Proceedings of the 5th WSEAS International Conference on Telecommunications and Informatics, Istanbul, Turkey, May 27-29, 2006 (pp57-61)

Modified SPIHT Image Coder For Wireless Communication

M. B. I. REAZ, M. AKTER, F. MOHD-YASIN
Faculty of Engineering
Multimedia University
63100 Cyberjaya, Selangor
Malaysia

Abstract: - The Set Partitioning in Hierarchical Trees (SPIHT) approach for still image compression is one of the most efficient embedded gray image compression schemes till date. The algorithm relies on a very efficient scanning cum bit-allocation scheme for quantizing the coefficients obtained by a wavelet decomposition of an image. In this paper, modifications of the SPIHT compressor without arithmetic coding have been presented. The simulation result has shown that the algorithm is suitable for wireless communication due to less memory requirement, fast with almost equal compression performance of peak signal to noise ratio (PSNR) compare to original SPIHT.

Key-Words: - SPIHT, Modified SPIHT, EBCOT, Sorting phase, Refinement phase.

1 Introduction

Natural images can be characterized by a linear combination of energy concentrated in pockets in both space and frequency. The wavelet transform is very well suited for transform coding of such images because it decorrelates the image both in space and frequency. Thus, it compact energy into a few low frequency and a few high frequency coefficients. Once a hierarchical wavelet representation of an image is obtained, the coding effort reduces to quantizing the coefficients as efficiently as possible. Recently there has been a plethora of research activity in wavelet based image coding. The efficiency of a wavelet-based compression scheme relies on the efficiency of specifying to the decoder which coefficients to quantize before which others, and of the corresponding bit allocation.

Shapiro [1] introduced the Embedded Zerotree Wavelet (EZW) scheme where the zerotree enables efficient prediction of significance information of the wavelet coefficients. Following this work, Said and Pearlman [2] developed an alternate scheme, called set partitioning in hierarchical trees (SPIHT) based on the same basic concepts. It was more effective in transmission of significance information to the decoder. Both the schemes relied on partial magnitude ordering of the wavelet coefficients, followed by progressive refinement, and produced embedded bitstreams. The transmission of ordering information is achieved by a subset partitioning approach that is duplicated at the decoder. The refinement is based on ordered bit plane transmission of the magnitudes of the coefficients previously ascertained as significant. Xiong et al [3] has developed a very efficient space-frequency quantization scheme that uses a rate distortion criterion to jointly optimize zerotree quantization and scalar frequency quantization.

Knipe et. al. [4] proposed vector quantization based SPIHT named VSPIHT. Later, Mukherjee and Mitra [5] studied broadly VSPIHT using successive refinement Voronoi lattice vector quantization yielding good quality of the reconstructed image in terms of Peak Signal to Noise ratio (PNSR). However, the higher complexity of the vector quantizer (VQ) design is needed and more memory space in system is required to store the codebook for the cost of the quality of the output image.

In order to code the images at very low bit rates, in this work, the efficient set-partitioning approach by Said and Pearlman has been adopted to partially order of wavelet coefficients and refine them successively using scalar quantization scheme. The modifications of the SPIHT compressor images have been presented exchanging the sorting and refinement phase. At first, the sorting phase with the refinement phase has been exchanged to save memory for status information. This scheme is more efficient than the original SPIHT [2]. This paper is organized as follows. Sections 2 discuss the original SPIHT algorithm and its modification version. Section 3 shows the experimental verification of the proposed algorithm. Section 4 concludes this paper.

2 Coding Methodology

The SPIHT algoritm is very efficient in transmission of ordering information, essentially involves a scalar

quantization operation. The essence of the set portioning is to first classify the elemental coding units based on their magnitude and then to quantize them in a successive refinement framework. The elemental coding unites are scalar wavelet coefficients.

2.1 Wavelet transformed images

While embedded zerotree like algorithms are applied, a wavelet transform is performed on the image. This results a multiscale representation. The transform reduces the correlation between neighboring pixels. The energy of the original image is concentrated in the lowest frequency band of the transformed image. Additionally, self similarities between different scales which result from the recursive application of the wavelet transform step to the low frequency band can be observed. Consequently, based upon these facts, good compression performance can be achieved if those coefficients are first transmitted which represent most of the image energy.

2.2 Bitplane coding

The overall encoding procedure is basically a kind of bitplane coding. Thereby the k_{th} bits of the coefficients constitute a bitplane. In general, a bitplane encoder starts coding with the most significant bit of each coefficient. When all those bits are coded, the next bitplane is considered until the least significant bits are reached. Within a bitplane the bits of the coefficients with largest magnitude come first. This ordering is illustrated in Fig.1 as bar chart [7]. The coefficients are shown in decreasing order from left to right. Each coefficient is represented with eight bit, where the least significant bit is in front. As a consequence, such a bitplane coding scheme encodes the important information in terms of compression efficiency first. On the other hand, the ordering information, which can scatter the compression effect, have to be stored in the system memory or transmitted via channel.

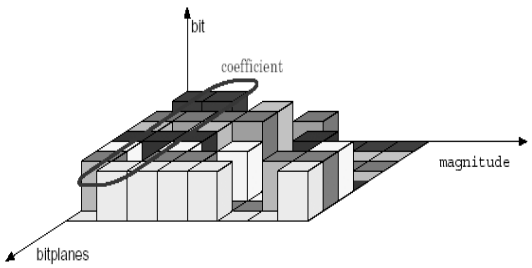

Fig.1 Bitplane coding

2.3 SPIHT coding scheme

The SPIHT algorithm is very efficient in transmission of ordering information, essentially involves a scalar quantization operation. As such, dependencies that exist between the neighboring coefficients are not exploited in full extent. The arithmetic coding enhancement of SPIHT indirectly exploits this redundancy to gain 0.3-0.6db over the non-arithmetic coded version. Said and Pearlman [2] have significantly improved the codec of EZW. The main idea is based on partitioning of sets, which consists of coefficients or representatives of whole subtrees. The coefficients of a wavelet transformed image are classified in three sets:

1. The list of insignificant pixels (LIP) which contains the coordinates of those coefficients which are insignificant with respect to the current threshold th.

2. The list of **s**ignificant pixels (LSP) which contains the coordinates of those coefficients which are significant with respect to threshold, and

3. The list of insignificant **s**ets (LIS) which contains the coordinates of the roots of insignificant subtrees.

During the compression procedure, the sets of coefficients in LIS are refined and if coefficients become significant they are moved from LIP to LSP.

The first difference to Shapiro's EZW algorithm is the distinct definition of the significance. Here, the root of the tree is excluded from the computation of the significance attribute, which can be explained more simply using the following notations.

For all $0 < m \leq \log_2 N$, let $h = h(m)$ be the set of the coordinates of the roots after m wavelet transform steps. Furthermore, let

$$O(i,j) = \begin{Bmatrix} (2i,2j) & , & (2i,2j+1) \\ (2i+1,2j) & , & (2i+1,2j+1) \end{Bmatrix} \qquad (1)$$

Be the sets of the coordinates of the children of the node (i, j), in the case that this node has children. The set D(i, j) is composed of the desendants of node (i, j), and

$$L(i,j) = D(i,j) - O(i,j) \qquad (2)$$

is the set of the decendants excluding the four children. Each element of LIS will have as attribute either A or B. If the type of any entry $(i, j) \in$ LIS is A, then the entry (i, j) represents set D(i, j). If the type is B then it represents L(i, j).

In general, the decoder duplicates the execution path of the encoder as it was also the case in Shapiro's algorithm [1]. To ensure this behavior, the coder sends the result of a binary decision to the decoder before a branch is taken in the algorithm. Thus, all decisions of the decoder are based on the received bits. The name of the algorithm is

composed of the words *set* and *partitioning*. The sets O(i, j), D(i, j) and L(i, j) were already mentioned. The set partitioning rules of Said and Pearlman [2] is addressed as follows:

- the initial partition is formed with the sets
  - {(i, j)} for all (i, j) ∈ H and
  - D(i, j) for all {(i, j)|(i, j) ∈ H with D(i, j) ≠ ∅}
- if Sk(D(i, j)) = 1 then D(i, j) is partitioned into L(i, j) and the four sets {(e, f)} ∈ O(i, j)
- if Sk(L(i, j)) = 1 then L(i, j) is partitioned into the four sets {(e, f)}, with (e, f) ∈ O(i, j).

It can be divided into three parts. The first part initializes the lists, computes, and output the initial threshold. k_{max} will be indicated as the maximum threshold.

Afterwards, the sorting phase and the refinement phase are executed (k_{max} + 1) times starting with bitplane k_{max} down to bit plane 0. In pass k, the sorting phase first outputs the kth bits of the coefficients in LIP. If the kth bit of some coefficient is 1, i.e., $S_k(i, j) = 1$, coefficient is significant with respect to k and insignificant with respect to (k + 1). Therefore, the sign of the coefficient has to be output.

Sorting phase: The elements of LIS are now processed using the set partitioning rules. First, the significance bit for set D(i, j) or L(i, j) is written out, respectively. The set corresponding to an element (i, j) of type A, which is now significant with respect to the current threshold, is partitioned into four single element sets {(e, f)} for each (e, f) ∈ O(i, j). Each of the coefficients (e, f) are appended to LIP or LSP, depending on there significance. The element (i, j) itself is changed to type B and moved to the end of the list LIS, if the set L(i, j) is non empty. The elements (i, j) of type B in LIS are refined to the four elements (e, f) ∈ O(i, j) of type A, if the $S_k(L(i, j)) = 1$.

Refinement phase: The refinement phase outputs the kth bit of each elements of list LSP, if it was not included in the last sorting phase.

The output of SPIHT encoder is binary thus the compression rate can be achieved exactly and arithmetic encoders for binary alphabets can be applied to further increase the compression efficiency.

2.4 Modifications to the SPIHT

In this section the modifications of the SPIHT compressor based on the partitioned approach to wavelet transform images has been presented. At first, the sorting phase with the refinement phase has been exchanged to save memory for status information. In the basic SPIHT algorithm, status information has to be stored for the elements of LSP specifying whether the corresponding coefficient has been added to LSP in the current iteration of the sorting phase. In the worst case all coefficients become significant in the same iteration. Consequently, we have to provide a memory capacity of q^2 bits to store this information. Here, q is the size of quadratic subimage. However, if we exchange the sorting and the refinement phase, we do not need to consider this information anymore. The compressed data stream is still decodable and it is not increased in size. Of course, it has to be considered that there is a reordering of the transmitted bits during iteration.

The more interesting case is the difference of the two algorithms during the coding of a bit plane. The maximum difference during the coding of the k_{th} bit plane can be estimated with lower and upper bounds l(k) and u(k). Obviously, the lower bound l(k) is zero and $u(k) = 4^k$. However, it is very difficult to distinguish the reconstructed images with the naked eye. To circumstantiate this remark, both algorithms have been applied on *Lena*. The mean squares error has been measured after each byte of compressed output.

3 Implementation and Results

The modified SPHIT coding scheme is applied to compress the 8bpp grey-level, 512×512, Lena image. Practical test have shown that the pyramid transformation does not have to be exactly unitary, so five-level pyramid has been used to be constructed with 7/5 filter bank. It is important to observe that the bit rates are not entropy estimate calculated from the actual size of the compressed file. By using progressive transmission ability, the sets of distortions are obtained from the same file. The decoder reads the same file bytes of the file calculated the inverse subband transform and then compared the recovered image with the original. The distortion is measured by the PSNR as given in equation (3).

$$PSNR = 10\log 10\left(\frac{255^2}{MSE}\right) \tag{3}$$

where MSE denotes the mean suare-error between the original and reconstructed images.

The performance of proposed SPIHT without arithmetic coding is compared with other popular algorithms such as EZW, original SPIHT, Embedded Block Coding with Optimized Truncation (EBCOT) [6]. Fig. 2 shows the PSNR

versus bit rate results for the Lena image with original SPIHT, EZW, EBCOT, proposed SPIHT without arithmetic coding. It is shown that SPIHT with arithmetic coding and EBCOT are almost overlapped with each other. The proposed SPIHT without arithmetic coding has gain the performance little less than SPIHT but higher than the EZW. The visual quality of the coded 512×512 Lena image is compared against original image. Fig.3 shows the visual quality of original Lena image and reconstructed image using the modified SPIHT algorithm at 0.0625bpp, 0.250bpp and 0.50bpp. Table 1 also shows the numerical comparison of the proposed algorithm with SPIHT.

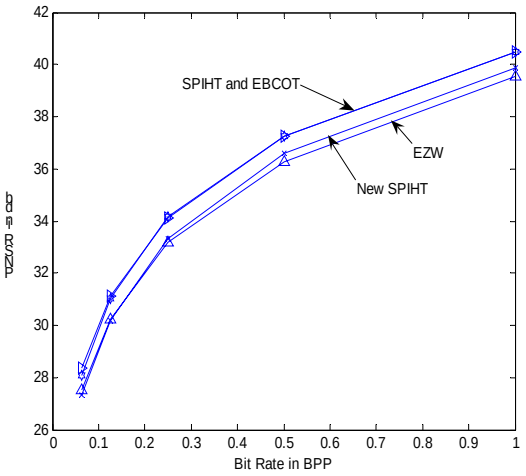

Fig. 2 PSNR versus bit rate results for the Lena image with original SPIHT, EZW, EBCOT, proposed SPIHT without arithmetic coding.

Table 1 Numerical comparison of SPIHT and proposed algorithm

bpp	SPIHT	Modified SPIHT
0.1000	29.8107	29.3202
0.2000	32.7202	32.2514
0.3000	34.5479	34.0331
0.4000	35.8422	35.4857
0.5000	36.8623	36.5939
0.6000	37.6650	37.3759
0.7000	38.2581	38.0491
0.8000	38.9390	38.7058
0.9000	39.5218	39.3437

4 Conclusion

Modified SPIHT without arithmetic coder has been demonstrated preserving the PSNR advantage of original SPIHT. It is fast and requires less system memory. The greatest challenge will be the hardware implementation of the three lists, LIP, LSP, and LIS. The reduced estimations of the memory requirements of these lists using modified SPIHT will certainly give a advantage for hardware implementation of such algorithm for any applications. Furthermore, the modified algorithm with arithmetic coding will enhance the efficiency for color image coding.

References:

[1] M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," IEEE Transaction on Signal Processing, vol. 41, pp. 3445-3462, December 1993.

[2] Amir Said, and William A. Pearlman, "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, issue 3, pp. 243-250, June 1996

[3] Z.Xiong, C. Herley, K. Ramchandran, and M. T. Orchard, "Space-frequency quantization for a space-varying wavelet packet coder," IEEE International Conference on Image Processing, pp. 614-617, Washington, DC, October 1995.

[4] J. Knipe, X. Li, and B. Han, "An improved lattice vector quantization scheme for wavelet compression," IEEE Transaction on Signal Processing, vol. 46, issue 1, pp. 239-243, January 1998.

[5] D. Mukherjee and S. K. Mitra, "Vector set partitioning with successive refinement Voronoi lattice VQ for embedded wavelet image coding," IEEE International Conference on Image Processing, vol. 1, pp. 107-111, Chicago, USA, October 1998.

[6] David Taubman, "High Performance Scalable Image Compression with EBCOT", IEEE transactions on image processing, vol. 9, issue 7, pp. 1158-1170, July 2000.

[7] Herrn Jörg Ritter, "Wavelet based image compression using FPGAs" http://nirvana.informatik.uni-halle.de/~molitor/CDROM1999/monographien/postscripts/02_ritter_diss.pdf

Original Image (Lenna 512×512)

Rate = 0.0625bpp, PSNR = 27.35db

Rate = 0.25bpp, PNSR = 33.3db

Rate = 0.5bpp, PSNR = 36.6db

Fig.3: Images obtained using modified SPIHT without arithmetic coding.