Wavelet-Based Embedded Rate Scalable Still Image Coders: A review

Farag Ibrahim Younis Elnagahy, B. Šimák

Embedded scalable image coding algorithms based on the wavelet transform have received considerable attention lately in academia and in industry in terms of both coding algorithms and standards activity. In addition to providing a very good coding performance, the embedded coder has the property that the bit stream can be truncated at any point and still decodes a reasonably good image. In this paper we present some state-of-the-art wavelet-based embedded rate scalable still image coders. In addition, the JPEG2000 still image compression standard is presented.

Keywords: JPEG2000, embedded image coding, grayscale, color, scalable, progressive, compression, discrete wavelet transform.

1 Introduction

The objective of an image compression algorithm is to exploit the redundancy in an image such that a smaller number of bits can be used to represent the image while maintaining an “acceptable” visual quality for the decompressed image. The redundancy of an image resides in the correlation of the neighboring pixels. For a color image, there is also a correlation, which can be exploited, between the color components [1, 2]. Different applications require different data rates for the compressed images and different visual qualities for the decompressed images. In some applications when browsing is required or transmission bandwidth is limited, progressive transmission is used to send images in such a way that a low quality version of the image is transmitted first at a low data rate [3]. Gradually, additional information is transmitted to progressively refine the image. A specific coding strategy known as embedded rate scalable coding is well suited for progressive transmission [4]. In embedded coding, all the compressed data is embedded in a single bit stream. The decompression algorithm starts from the beginning of the bit stream and can terminate at any data rate. A decompressed image at that data rate can then be reconstructed. In embedded coding, any visual quality requirement can be fulfilled by transmitting the truncated portion of the bit stream. To achieve the best performance the bits that convey the most important information need to be embedded at the beginning of the compressed bit stream [4].

The remainder of this paper is organized as follows: in Section 2, the classifications of the image coder are presented. Section 3 presents the discrete wavelet transform. Section 4 presents imbedded image coding in the wavelet domain. The objective and subjective image quality measures are presented in Section 5. Section 6 provides a review of the main algorithms and recent publications in embedded scalable grayscale image compression. Embedded scalable color image compression codecs are presented in Section 7. In Section 8 the JPEG2000 still image compression standard is presented. Finally, conclusions are drawn in Section 9.

2 Image coder classifications

Image coders can be classified according to the organization of the compressed bit stream into scalable image coders

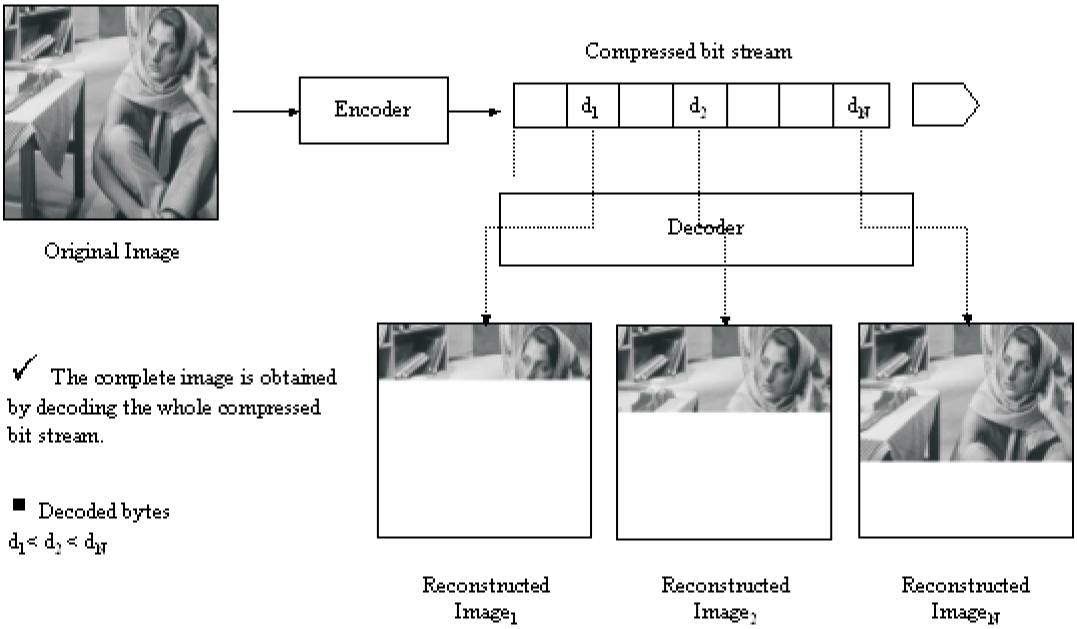


Fig. 1: Non-scalable image coder

and non-scalable image coders. In non-scalable image coders, the complete image is only obtained by decoding the entire compressed bit stream. Truncating the compressed bit stream during the decoding process will produce an incompletely reconstructed image, as shown in Fig. 1. Scalable image coders are coders that allow the image data to be compressed once and then decompressed at multiple data rates or decompressed at different resolutions of the image. Resolution refers to the size of the reconstructed image. A scalable image coder that is able to decode different resolutions of the image from the compressed bit stream is called a resolution scalable coder, while a scalable image coder that has the ability to decode a full resolution image with a certain bit rate from the compressed bit stream is called a rate (SNR) scalable coder. Both types of image scalable (resolution/SNR) coders are further classified into embedded and non-embedded image coders.

In an embedded resolution scalable image coder, lower resolution of the image is obtained by just decoding the first portion of the compressed bit stream, as shown in Fig. 2. In a non-embedded resolution scalable image coder, lower resolution of the image is obtained by decoding certain portions of the compressed bit stream, as shown in Fig. 3. In both cases no complete decoding of the whole compressed bit stream is needed. In an embedded rate (SNR) scalable image coder, a lower quality of the image is obtained by just decoding the first portion of the compressed bit stream. By decoding more and more bits we can achieve a higher quality

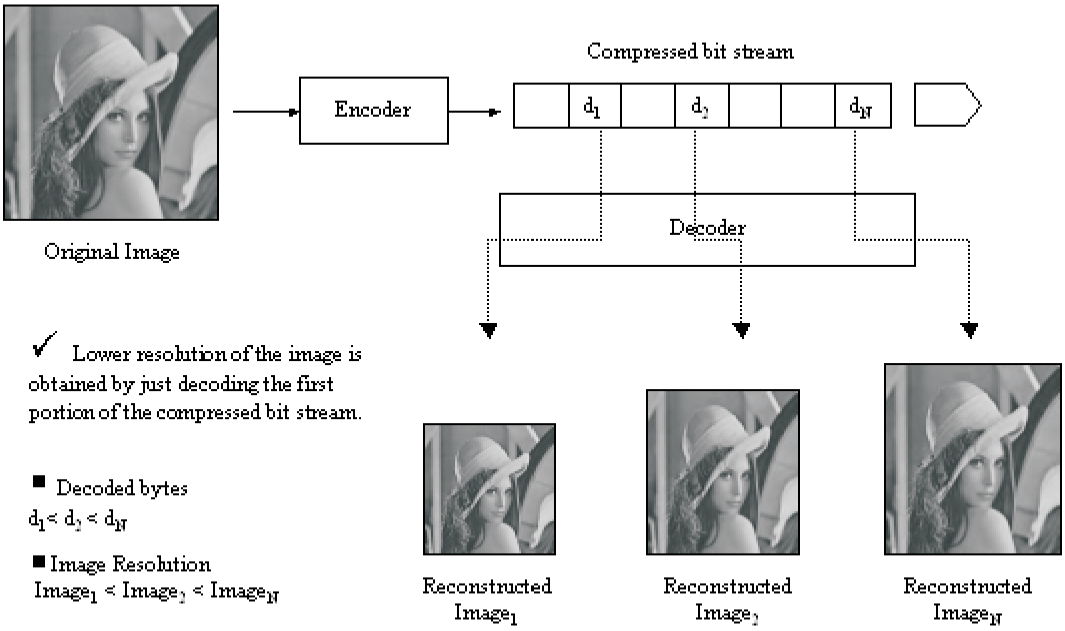


Fig. 2: Embedded resolution scalable image coder

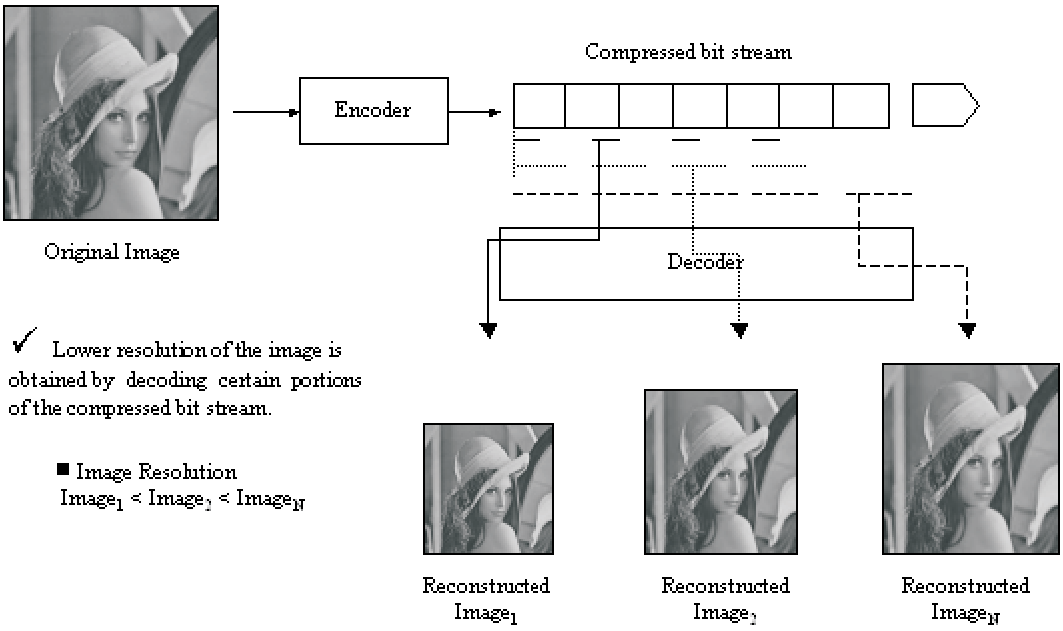


Fig. 3: Non-embedded resolution scalable image coder

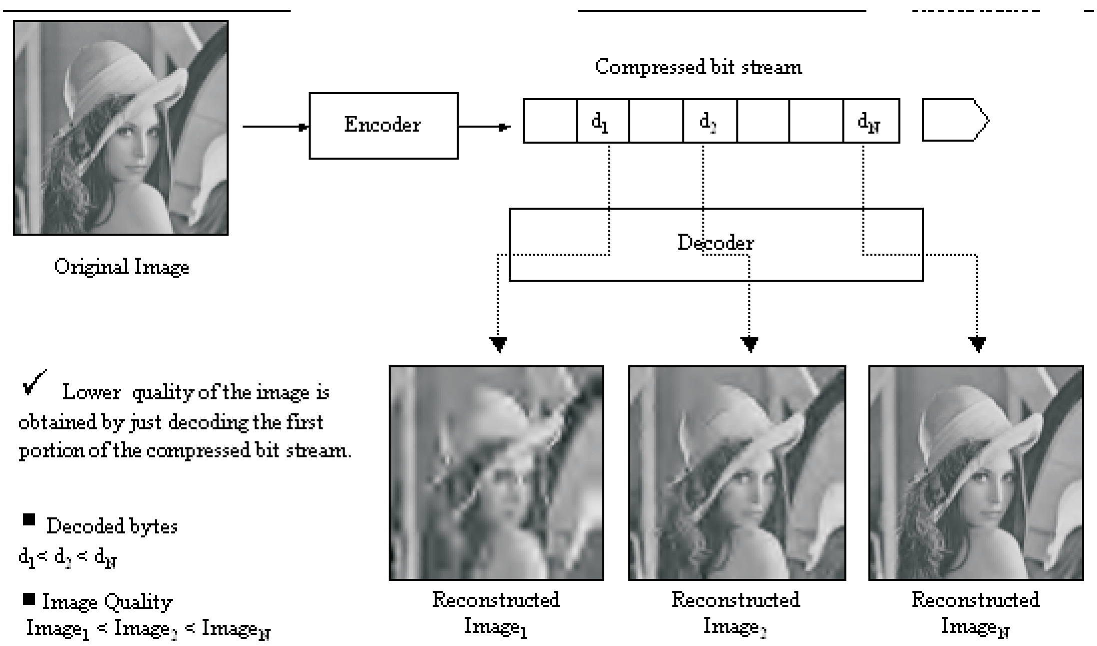


Fig. 4: Embedded SNR scalable image coder

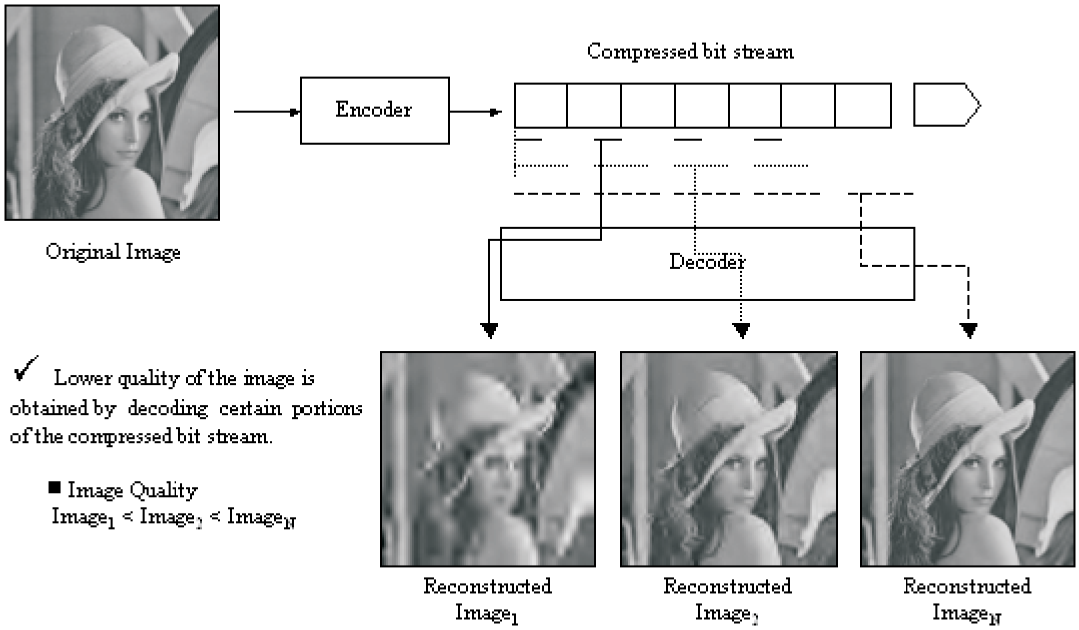


Fig. 5: Non-embedded SNR scalable image coder

image (higher SNR), as shown in Fig. 4. In non-embedded rate (SNR) scalable image coder, however, a lower quality of the image is obtained by decoding certain portions of the compressed bit stream, as shown in Fig. 5. In both cases no complete decoding of the whole compressed bit stream is needed. It is important to note that if a compression technique produces an embedded bit stream, this implies that the technique is scalable, because embedded bit streams can be truncated at any point during decoding. However, not all scalable compression techniques are embedded.

Resolution scalability can be achieved by encoding whole wavelet sub-bands one after the other, without interleaving bits from different sub-bands. SNR scalability can be achieved by distributing encoded bits from one sub-band into the whole bit stream in an optimal way. Another very interesting property of some coders is the ability to decode only certain regions of an image; this property is called random access. A coder can support the decoding of arbitrarily shaped regions or of any rectangular shaped region. Region of interest encoding and decoding is also related to random access. For region of interest encoding, only an arbitrarily shaped region of the input image is encoded, and the rest of the image, which does not fall into the region of interest, is discarded. Random access can be achieved by independently encoding portions of the whole image. All image coders can support random access by tiling the image and encoding each tile independently of the rest.

3 Discrete wavelet transform

A wavelet transform corresponds to two sets of analysis/synthesis digital filters, g/~g and h/~h, where h is a low pass filter (LPF) and g is a high pass filter (HPF). In two dimensions, filtering is usually applied in both horizontally and vertically. Filtering in one direction results in decomposing the image in two components. There is a total of four components after vertical and horizontal decompositions. We will refer to these components as image sub-bands, LL, HL, LH, and HH. By using filters g and h, an image can be decomposed into four sub-bands. This is the first level of the wavelet transform (Fig. 6). One of the four sub-bands, the LL sub-band, will contain low pass information, which is essentially a low resolution version of the image. Sub-band HL will contain low pass information vertically and high pass information horizontally, and sub-band LH will contain low pass information horizontally and high pass information vertically. Finally, sub-band HH will contain high pass information in both directions [5,6]. The operations can be repeated on the low-low (LL) sub-band for some number of levels, D, producing a total of 3-D+1 sub-bands whose samples represent the original image. The total number of samples (coefficients) in all sub-bands is identical to that in the original image. Thus, a typical 2-D discrete wavelet transform used in image processing will generate a hierarchical pyramidal structure, as shown in Fig. 7. The LL sub-band at the highest level can be classified as most important, and the other ‚details' sub-bands can be classified as of lesser importance, with the degree of importance decreasing from the top of the pyramid to the sub-bands at the bottom. The inverse wavelet transform is obtained by reversing the transform process and replacing the analysis filters by the synthesis filters and using up-sampling (Fig. 8). An example illustrating three level octave-band decomposition is shown in Fig. 9.

4 Embedded image coding

The wavelet transform can decorrelate the image pixel values and result in frequency and spatial-orientation separation. The transform coefficients in each sub-band exhibit unique statistical properties that can be used for encoding the image. The sub-band LL can be encoded alone and is added to the encoded bit stream information about the sub-bands HL, LH, HH. Then the decoder is responsible for deciding whether to decode only LL or all sub-bands. Sub-band data are usually real numbers, therefore they require a significant number of bits in their representation, and do not directly offer any form of compression. The first approach towards compressing wavelet data is to apply a quantizer in all the coefficients, where uniform and dead-zone quantizers are the most common choices. After quantization an entropy coder is applied to compress the quantization indices.

In embedded coding, a wavelet transform is usually used to decorrelate the image pixels and achieve frequency and spatial-orientation separation. Assuming that the wavelet transform $c = \Omega(p)$, where p is the collection of image pixels and c is the collection of transformed coefficients, is unitary, the distortion of the image after decompression is

$$D(p) = D(c) = \sum_i D(c_i) .$$

The distortion measure is defined as the summation of the distortion at each pixel. The greatest distortion reduction can be achieved if the transformed coefficient with the largest magnitude is coded with infinite precision. Thus attempts have been made to encode the transformed pixels with larger magnitudes first. Furthermore, in order to distribute the bits strategically such that the decoded image will look "natural" at any data rate, progressive refinement or bit-plane coding is used.

Thus in the coding procedure, multiple passes through the data are adopted. In the first pass, those transformed pixels that are larger than a pre-selected threshold are added to the significance list and coded to the precision of the threshold. In the following passes, the threshold is halved and the pixels already in the list are refined to one more bit of precision. Meanwhile more transformed pixels that are larger than the (halved) threshold are added to the list. It is critical that the positions of the enlisted pixels be encoded efficiently. It has been observed that large number of transform coefficients is insignificant in comparison with the threshold. These coefficients will be quantized to zero, which will not reduce the distortion. Thus spending more bits on coding these insignificant coefficients results in lower efficiency [7]. Imbedded image coding makes more efficient use of a communications channel, because images become visible more quickly, reducing the apparent retrieval time. Moreover, progressive transmission enables efficient browsing of image databases: based on a quickly obtained preview of an image, a user might decide to terminate the transmission and proceed to the next image. Alternatively, a user might select a low or medium resolution default for image retrieval. In this way, images of less value are not transmitted at full resolution. Because of the small volume of data required to transmit relatively high quality previews of an image, the savings in access time can be significant when browsing large databases of images.

5 Measures of image quality

Two mathematical formulas are commonly used to compare the various image compression techniques; they are the Mean Square Error (*MSE*) and the Peak Signal to Noise Ratio (*PSNR*). The *MSE* is the cumulative squared error between the compressed and the original image, whereas *PSNR* in decibels (dB) is a measure of the peak error. The mathematical formulae for the two are:

$$MSE = \frac{1}{MN} \sum_{y=1}^{M} \sum_{x=1}^{N} \left[I(x,y) - I'(x,y) \right]^2 ,$$

$$PSNR = 10 \log \left(\frac{255^2}{MSE} \right) .$$

Where $I(x, y)$ is the original image, $I'(x, y)$ is the approximated version (which is actually the decompressed image) and M, N are the dimensions of the images. A lower value for *MSE* means less error, and as seen from the inverse relation between *MSE* and *PSNR*, this translates to a high value of *PSNR*. Logically, a higher value of *PSNR* is good because it means that the ratio of signal-to-noise is higher. Here, the 'signal' is the original image, and the 'noise' is the error in

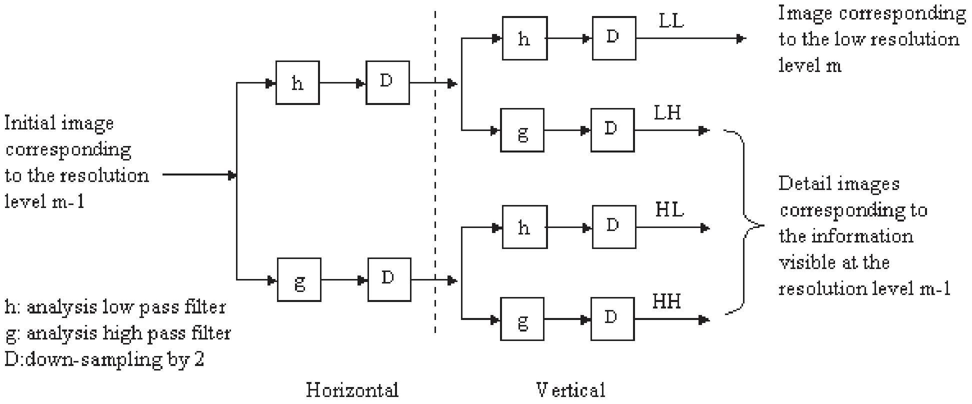
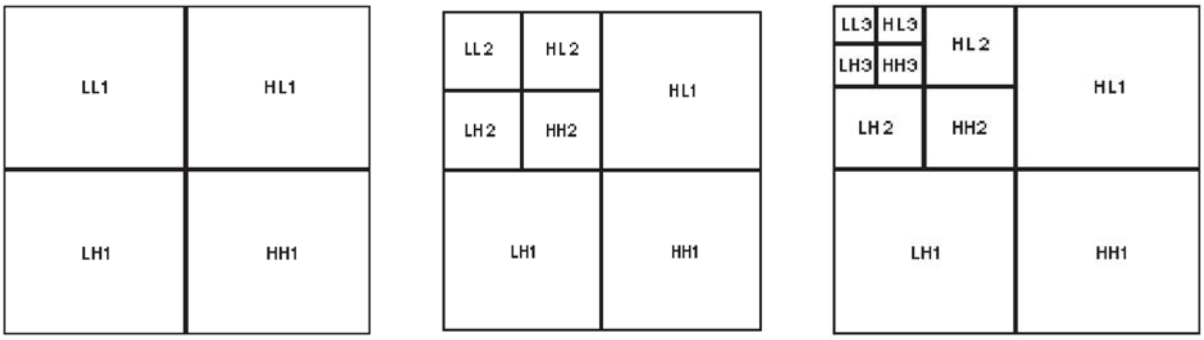


Fig. 6: One level of wavelet transform decomposition

reconstruction. The nominator (255) in formula 2 is the maximum decimal value of an unsigned 8-bit image. A high *PSNR* value does not always correspond to an image with perceptually high quality. Another measure of image quality is the mean opinion score, where the performance of a compression process is characterized by the subjective quality of the decoded image. For example, a five-point scale such as very annoying, annoying, slightly annoying, perceptible but not annoying, and imperceptible might be used to characterize the impairments in the decoder output [8]. More details about the performance of image quality measures can be found in [9].



(a) Single level decomposition

(b) Two levels decomposition

(c) Three levels decomposition

Fig. 7: Pyramidal decomposition of an image

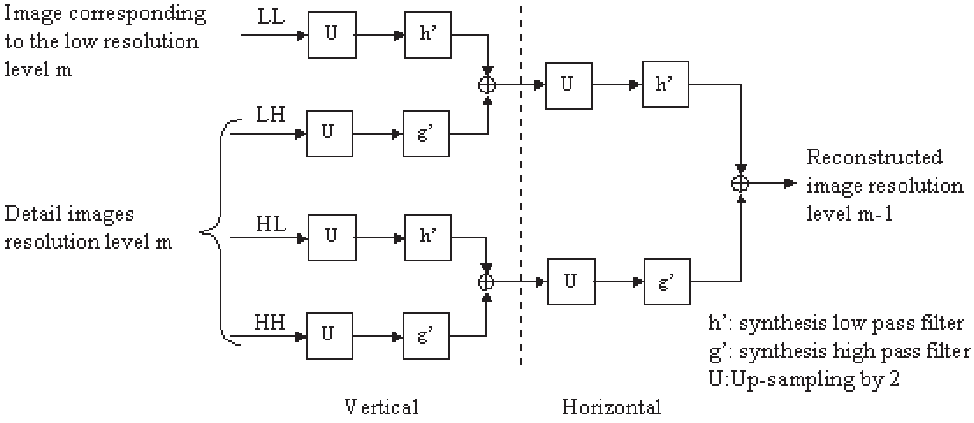
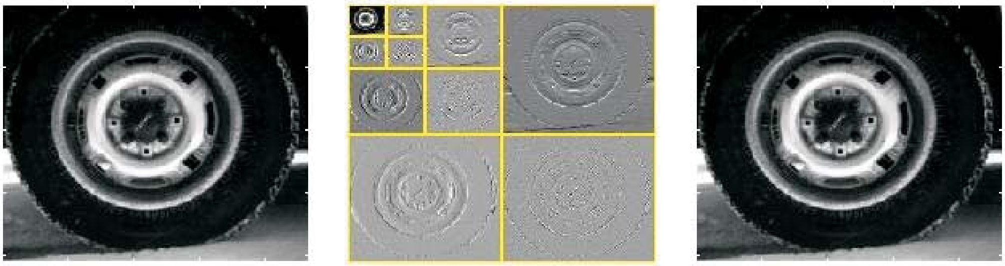


Fig. 8: One level of the wavelet transform reconstruction



(a) Original image

(b) Three level octave-band decomposition

(c) Synthesized image

Fig. 9: An example of wavelet decomposition

6 Wavelet-based embedded rate scalable grayscale image compression algorithms

Several techniques have been proposed to achieve rate scalability in still image compression. Two of the most important techniques are Shapiro's Embedded Zerotree Wavelet (EZW) [4], and Said and Pearlman's Set Partitioning in Hierarchical Trees (SPIHT) [10]. Both make use of "spatial orientation trees" (SOT). Spatial orientation trees are structures that use quad-tree representations of sets of wavelet coefficients that belong to different sub-bands, but have the same spatial location. This structure, which can be efficiently represented by one symbol, has been used extensively in rate scalable image and video compression. EZW, SPIHT, and other recent works are described in the following sub-sections.

6.1 Embedded zerotree wavelet (EZW) algorithm [4]

The EZW algorithm is based on four key concepts:

- A discrete wavelet transform or hierarchical sub-band decomposition.
- Prediction of the absence of significant information across scales by exploiting the self-similarity inherent in images.
- Entropy-coded successive approximation quantization.
- Universal lossless data compression, which is achieved via adaptive arithmetic coding [11].

The Embedded zerotree wavelet (EZW) algorithm exploits the interdependence between the coefficients of the wavelet decomposition of an image by grouping them into SOTs. This is based on the following observations:

- Most of the energy is concentrated at the low frequency sub-bands of the wavelet decomposition.
- If the magnitude of a wavelet coefficient in the lowest sub-band of a decomposition is insignificant with respect to a threshold, then it is more likely that wavelet coefficients having the same spatial location in different sub-bands will also be insignificant.
- The standard deviation of the wavelet coefficients decreases when proceeding from the lowest to the highest sub-bands of the wavelet pyramid.

The combination of the above observations allows for the coding of a large number of insignificant wavelet coefficients by coding only the location of the root coefficient to which the entire set of coefficients is related. Such a set is commonly referred to as a "zerotree". The SOT used in EZW is shown in Fig. 10. The root of the tree is the coefficient located in the lowest sub-band (LL) of the decomposition. Its descendants are all the other coefficients in the tree. The wavelet coefficient in the LL sub-band has three children. Other coefficients, except for those in the highest frequency sub-bands, have four children.

The EZW algorithm consists of successive approximation of wavelet coefficients, and can be summarized as follows:

Let $f(m; n)$ be a grayscale image, and let $W[f(m; n)]$ be the coefficients of its wavelet decomposition.

1) Set the threshold $T = \max\left(W\left[f(m;n)\right]\right)/2$
2) *Dominant pass*:

- Compare the magnitude of each wavelet coefficient in a tree, starting with the root, to threshold T.
- If the magnitudes of all the wavelet coefficients in the tree are smaller than threshold T, then the entire tree structure (that is, the root and all its descendants) is represented by one symbol. This symbol is known as the Zerotree (ZTR) symbol.
- Otherwise, the root is said to be "significant" (when its magnitude is greater than T), or "insignificant" (when its magnitude is less than T). A significant coefficient is represented by one of two symbols, POS or NEG, depending on whether its value is positive or negative. The magnitude of significant coefficients is set to zero to facilitate the formation of Zerotree structures. An insignificant coefficient is represented by the symbol IZ "isolated zero" if it has some significant descendant.
- This process is carried out such that all the coefficients in the tree are examined for possible sub-zerotree structures.

3) *Subordinate pass*: The significant wavelet coefficients in the image are refined by determining whether their magnitudes lie within the interval $[T; 3T/2)$, represented by the symbol LOW, or the interval $[3T/2; 2T)$, represented by the symbol HIGH.
4) Set $T = T/2$, and go to Step 2. Only the coefficients that have not yet been found to be significant are examined.

This coding strategy is iterated until the target data rate is achieved. The order in which the coefficients are examined is

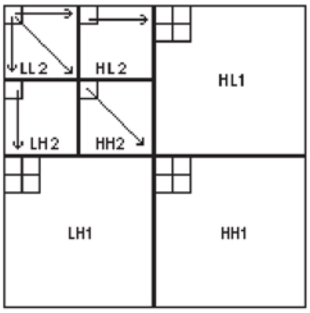


Fig. 10: Parent-descendant relationships in the spatial-orientation tree of the EZW algorithm

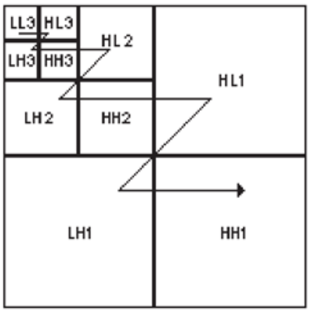


Fig. 11: Scanning order of the sub-bands for encoding a significance map (EZW)

predefined and known by the decoder, as shown in Fig. 11. The initial threshold is encoded in the header of the bit stream, followed by the resulting symbols from the dominant and subordinate passes, which are entropy coded using an arithmetic coder [11]. An important element of EZW is the “significance map". The significance map is used to code the position of the significant coefficients in the wavelet decomposition. The simplest significance map is that where a significant coefficient is accompanied by the actual coordinates of its location. In EZW, the significance map is determined by having both the encoder and the decoder know the scanning order of the coefficients. In essence, the dominant pass of EZW determines of the significance map, whereas the subordinate pass refines the wavelet coefficients that have been found significant. In consequence, the reduction of distortion during the dominant pass is smaller than during the subordinate pass. EZW was developed for grayscale images. However it has been used for color images by using EZW on each of the color components separately.

6.2 Set partitioning in hierarchical trees (SPIHT) [10]

Said and Pearlman [10] investigated different tree structures that improved the quality of the decomposition. Using Set Partitioning in Hierarchical Trees (SPIHT), the coefficients are divided into partitioning subsets using a known ordering in the SOT. Then, each subset is classified as significant or insignificant according to a pre-specified rule based on the magnitude of the coefficients in the subset. In the SOT, the descendants (a group of 2×2 adjacent coefficients) correspond to the pixels of the same spatial orientation in the next finer level of the decomposition. Fig. 12 shows how the spatial orientation tree is defined in a pyramid constructed with recursive four-sub-band splitting. One wavelet coefficient in the LL sub-bands (noted with “*”) does not have a child. The other coefficients, except for those in the highest frequency sub-bands, have four children. In contrast to Shapiro’s Zerotree, one coefficient in each group does not have descendants. SPIHT groups the wavelet coefficients into three lists: a list of insignificant sets (LIS), a list of insignificant pixels (LIP), and a list of significant pixels (LSP). The SPIHT algorithm can be summarized as follows:

1) Initialize the LIS to the set of sub-tree descendants of the nodes in the highest level, the LIP to the nodes in the highest level, and the LSP to an empty list.
2) *Sorting pass*:

- Traverse the LIP testing the magnitude of its elements against the current threshold and representing its significance by 0 or 1. If a coefficient is found to be significant, its sign is coded and is moved to the LSP.
- Examine the LIS and check the magnitude of all the coefficients in that set. If a particular set is found to be significant, it is then partitioned into subsets and tested for significance. Otherwise, a single bit is appended to the bit stream to indicate an insignificant set.

3) *Refinement pass*: Examine the LSP, excluding coefficients added during the sorting pass. This pass is accomplished by using progressive bit plane coding of the ordered coefficients.
4) Set $T = T/2$, and go to Step 2.

The process is repeated until the target data rate is achieved. It is important to note that the locations of the coefficients being refined or classified are never explicitly transmitted. This is because all branching decisions made by the encoder as it searches throughout the coefficients are appended to the bit stream. The output of the sorting-refinement process is then entropy coded [11].

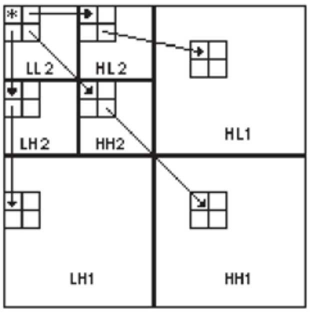


Fig. 12: Parent-descendant relationships in the spatial-orientation tree in the SPIHT algorithm

6.3 Space frequency quantization (SFQ) algorithm [12]

In [12], a wavelet-based image compression scheme is presented. This iterative algorithm seeks to minimize the distortion measure (*MSE*) by successively pruning a tree for a given target rate. The survivor nodes in the tree are quantized and sent in the data stream along with significance map infor-

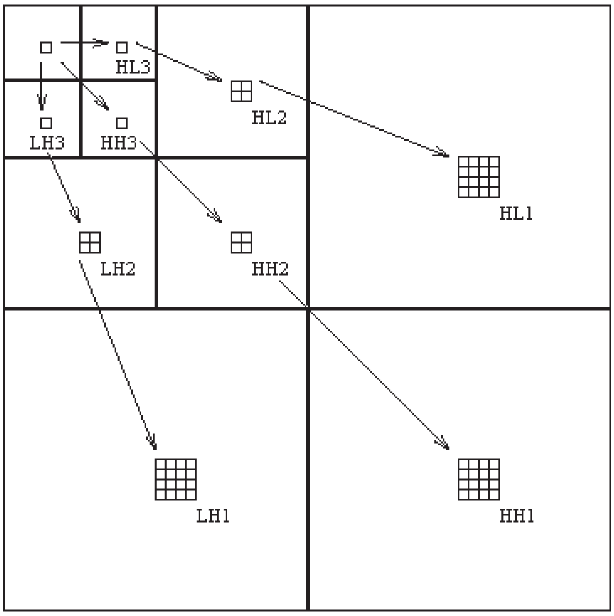


Fig. 13: SFQ spatial wavelet coefficient tree

mation. To further compress the stream, this map is predicted on the basis of statistical information. The spatial wavelet coefficient tree structure used in the SFQ algorithm is shown in Fig. 13. A spatial wavelet coefficient tree is defined as the set of coefficients from different sub-bands that represent the same spatial region in the image. The arrows in Fig. 13 identify the parent-children dependencies in a tree. The lowest frequency sub-band of the decomposition is represented by the root nodes (top) of the tree, the highest frequency sub-bands by the leaf nodes (bottom) of the tree, and each parent node represents a lower frequency component than its children. Except for a root node, which has only three children nodes, each parent node has four children nodes, the 2×2 region of the same spatial location in the immediately higher frequency sub-band. The SFQ coder has the goal of jointly finding the best combination of spatial zerotree quantization choice and the scalar frequency quantizer choice. The block diagram of this coder is shown in Fig. 14 (assuming a 2-level wavelet decomposition). The SFQ paradigm is conceptually simple: throw away, i.e. quantize to zero, a subset of the wavelet coefficients, and use a single simple uniform scalar quantizer on the rest. All nine possible complementary subsets for each depth-2 spatial wavelet coefficient tree are listed in the spatial zerotree quantization block, and there are three possible scalar quantizer choices in the frequency scalar quantization block. The results show a gain of about 0.5 dB in PSNR versus SPIHT for grayscale images.

6.4 Multi-scale zerotree wavelet entropy (MZTE) coding algorithm [13]

This paper describes the texture representation scheme adopted for MPEG-4 synthetic/natural hybrid coding (SNHC) of texture maps and images. The scheme is based on the concept of the multiscale zerotree wavelet entropy (MZTE) coding technique. MZTE was rated as one of the top five schemes in terms of compression efficiency in JPEG2000 November 1997, from among 27 submitted proposals. This scheme provides many levels of scalability layers in terms of either spatial resolution or picture quality. The MZTE coding technique is based on zerotree entropy (ZTE) coding, but it utilizes a new framework to improve and extend the ZTE method to achieve a fully scalable yet very efficient coding technique. In this scheme, the low-low sub-band is separately encoded. To achieve a wide range of scalability levels efficiently, as needed by the application, the other sub-bands are encoded using the multi-scale zerotree entropy coding scheme. This multi-scale scheme provides a very flexible approach to support the right tradeoff between layers and types of scalability, complexity, and coding efficiency for any

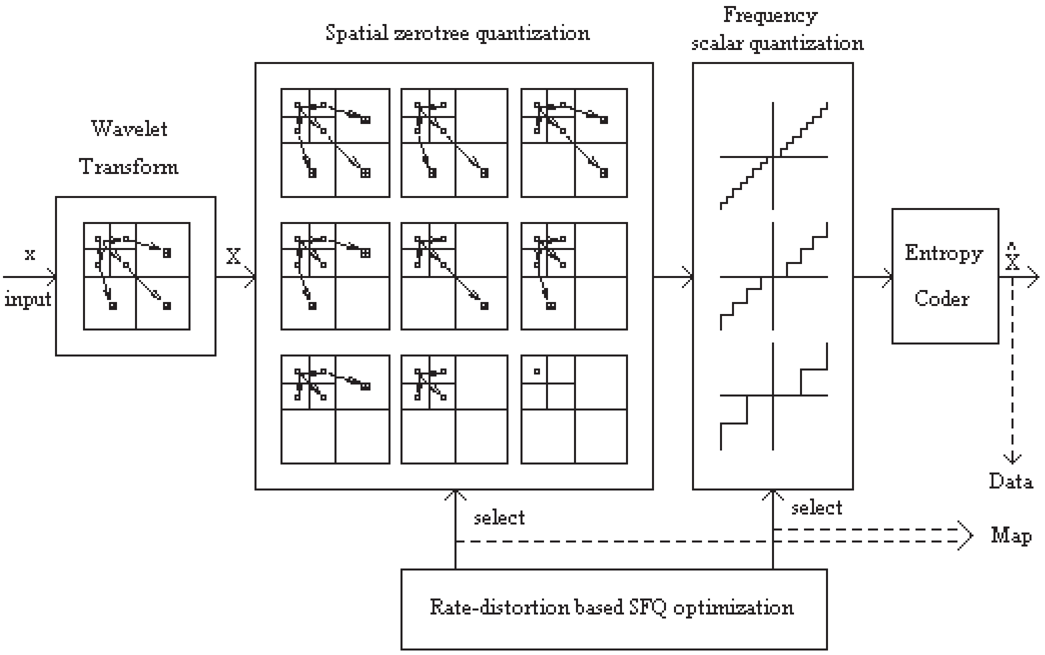

Spatial zerotree quantization
Frequency scalar quantization
Wavelet Transform
x
input
X
Entropy Coder
X̂
Data
select
select
Map
Rate-distortion based SFQ optimization

Fig. 14: Block diagram of the SFQ coder

multimedia application. Fig. 15 shows the concept of this technique.

The wavelet coefficients of the first spatial (and/or quality) layer are first quantized with the quantizer Q_0. These quantized coefficients are scanned using the zerotree concept, and then the significant maps and quantized coefficients are entropy coded. The output of the entropy coder at this level, BS0, is the first portion of the bitstream. The quantized wavelet coefficients of the first layer are also reconstructed and subtracted from the original wavelet coefficients. These residual wavelet coefficients are fed into the second stage of the coder, in which the wavelet coefficients are quantized with Q_1, zerotree scanned, and entropy coded. The output of this stage, BS1, is the second portion of the output bitstream. The quantized coefficients of the second stage are also reconstructed and subtracted from the original coefficients. As shown in Fig. 15, $N+1$ stages of the scheme provide $N+1$ layers of scalability. Each level represents one layer of SNR quality, spatial scalability, or a combination of both. In MZTE, the wavelet coefficients are scanned in either the tree-depth scanning for each scalability layer or in the sub-band by sub-band fashion, from the lowest to the highest frequency sub-bands. The wavelet coefficients are quantized by a uniform and midstep quantizer with a dead zone equal to the quantization step size as closely as possible to each scalability layer. The results show that MTZE performs about 2.5 dB better than baseline JPEG for the luminance component, and about 6 dB better for the chrominance components.

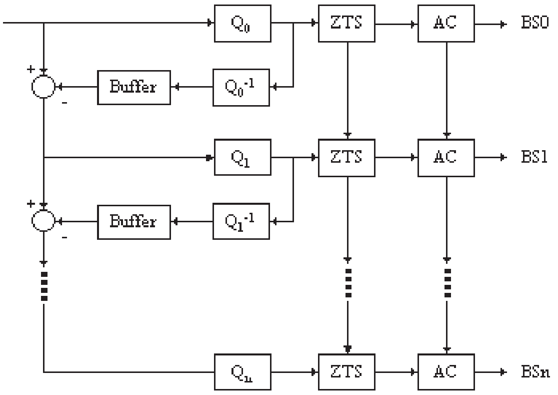


Fig. 15: MZTE encoding structure

6.5 *Embedded block coding with optimized truncation of the embedded bit-streams (EBCOT) algorithm [14–16]*

These papers describe the image compression scheme adopted for JPEG2000. The scheme offers rate and spatial scalability with excellent compression performance. In embedded block coding with optimized truncation of the embedded bit-streams (EBCOT), each sub-band is partitioned into relatively small blocks of wavelet coefficients, called code blocks. EBCOT generates an embedded bit stream separately for each code block. These embedded bit streams can be truncated independently to different lengths. The code-blocks are of 32×32 or 64×64 samples each, and provide random access to the image. The actual block coding algorithm which generates a separate embedded bit stream for each code block is combined with bit plane coding and adaptive arithmetic coding. Each code block is again divided into 2-D sequences of sub-blocks, whose size is 16×16. Then, for each bit plane, the significance map of sub-blocks is firstly encoded through quadtree coding. Then, four different primitive coding operations are used to encode those significant sub-blocks. The operations are zero coding (ZC), run-length coding (RLC), sign coding (SC) and magnitude refinement (MR). Finally, EBCOT organizes the final big stream in layers with optimized truncation so as to make it both resolution and SNR scalable. The layered bit stream may be truncated at any point during decoding. EBCOT uses the Daubechies (9,7) wavelet filter [17] with a five level transform, though JPEG2000 can use other filters. The results show performance gains of about 0.5 dB against SPIHT at various data rates.

7 Wavelet-based embedded rate scalable color image compression algorithms

Both EZW and SPIHT were originally designed for grayscale images. A straightforward application to color images is to code the transformed data from each spectral channel independently without exploiting any correlation that might exist between the spectral channels. The sub-sections below describe the EZW and SPIHT algorithms for color image compression.

7.1 *Color image compression using an embedded rate scalable approach (CEZW) [7]*

This paper presents a wavelet based coding algorithm for color images using a luminance/chrominance color space. Data rate scalability is achieved by using an embedded coding scheme, which is similar to Shapiro's embedded zerotree wavelet (EZW) algorithm [4]. In a luminance/chrominance color space, the three color components have little statistical correlation. However, it is observed that at the spatial locations where chrominance signals have large transitions it is highly likely that the luminance signal will have large transitions. This interdependence between the color components is exploited in this algorithm. In this algorithm, the YUV space is used. The algorithm is developed on the basis of Shapiro's algorithm. The SOT is established as follows: the original SOT structure in Shapiro's algorithm is applied to all three-color components. Each chrominance node is also a child node of the luminance node of the same location. Thus each chrominance node has two parent nodes: one is of the same chrominance component in a lower frequency sub-band; the other node is of the luminance component. A diagram of the SOT is shown in Fig. 16, where the tree is developed on the basis of the tree structure in Shapiro's algorithm, and YUV color space is used. In this algorithm, the coding strategy is similar to Shapiro's algorithm, except for the following changes in the dominant pass. (1) The luminance component is first scanned. For each luminance pixel, all descendents, including those of the luminance component and those of the chrominance components, are examined and appropriate

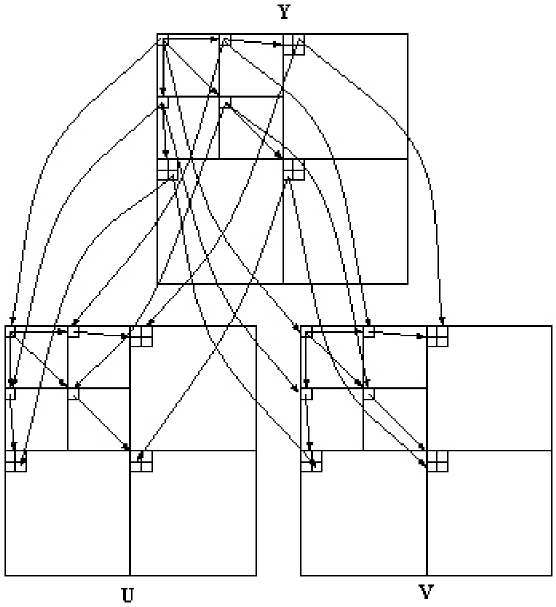


Fig. 16: Parent descendent relations in CEZW algorithm

symbols are assigned. (2) The two chrominance components are alternately scanned. The pixels that have already been encoded while scanning the luminance component are not examined. The subjective experiments showed that this algorithm produces images with the same quality as those from Said and Pearlman's algorithm (http://ipl.rpi.edu:80/SPIHT/) at the same data rate. However the peak-signal-to-noise (*PSNR*) ratio is lower than that from Said's algorithm, where the *PSNR* is obtained as

$$PSNR = 10\lg\left(\frac{255^2}{(MSE(Y)+MSE(U)+MSE(V))/3}\right)$$

7.2 Embedded color image coding using SPIHT with partially linked spatial orientation trees (CSPIHT) [18]

This paper describes a variation of the set partitioning in hierarchical trees (SPIHT) scheme for color image coding. By using partially linked spatial orientation tree structures across different spectral planes, the new color-SPIHT scheme is able to embed both chrominance and luminance data in the coded bit stream. The performance is comparable to that of a SPIHT-based coding scheme but with significantly lower computational complexity. In CSPIHT, SPIHT's SOT structure [10] is used for all spectral planes. To create a comprehensive SOT structure across the different spectral planes, luminance nodes in the LL sub-band that do not have any offspring will have descendants in the LL sub-bands of the chrominance planes, as shown in Fig. 17. The results obtained are compared to Said and Pearlman's algorithm (SPIHT KLT) (http://ipl.rpi.edu:80/SPIHT/), which performs the Karhunen–Loeve Transform (KLT) [19] on the spectral components of the image before coding the decorrelated color planes independently using the SPIHT scheme. For the case of four and five-level DWT decomposition, CSPIHT has significantly better PSNR performance than SPIHT KLT, especially at low bit-rates (up to 6 dB improvement in performance).

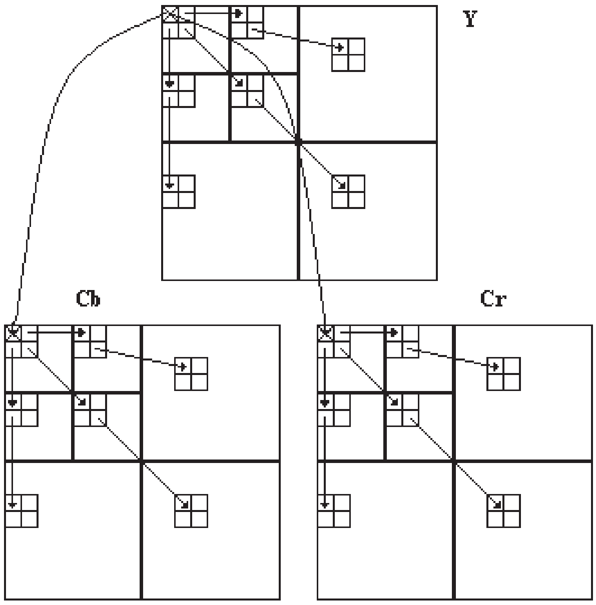


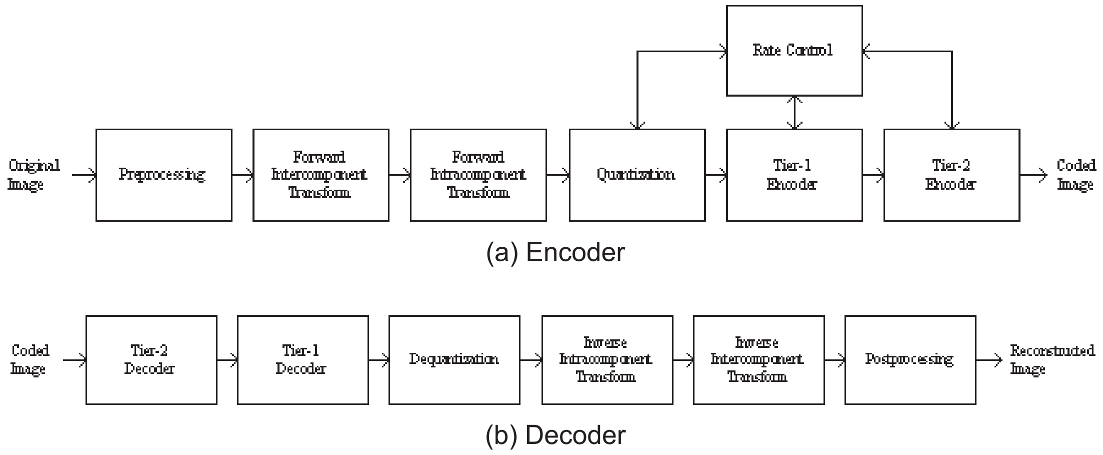
Fig. 17: Linking SOT structures in different spectral planes in the CSPIHT scheme

8 JPEG2000 still image compression standard

JPEG2000 [20] is the new international standard for still image compression. The JPEG2000 standard is based on wavelet/sub-band coding techniques [17, 21] and supports lossy and lossless compression of single-component (e.g., grayscale) and multi-component (e.g., color) imagery. In order to facilitate both lossy and lossless coding in an efficient manner, reversible integer-to-integer [22–24] and nonreversible real-to-real transforms are employed. To code transform data, the codec makes use of bit-plane coding techniques. For entropy coding, a context-based adaptive binary arithmetic coder [11] is used. In addition to this basic compression functionality, however, numerous other features are provided, including:

1) progressive recovery of an image by fidelity or resolution;
2) region of interest coding, whereby different parts of an image can be coded with differing fidelity;
3) random access to particular regions of an image without needing to decode the entire code stream;
4) a flexible file format with provisions for specifying opacity information and image sequences and
5) good error resilience.

Due to its excellent coding performance and many attractive features, JPEG2000 has a very large potential application base. Some possible application areas include: image archiving, Internet, web browsing, document imaging, digital photography, medical imaging, remote sensing, and desktop publishing. The JPEG2000 core compression algorithm is primarily based on embedded block coding with optimized



(a) Encoder

(b) Decoder

Fig. 18: The general structure of the JPEG2000 codec

truncation (EBCOT) of the embedded bit-streams [14–16]. The EBCOT algorithm provides superior compression performance and produces a bit-stream with features such as resolution and SNR scalability and random access.

8.1 JPEG2000 codec structure

The general structure of the JPEG2000 codec is shown in Fig. 18. The input to the encoding process is an image consisting of one or more components. Before any further processing takes place, an image is partitioned into one or more disjoint rectangular regions called tiles. This is done when the original image is quite large in comparison to the amount of memory available to the codec.

In the preprocessing step, each image component has its sample values adjusted by an additive bias, in a process called DC level shifting. The bias is chosen such that the resulting sample values have a nominal dynamic range (approximately) centered about zero. The RGB color space is transformed to YCrCb color space in the forward intercomponent transform step. In the forward intracomponent transform step, transforms that operate on individual components can be applied. The particular type of operator employed for this purpose is the wavelet transform. The resulting wavelet coefficients are quantized in the quantization step. A different quantizer is employed for the coefficients of each sub-band. In the case of lossless coding, reversible transforms must be employed and all quantizer step sizes are forced to be one. In the tier-1 coding step, the quantizer indices for each sub--band are partitioned into code blocks and each of the code blocks is independently embedded coded. The coding is performed using the bit-plane coder. There are three coding passes per bit plane. These passes are the significance pass, the refinement pass, and the cleanup pass, respectively. In the tier-2 encoding step, the coding pass information generated during tier-1 is packaged into data units called packets, in a process referred to as packetization. The packetization process imposes a particular organization on coding pass data in the output code stream. This organization facilitates many of the desired codec features cited above. The rate control can be achieved through two distinct mechanisms: 1) the choice of quantizer step sizes, and 2) the selection of the subset of coding passes to include in the code stream.

The decoder structure essentially mirrors that of the encoder. That is, with the exception of rate control, there is a one-to-one correspondence between the functional blocks in the encoder and decoder. Each functional block in the decoder either exactly or approximately inverts the effects of its corresponding block in the encoder. Since the tiles are coded independently of one another, the input image is (conceptually, at least) processed one tile at a time. In the sections that follow, each of the above processes is briefly explained. For more details about the whole processes, see [20].

8.2 Preprocessing/postprocessing

The codec expects its input sample data to have a nominal dynamic range that is approximately centered about zero. The preprocessing stage of the encoder simply ensures that this expectation is met. Suppose that a particular component has P bits/sample. The samples may be either signed or unsigned, leading to a nominal dynamic range of $[-2^{P-1}, 2^{P-1}-1]$ or $[0, 2^{P}-1]$, respectively. If the sample values are unsigned, the nominal dynamic range is clearly not centered about zero. Thus, the nominal dynamic range of the samples is adjusted by subtracting a bias of 2^{P-1} from each of the sample values. If the sample values for a component are signed, the nominal dynamic range is already centered about zero, and no processing is required. The postprocessing stage of the decoder essentially undoes the effects of preprocessing in the encoder. If the sample values for a component are unsigned, the original nominal dynamic range is restored. Lastly, in the case of lossy coding, clipping is performed to ensure that the sample values do not exceed the allowable range.

8.3 Intercomponent transform

In the encoder, the preprocessing stage is followed by the forward intercomponent transform stage. Here, an intercomponent transform can be applied to the tile-component data. Such a transform operates on all of the components together, and serves to reduce the correlation between components, leading to improved coding efficiency. Only two intercomponent transforms are defined in the base-line JPEG-2000 codec: the irreversible color transform (ICT) and the reversible color transform (RCT). The ICT is nonreversible and real-to-real in nature, while the RCT is reversible and integer-to-integer. Both of these transforms essentially map image data from the RGB to the YCrCb color space. The ICT may only be used in the case of lossy coding, while the RCT can be used in either the lossy or lossless case. After the intercomponent transform stage in the encoder, data from each component is treated independently. The ICT is nothing more than the classic RGB to YCrCb color space transform. The RCT is simply a reversible integer-to-integer approximation to the ICT (similar to that proposed in [24]). The inverse intercomponent transform stage in the decoder essentially undoes the effects of the forward intercomponent transform stage in the encoder.

8.4 Intracomponent transform

The intercomponent transform stage in the encoder is followed by the intracomponent transform stage. In this stage, transforms that operate on individual components can be applied. The particular type of operator employed for this purpose is the wavelet transform. Through the application of the wavelet transform, a component is split into numerous frequency bands (i.e., sub-bands). Due to the statistical properties of these sub-band signals, the transformed data can usually be coded more efficiently than the original untransformed data. Both reversible integer-to-integer [22, 23, 25–27] and non-reversible real-to-real wavelet transforms are employed by the baseline codec. The inverse intracomponent transform stage in the decoder essentially undoes the effects of the forward intracomponent transform stage in the encoder.

8.5 Quantization/dequantization

In the encoder, after the tile-component data has been transformed (by intercomponent and/or intracomponent transforms), the resulting coefficients are quantized. Quantization allows greater compression to be achieved, by representing transform coefficients with only the minimal precision required to obtain the desired level of image quality. Quantization of transform coefficients is one of the two primary sources of information loss in the coding path. Transform coefficients are quantized using scalar quantization with a deadzone. A different quantizer is employed for the coefficients of each sub-band, and each quantizer has only one parameter, its step size. In the decoder, the dequantization stage tries to undo the effects of quantization. Unless all of the quantizer step sizes are less than or equal to one, the quantization process will normally result in some information loss, and this inversion process is only approximate. The baseline codec has two distinct modes of operation, referred to as the integer mode and the real mode. In the integer mode, integer-to-integer transforms are employed. The quantization step is fixed to one. Lossy coding is still achieved by discarding bit-planes. In the real mode, real-to-real transforms are employed. Quantization steps are chosen in conjunction with rate control. In this mode, discarding bi-planes or changing the size of the quantization step, or both, achieves lossy compression.

8.6 Tier-1 coding

After quantization is performed in the encoder, tier-1 coding takes place. This is the first of two coding stages. The quantizer indices for each sub-band are partitioned into code blocks and each of the code blocks is independently coded. The coding is performed using the bit-plane coder. For each code block, an embedded code is produced, comprised of numerous coding passes. The output of the tier-1 encoding process is, therefore, a collection of coding passes for the various code blocks. On the decoder side, the bit-plane coding passes for the various code blocks are input to the tier-1 decoder, these passes are decoded, and the resulting data is assembled into sub-bands.

The tier-1 coding process essentially involves bit-plane coding. After all of the sub-bands have been partitioned into code blocks, each of the resulting code blocks is independently coded, using a bit-plane coder. There are three coding passes per bit plane. These passes, in order are as follows:

1) significance,
2) refinement,
3) cleanup.

All three types of coding passes scan the samples of a code block in the same fixed order. The bit-plane encoding process generates a sequence of symbols for each coding pass. Some or all of these symbols may be entropy coded. For the purposes of entropy coding, a context-based adaptive binary arithmetic coder is used [28]. For each pass, all of the symbols are either arithmetically coded or raw coded (i.e., the binary symbols are emitted as raw bits with simple bit stuffing). The arithmetic and raw coding processes both ensure that certain bit patterns never occur in the output, allowing such patterns to be used for error resilience purposes. The following subsections present these three passes.

8.6.1 Significance pass

The first coding pass for each bit plane is the significance pass. This pass is used to convey the significance and (as necessary) sign information for samples that have not yet been found to be significant and are predicted to become significant during the processing of the current bit plane. The samples in the code block are scanned in a predefined order. If a sample has not yet been found to be significant, and is predicted to become significant, the significance of the sample is coded with a single binary symbol. If the sample also happens to be significant, its sign is coded using a single binary symbol. If the most significant bit plane is being processed, all samples are predicted to remain insignificant. Otherwise, a sample is predicted to become significant if any 8-connected neighbor has already been found to be significant. As a consequence of this prediction policy, the significance and refinement passes for the most significant bit plane are always empty (and need not be explicitly coded). The symbols generated during the significance pass may or may not be arithmetically coded. If arithmetic coding is employed, the binary symbol conveying significance information is coded using one of nine contexts. The particular context used is selected on the basis of the significance of the sample's 8-connected neighbors and the orientation of the sub-band with which the sample is associated (e.g., LL, LH, HL, HH). In the case that arithmetic coding is used, the sign of a sample is coded as the difference between the actual and predicted sign. Otherwise, the sign is coded directly. Sign prediction is performed using the significance and sign information for 4-connected neighbors.

8.6.2 Refinement pass

The second coding pass for each bit plane is the refinement pass. This pass signals subsequent bits after the most significant bit for each sample. If a sample was found to be significant in a previous bit plane, the next most significant bit of that sample is conveyed using a single binary symbol. Like the significance pass, the symbols of the refinement pass may or may not be arithmetically coded. If arithmetic coding is employed, each refinement symbol is coded using one of three contexts. The particular context employed is selected according to whether the second MSB position is being refined and the significance of 8-connected neighbors.

8.6.3 Cleanup pass

The third (and final) coding pass for each bit plane is the cleanup pass. This pass is used to convey significance and (as necessary) sign information for those samples that have not yet been found to be significant and are predicted to remain insignificant during the processing of the current bit plane. Conceptually, the cleanup pass is not much different from the significance pass. The key difference is that the cleanup pass conveys information about samples that are predicted to remain insignificant, rather than those that are predicted to become significant. Algorithmically, however, there is one important difference between the cleanup and significance passes. In the case of the cleanup pass, samples are sometimes processed in groups, rather than individually as with the significance pass.

8.7 Tier-2 coding

In the encoder, tier-1 encoding is followed by tier-2 encoding. The input to the tier-2 encoding process is the set of bit-plane coding passes generated during tier-1 encoding. In tier-2 encoding, the coding pass information is packaged into data units called packets, in a process referred to as packetization. The resulting packets are then output to the final code stream. The packetization process imposes a particular organization on coding pass data in the output code stream. This organization facilitates many of the desired codec features, including rate scalability and progressive recovery by fidelity or resolution. In the decoder, the tier-2 decoding process extracts the various coding passes from the code stream (i.e., depacketization) and associates each coding pass with its corresponding code block.

8.8 Rate control

In the encoder, rate control can be achieved through two distinct mechanisms:

1) the choice of quantizer step sizes,
2) the selection of the subset of coding passes to include in the code stream.

When the integer coding mode is used (i.e., when only integer-to-integer transforms are employed) only the first mechanism may be used, since the quantizer step sizes must be fixed at one. When the real coding mode is used, then either or both of these rate control mechanisms may be employed. When the first mechanism is employed, the quantizer step sizes are adjusted in order to control the rate. As the step sizes are increased, the rate decreases, at the cost of greater distortion. Although this rate control mechanism is conceptually simple, it does have one potential drawback. Every time the quantizer step sizes are changed, the quantizer indices change, and tier-1 encoding must be performed again. Since tier-1 coding requires a considerable amount of computation, this approach to rate control may not be practical in computationally-constrained encoders. When the second mechanism is used, the encoder can elect to discard coding passes in order to control the rate. The encoder knows the contribution that each coding pass makes to the rate, and can also calculate the distortion reduction associated with each coding pass. Using this information, the encoder can then include the coding passes in order of decreasing distortion reduction per unit rate until the bit budget has been exhausted. This approach is very flexible in that different distortion metrics can be easily accommodated (e.g., mean squared error, visually weighted mean squared error, etc.). For a more detailed treatment of rate control, the reader is referred to [14] and [20].

8.9 Region of interest coding

The codec allows different regions of an image to be coded with differing fidelity. This feature is known as region-of-interest (ROI) coding. When an image is to be coded with an ROI, some of the transform coefficients are identified as being more important than the others. The coefficients of greater importance are referred to as ROI coefficients, while the remaining coefficients are known as background coefficients. ROI coefficients are encoded with greater precision than the background coefficients. For more information on ROI coding, the reader is referred to [29, 30].

9 Conclusions

This paper has presented a review on some embedded rate scalable image codecs. The embedded zerotree wavelet (EZW) coder is often used as a performance reference. Similar visual performance, with the added advantage of low arithmetic complexity, is achieved by set partitioning in the hierarchical trees algorithm (SPIHT). These algorithms use a tree structure or lists to detect and exploit similarities. The precise rate control that is achieved with these algorithms is a distinct advantage. The user can choose a bit rate and encode the image to exactly the desired bit rate. EZW and SPIHT offer only SNR scalability. The complexity of the SFQ algorithm lies mainly in the iterative zerotree pruning stage of the encoder, which can be substantially reduced with fast heuristics based on models rather than actual R-D data, which is expensive to compute. The SFQ coder has important characteristics. First, SFQ is built around a linear transform that allows signal energy to be compacted both in frequency and in space, and quantization modes designed to match this characterization. Second, SFQ provides a framework for optimizing (in the rate-distortion sense) the application of the quantization modes available to it.

The EBCOT image compression algorithm offers state-of-the-art compression performance together with an unprecedented set of bit-stream features, including resolution scalability, SNR scalability and a “random access” capability. All features can coexist-exist within a single bit-stream without substantial sacrifices in compression efficiency. The MZTE algorithm describes the texture representation scheme adopted for MPEG-4 synthetic/natural hybrid coding (SNHC) of texture maps and images. The scheme is based on the concept of the multi-scale zerotree wavelet entropy (MZTE) coding technique, which provides many levels of scalability layers in terms of either spatial resolutions or picture quality. MZTE, with three different modes (single-Q, multi-Q, and bilevel), provides much improved compression efficiency and fine-gradual scalabilities, which are ideal for hybrid coding of texture maps and natural images. The MZTE scheme has been adopted as the baseline technique for the visual texture coding profile in both the MPEG-4 video group and SNHC group. MZTE was also rated as one of the top five schemes in terms of compression efficiency in the JPEG2000 November 1997 evaluation, among 27 submitted proposals.

CEZW is a coding algorithm for color images that uses a luminance/chrominance color space. In CEZW algorithm data rate scalability is achieved by using an embedded coding scheme, which is similar to the EZW algorithm. The CEZW algorithm does not require image-dependent transforms such as KL transforms to decorrelate the color components. A spatial-orientation tree that links not only the frequency bands but also the color channels is used for scanning the wavelet coefficients, such that the interdependence between different the color components in the LC spaces is automatically exploited. The CSPIHT scheme offered a simple solution to embed both luminance and chrominance data into a single coded data stream. If an image is able to go through a maximum of levels of DWT decomposition, CSPIHT provides comparable performance to SPIHT KLT in terms of quality of reconstruction. For much fewer than decomposition levels for the same image, CSPIHT offers much better quality of reconstruction without the need to perform the computationally intensive KLT on the spectral components of the image. The advantage in the performance of CSPIHT lies in the low bit-rate coding and DWT maps with large LL sub-band dimensions.

The JPEG2000 standard for still image compression is presented. The JPEG2000 standard supports lossy and lossless compression of single-component and multi-component imagery. In addition to this basic compression functionality, it has many other features such as: progressive recovery of an image by fidelity or resolution, region of interest coding, whereby different parts of an image can be coded with differing fidelity, random access to particular regions of an image without needing to decode the entire code stream, a flexible file format with provisions for specifying opacity information and image sequences, and good error resilience.

The performance of all mentioned algorithms can be improved by using efficient sign coding and estimation of zero-quantized coefficients, as presented in [31].

References

[1] Netravali A. N., Rubinstein C. B.: "Luminance Adaptive Coding of Chrominance Signals." *IEEE Transactions on Communications*, COM 27, 1979, p. 703–710.

[2] Limb J. O., Rubinstein C. B.: "Plateau Coding of the Chrominance Component of Color Picture Signals." *IEEE Transactions on Communications*, COM 22, 1974, p. 812–820.

[3] Rabbani M., Jones P. W.: *Digital image compression techniques*. Bellingham, Washington: SPIE Optical Engineering Press, 1991.

[4] Shapiro J. M.: "Embedded Image Coding Using Zerotrees of Wavelet Coefficients." *IEEE Transactions on Signal Processing*, Vol. **41** (1993), p. 3445–3462.

[5] Shen K., Delp E. J.: "Wavelet Based Rate Scalable Video Compression." *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. **9** (1999), p. 109–122.

[6] Frazier M. W.: *An introduction to wavelets through linear algebra*. Springer-Verlag New York Inc., 1999.

[7] Shen K., Delp E. J.: *Color Image Compression Using an Embedded Rate Scalable Approach*. Proc. IEEE Int. Conf. Image Processing, Santa Barbara, CA, Oct. 26–29, 1997, III-34–III-37.

[8] Bhaskaran V., Konstantinides K.: *Image and video compression standards: Algorithms and architectures* (second edition). Kluwer Academic Publisher, 1997.

[9] Eskicioglu A. M., Fisher P. S.: "Image Quality Measures and Their Performance." *IEEE Transactions on Communications*, Vol. **43** (1995), p. 2959–2965.

[10] Said A., Pearlman W. A.: "A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees." *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. **6** (1996), p. 243–250.

[11] Witten H., Neal R., Cleary J. G.: "Arithmetic Coding for Data Compression." *Comm. ACM*, Vol. **30** (1987), p. 520–540.

[12] Xiong Z., Ramchandran K., Orchard M. T.: "Space--Frequency Quantization for Wavelet Image Coding." *IEEE Transaction on Image Processing*, Vol. **6** (1997), p. 677–693.

[13] Sodagar I. et al: "Scalable Wavelet Coding for Synthetic/Natural Hybrid Images." *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. **9** (1999), p. 244–254.

[14] Taubman D.: "High performance scalable image compression with EBCOT." *Proc. of IEEE International Conference on Image Processing*, Kobe (Japan), Vol. **3** (1999), p. 344–348.

[15] Taubman D.: "High performance scalable image compression with EBCOT." *IEEE Trans. on Image Processing*, Vol. **9** (2000), p. 1158–1170.

[16] Taubman D. et al: "Embedded block coding in JPEG 2000." *Signal Processing: Image Communication*, Vol. **17** (2002), p. 49–72.

[17] Antonini M. et al: "Image Coding Using Wavelet Transform." *IEEE Transactions on Image Processing*, Vol. **1** (1992), p. 205–220.

[18] Kassim A. A., Lee W. S.: "Embedded Color Image Coding Using SPIHT with Partially Linked Spatial Orientation Trees." *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. **13** (2003), p. 203–206.

[19] Kouassi R. K. et al: "Application of the Karhunen-Loeve Transform for Natural Color Images Analysis." Conf. Record 3, *1st Asilomar Conf. Signals, Systems & Computers*, Vol. **2** (1997), p. 1740–1744.

[20] ISO/IEC 15444-1: Information technology – JPEG2000 image coding system – Part 1: Core coding system, 2000.

[21] Lewis A. S., Knowles G.: "Image compression using the 2-D wavelet transform." *IEEE Trans. on Image Processing*, Vol. **1** (1992), p. 244–250.

[22] Calderbank A. R. et al: "Wavelet transforms that map integers to integers." *Applied and Computational Harmonic Analysis*, Vol. **5** (1998), p. 332–369.

[23] Adams M. D.: *Reversible Integer-to-Integer Wavelet Transforms for Image Coding*. Ph.D. thesis, Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada, Sept. 2002, Available online from http://www.ece.uvic.ca/ mdadams.

[24] Gormish M. J. et al: *Lossless and nearly lossless compression of high-quality images*. Proc. of SPIE, San Jose, CA, USA, 3025, 1997, p. 62–70.

[25] Chao H., Fisher P., Hua Z.: *An approach to integer wavelet transforms for lossless image compression*. Proc. of International Symposium on Computational Mathematics, Guangzhou, China, 1997, p. 19–38.

[26] Adams M. D.: *Reversible wavelet transforms and their application to embedded image compression*. M.Sc. thesis, Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, Canada, Jan. 1998, Available from http://www.ece.uvic.ca/ mdadams.

[27] Adams M. D., Kossentini F.: "Reversible integer-to-integer wavelet transforms for image compression: Performance evaluation and analysis." *IEEE Trans. on Image Processing*, Vol. **9** (2000), p. 1010–1024.

[28] ISO/IEC 14492-1: Lossy/lossless coding of bi-level images, 2000.

[29] Christopoulos C., Askelof J., Larsson M.: "Efficient methods for encoding regions of interest in the upcoming JPEG2000 still image coding standard." *IEEE Signal Processing Letters*, Vol. **7** (2000), p. 247–249.

[30] Askelof J., Carlander M. L., Christopoulos C.: "Region of interest coding in JPEG2000." *Signal Processing: Image Communication*, Vol. **17** (2002), p. 105–111.

[31] Deever A. T., Hemami S. S.: "Efficient Sign Coding and Estimation of Zero-Quantized Coefficients in Embedded Wavelet Image Codecs." *IEEE Transactions on Image Processing*, Vol. **12** (2003), p. 420–430.

Doc. Ing. Boris Šimák, CSc.
e-mail: simak@feld.cvut.cz

Eng. Farag Ibrahim Younis Elnagahy, MSc.
E-mail: faragelnagahy@hotmail.com

Department of Telecommunications Engineering

Czech Technical University in Prague
Faculty of Electrical Engineering
Technická 2
166 27 Praha 6, Czech Republic