Listless Block Tree Coding with Discrete Wavelet Transform for Embedded Image Compression at Low Bit Rate

Chandandeep Kaur
UIET, Panjab University
Chandigarh [India]

Sumit Budhiraja
UIET, Panjab University
Chandigarh [India]

ABSTRACT

Listless Block Tree Coding (LBTC) is Wavelet Block Truncation Coding (WBTC) algorithm without using lists. LBTC is evolved from two variants of Set Partitioning In Hierarchal Trees (SPIHT) which are Wavelet based Block Tree Coding (WBTC) and No list SPIHT (NLS). WBTC works on blocks with varying root sizes instead of pixels so it lowers the memory requirement as compared to SPIHT and uses three ordered auxiliary lists to keep track of the significant coefficients similar to SPIHT. SPIHT uses lists which keep on increasing at each pass and require o lot of memory, so another variant of SPIHT is NLS which uses a state table to store the significant coefficients. This state table uses four bits for each coefficient. In LBTC, listless variant of WBTC is implemented using the concept of NLS in which instead of lists, markers are used. This paper presents the proposed algorithm in which LBTC is combined with DWT and the quality of compressed image is improved significantly by optimizing the Peak Signal to Noise Ratio (PSNR) and Mean Square Error (MSE). It outperforms other compression methods by a wide margin in terms of PSNR and MSE though the time taken by the algorithm is more than LBTC.

General Terms

Algorithm, Bit rate, Compression, Markers, Memory Management, Optimize, Root Size

Keywords

SPIHT, NLS, Block Tree, DWT, PSNR, MSE, EZW

1. INTRODUCTION

SPIHT [1] is an image compression method introduced by Said and Pearlman in 1996 in which the basic principle is to use set partitioning so that spatial self similarity across different scales can be exploited. SPIHT by using parent–child relationship order the wavelet coefficients into sets and check their significance at successively finer quantization levels. The complexity of SPIHT cannot be estimated easily as it depends upon the source image and decomposition depth [2]. Even if the compression ratio is noticeably high, the quality of reconstructed image is very good. The basic idea of SPIHT is that, it applies progressive coding and image is processed at decreasing threshold at each level. Main concept is of zero trees [3], which consider relationship between coefficients in subbands at different levels. If in a particular subband a coefficient at the highest level of transform is considered insignificant when compared with a predetermined threshold, its descendants at the lower levels will also be insignificant which means only a single symbol is enough to code many insignificant coefficients. SPIHT is an efficient image compression technique as it's a fully embedded codec with progressive image transmission [4] and error protection [5]. It has another advantage that it gives really good results even if a small part of file is downloaded and a compact output bit stream can be produced which will have large bit variation so does not require any additional entropy coding. Still it has many disadvantages as significant image distortion can be introduced by just a single bit if it is at critical location. It has bit synchronization property and leakage in a bit can lead to complete misinterpretation from decoder side. So various improvements have been done in the SPIHT in terms of various parameters as speed [6,7], redundancy [8], quality [9], error resilience [10,11,12], complexity [13], memory requirement [14] and compression ratio [15]. One of the variants of SPIHT, No List SPIHT (NLS) [16] improves speed by replacing the list structure of SPIHT with a state table to store the significant coefficients. In SPIHT three lists are used to store the significant coefficients and as the list nodes are increasing at each pass and grow exponentially, a lot of memory is required, hence increasing the complexity and also making it undesirable for hardware applications. The state table of NLS has four bits for a single coefficient for tracking the set partitions. NLS is a low complexity image codec with performance nearly same as SPIHT though with a drawback that at lower bit rates, the coding performance is not efficiently used in these codecs. Insignificant coefficients in NLS are increased at each pass and are coded so with an increase of non zero rate, the zero distortion reduces [17]. Another improvement in SPIHT in terms of memory requirement is introduction of WBTC [18] algorithm which uses block trees compared to SPIHT and is based on Spatial Orientation Trees (SOT) [19]. It fuses both zero tree and zero block algorithms. Significant blocks are located in WBTC by using the idea of SPIHT and each block has significant coefficients which are located using the concept of quad tree partitioning [20]. As a block tree is created in WBTC so a single SOT replaces various SOTs of SPIHT. WBTC uses three ordered auxiliary lists to store the significant coefficients which make it inefficient for hardware implementation as it requires a lot of memory management as the list nodes grow at a high rate on each pass. So LBTC [21], the listless implementation of WBTC is introduced which has markers to store the significant coefficients at each pass instead of lists. LBTC is combined with DWT to further improve the quality of compressed image by optimizing various parameters as PSNR and MSE though the time taken for the execution of both algorithms is same.

2. LISTLESS BLOCK TREE CODING

LBTC algorithm is evolved by combining the concept of two variants of SPIHT which are WBTC and NLS. From the concept of NLS, fixed size array is used to store the state

information. This is an array of coefficient values in which each coefficient has four bits. LBTC uses eight markers to store the significant coefficients [22]. In this a coefficient is not addressed by two coordinates but linear indexing is used hence a one dimensional array is formed by mapping two dimensional array of coefficients. Using the concept of Morton scan sequencing, shown in figure 1 [23] insignificant block trees are skipped efficiently. Encoding process of this algorithm is performed bit plane by bit plane starting from the most significant bit plane to the least significant bit plane.

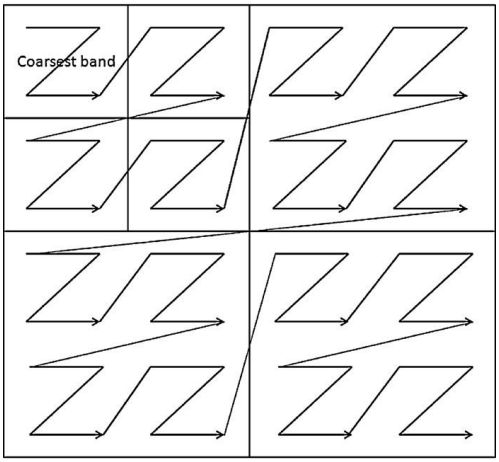

Fig 1: Mortan Scan Sequence

2.1 The Proposed LBTC-DWT Algorithm

While using DCT, blocks are formed by dividing an image and each block goes through transformation. This creates a series of frequency components corresponding to detail levels of an image [24]. Significant coefficients are stored using some coding technique. DWT performs multi-scale image decomposition and reveals data redundancy very effectively by applying filtering and sub sampling. Data is compressed using some coding technique. There are two ways to implement DWT. Convolution (filtering) is applied in first method which handles boundaries in an appropriate manner and the second method is a fast lifting approach, which gives same result as the first approach by applying filters in a different way and leads to significant saving of computations and memory. As an image is a two dimensional array of pixels, the transform is applied rows and columns wise. So when the transform is applied at first level, four subbands arise, one is low-pass subband which contains the coarse approximation of the source image and is represented as LL and there are three high-pass subbands which exploit details of an image across various directions – HL is for horizontal, LH is for vertical and HH is for diagonal details. At the second level of the transform, the LL band is used for further decomposition and respective four subbands develop form it[25]. The three level decomposed image is as shown in figure 2. LBTC-DWT codec provides efficient image compression by improving the quality of compressed image in terms of various parameters as PSNR and MSE [26].

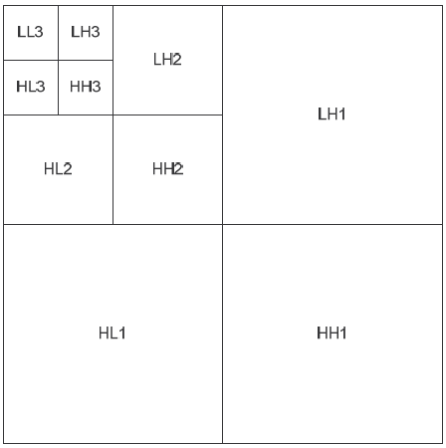

Fig 2: Three level wavelet decomposition

Markers used in LBTC are:

BIP- pixel is insignificant for current bit plane

BNP- newly significant pixel and will not be refined for current bit plane

BSP- pixel is significant and will be refined for current bit plane

BDS- first child pixel in a single tree consisting all descendants

BCP- same as BIP but it will be tested for significance in the current pass

SD- first child pixel in a composite tree consisting descendants

SG- first grandchild pixel in a tree of grand descendants

SN- used on the main nodes at each generations of an insignificant block tree

2.2 Steps of LBTC-DWT Encoder algorithm

2.2.1. Set initial threshold:

T= $\log_2 \max(c(i,j))$, a coefficient s is significant if $s \geq 2^T$. At decoder, when a significant bit arrive it's reconstructed as $\pm 1.5 \times 2^T$. 2^T-1 is added to its reconstructed value if the significant bit is in current pass otherwise subtarcted

2.2.2 Insignificant pixel pass:

For a pixel at i^{th} position with i=0 and i≤ I;

- if insignificant, it is marked as BIP and d= val[i] and s is output

- Further check the significance and mark it as BNP if significant and output d= sign[i] and move to next coefficient

- otherwise skip to next block

2.2.3 Insignificant set pass:

- If block tree of descendants is significant, mark it as SD

- Quad split and mark as BDS

- Set of grand descendants marked as SG

- If grand descendant block tree is significant, it's quad split and and marked as SD else skip to next grand descendant block tree

- After reaching to a single tree, check it's significance and mark as BCP if significant otherwise skip to next tree

- The significance of each coefficient inside the single tree is checked and marked as BNP if significant else BI

2.2.4 Refinement pass

- The pixels marked as BSP are refined in the same pass

- The pixels marked as BNP are significant for next pass and are marked as BSP in next pass and then refined

- If a pixel is marked as BIP, it is skipped and move to the next block

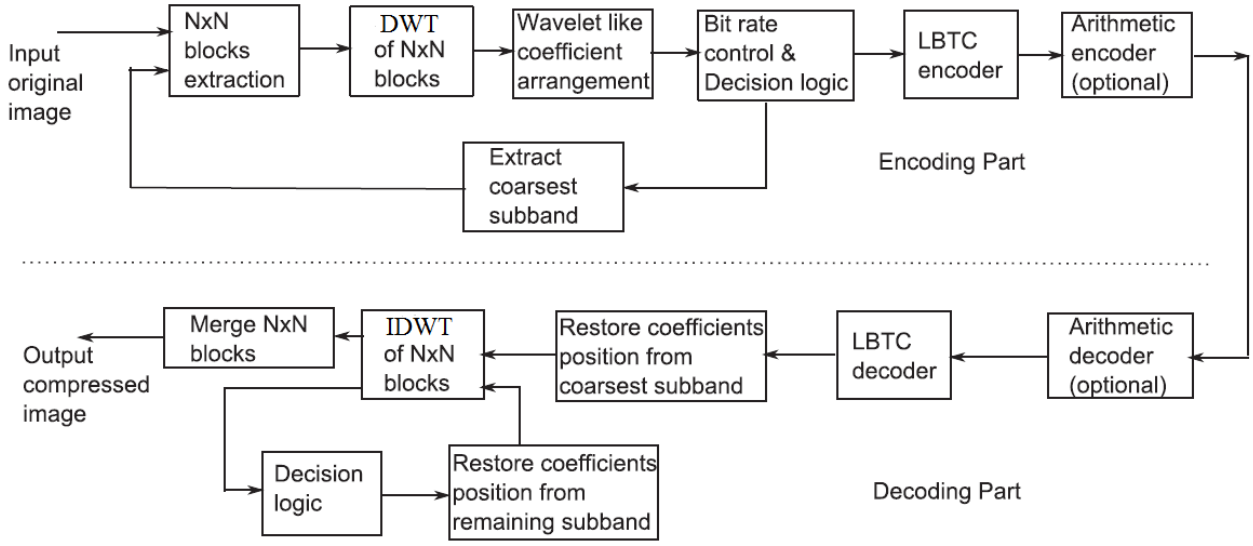

Fig 3: Block Diagram of Proposed LBTC-DWT Algorithm

3. RESULTS AND PERFORMANCE ANALYSIS

The performance of LBTC-DWT is compared with other compression methods as LBTC, WBTC, NLS and SPIHT. The proposed algorithm is found to be better than other methods by a wide margin in terms of various parameters as memory requirement, image quality measured by PSNR and MSE.

3.1 Memory Requirement

Number of block trees required by LBTC is less than NLS and SPIHT. As if a root block is of size (2×2), (4×4), and (8×8) LBTC requires 1/4, 1/16 and 1/64 times lesser block trees respectively when compared to NLS. So as compared to NLS, the initial cost required by LBTC is 25%, 6.25% and 1.25% less respectively. Due to block based nature of WBTC used in LBTC, the number of coefficients arre more than list nodes. So the total memory occupied by LBTC is less than SPIHT which is almost one third of the memory required by SPIHT. The number of coefficients in the DC band is $I_{dc}=R_{dc} \times C_{dc}$, where $R_{dc} = R\times 2^{-L}$, $C_{dc} = C\times 2^{-L}$. R is the number of rows, C is the number of columns and L is the number of subband decomposition levels. So the total memory required by SPIHT is

$M_{SPIHT} = RC /4 \times (8W + 1)$

And the memory required by LBTC is

$M_{LBTC} = IW + RC/ 2$

$M_{WBTC}= \sim 0.95\times M_{SPIHT}$

3.2 Image Quality

Quality of the compressed image is measured by analyzing various parameters as PSNR and MSE. These parameters are improved by a large extent using the proposed algorithm when compared to other compression methods. Table 1 shows the comparison of various algorithms in terms of PSNR and MSE for gray scale Lena and Barbara image respectively. Table 2 shows the comparison PSNR and MSE of colored Lena and Barbara images with LBTC-DWT algorithm. Figure 4(a) and Figure 4(b) shows the colored original and compressed images of Lena using the proposed LBTC-DWT algorithm.

Table 1. Comparison of PSNR Values of LBTC-DWT, LBTC, WBTC, NLS, SPIHT For Gray Scale Lena and Barbara Image (in dB)

Image	Algorithm	PSNR	MSE
Lena	LBTC-DWT	48.35	42.25
	LBTC	24.67	54.67
	WBTC	22.89	56.76
	NLS	21.97	57.12
	SPIHT	21.94	57.34
Barbara	LBTC-DWT	46.56	44.56
	LBTC	22.95	55.23
	WBTC	20.32	57.13
	NLS	20.53	58.67
	SPIHT	19.84	58.95

Table 2. Comparison Of PSNR And MSE Of Gray Scale And Colored Images Using LBTC-DWT Algorithm

Image	PSNR	MSE
Lena (gray)	48.35	42.25
Lena (colored)	82.51	24.23
Fruit (gray)	47.56	43.86
Fruit (colored)	80.86	25.36

Fig 4(a): Original Lena Image

Fig 4(b): Compressed Image By LBTC-DWT Algorithm

Figure 5(a) and 5(b) shows the colored original and fruit images using LBTC-DWT algorithm. From the analysis done by observing the PSNR and MSE values of gray scale Lena and Barbara images using various algorithms its shown that the proposed algorithm outperforms other compression methods by a wide margin by optimizing the PSNR and MSE values to large extent. Also the algorithm performs very efficiently for colored images as the quality of colored images after compression is lot better than gray scale images. In future the prposed algorithm can be combined with neural networks to furthur optimise the quality of compressed image.

Fig 5(a): Original Fruit Image

Fig 5(b): Compressed Fruit Image By LBTC-DWT Algorithm

4. REFERENCES

[1] Said A, Pearlman WA., "A New fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", IEEE Trans Circuits Syst Video Tech., Jun. 1996;6(3):243–50

[2] J.M Shapiro, "Embedded Image Coding Using Zero Trees of Wavelet Coefficients", IEEE trans on signal processing, Dec. 1993;(41):3445-3462

[3] W.K.Lin, N.Burgess, " Listless Zero Tree Coding for Color Images", in Proc of 32nd Asilomar Conf on Signals, Sys. And Computers, Nov. 1998;(1):231-235

[4] C.L Tung, T.S Chen, " A New Improvement of SPIHT Progressive Image Transmission", Proc of IEEE 5th Int. Symposium on Multimedia Software Engineering, Jun. 2003;(7)

[5] E.Khan, M.Ghanabari, "Error Detection And Correction Of Transmission Errors In Spiht Coded Images", IEEE, ICIP, Jun. 2002:689-692

[6] H.Minghe, Z.Cuixiang, " Application Of Improved SPIHT for Multispectral Image Compression", 5th Int. Conf. On Computer Science & Education, China, Aug. 2010:1058-1061

[7] Y.Jin, H.Lee, "A Block-Based Pass-Parallel SPIHT Algorihtm", IEEE Tran Circuits And Systems Video Tech.,(22) July 2012

[8] L.Zhu, Y.Yang, "Embeded Image Compression Using Differential Coding and Optimization Method", IEEE, 2011

[9] J.Zhu, S.Lawson, "Improvements to Spiht for Lossy Image Coding", IEEE, Jan. 2001

[10] S.Zaibi, V.Kerbaol, "Efficient Source and Channel Coding for Progressive Image Transmission over Noisy Channels", IEEE, Feb. 2002

[11] M.A.Khan, E.Khan, "Error Resilient Technique for SPIHT Coded Color Images", IEEE, Sept. 2009

[12] Y.Hue, W.A Pearlman, "Progressive Significance Map and Its Application to Error-Resilient Image Transmission", IEEE Trans. Image Processing, (21) No. 7, July 2012

[13] P.Singh, M.N.S.Swamy, "Block Tree Partitioning for Wavelet Based Color Image Compression", IEEE, ICASSP, Jun. 2006

[14] Y.Sun, H.Zhang, "Real-Time Implementation of a New Low-Memory SPIHT Image Coding Algorithm Using DSP Chip", IEEE Trans. Image Processing, (11)9, Sept. 2002

[15] T.Brahmi, A.Melit, "Improvements to SPIHT for Lossless Image Coding", IEEE, Jun. 2006

[16] F.W.Whheler, W.A,Pearlman, "SPIHT Image Compression Without Lists", in Proc IEEE, ICASSP, (4),Jun. 2000:2047-2050

[17] J.W.Han, M.C.Hwang, "Vector quantizer based block truncation coding for color image compression in LCD overdrive," IEEE Trans. Consumer Electron., (54)4, Nov. 2008, pp.1839–1845

[18] R.Praba1, C.Vasanthanayaki, "Enhanced Wavelet Block Tree Based Image Coding Algorithm", Int. Conf. on Control, Automation,Communication And Energy Conservation, Jun. 2009

[19] Pearlman W.A., Islam A., Nagaraj N., Said A., "Efficient low complexity image coding with set-partitioning embedded block coder', IEEE Trans. Circuits Syst. Video Technol., 2004(14) pp. 1219–1235

[20] Munteanu A., Cornelis J, "Wavelet Image Compression – The Quadtree Coding Approach. IEEE Trans. on Information Technology.in Biomedicine,1999 (3), :176–18

[21] R.K.Senapati, U.C.Pati, "Listless block-tree set partitioning algorithm for very low bit rate embedded image compression", International Journal of Electronics and Communications (AEÜ), 2012

[22] Liu G., Zeng X., "A novel direction-adaptive wavelet based image compression.", Int J Electron Commun, Elsevier 2010;64(6):531–9

[23] Chai BB., Vass J., "Significance-linked connected component analysis for wavelet image coding", IEEE Trans Image Process 1999;8(6):774–84

[24] Pan H., Siu WC., "A fast and low memory image coding algorithm based on lifting wavelet transform and modified SPIHT", Signal Process: Image Commun., 2008;23(3):146–61

[25] Douak F., Benzid R., "Color image compression algorithm based on DCT transform combined to an adaptive block scanning. Int J Electron Commun, Elsevier 2011;65(1):16–26.

[26] Davis GM., Chawla S., "Image coding using optimised significance tree quantization" in IEEE data compression conference. 1997. p. 387–96.