration of the algorithm, the best (Type A or single coefficient) set B with the smallest marginal return of $\lambda_{\min}$, residing at coordinates (i^*,j^*) is pruned by setting $S_{\lambda,P}(B) \leftarrow 0$ for as long as $\lambda_{\min} < \lambda$. Here, the indicator function $S_{\lambda,P}(.)$ denotes the significance of its argument with respect to λ, the rate-distortion threshold, and T_p, the magnitude threshold of the final pass of the first run. If $\lambda_{\min} \geqslant \lambda$ the pruning iterations are terminated. As set B is pruned, $\Delta D_B \leftarrow 0$, $\Delta R_B \leftarrow 0$ so that the increase in distortion and decrease in rate for prospective pruning of an ancestor set of set B may be properly computed in the future by means of Eqs. (2) and (3). The coordinate pair (m,n) denotes the parent of the coordinate pair (i^*,j^*) of the pruned set B. In order to employ Eq. (3), $\tilde{R}_{D(m,n)}$, the rate required to code the significance decisions of the sets whose immediate ancestor (parent) is $D(m,n)$ (neighbors of the pruned set B), also needs to be updated after set B is pruned. Update of $\tilde{R}_{D(m,n)}$ is achieved by adding a precomputed quantity $\Delta\tilde{R}_B$ to $\tilde{R}_{D(m,n)}$. In other words, $\Delta\tilde{R}_B$ is the difference in $\tilde{R}_{D(m,n)}$ before and after the pruning of set B. Furthermore, after set B is pruned, $\{\Delta\tilde{R}_{c(i,j)} : (i,j) \in \mathrm{O}(m,n), S_{\lambda,P}(c(i,j)) = 1\}$, the differences in $\tilde{R}_{D(m,n)}$ before and after prospective pruning of significant single coefficient sets that are neighbors of set B, and $\{\Delta\tilde{R}_{D(i,j)} : (i,j) \in \mathrm{O}(m,n), S_{\lambda,P}(D(i,j)) = 1\}$, the differences in $\tilde{R}_{D(m,n)}$ before and after prospective pruning of significant Type A sets that are neighbors of set B, are precomputed to be used in future iterations.

We can compute $\Delta\tilde{R}_B$ by estimating $\tilde{R}_{D(m,n)}$ before and after the pruning of set B. $\tilde{R}_{D(m,n)}$, in turn, may be estimated after the first run from the number of passes, $N_{D(i,j)}$, for which Type A set $D(i,j)$ is significant, and the number of passes, $N_{c(i,j)}$, for which single coefficient set $c(i,j)$ is significant, for all coordinate

pairs (i,j) that are offspring of the coordinate pair (m,n). Let $N_{C_k} = \underset{(i,j):(i,j)\in O(m,n)}{k'\text{thmax}} \{N_{c(i,j)}\}$ and $N_{D_k} = \underset{(i,j):(i,j)\in O(m,n)}{k'\text{thmax}} \{N_{D(i,j)}\}$. We employ the following formula to estimate $\tilde{R}_{D(m,n)}$:

$$\begin{aligned}\tilde{R}_{D(m,n)} = {} & 4(\max\{N_{C_1}, N_{D_1}\} - N_{C_1} + (N_{C_1} > 0)) + 3(N_{C_1} > 0)\,(N_{C_1} - N_{C_2} - (N_{C_2} == 0)) \\ & + 2(N_{C_2} > 0)\,(N_{C_2} - N_{C_3} - (N_{C_3} == 0)) + (N_{C_3} > 0)\,(N_{C_3} - N_{C_4} - (N_{C_4} == 0)) \\ & + \max\{0, N_{C_1} - N_{D_1} + (N_{D_1} > 0)\} \\ & + (N_{D_1} > 0)(4 + 3(N_{D_1} - N_{D_2} - (N_{D_2} == 0)) + 2(N_{D_2} > 0)(N_{D_2} - N_{D_3} - (N_{D_3} == 0)) \\ & + (N_{D_3} > 0)(N_{D_3} - N_{D_4} - (N_{D_4} == 0))).\end{aligned}$$

In the above, the first two lines account for the number of bits expended to represent the significance decisions for the 4 single coefficient sets. The third line adds the contribution of the number of bits needed to code the significance decisions for the Type B sets. The final line is due to the number of bits needed to code the significance decisions for the Type A sets. The above estimate becomes exact when the set significance decisions are memoryless, fixed-length coded.

Finally, we address the choice for λ, the rate-distortion threshold for pruning the sets. Ideally, λ should be such that the desired rate is achieved. However, given P passes in the first run, the value of λ is restricted by the quantizer step size, T_p, used in the Pth refinement pass. An estimate for the marginal return of the $P-1$th refinement pass can be formed as the ratio of the expected value of ΔD to the expected value of ΔR for the $P-1$th refinement pass. This ratio is approximately T_P^2 if the quantizer distortion is modelled by a random variable uniformly distributed within the range $[-T_P, T_P]$ for the $P-1$th refinement pass, and within the range $[-2T_P, 2T_P]$ for the $P-2$th refinement pass. If the refinement bits are arithmetic coded this ratio is expected to be slightly larger. Note that λ should not significantly exceed T_P^2; otherwise coding the $P-1$th refinement pass is suboptimal. We cannot remove the $P-1$th refinement pass without removing the Pth pass completely. On the other hand, any value for λ significantly smaller than T_P^2 is also suboptimal since one can do better in this case by further pruning sets with marginal returns up to T_P^2. Hence, given a target number of bits for encoding the image, we perform the first run with the smallest number of integral passes, P, such that the number of generated bits just exceeds the target number of bits. The generated significance map is pruned with $\lambda = T_P^2$. Next, we perform the second run of SPIHT with P passes. If the target number of bits can be achieved we stop. Otherwise, we increment P and repeat the above procedure for the new P starting with the first run. This way the final bitstream can achieve the target number of bits by employing the maximal value for λ which is optimal for the $P-1$th refinement pass.

3.2. Joint coding of coefficient signs and significances

As stated in [20], adjacent coefficients (of the same subband) in the low-pass filtering direction tend to have positively correlated signs and adjacent coefficients (of the same subband) in the high-pass filtering direction tend to have negatively correlated signs. These correlation properties may be exploited by joint coding the signs as well as significances of a 2×2 block of coefficients.

Each previously insignificant coefficient of the horizontally and vertically oriented subbands (i.e. LH,HL,LLLH,…) can be classified as one of insignificant, or positively signed significant, or negatively signed significant. The sign/significance classes for the previously insignificant coefficients of the 2×2 block may be mapped by means of a predetermined scan order to a single symbol that is arithmetic coded. There can be 3^m symbols for m previously insignificant coefficients in a 2×2 block. Of these 3^m symbols, those corresponding to the above mentioned positive or negative correlations between signs tend to occur more frequently.

Joint arithmetic coding of coefficient signs and significances is applied only in the horizontally and the vertically oriented subbands. The coefficient signs are independently fixed length coded in the diagonally oriented subbands. The scan orders for the horizontally and vertically oriented subbands are depicted in Fig. 2. The first coefficient scanned in each block contributes 2×3^m, 3^m or 0 to the symbol value and last coefficient visited contributes 2, 1 or 0 to the symbol value depending on class. These scan orders are used for $m \in \{1, 2, 4\}$. For $m = 3$ the scan order used is different. In this case, the significance and sign information of the coefficient which is diagonally opposite the previously significant coefficient is coded last.

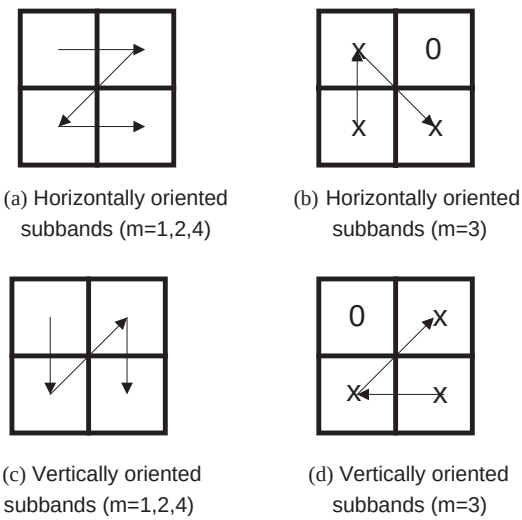
As depicted in Fig. 3, the $m' = 4 - m$ previously significant coefficients designate the context for conditioning the adaptive arithmetic coder in the following manner: One arithmetic model (Model 0) is used when $m' = 0$. When $m' = 1$, one arithmetic model (Model 1) is used for all

$$\binom{4}{m'} = 4$$

possible ways of finding one significant coefficient in the 2×2 block. When $m' = 2$ there exist

$$\binom{4}{m'} = 6$$

unique ways of finding two significant coefficients in the 2×2 block. In this case, there is no need to distinguish between the two possibilities each of finding the two significant coefficients as vertical neighbors,



(a) Horizontally oriented subbands (m=1,2,4)

(b) Horizontally oriented subbands (m=3)

(c) Vertically oriented subbands (m=1,2,4)

(d) Vertically oriented subbands (m=3)

Fig. 2. Scan orders used for mapping coefficient sign/significance classes to a single symbol. For horizontally oriented subbands, the scan order in (a) is used when $m = 1, 2, 4$. The sign and significance of the first visited coefficient determines the most significant ternary digit of the coded symbol. The sign and significance of the last visited coefficient determines the least significant ternary digit of the coded symbol. For example, when $m = 2$ and the upper left and lower right coefficients are previously insignificant, the upper right coefficient yields the most significant ternary digit and the lower left coefficient yields the least significant ternary digit. With the convention that insignificance is assigned 2, $+$ve significance is assigned 1, and $-$ve significance is assigned 0, if the upper right coefficient is deemed $+$ve significant and the lower left one insignificant, then the coded symbol will be $(12)_3$ in this case. For $m = 3$, the scan order is slightly changed to allow the coefficient diagonally opposite the previously significant coefficient to yield the most significant ternary digit, as shown in (b). For vertically oriented subbands, the top-right and bottom-left coefficients are interchanged prior to using scan order in (c) when $m = 1, 2, 4$, and using scan order in (d) when $m = 3$.

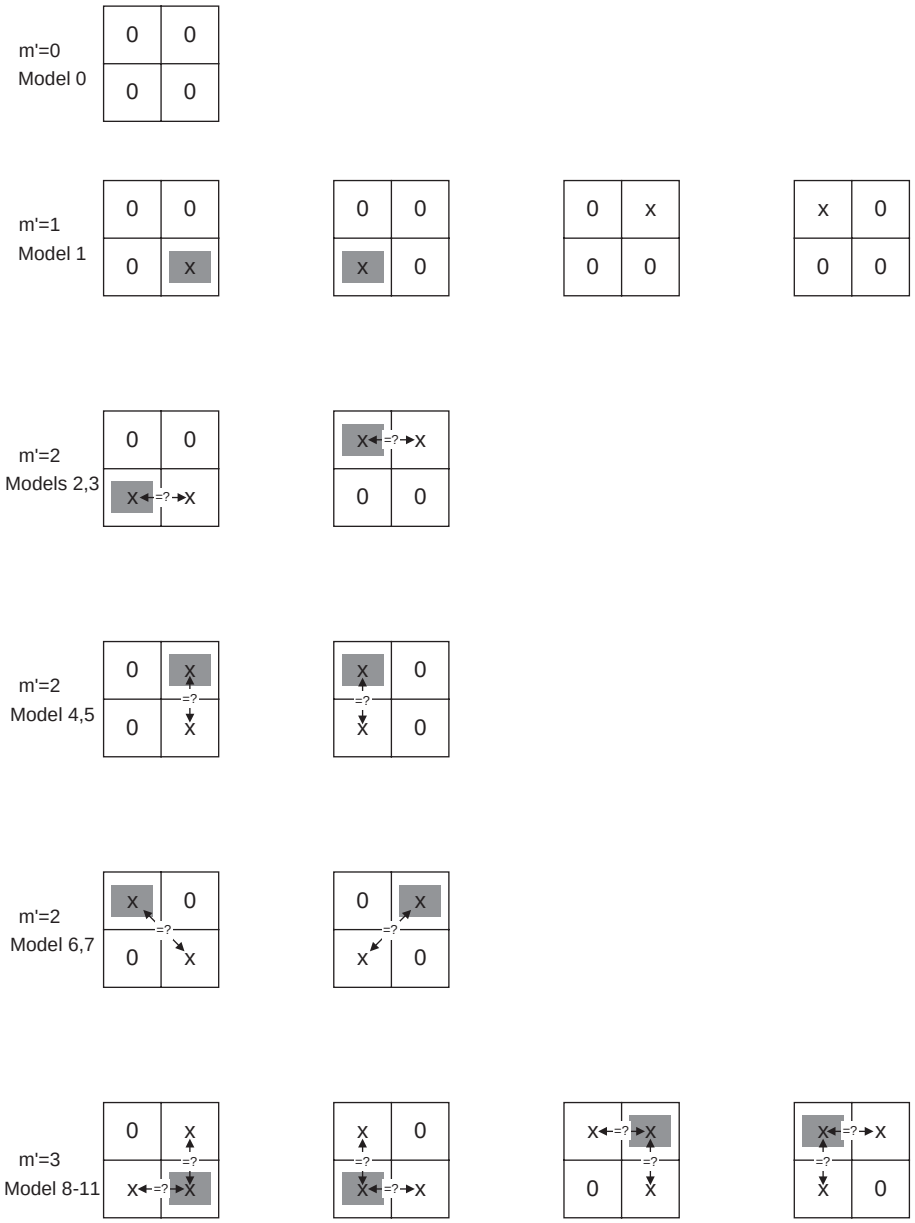


Fig. 3. In a 2×2 block of coefficients, $m' = 4 - m$ represents the number of previously significant coefficients. Each previously significant coefficient is shown as "x". The remaining coefficients, each of which is shown as "0", are coded for significance and sign by employing 3^m symbols. The configurations on the same row are coded using one of the arithmetic models indicated on the left. The exact arithmetic model used is based on the knowledge of the signs of pair(s) of coefficients (each pair indicated by arrows) matching or not matching. For $m' = 1, 2, 3$ the sign of the shaded coefficient is taken as a reference (i.e. if a coefficient deemed significant in the current pass has a sign that matches the shaded coefficient's sign the ternary digit encoded into the symbol is 1, otherwise it is 0).

horizontal neighbors or diagonal neighbors in the 2×2 block. However, since one must distinguish between the two possibilities of the signs of the two significant coefficients matching and not matching, 6 arithmetic models (Models 2–7) are used when $m' = 2$. Finally, for $m' = 3$ there exist $\binom{4}{m'} = 4$ possible ways of finding three significant coefficients in the 2×2 block. Again there is no need to distinguish between these ways, but one must distinguish among the four possibilities of the signs of the vertically adjacent coefficients matching and not matching, and the signs of the horizontally adjacent coefficients matching and not matching. Hence, when $m' = 3$, the total number of arithmetic models used is 4 (Models 8–11).

When $m' = 0$, the signs of the coefficients deemed significant in the current pass are coded by taking the positive sign as reference. When $m' = 1$, the signs of the coefficients deemed significant in the current pass are coded by taking the sign of the single previously significant coefficient as reference. When $m' = 2$, the signs of the coefficients deemed significant in the current pass are coded by taking as reference the sign of the previously significant coefficient which is first encountered in the scan order. Finally, when $m' = 3$, the sign of the coefficient deemed significant in the current pass is coded by taking as reference the sign of the previously significant coefficient which is diagonally opposite.

3.3. Different arithmetic models for Type B sets rooted at different pyramid levels

Since, in general, the entropy of a random variable upper bounds its conditional entropy, the entropy of the significance decisions for the Type B set $L(i,j)$ can be expected to decrease when conditioned on the level of the SOT $l(i,j)$ at which the coordinate pair (i,j) is located (Type B set is rooted), i.e.

$$H(S_p(L(i,j))) \geq H(S_p(L(i,j))|l(i,j)).$$

We have determined that the statistical dependence between $l(i,j)$ and $S_p(L(i,j))$, the symbol representing the significance decision for the Type B set, is strong enough and can be exploited by the use of a separate arithmetic model for each level of the SOT. The increase in overhead due to the initialization of more than one arithmetic model is more than offset by the reduction in entropy (due to conditioning) at moderate to high coding rates.

To justify the claim of strong statistical dependence, we show in Fig. 4 histograms of the relative significance/insignificance frequencies of Type B sets rooted at each of the six levels of the SOT at the end of the 10th pass for the Lenna 512×512 monochrome image. Type B sets rooted at higher levels (small level index in Fig. 4) of the SOT are more likely to be significant.

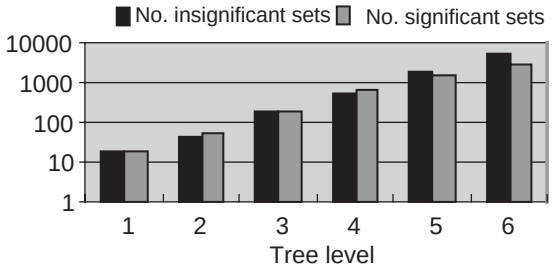


Fig. 4. Number of significant and insignificant Type B sets rooted at each of the six levels of the SOT at the end of 10 passes on Lenna512. Sets rooted at the higher levels (Levels 1–4) are more likely to be significant than those rooted at the lower ones (Levels 5–6).

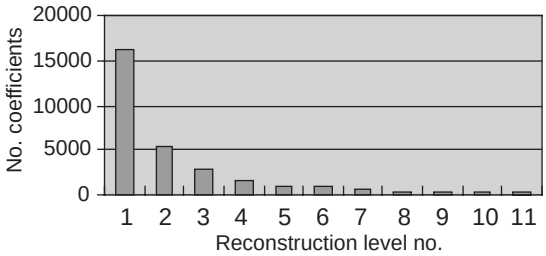


Fig. 5. Histogram of the number of coefficients getting mapped to the 11 lowest magnitude nonzero reconstruction levels at the end of 10th pass for Lenna512. (Reconstruction level i has magnitude $(i+0.5)T_p$.)

3.4. Exploitation of the nonuniformity in the probability mass function of the successive approximation quantizer by arithmetic coding

The probability mass function of the wavelet coefficients peaks at zero magnitude and declines with magnitude. The probabilities of the output levels of the successive approximation quantizer used in the refinement pass are therefore unequal. To support this claim, the frequencies of the lowest reconstructed coefficient magnitudes for the Lenna512 image at the end of the 10th pass are shown in Fig. 5. If we examine an adjacent pair of magnitude frequencies we see that there is a bias towards the larger magnitude. Since the successive approximation quantizer of the last pass distinguishes between the two magnitudes of each pair, the output of the successive approximation quantizer is biased towards the level corresponding to the larger magnitude.

In order to take advantage of this, the bi-level output of the successive approximation quantizer may be arithmetic coded. Specifically as shown in Fig. 6, if the sign of the reconstructed coefficient is positive, Symbol 0 is coded to refine the reconstruction value by adding $-T_p/2$ and Symbol 1 is coded to refine the reconstruction value by adding $T_p/2$. If the sign of the reconstructed coefficient is negative, Symbol 0 is coded to refine the reconstruction value by adding $T_p/2$ and Symbol 1 is coded to refine the reconstruction value by adding $-T_p/2$. This ensures that the high probability level is coded with Symbol 0 for a negative refinement and the low probability level is coded with Symbol 1 for a positive refinement in magnitude.

Successive approximation quantization is a simple way of implementing uniform quantization. When the output indices of a uniform quantizer are entropy coded, the rate-distortion performance is equivalent to the rate distortion performance of an optimal quantizer even at low rates [5]. Therefore, one cannot expect to achieve significantly better scalar quantization performance by increasing the complexity of successive approximation quantization beyond the entropy coding of the quantizer output indices.

4. Experimental results

This section presents the coding results obtained with the SPIHT algorithm incorporating the proposed enhancements mentioned in the previous section and compares them with the corresponding coding results obtained with the original SPIHT algorithm. Coding tests have been performed on the monochrome, 8 bpp, 512×512 'Lenna', 'Barbara' and 'Goldhill' images. A 6-level pyramidal subband decomposition with 9/7 tap biorthogonal filters and mirror extension at the edges has been employed as in [17]. The bit-rates are calculated from the actual size of the compressed files. Distortion is computed from reconstructed pixel intensity values at 8 bpp precision.

Table 1 compares the PSNR vs. Rate performance of the original SPIHT algorithm with the original SPIHT algorithm incorporating the significance map pruning method of Section 3.1 for the three still

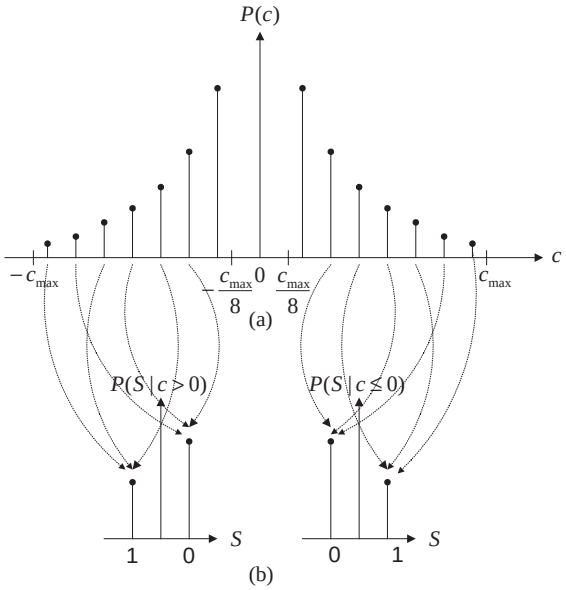


Fig. 6. (a) Illustration of the probability mass function for the reconstructed coefficients at the end of 3 passes ($p = 3$). Probabilities decline with magnitude. (b) Symbols are assigned to the output of the bilevel quantizer used in the refinement pass based on the sign of the reconstruction values (S:Symbol). High probability level is coded with $S = 0$ for a negative and low probability level is coded with $S = 1$ for a positive refinement in magnitude.

Table 1
PSNR vs. Rate for original SPIHT and original SPIHT with the enhancement of Section 3.1 (binary coding)

	Number of passes	Rate	MSE	
			Original SPIHT	Original SPIHT + 3.1
Lenna512	11	1.0	39.98	40.02
	10	0.5	36.84	36.87
	9	0.25	33.70	33.73
	8	0.125	30.72	30.76
Barbara	10	1.0	36.42	36.46
	9	0.5	31.25	31.27
	8	0.25	27.31	27.35
	7	0.125	24.62	24.98
Goldhill	10	1.0	36.00	36.10
	9	0.5	32.71	32.76
	9	0.25	30.22	30.23
	8	0.125	28.27	28.30

images. The results in this table were obtained by binary coding of significance symbols and coefficient signs. As described in the last paragraph of Section 3.1, the SPIHT algorithm executes an integral number of passes, P, reported in column 2 in the first run such that when the significance map is pruned with the

rate-distortion threshold $\lambda = T_P^2$, the target number of bits is just exceeded. By observing the pruning decisions, the actual bitstream is generated in the second run up to the bit-rate reported in column 3. For the most part, we observe a gain of about 0.04 dB. However, there are cases for which the improvement is as much as 0.36 dB. Table 2 compares the PSNR vs. Rate performance of the original SPIHT algorithm with the original SPIHT algorithm incorporating the enhancements of Section 3.2–3.4 for the three still images. Since all of these three enhancements utilize the arithmetic coding tool, binary coding was not an option to consider here. In this table, the PSNR gain with the enhancements of Section 3.2–3.4 is seen to increase by increasing rate. There is very little or no PSNR gain at very low rates since all the enhancements increase the number of arithmetic models. The initial distribution of the symbol frequencies in an arithmetic model is uniform, and the first few bits are inefficiently coded with each arithmetic model. For example, with the enhancement of Section 3.4, as many as $3^4 = 81$ arithmetic models are employed when all 4 coefficients of a 2×2 block are previously insignificant and need to be jointly coded. The original SPIHT algorithm employs only $2^4 = 16$ arithmetic models in this case. This enhancement can therefore yield an advantage only at moderate to high rates (> 0.25 bpp).

Table 3 compares the PSNR vs. Rate performance of the original SPIHT algorithm with the original SPIHT algorithm incorporating the enhancements of Section 3.1–3.4 combined. The results in this table were obtained by arithmetic coding of significance symbols and coefficient signs. Note that even though the estimates of ΔR and $\tilde{R}_{D(m,n)}$ terms are not exact (estimates based on binary coding), a comparison with Table 1 results reveals a performance gain similar to that obtained for binary coding significance symbols and coefficient signs.

Original 'Barbara' image is shown as the top-left image in Fig. 7. A region of interest of high detail is demarked by a grey box and a blow up of this region is shown in the top-right image of Fig. 7. The reconstruction of this region with the original SPIHT algorithm, and with the SPIHT algorithm incorporating all four enhancements are shown in the bottom-left and bottom-right images, respectively, of Fig. 7. Although not conspicuous at first glance, there are at least three areas of detail in this region that are better reconstructed with all four enhancements turned on.

Table 2
PSNR vs. Rate for original SPIHT and original SPIHT with the enhancements of Sections 3.2–3.4 incorporated one at a time (arithmetic coding)

	Rate (bpp)	PSNR			
		Original	3.2	3.2 and 3.3	3.2–3.4
Lenna512	0.0625	28.38	28.39	28.40	28.32
	0.125	31.10	31.11	31.12	31.10
	0.25	34.12	34.15	34.15	34.17
	0.5	37.21	37.24	37.25	37.28
	1.0	40.41	40.45	40.45	40.47
Barbara	0.0625	23.38	23.39	23.40	23.36
	0.125	24.93	24.93	24.94	24.92
	0.25	27.65	27.67	27.67	27.66
	0.5	31.69	31.78	31.79	31.79
	1.0	36.91	37.00	37.01	37.07
Goldhill	0.0625	26.73	26.74	26.74	26.73
	0.125	28.48	28.50	28.50	28.51
	0.25	30.56	30.59	30.60	30.62
	0.5	33.13	33.15	33.16	33.21
	1.0	36.55	36.59	36.59	36.65

Table 3
PSNR vs. Rate for original SPIHT and original SPIHT with the enhancements of Sections 3.1–3.4 combined (arithmetic coding)

	Number of passes	Rate	MSE	
			Original SPIHT	Original SPIHT with 3.1–3.4
Lenna512	11	1.0	40.41	40.50
	10	0.5	37.21	37.34
	9	0.25	34.12	34.24
	8	0.125	31.10	31.17
Barbara	10	1.0	36.91	37.22
	9	0.5	31.69	31.83
	8	0.25	27.65	27.72
	7	0.125	24.93	24.93
Goldhill	10	1.0	36.55	36.75
	9	0.25	30.56	30.61
	9	0.5	33.13	33.22
	8	0.125	28.48	28.56

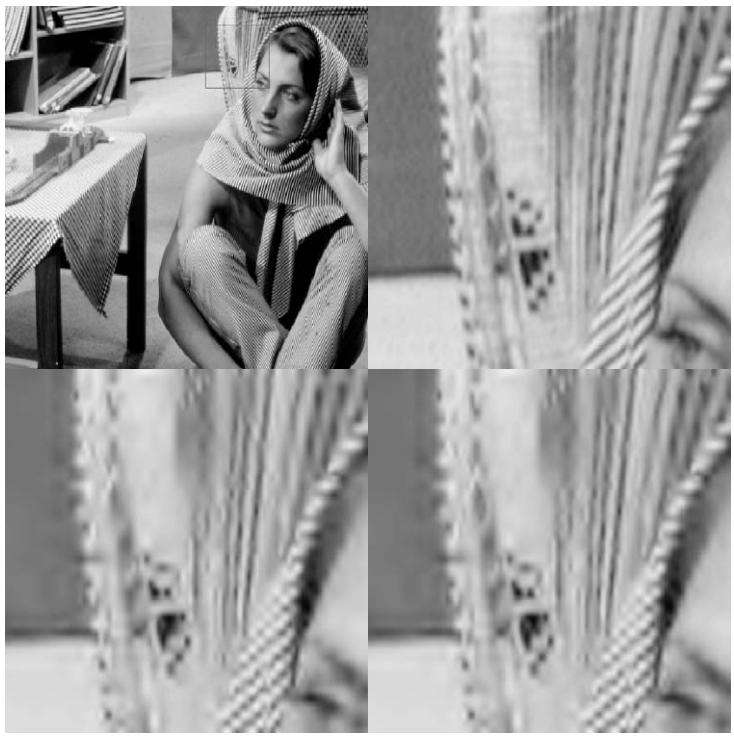


Fig. 7. Top-Left: Original Barbara and a region of interest marked as a grey box, Top-Right: Detail inside region of interest for the original image, Bottom-Left: Reconstruction with SPIHT inside region of interest (Rate = 0.5 bpp), Bottom-Right: Reconstruction with SPIHT with all enhancements turned on inside region of interest (Rate = 0.5 bpp).

5. Discussion

The R-D optimized significance map pruning method is perhaps the most controversial contribution to the original algorithm given the added complexity on the encoder side. However, we note that the decoder side complexity does not change with this enhancement since the generated bitstream can be decoded with the original SPIHT decoder. This could be useful in applications such as digital picture archiving on a web site where decoding is time critical but encoding is not. For these applications the slight increase in picture quality could be well worth the considerable complexity added to the encoding process.

Subband-Block Hierarchical Partitioning (SBHP) coder of [4,14], the low complexity coding option of JPEG2000, utilizes quadtrees in a way similar to SPIHT utilizes SOT's. We envision that the R-D optimized pruning method could be applied to the sets of SBHP (S sets of SPECK [6]) to optimize their significance decisions for the applications mentioned above.

An idea similar to joint arithmetic coding of signs and significances already exists in EBCOT. This together with the enhancement of Section 3.3 are also of no use in the SBHP framework where arithmetic coding is forgone for the sake of low computational complexity. However, EZBC [18], which integrates adaptive arithmetic coding to SPECK to achieve coding performance rivalling that of EBCOT, could benefit from the joint arithmetic coding of signs and significances, if this enhancement is adapted to its coding algorithm.

Our work also demonstrates that entropy coding of refinement bits can yield compression gains contrary to popular belief. Suitable arithmetic or Huffman coding of the refinement bits may enhance the performance of EBCOT, EZBC or SBHP coders.

6. Conclusions

In this paper, we have first introduced a method of pruning the significance maps generated by the SPIHT algorithm for improving its rate-distortion performance. In this method, some of the sets of coefficients that are deemed significant with respect to the smallest magnitude threshold used in the final pass of a first run of SPIHT algorithm are deemed insignificant if their marginal returns are below a rate-distortion threshold. We have also introduced three low complexity enhancements that exploit traits common to the real world images. Generally speaking, we have made use of the fact that the distribution of either the symbol data itself or its partitioning into subsets defined by context is largely non-uniform and can be exploited by the advanced use of arithmetic coding for a coding performance advantage at moderate to high coding rates. When combined, the enhancements yield modest rate-distortion performance gain for the test images at various bit-rates.

Although the enhancements proposed in this paper does not bring SPIHT's performance level on a par with the performance levels of some of the other wavelet-based image coding algorithms [18,1], similar performance gains might be obtained when one or more of the enhancements proposed in this paper are adapted to these algorithms. For example, the enhancements of Section 3.1,3.2 and 3.4 might be applicable to the EZBC algorithm of [18] which employs a different quadtree structured set partitioning for each subband.

Acknowledgements

The author would like to thank Prof. W.A. Pearlman of Rensselaer Polytechnic Institute who contributed to the earlier stages of this work ([2]) and provided the original SPIHT source code.

References

[1] B.A. Banister, T.R. Fischer, Quadtree classification and TCQ image coding, IEEE Trans. Circuits Syst. Video Technol. 11(1) (January 2001) 3–8.

[2] U. Bayazit, W.A. Pearlman, Algorithmic Modifications to SPIHT, in: Proceeding of the International Conference on Image Processing, Thessaloniki, Greece, 2001.

[3] P.A. Chou, T. Lookabaugh, R.M. Gray, Optimal pruning with applications to tree-structured source coding and modelling, IEEE Trans. Inform. Theory 35 (2) (March 1989) 299–315.

[4] C. Chrysafis, A. Said, A. Drukarev, A. Islam, W.A. Pearlman, SBHP—a low complexity wavelet coder, in: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2000), Istanbul, Turkey, June 5–9, 2000.

[5] N. Farvardin, J.W. Modestino, Optimum quantizer performance for a class of non-Gaussian memoryless sources, IEEE Trans. Inform. Theory IT-30(3) (May 1984) 485–497.

[6] A. Islam, W.A. Pearlman, An embedded and efficient low complexity hierarchical image coder, in: Proceedings of SPIE Conference on Visual Communications and Image Processing, San Jose, CA, 1999, pp. 294–305.

[7] R.L. Joshi, T.R. Fischer, Image subband coding using arithmetic and trellis coded quantization, IEEE Trans. Circuits Systems Video Technol. 5 (6) (1995) 515–523.

[8] B.-J. Kim, Z. Xiong, W.A. Pearlman, Low bit-rate scalable video coding with 3D set partitioning in hierarchical trees (3D SPIHT), IEEE Trans. Circuits Systems Video Technol. 10 (December 2000) 1365–1374.

[9] J. Li, S. Lei, An embedded still image coder with rate-distortion optimization, IEEE Trans. Image Process. 8 (7) (July 1999) 913–924.

[10] Z. Lu, D.Y. Kim, W.A. Pearlman, Wavelet compression of ECG signals by the set partitioning in hierarchical trees algorithm, IEEE Trans. Biomed. Eng. 47 (7) (July 2000) 849–856.

[11] D. Marpe, et al., A two layer wavelet based algorithm for efficient lossless and lossy image compression, IEEE Trans. Circuits Syst. Video Technol. 10 (7) (October 2000) 1094–1102.

[12] S.A. Martucci, I. Sodagar, T. Chiang, Y.-Q. Zhang, A zerotree wavelet video coder, IEEE Trans. Circuits Syst. Video Technol. 7 (1) (February 1997) 109–118.

[13] D. Mukherjee, S.K. Mitra, Vector set partitioning with classified successive refinement VQ for embedded wavelet image and video coding, in: Proceedings of the IEEE International Conference on Acoustic Speech & Signal Processing, Seattle, WA, May 1998, pp. 2809–2812.

[14] W.A. Pearlman, A. Islam, N. Nagaraj, A. Said, Efficient, low-complexity image coding with a set-partitioning embedded block coder, IEEE Trans. Circuits Syst. Video Technol., in press.

[15] A. Said, W.A. Pearlman, A new, fast, and efficient image codec based on set partitioning in hierarchical trees, IEEE Trans. Circuits Syst. Video Technol. 6 (3) (June 1996) 243–250.

[16] A. Said, W.A. Pearlman, An image multiresolution representation for lossless and lossy compression, IEEE Trans. Image Process. 5 (September 1996) 1303–1310.

[17] J.M. Shapiro, Embedded image coding using zeroes of wavelet coefficents, IEEE Trans. Signal Process. 41 (December 1993) 3445–3462.

[18] Shih-Ta Hsiang, J.W.Woods, Embedded image coding using zeroblocks of subband/wavelet coefficients and context modeling, MPEG-4 Workshop and Exhibition at ISCAS 2000, Geneva, Switzerland, May 2000.

[19] I. Sodagar, H.J. Lee, P. Hatrack, Y.-Q. Zhang, Scalable wavelet coding for synthetic/natural hybrid images, IEEE Trans. Circuits Syst. Video Technol. 9 (2) (March 1999) 244–254.

[20] D. Taubman, High performance scalable image compression with EBCOT, IEEE Trans. Image Process. 9 (7) (July 2000) 1158–1170.

[21] I.H. Witten, R.M. Neal, J.G. Cleary, Arithmetic coding for data compression, Commun. ACM 30 (6) (June 1987) 520–540.

[22] Z. Xiong, O. Guleryuz, M.T. Orchard, A DCT-based embedded image coder, IEEE Signal Process. Lett. 3 (November 1996) 289–290.

[23] Zhihai He, Tian-Hu Yu, S.K. Mitra, Fast image coding using blockwise binary classification and trellis coded quantization, in: Proceedings of the 34th Asilomar Conference on Signals, Systems, and Computers, October 2000.