EMBEDDED LINKED SIGNIFICANT TREE WAVELET IMAGE CODER

Tanzeem Muzaffar and Tae-Sun Choi
Gwangju Institute of Science and Technology,
Gwangju, Republic of Korea

ABSTRACT

This paper presents a new linked significant tree (LST) wavelet coding method for efficient compression of images. The proposed method links all the significant coefficients together within a wavelet tree to facilitate encoding algorithm. Insignificant parents with at least one significant child found in the wavelet data are changed to significant to link them together. Their location is saved for identification, and they are then treated just like other significant coefficients during encoding. LST encoding process is performed next on the processed output for compression. It exploits the fact that since all the significant coefficients within a wavelet tree are linked together, no descendent of an isolated coefficient can be significant any more. This eliminates the need to check descendents of those coefficients that are found as insignificant. Locations of converted coordinates are sent to the decoder after each encoding pass, in order to change these values back to original during reconstruction. Better compression is achieved when the proposed method is applied to smooth images.

1. INTRODUCTION

Wavelet transform has become a very effective tool for the compression of images. Many algorithms have been proposed [1,2,6] in literature that employ wavelet transform to compress image data. By deep analysis, it has been observed that most of the wavelet transformed image data lies in descending order within a subband tree at different levels. Coefficients at higher level subbands(parents) are generally of larger values compared to coefficients at lower levels(children) of the same orientation. This assumption is exploited and parent-child relationship between subbands is used to achieve high compression. Embedded zerotree wavelet coding (EZW) by Shapiro[1] was the first to apply this relation for compression of images. Since then, many improvements have been proposed in wavelet compression techniques to make it more efficient [2,3,4,5]. SPIHT [2], another well known method was proposed later by Said and Pearlman with improved performance and faster processing.

Although the assumption that wavelet data lies in descending order within a subband is valid in most of the cases, but still there are some coefficients that do not follow it. There are some children that are of larger value than their parents. This type of coefficient i.e. a significant child with insignificant parent is termed as an isolated significant. Extra processing is required to find the isolated significant coefficients, and thus more bits are required to code a given image. This paper presents a new method that first links all the significant coefficients within the wavelet transformed data prior to compression. Then, LST encoding method is applied to the processed data for efficient encoding.

The organization of this paper is as follows: Section 2 describes the proposed LST encoding algorithm for efficient compression of an image. Section 3 shows experimental results and discussion. Finally, the paper provides its concluding remarks in section 4.

2. PROPOSED ALGORITHM

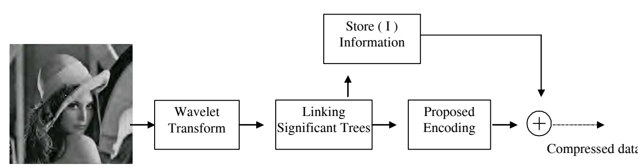
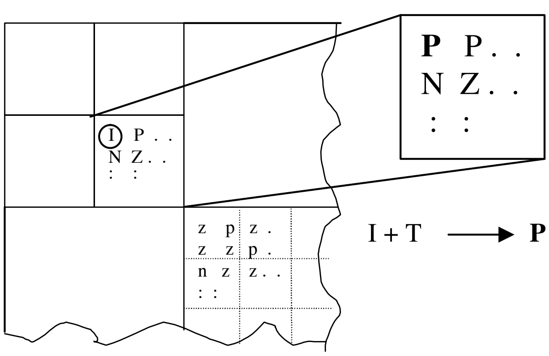


Fig. 1 Block diagram of proposed image coding algorithm

Natural images are generally smooth, so its wavelet coefficients have a certain order of decay in scale and space. But still there are several isolated significant coefficients (i.e. significant coefficient with insignificant parent) present in the wavelet transform data of an image. These coefficients reduce the coding efficiency of the algorithm. In this paper, this issue is reviewed and a new compression method is proposed which first links all the significant coefficients together within a wavelet tree at same orientation, and then applies Linked Significant Tree (LST) encoding to increase compression efficiency. LST proposed in this paper, exploits the fact that if a coefficient is insignificant, all its descendents will be insignificant and need not be checked. Figure 1 shows block diagram of proposed linked significant tree (LST) wavelet image coding algorithm. Each block of the proposed LST coder is described in detail below:

2.1 Linking Significant Coefficients:

In order to connect and link all the significant coefficients together, isolated significant coefficients are first searched in the transformed data using current threshold value, and their insignificant parents 'I' are marked for conversion as 'I=1'. The proposed method adds current threshold value to all the insignificant coefficients in the wavelet tree that lies above the isolated significant coefficient. If the insignificant parent 'I' is selected for conversion, the proposed algorithm checks its initial value(sign) and changes its value accordingly. Current threshold value is added to the coefficient 'I' when it is positive, whereas threshold is subtracted in case of negative valued coefficients. The addition (or subtraction) of current threshold in the original value makes it significant for that threshold, and in this way all the intermediate coefficients marked as 'I=1' with isolated significant child become significant. This method is repeated for all the threshold values, so that no more isolated significant coefficients exist in the transformed data. Figure 1 below shows conversion of insignificant parent of significant coefficient 'I' to significant symbol.



I = Insignificant parent coeff P/N = Significant coeff
T = Current threshold value Z = Zerotree coefficient

Fig. 2 Conversion of insignificant parent of significant coefficient 'I' to significant symbol.

A new Linked Significant Tree wavelet image coding algorithm is proposed to encode the processed wavelet data. Since all the significant coefficients within a subband tree are now linked together with no isolated significant coefficient left in the transformed image, the proposed encoder uses this property to improve compression results. It uses three lists to encode the linked wavelet data i.e. Pixel List(PL), children List(CL) and Significant List(SL). PL checks significant pixels, CL searches significant children of significant pixels, and SL stores locations of already significant coefficients in the order of occurrence along with information of converted coefficients.

At the start of encoding process, PL is initialized with all the locations in highest level subband of transformed image, whereas CL and SL are initialized to zero. LST encoder first checks all the entries in PL in its *significant pass*. If any coefficient is found significant (i.e, coefficient value >= threshold); the encoder transmits a '1' followed by its sign bit, removes its entry from the PL and adds it to SL. The location of significant coefficient is also added to CL to check significance of its children afterwards. Otherwise, if the coefficient in PL is found as insignificant (i.e. coefficient value < threshold), the encoder transmits a '0' and moves on to next entry in the list. When all the entries in PL are searched, LST algorithm starts checking CL entries to find significant coefficients in its 4 children. One bit for O(x,y) is used to indicate significance present in the 4 children of coefficient(x,y). If any child is found significant, the algorithm transmits O(x,y) as '1' showing that at least one of its child is significant. It then encodes significance of all its 4 children immediately afterwards, i.e. sends '1' along with sign bit for significant child, and '0' for insignificant child. The entry of CL is removed from the list and split into 4 children. Insignificant children locations are moved to PL, whereas significant children are added to SL, as well as at the end of CL to further search significant children of this newly found significant coefficient. Entries of insignificant children are not added to CL to search significance because no significant coefficient exists after insignificant coefficient within a linked significant wavelet tree. If no child at the CL location is found significant, send O(x,y) as '0' and the algorithm moves on to next entry of CL without encoding significance of its children. This process continues till all the CL entries are tested for significance.

Once the encoder scans all the entries of PL and CL for a certain threshold value, it sends locations of newly found significant coefficients using SL in the *update pass*. Significant List (SL) serves two purposes: first it holds locations of significant coefficients (x,y), and secondly it stores information about changed significant coefficients in the order of occurrence to keep track of converted coefficients. Converted coefficients marked as 'I=1' are stored with a '1' in SL for their identification. This stored information of 'I' locations is transmitted to decoder in this pass, which is used to revert the values for proper reconstruction during decoding. Decoder identifies converted significant coefficients 'I' using identical SL maintained there, and changes them back to their original value. *Update pass* is followed by *refinement pass* where algorithm sends refinement bits of already significant coefficients. For this purpose, one next bit value of locations stored in SL are sent to decoder. Afterwards, the encoder starts the *significant pass* again and searches PL and CL again with a reduced threshold value, followed by *update* and *refinement passes*. The process is repeated till the desired bitrate is achieved. Figure 3 shows some common scenarios of coefficient distribution in wavelet data, and their coding results using LST compared with SPIHT.

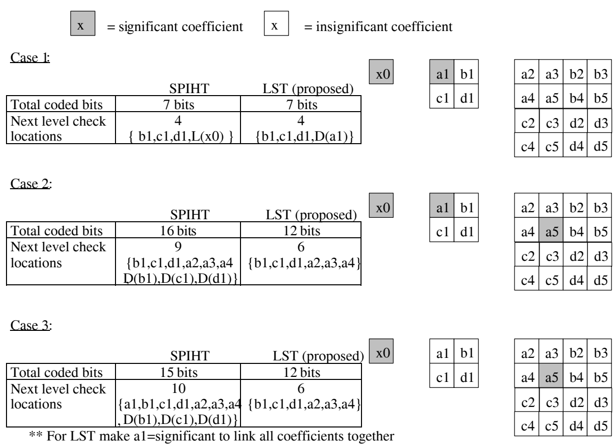


Fig. 3 Some common scenarios of coefficient distribution in wavelet data

2.2 Coding Isolated-zero Locations 'I':

The proposed method encodes the isolated zero information 'I' among newly found significant coefficients in the *update pass*. Significant coefficient locations found in the current *significant pass* are first stored in significant list (SL) in the order of occurrence. These significant coefficients of current pass in the list then search for significance among its children (if any). One bit of 'I' information is sent for those coefficients that have at least one significant child. This is done to differentiate isolated zero coefficients 'I' that are temporarily converted to significant, from real significant coefficients of the current pass. A '0' is sent to the decoder when a real significant coefficient is present, whereas a '1' bit is encoded for 'I' coefficients. The prior checking of significant children before sending 'I' information is done because converted coefficients 'I' can only exist when coefficient has significant child(ren) present in the wavelet tree. Therefore extra bit for 'I' information is sent only when coefficient's significant child exists, thus significantly improving the encoding of 'I' information to identify isolated zeroes among the significant pixels. Figure 4 shows an example of subband tree with significant coefficients. In this case, a total of 2 bits of 'I' information is sent, i.e, one bit each for coefficients 'a' and 'b' (assuming 'a' to be a newly found significant coefficient), as they are the only significant coefficients with significant children.

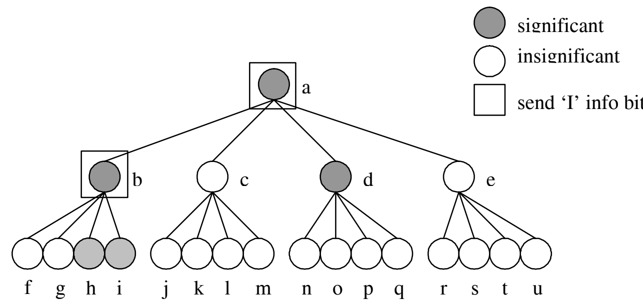


Fig. 4 Coding Isolated zero locations 'I' among significant symbols in subband tree

With the help of identical SL maintained at decoder, this technique is used to identify all 'I' symbols among newly found significant symbols of the current pass, and original data can be recovered by simply eliminating first nonzero MSB bit (threshold value at the time when 'I' was made significant) of these coefficients. Figure 5 shows the process of converting 'I' coefficient values back to their original values.

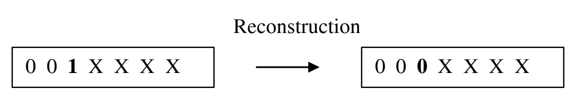


Fig. 5 Converting values back to original by removing first non-zero MSB bit during reconstruction.

2.3 Overhead:

The proposed algorithm uses a new approach to reduce number of symbols in zerotree based coding. The overhead in the proposed algorithm is that after 'I' becomes significant, it starts sending remaining bits in subordinate passes, these bits are useless until true significant bit is sent for that coefficient. Still, bits saved by removing isolated zeroes are greater than what is sent as overhead.

3. EXPERIMENTAL RESULTS

The proposed algorithm was implemented in software and experiments were performed to verify the results. Several 8-bit grayscale images were used for experimental purposes. The input image was first decomposed into five-level wavelet transform. Processing of transformed output was done before compression to link all significant coefficients in a wavelet tree. Linked Significant Tree wavelet coding methods was applied next on the resultant data. The algorithm used z-scan coding approach to scan significant pixels starting from highest level. Locations of converted coefficients 'I' were also sent to decoder to ensure proper reconstruction. Finally, reconstruction was done at receiver to decode the coded bitstream data.

When this method was applied to 512x512 gray-scale images, better compression was achieved compared to SPIHT algorithm, for similar reconstructed image quality. Results of encoded bitstream of the proposed LST encoding method are compared with that of original SPIHT and shown in Table 1, whereas their reconstructed PSNR results are stated in Table 2. These results show that the proposed LST coder improves the compression ratio of an image especially when they are applied to images with smooth texture. Figure 6 shows reconstructed gray-scale images after compression using the proposed method.

Table 1 Comparative results of LST with SPIHT

No. of Passes	Image (512x512	SPIHT (bytes)	LST (bytes)	PSNR (dB)	Gain (%age)
6	Pepper	3071	2994	29.47	2.50 %
	Lamp	1364	1362	32.98	0.15 %
	Tiffany	3310	3266	27.70	1.33 %
	Cow3d	2451	2324	38.75	5.18 %
	Splash	1915	1931	31.87	-0.83 %
7	Pepper	6258	6071	32.38	2.99 %
	Lamp	2872	2822	36.18	1.74 %
	Tiffany	8348	8205	30.76	1.71 %
	Cow3d	4969	4694	43.16	5.54 %
	Splash	3828	3750	35.62	2.03 %

Table 2 PSNR Comparison of LST with SPIHT

Image	Bitrate(bpp)	0.1	0.2	0.3	0.4	0.5
Pepper	LST	29.78	32.62	34.00	35.03	35.49
	SPIHT	29.66	32.50	33.96	34.91	35.61
Splash	LST	34.81	38.24	40.06	41.44	42.12
	SPIHT	34.71	38.04	40.07	41.15	42.16
Lamp	LST	36.74	40.68	43.09	44.85	46.20
	SPIHT	36.68	40.56	42.94	44.81	46.09
Cow3d	LST	40.44	45.72	49.76	52.74	54.46
	SPIHT	40.12	45.21	49.30	52.17	54.07
Squares	LST	53.44	66.88	-	-	-
	SPIHT	53.05	66.43	-	-	-

(a) Lamp (36.18 dB)   (b) Pepper (32.38 dB)

Fig. 6 Reconstructed images after 7th encoding pass

4. CONCLUSIONS

A new linked significant tree (LST) based wavelet image coding method is proposed. It first links all the significant coefficients together in the wavelet transform data, and then compress it using LST image coding algorithm for better results. Experiments show improvement in results when the proposed method is applied to smooth images. Complexity of the coding algorithm is also reduced for easy implementation and fast execution, a property useful in hardware design.

ACKNOWLEDGEMENT

This work was supported by the Korea Research Foundation Grant (KRF-2003-041-D20470).

REFERENCES

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

[2] A. Said, W. Pearlman, " A new, fast and efficient image codec based on set partitioning in hierarchical trees", IEEE Transactions on Circuits and Systems for Video Technology, pp 243-250, vol 6, Jun 1996.

[3] J. Wu, C. Oliver, C. Chatellier, "Embedded zerotree runlength wavelet image coding" ICIP Proceedings, pp 784-787, Oct 2001.

[4] C. Hsieh, Y. Chen, Y. Wu, F. Kuo, "Embedded image compression using the important oriented tree", ISMSE Proceedings, pp 432-436, Dec 2000.

[5] I. Sodagar, H. Lee, Y. Zhang, "Scalable wavelet coding for synthetic natural and hybrid coding", IEEE Transactions on Circuits and Systems for Video Technology, Mar 1999.

[6] D. Taubman, "High performance scalable image compression with EBCOT", IEEE Transaction on Image Processing, pp 1158-1170, vol 7, Jul 2000.

[7] http://cmp.felk.cvut.cz/~fojtik/gallery/gray.htm.