IMPROVED SET PARTITIONING IN HIERARCHICAL TREE (SPIHT) ALGORITHM FOR IMAGE DATA COMPRESSION

V. K Singh
Pennar Infotech
#307, 5306, N. Cumberland Ave., Chicago, IL- 60656, USA
E-mail: vinay_k_singh@hotmail.com

Ajay Deshpande
Pennar Infotech
#307, 5306, N. Cumberland Ave., Chicago, IL- 60656, USA
E-mail: ajjudesh@hotmail.com

1. Introduction

Lot of work has been reported on the wavelet transform based compression techniques during last few years. Selection of appropriate mother wavelet as well as appropriate coding method suited for wavelet coefficients are two major issues addressed by various researchers[1][2]. A major breakthrough in encoding scheme for wavelet coefficients is reported by J.M.Shapiro[3] using Zero-tree method. Another implementation of this concept by Amir Said and Willam A. Pearlman [4] called SPIHT not only provided better explanation of its working but also improved the performance significantly by doing away with the transmission of the symbols for various types of nodes. Recently, Hough Jyh Wang et al have proposed multi threshold wavelet codec (MTWC) [5] where all sub bands are not quantized at every threshold, instead sub-band for quantization is selected based upon largest coefficient present which results in better rate distortion performance.

In SPIHT, crucial parts of the coding process – the way the subsets of the coefficients are partitioned and how the significance information is conveyed – are fundamentally different from the zero-tree method. Unlike Zero-Tree, here symbols for various types of nodes are not transmitted. Instead type of node is determined by the comparison result of a particular threshold. Moreover in the EZW, Arithmetic coding of the bit stream was carried out to achieve better performance. Here the subset partitioning is so effective and the significance information is so compact that even binary un-coded transmission achieves about the same or better performance than the zero- tree algorithm. However, the utilization of arithmetic coding can further reduce the mean-square error or increase the peak signal to noise ratio(PSNR) by 0.3 to 0.6 dB for the same rate or compressed file size and achieve results which are equal to or superior to any previously reported results, regardless of complexity. The transmitted code or compressed image file is completely embedded, so that a single file for an image at a given code rate can be truncated at various points and decoded to give a series of reconstructed images at lower rates. Here a new improved SPIHT algorithm is proposed which results in improvement in rate distortion performance. Section 2 briefs the improved SPIHT algorithm. Experimental results are discussed in section 3 and section 4 concludes.

2. Improved SPIHT algorithm

The following set of coordinates is used to present the SPIHT coding method.

- $O(i,j)$: set of co-ordinates of all offspring of node (i, j) ;
- $D(i,j)$: set of co-ordinates of all descendants of the node (i, j) ;
- H : set of co-ordinates of all spatial orientation tree roots (nodes in the highest pyramid level) ;
- $L(i,j) = D(i,j) - O(i,j)$

For instance, except at the highest and the lowest pyramid levels, we have

$O(i,j) = \{ (2i, 2j), (2i, 2j+1), (2i+1, 2j), (2i+1, 2j+1) \}$

We use parts of the spatial orientation trees as the partitioning subsets in the sorting algorithm. The set partitioning is simply :

- The initial partition is formed with the sets {(i,j)} and D(i,j), for all (i,j)∈H
- If D(i,j) is significant then it is partitioned into L(i,j) plus the four single element sets with (k,l)∈O(i,j)
- If L(i,j) is significant then it is partitioned into the four sets D(k,l) with (k,l)∈O(i,j)

2.1 Coding algorithm

Since the order in which the subsets are tested for significance is important, in a practical implementation the significance is stored in three ordered lists, called list of insignificant sets (LIS), list of insignificant pixels (LIP) , and list of significant pixels (LSP). In all lists each entry is identified by a coordinate (i,j), which in the LIP and LSP represents individual pixels and in the LIS represents either the set D(i,j) or L(i,j). To differentiate between them we say that a LIS entry is of type A if it representsD(i,j) and of type B if it represents L(i,j) . During the sorting pass, the pixels in the LIP, which were insignificant in the previous pass, are tested, and those that become significant are moved to the LSP. Similarly, sets are sequentially evaluated following the LIS order, and when a set is formed to be significant, it is removed from the list and partitioned. The new subsets with more than one element are added back to the LIS, while the single coordinated sets are added to the end of LIP or the LSP, depending whether they are insignificant or significant, respectively. The LSP contains the coordinates of the pixels that are visited in the refinement pass.

2.1.1 Encoder

There are four steps in the proposed implementation similar to those in Pearlman's implementation. The differences between the Pearlman's SPIHT Algorithm and the proposed algorithm are italicized .

2.1.1.1 Initialization

(a) Output $n = \log_2(\max (i,j) [C(i,j)])$ while n is minimum number of bits required to code the coefficient with maximum magnitude in the image

(b) Add (0,1), (1,0), (1,1) coordinates to the LIS (List of Insignificant Set)

2.1.1.2 Sorting Pass

- *For each of (0,0), (0,1), (1,0), (1,1) pixels*

  Output $S_n(i,j)$: $S_n(i,j) = 1$ if $C(i,j) >= 2^n$

  0 otherwise

  If $S_n(i,j) = 1$ output the sign of C(i.j)

  Update the C(i,j) as $| C(i,j) | = | C(i,j) | - 2^n$

  Add this coefficient to List of Significant Pixels (LSP)

  Replace the Coefficient value by Zero in the array

- For each entry in LIS

  Output $S_n(D(i,j))$: $S_n(D(i,j) = 1$ if $\max (| C (i,j) |) >= 2^n$

  0 otherwise, where (i,j)∈Set

  If $S_n(D(i,j)) = 1$

  For each offspring (k,1) of (i,j): O(i,j) = { (2i,2j), (2i,2j+1), (2i+1,2j), (2i+1,2j+1)}

  *output $S_n(k,l)$

  *If $S_n(k,l) = 1$

  Output sign of C(k,l)

  Update the value of $| C(k,l) |$ *as* $| C(k,l) | = | C (k,l) | - 2^n$

  Move C(k,l) to LSP

  Replace the coefficient values by Zero in the array

  If L(i,j) = ϕ remove (i,j) from the LIS

  Else

  Output $S_n(L(i,j))$: L (i,j) = D(i,j)-O(i,j)

If $S_n(L(i,j)) = 1$

Add each $(k,l) \in O(i,j)$ to the end of LIS

Remove (i,j) from the LIS

2.1.1.3 Refinement Pass

For all coefficients listed in LSP except those added during current pass, output S_n*, based upon current threshold and update the values in LSP as* $C_{new} = C_{old} - 2^n$ *, if* $S_n = 1$*.*

2.1.1.4 Quantization Step

Decrement n by 1 and go to 2.1.1.1(b)

2.1.2 Decoder

As the branching path in the encoder is based is upon S_n, which is calculated based on C(i,j) and n, decoder algorithm can be duplicated from the encoder simply by replacing 'output' by 'input' for execution of branching path, and 'subtraction' by 'addition' for updation of coefficient values in the encoder(during sorting and refinement passes).

3. Numerical Results

It can be seen from the above discussion of the proposed new implementation of SPIHT algorithm that by replacing the coefficient value for the pixel found to be significant in present sorting pass by zero, many Zero-Trees or insignificant sets are generated in next sorting pass, which causes in saving of many bits in encoding zeros. Also there is no need of maintaining LIP list thus reduces complexity in algorithm and n^{th} bits need not be transmitted for all pixels listed in LIP which are already transmitted during the sorting of significant set as bits for offspring. Hence it provides some gain in the compression. Also due to reduced complexity and by maintaining only two lists of LIS and LSP instead of four lists(LIP, LIS_a, LIS_b, LSP) in original algorithm, computational time is reduced. The algorithm was tested for the monochrome 8bpp, 512 x 512, Lena Image of Fig. 1(a) . We have used the same 6 level pyramids, constructed with 9/7-tap filter of [6] and using reflection extensions at the image edges. The sets of distortions are obtained from the same compressed file as the technique is fully embedded. Fig. 1(b) and 1(c) show the Lena picture reconstructed from the compressed file using Pearlman's and our implementation at 0.16 and 0.10 bpp respectively. It may be seen that the reconstructed image at 0.10bpp using our implementation has sharper boundaries compared to those in the reconstructed at 0.16 bpp using Pearlman's implementation. There is also no need to use the complex arithmetic coding with the proposed new implementation of SPIHT which hardly provides gain of 0.5db in PSNR at the cost of complexity and increased execution time at least by a factor of 2.

4. Conclusions

The new algorithm further simplifies the already highly optimized SPIHT algorithm by not involving any complex mathematics, while results surpass those obtained using all the zero-tree based technique reported so far, including MTWC. It is also embedded in nature thus suitable for progressive transmission. The proposed technique being simple and thus faster and having better performance is used for studying the effect of compression on various remote sensing applications.

References

1. Michael L. Hilton, Bijorn D. Jewerth and Ayan Sengupta, "Compressing still and moving images with wavelets", Multimedia Systems, Vol. 2, No. 3.
2. John D. Villesenor, Benjamin Belzer and Judy Lico, "Wavelet filter evaluation for image compression", IEEE Trans. on Image Processing, Vol. 4, No 8, Aug. 1995.
3. J. M. Shapiro, "Embedded image coding using zero-tree wavelets coefficients", IEEE Transaction on Signal Processing, Vol. 41, pp. 3445-3462, December 1993.
4. A. Said and W.A. Pearlman, "A new fast and efficient image codec based upon set partitioning in hierarchical trees", IEEE Trans. on Circuits and Systems for Video Tech., Vol. 6, June 1996.

5. Hough Jyh Wang, Yi Lang and C C Jay Kuo, " Multi-threshold wavelet codec (MTWC)", IEEE International Conference on Image Processing, Sept. 1997.
6. M. Antonini, M. Barland, P. Mathien and I. Daubechies, " Image coding using wavelet transform", IEEE Trans. on Image Processing, Vol. 1, pp. 205 – 222, April 1992.

Fig. 1(a) Original Image Lena

Fig. 1(b) Reconstructed from the compressed file using Pearlman's SPIHT at 0.16 bpp

Fig. 1(c) Reconstructed from the compressed file using improved SPIHT at 0.1 bpp