Block Tree Partitioning for Wavelet Based Color Image Compression

Pramit Singh, M.N.S. Swamy, R.Agarwal
Department of Electrical and Computer Engineering
Concordia University,
Montreal- Quebec-Canada

ABSTRACT

This paper presents a new algorithm for wavelet based image compression by exploiting zero tree concept in the wavelet decomposed image. The algorithm has a big edge over previously developed wavelet based image compression algorithms in that it utilizes inter and intra band correlation simultaneously, something that previous algorithms failed to exploit. Besides the improvement in coding efficiency, the algorithm also uses significantly lower memory for computation and coding thereby reducing the complexity of the algorithm. The striking feature of the algorithm is pass independent coding that makes it suitable for application to error protection schemes and makes it less vulnerable to data loss due to noisy communication channel. The algorithm codes all the color bands independently thus enabling differential coding for the color information. The paper starts with the discussion of the concept underlying the algorithm and then sees the algorithm in a broader light. Comparisons have been made to SPIHT, the well known zero tree coder for wavelet based image compression in terms of coding efficiency, memory requirements and error resiliency.

1. INTRODUCTION

Due to energy compaction property of transform coding, it is common that a large percentage of coefficients are quantized to zero value after coarse quantization. Some specially designed zero tree coding methods can thus substantially improve efficiency of an image coding system. For embedded wavelet image coding, a special data structure like a tree is created so that a large number of insignificant coefficients can be compactly grouped in an insignificant set (or zero set). This is known as spatial orientation tree (SOT) and insignificant set is called zero tree. An attractive feature of the zero tree coding is the spatial orientation of the data in the tree. As all subband coefficients from a zero tree are approximately related to same input image area. Hence it is particularly efficient for exploiting the correlation in the spatial domain, such as region of interest (ROI) and memory constrained image coding.

Among the various attempts to improve performance of zero tree coding, set partitioning in hierarchical trees (SPIHT) developed by Said and Pearlman [3] is particularly successful. Combined with efficient list management, SPIHT algorithm provides quality embedded bit-stream. It gives substantial improvement over earlier methods like EZW in both speed and compression performance and is widely recognized as the benchmark wavelet coder in the image coding community. SPIHT uses wavelet subband decomposition and imposes a quad tree structure across the subbands in order to exploit the inter-band correlation.

The capability to encode a large image without storing the entire image in memory is an important feature in the JPEG2000 requirements specification. Though SPIHT [1] was a contender for JPEG2000, but due to the use of pixel/set lists that significantly increases the computational memory requirements, it was subsequently rejected. Besides SPIHT is highly error sensitive which makes it highly unsuitable for real world storage and communication. Algorithms like (EBCOT) [3] , (included in JPEG2000) restrict the memory usage through encoding of pixel blocks but they are of higher complexity due to the use of adaptive arithmetic coding, multiple coding passes & use of rate distortion optimizers. Other algorithms like No List SPIHT [4] try to replace the lists in SPIHT but demand fixed memory arrays independent of the required bit rate. This makes them highly inefficient for low bit rate coding.

The new algorithm introduced in this paper requires significantly less working memory by refreshing the coding information for each bit plane. The algorithm works with only two sets of addressing information for the image data which are reinitialized after each bit plane is coded.

The paper gives a brief overview of zero tree coding as applied in SPIHT, then the new algorithm has been discussed in section 3. Part 4 of the paper highlights the advantages of this algorithm as compared to SPIHT in terms of memory efficiency, error resiliency and coding efficiency.

2. SPIHT

The SPIHT is a combination of bit plane based coding, uniform quantization and an optional entropy coding. It is based on hierarchical set partitioning, which can be thought as a divide and conquer strategy. In a set partitioning coder, a spatial orientation tree (SOT) is created such that a large number of insignificant coefficients can be compactly grouped in a set. A partitioning rule is used to divide a given set into smaller subsets so that insignificant coefficient can be efficiently isolated.

The SPIHT algorithm imposes a hierarchical quad-tree data structure on a wavelet transformed image and groups the wavelet coefficients into three lists: The list of insignificant sets (LIS), the list of insignificant pixels (LIP), and the list of significant pixels (LSP). An initial threshold is chosen as $T(0)=2^{n0}$, where n_0 is selected such that the largest pixel magnitude, $T(n)= 2^{n0-n}$, where n=0,1,2 …..

Since the thresholds are a power of two, the encoding methods can be thought of as "bit-plane" encoding of wavelet coefficients. The pixels with magnitude satisfying $T(n) < |x| < 2T(n)$ are identified as "significant" and their position and sign bit encoded. This process is called a sorting pass. Then every pixel with magnitude at least 2T(n) is "refined" by encoding the nth most significant bit. This is called a refinement pass. The encoding of significant pixel position, and scanning of pixels for refinement, is efficiently accomplished using the LIP, LIS and LSP. The SPIHT algorithm can be found in [3].

Some of the drawbacks of SPIHT are obvious. Firstly, lists are initialized only once in the beginning and they continue to increase during the encoding and decoding process. This increases the memory requirement. Second, once an element enters into LIP at least one 'zero' bit per bit-plane is generated until it becomes significant and transferred to LSP. Third, encoding and decoding of a bit-plane very much depend on the previous bit-planes as lists updated in previous bit-planes are used in subsequent bit-planes.

3. BLOCK PARTITIONING CODER

In SPIHT, the lists are initialized only in the beginning and they continue to grow during the encoding or decoding process. This increases the memory requirements. Secondly once an element enters the LIP, one zero bit per plane would definitely be generated until the bit becomes significant and is moved over to LSP. Thirdly, the encoding and decoding process follows a complex sequence which depends upon and utilizes the coding information of previous bit planes. So even a single bit error during any coding pass would completely destroy the whole decoding sequence and hence would result in complete destruction of image data. Besides these practical problems, the SPIHT algorithm completely overlooks the intraband correlation (approximately 70%) present in the wavelet decomposed image which could otherwise enhance the coding efficiency.

The algorithm proposed in this paper for color image compression makes the coding process bit plane independent. Besides it also exploits intraband correlation among the wavelet decomposed bands which improves coding efficiency. Other features such as significantly less memory requirements and pass independence mark significant improvements over SPIHT.

After the 'N_d' level of wavelet decomposition of input luminance and chrominance bands, depending upon the value of the largest wavelet coefficient, the initial threshold is calculated as 2^n where $n = \left\lfloor \log_2 \left(\max_{\forall (i,j)} |c_{i,j}| \right) \right\rfloor$.

The individual LL-sub-bands of the wavelet decomposed color bands are coded separately. The coefficients of LL-sub-bands are represented in (n+1) bit sign-magnitude form. In the first (most significant) bit-plane, the sign bit and the most significant bit from magnitude of the LL sub-band coefficients are embedded in the bitstream. In the subsequent bit-planes, only the n^{th} most significant bits of LL-sub-band coefficient are transmitted. In order to encode the remaining sub-band coefficients, they are linked through spatial orientation block tree with their roots as the three quad blocks of LL band. To define a uniform child-parent relationship, it is assumed that first quarter of LL-sub-band coefficients have no descendents and remaining three-quarter of the coefficients have their children in wavelet decomposed sub-bands of correspondingly same orientation. As luminance band of the color image has the maximum information, it is the first one to be coded. Also offsets for chrominance bands are defined in the header and thus their coding starts only when they become significant.

Unlike SPIHT which needs three lists, BPC needs only two lists namely **S**et of **I**nsignificant **B**locks **(SIB)** and **S**et of **Q**uad partitioned **B**locks **(SQB).** SIB contains the information of the blocks or trees which are yet to be tested for significance. Each entry in these lists consists of the following information

1) *Address of the block*
2) *Band information: - whether it belongs to the luminance or chrominance bands.*
3) *Size of the block*
4) *Type information: - whether it has to be treated as a block or tree. 'Type'=1 denotes tree, 'Type'=2 denotes block.*

The following terms have been used consistently to explain the algorithm:

1) ***Parent block: -*** A pixel block of size $m \times n$ with address (x, y).
2) ***Child block: -*** A block of size $2m \times 2n$ with address $(2x,2y)$.
3) ***Offspring blocks: -*** Offsprings of parent block with addresses $(2x,2y)$, $(2x+m,2y)$, $(2x,2y+n)$, $(2x+m,2y+n)$ of size $m \times n$ each.
4) ***Quad Blocks: -*** Partitioned blocks of parent block with addresses $(x, y), (x+\frac{m}{2}, y)$, $(x, y+\frac{n}{2}), (x+\frac{m}{2}, y+\frac{n}{2})$ of size $\frac{m}{2} \times \frac{n}{2}$ each.
5) ***Descendant blocks: -*** All offspring blocks of the parent block and their corresponding offspring making a hierarchical tree structure [1] .

To begin with, all LL sub bands are coded. The SIB is initialized and processed in following steps.

Step1:- Initialise SIB with three quad partitioned bloks of LL sub bands (which have trees) and label them as "type 1" entries. Start checking the entries.
Step 2:- If entry is of "type1" then goto step 3, else goto step 5.
Step 3:- If tree exists for the entry then proceed to next step else check next entry.
Step 4:- If the tree is significant then replace it by its child blocks and label them as "type 2" entries, else check next entry in SIB.
Step 5:- Check the Block status. If significant then partition it into quad blocks labeled as "type 2" and call SQB, else replace it by its child block as "type 2" entry.
Step 6:- Proceed to next unchecked entry and continue till list is empty.
Once a call for SQB is made and SQB is initialized, the processing starts from the first entry as:-
Step 1:- If the entry is a pixel then change its label as "type 1", code it and goto next entry.
Step 2:- Check the significance of the block. Change its label to "type 1" if insignificant else append SQB with its offspring blocks labeled as "type 2".
Step 3:- Proceed to next entry and continue above steps till all entries are checked.
Step 4:- Send all SQB entries of "type 2" with significant trees to SIB and reset SQB.
Step 5:- Return to step 6 in SIB.
Once this whole procedure is carried out and all entries in SIB have been coded, the threshold is reduced by a factor of 2 and the SIB is initialized again for the coding of next biit plane. Some coefficients might become significant in more than one pass depending upon their absolute values, so to avoid multiple transmissions of their sign bits, a record of the sign bit transmission is maintained for all the three color bands.
It is observed that usually for the chrominance bands, the highest value of the coefficients is significantly lower than that of luminance band. So to avoid redundant information, the coding of chrominance bands is started only when these bands become significant. This information is transmitted in the image header.
At the decoder, exactly the reverse process is performed. At the arrival of first significant bit of a coefficient, it is reconstructed as $\pm 1.5 \times 2^n$. The decoder adds or subtract $2^{n-1}$to its current reconstructed value depending upon whether it inputs significant bit in the current pass or not.

4.COMPARISONS

The major features that underline the quality of any image compression algorithm are its compression efficiency, memory requirements and its ability to withstand noisy conditions.
BPC outclasses all zero tree coders in these respects. Comparisons have been made to SPIHT, as SPIHT is considered to be the benchmark in zero tree coding.

4.1 Memory requirements

Zero tree coders require fixed and variable memories to store and process the images. As fixed memory is same for any image, comparisons have been made for variable memory.
In SPIHT three linked lists are used namely LIP, LSP and LIS. Each entry in LIP and LSP is a single coordinate of a wavelet coefficient whereas LIS also requires type (A or B) information to distinguish nodes.
Let, C be total number of pixels in the image.
N_{LIP} = number of entries in LIP.
N_{LSP} = number of entries in LSP.
N_{LIS} = number of entries in LIS.
b= number of bits to store addressing information of a coefficient.
M_{SPIHT}= total memory required in SPIHT (in bits).
Then,

$$M_{SPIHT} = (b+1) N_{LIP} + (b+2) N_{LIS} + (b+1) N_{LSP} \quad (1)$$

Where an extra bit per element in LIS is needed for defining the "type" of entry and an extra bit is needed in all three lists to identify the color band for the pixel.
In the worst case,

$N_{LIP} + N_{LSP} = C$
$N_{LIS} = C/4$ (coefficients having no descendents (the highest frequency subbands) will never enter into

LIS.). Thus the maximum working memory requirement in SPIHT is

$$M_{SPIHT}^{\max} = (5b+6)\frac{C}{4} \qquad (2)$$

In the proposed algorithm only two lists have been used, but as entries correspond to address of blocks so they require twice as much memory. But as the maximum number of entries would only be a quarter of the total number of pixels, the effective bits per element would be 2(b-1). But two extra bits are also required for identifying the 'type' of entry so effective bits per element = 2b Therefore

M(new algo)= 2b.C /4 (3)

Therefore, the algorithm has been able to reduce the memory requirements by a memory reduction factor,

$$MRF = \frac{5b+6}{2b} \qquad (4)$$

For instance if we have an image of size 512x512, then b=9. Hence this algorithm would need ***three times less memory than SPIHT***.

4.2 Coding Efficiency

The new algorithm proposed in this paper, exploits the intraband correlation of the wavelet bands in the wavelet decomposed image by coding the zero trees of blocks of pixels unlike zero trees of pixels as in SPIHT. It has been observed that correlation among neighboring pixels in any band of wavelet decomposed image is around 70%. Due this high correlation, if certain pixel is below a threshold, it is highly probable that its neighboring pixels would also be below that level. So instead of transmitting individual zeros for these pixel trees, this algorithm transmits a single zero for this whole tree of blocks therefore exploiting the intraband correlation along with interband correlation.

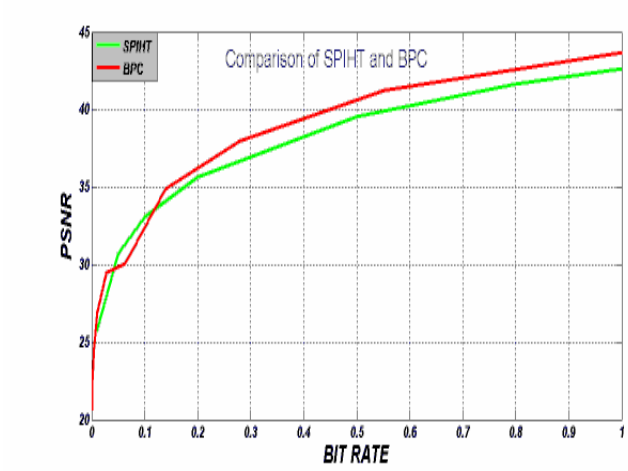

Figure:- Rate Distortion Curve for BPC & SPIHT

The coding efficiency of the algorithm is reflected in the bit rate curve which clearly shows that BPC generates higher PSNR than SPIHT at various bit rates. The curve has been plotted for color lena image of size 512x512.

4.3 Error Resiliency

The inherent problem in SPIHT is that it uses complex coding scheme that carries forward coding information of previous bit planes. As the encoder and decoder follow the same sequence, any transmission error would result in the whole image data getting corrupt.

This algorithm has made the coding process pass independent. The SIB and SQB are refreshed after each bit plane coding. Even if a transmission error occurs in any bit plane, it won't be propagated to the next bit planes. Hence the decoder would still be able to produce the image though the quality would be suffered.

5. CONCLUSION

A new algorithm has been developed that exploits the inter and intra band correlation in the wavelet decomposed image. The algorithm gives better performance than SPIHT, the benchmark zero tree coder, in terms of memory requirements, coding efficiency and error resiliency.

6. REFERENCES

[1] A. Said and W. A. Pearlman. "A new, fast, and efficient image codec based on set partitioning in hierarchal trees", *IEEE Trans. on Circuits and Systems for Video Technology*, 6(3): pp. 243–250, June 1996.

[2] J. M. Shapiro. "Embedded image coding using zerotrees of wavelet coefficients", *IEEE Trans. on Signal Processing*, 41(12): pp. 3445–3462, December 1993.

[3] D. Taubman, "High performance scalable image compression with EBCOT", *IEEE Transaction on Image Processing*, Vol. 9, pp. 1158-1170, July 2000.

[4] F. W. Wheeler and W. A. Pearlman, "SPIHT image compression without lists", *IEEE International Conference on Acoustics, Speech and Signal Processing, ICAASP 2000*, Vol. 6, pp 2047-2050, September 2000.

[5] F. W. Wheeler and W. A. Pearlman, "Low memory packetized SPIHT image compression", *33rd Asilomar Conference on Signals, Systems and Computers*, Vol. 2, pp 1193-1197, October 1999.

[6] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. "Image coding using wavelet transform", IEEE Trans. on Image Processing, Vol. 1, No. 2, pp. 205–220, April 1992.