Performance Improvement of EZW Encoding through Parallelization

Pradeep Ch
Computer Engineering Department
Indian Institute of Technology (BHU), Varanasi

Ravi Shankar Singh, Ph.D
Computer Engineering Department
Indian Institute of Technology (BHU), Varanasi

ABSTRACT
During the past few decades the wavelet transform is more and more widely used in image and video compression. One of the well known progressive encodings in image compression is the "Embedded Zerotree wavelet" (EZW) encoding, which involves the wavelet transform. As of today the parallelization of the wavelet transform is abundantly investigated, So this work deals with the parallelization of the encoding part itself. The OPENMP programming model is used for implementing the parallel version. Both the sequential and parallel versions of EZW encoding are presented along with their performance.

General Terms
Progressive encoding, EZW, zerotree, parallelization

Keywords
EZW parallelization, bitplane coding, progressive encoding, zerotree, dominant pass

1. MOTIVATION
Today multiple-core processors have become very common. Even handheld devises also have multiple-cores. This hardware will be best utilized by programs amenable to parallel computing. For some algorithms it is not possible to parallelize entire algorithm. For those types of algorithms we can only parallelize fraction of the algorithm. But if that fraction includes the heavy computation then parallelizing the fraction also produces good results. Here efforts have been made to parallelize the Dominant pass, which is part of the embedded zerotree wavelet encoding.

2. INTRODUCTION
EZW [1] is a progressive encoding technique used in wavelet based image compression schemes. It was originally designed to operate on images or 2D-signals but it can also be operated on other dimensional signals. The EZW encoding involves three steps. First step involves reading an input image which is nothing but a two dimensional matrix containing pixel values. Typically pixel values are integer values. In first step image undergoes wavelet transformation [2]. The output of it is a transformed image, which is a matrix containing different sub-bands. Now the transformed matrix contains real values or integer values depending on the wavelet filters used for transformation. If CDF 9/7 [3] filters are used in wavelet transformation, then the output would be of real values (the precision of the value depends on the system architecture). If CDF 5/3 [4] filters are used then output would be of integer values [5,6]. The second step involves quantization of transformed matrix. Quantization is used to reduce the number of bits used to represent the values in the matrix. The output of this step is an integer valued matrix. Quantization is optional for lossless wavelet transformation [6], i.e. in case of integer valued wavelet transformation. Quantization is an irreversible step, so information loss may occur in this step. From this point the integer values produced from quantization step are called as coefficients. The third step involves progressive encoding of the coefficients using Embedded Zerotree wavelet (EZW) algorithm. The typical structure of EZW encoding and decoding are given in figure-1.

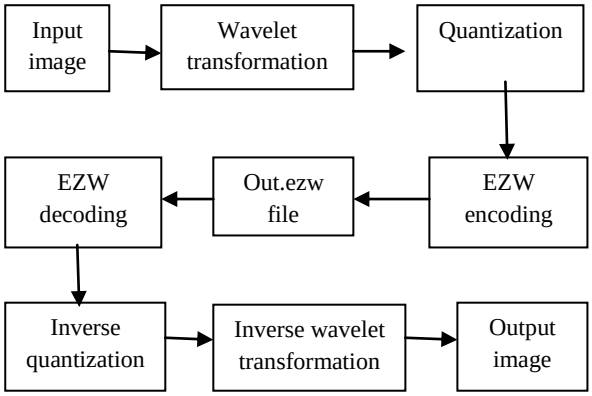

Figure-1: The typical structure of an EZW encoding and decoding

3. EMBEDDED ZEROTREES OF WAVELET TRANSFORMS

3.1 EZW Encoding
The coefficients are of two types - approximation coefficients and wavelet coefficients [7]. Approximation coefficients give low resolution of the image. Next higher resolution of the image can be produced using wavelet coefficients of same scale as that of approximation coefficients. From this produced image another next higher resolution image is generated using next low scale wavelet coefficients. Wavelet coefficients add details to the lower resolution image, to get higher resolution image [7]. Low valued coefficients don't have much importance as they add very little to the human perception of the image. But high valued coefficients are very important as they identify the boundaries or edges in higher resolution image [8].

EZW is progressive encoding. Like any progressive encoding it includes multiple scans over the transformed image. Each scan consists of two passes. First pass is called dominant pass and the second pass is called subordinate pass. The dominant pass identifies the coefficients which are, in absolute value, greater than the certain threshold value. These coefficients are known as significant coefficients. The absolute value of these significant coefficients will be added to the subordinate list and their values are replaced by zero in the transformed image. This will prevent them from being encoded again. If the dominant pass follows a scan order, then just using a very

few bits it can specify which coefficients are significant [1, 9]. The scanning order that was followed in this process is shown in figure-2.

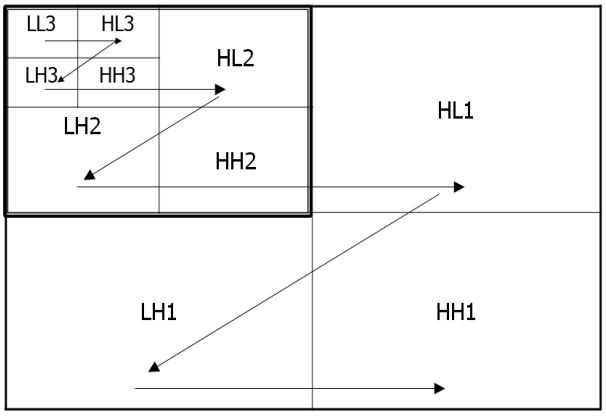

Figure-2: The scanning order of coefficients

By using following observations, the EZW compresses the data. Most of the lower scale sub-band coefficients are zero or closer to zero. Another important observation is the energy in the sub-bands decreases as the scale decreases. So the wavelet coefficients will have smaller value in the low scale sub-bands than in high scale sub-bands.

A coefficient in high scale sub-band is spatially related to four descendant coefficients in next lower scale sub-band. As shown in the figure-3. Every root coefficient has four coefficients as its leaves.

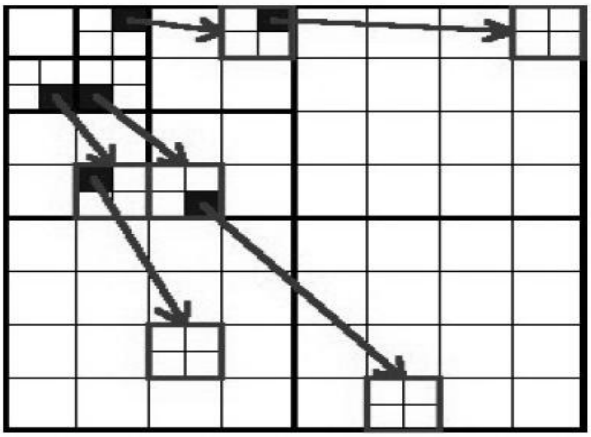

Figure-3: Relation of coefficient with other lower scale sub-band coefficients

A zero tree is a quad-tree of which all coefficients are equal to or smaller than the root. The tree is encoded with a single symbol and reconstructed by the decoder as a quad-tree filled with zeroes. To clutter this definition, also add that the root has to be smaller than the threshold against which the wavelet coefficients are currently being measured. In that case, it is efficient to encode all of the coefficients with a single symbol. For example, a single zero tree symbol in HH3 encodes 21 coefficients (1(HH3) + 4(HH2) + 16(HH1)). So in the dominant pass, if the coefficient is larger than the threshold a P (positive) is coded, if the coefficient is smaller than the negative of the threshold, a N (negative) is coded. If the coefficient is the root of a Zerotree, a T is coded and finally, if the coefficient is smaller than the threshold but it is not the root of a Zerotree, a Z (isolated zero) is coded.

The subordinate pass scans over the subordinate list, populated by dominant pass and generates a symbol for each coefficient in the list specifying roughly what the coefficient value is instead of exactly what the coefficient value is.

To fully reconstruct the image the scan is repeated several times, each time with a threshold value lower than the previous threshold value. If this sequence is a sequence of power of two, it is called bitplane coding since the threshold in this case corresponds to the bits in the binary representation of the coefficients. If bit plane coding was employed then in subordinate pass, it all comes down to outputting the next most significant bit of values in the subordinate list, and initial threshold t_0 will be $t_0 = 2^{\text{floor}(\log_2(\text{MAX}(|\vartheta(X,Y)|)))}$. Here MAX(.) means the maximum coefficient value in the image and $\vartheta(X,Y)$denotes the coefficient. A detail presentation on EZW was given in [1,9,10].

3.2 EZW Algorithm

```
//EZW encoding
threshold = initial_threshold;
do {
        dominant_pass(image);
        subordinate_pass(image);
        threshold = threshold/2;
}while (threshold > minimum_threshold)

//Dominant pass
initialize_fifo();
while (fifo_not_empty){
        get_coded_coefficient_from_fifo();
        if coefficient was coded as P, N or Z then {
                code_next_scan_coefficient();
                put_coded_coefficient_in_fifo();
                if coefficient was coded as P or N then {
                    add abs(coefficient) to subordinate list;
                    set coefficient value to zero;
                }
        }
}

//Subordinate pass
subordinate_threshold = current_threshold/2;
for all elements on subordinate list do{
        if (coefficient > subordinate_threshold) {
            output a one;
            coefficient = coefficient -subordinate_threshold;
        }
        else output a zero;
}
```

Here FIFO is used to keep track of the identified Zerotrees. Before entering dominant pass loop, FIFO has to be initialized by adding the first quad-tree root coefficients code.

4. PARALLELIZATION OF EZW ENCODING

4.1 Scope of Parallelization

For each component EZW encoding is done independently. One way of parallelizing is assigning one processing element (PE) to encode each component. The problem with this approach is that it isn't scalable. i.e., if there are three

components (RGB) and four PEs then fourth PE isn't of any use. As the EZW encoding involves several scans with various threshold values the other approach is to assign a PE to every scan and run them in parallel. The problems with this approach are scalability and algorithm modification. The maximum number of scans is equal to the number of bits used to represent the coefficients. So it is not scalable beyond that. Every scan has a sequential dependency on the scans before it. And final option is parallelizing the dominant and subordinate passes. The majority of the time is spent in the dominant pass among the both passes.

The main constraint of the dominant pass is that it should follow a scan order while producing codes for coefficients. Calculations of codes for coefficients in same sub-band are independent of each other. Using this fact, we can parallelize the dominant pass. The list used in the subordinate pass grows dynamically, so Parallelizing subordinate pass doesn't give any better result.

4.2 Parallelization of Dominant pass

In sequential execution, one FIFO and one LIST are used. For parallel execution, if N PEs are available then 6N-FIFOs and (N+1)-LISTs will be used. Each PE is assigned with 6 FIFOs and 1 LIST. Let's name these six FIFOs with the each PE as N1, N2, N3, N11, N21 and N31. Each PE distributes its FIFOs to the sub-bands as shown in figure-4. These FIFOs are used to hold the codes of the coefficients of the corresponding sub-band. The LIST is used to temporarily store the significant coefficients which will be added to the subordinate's list (MAIN LIST) after the scanning of each sub-band.

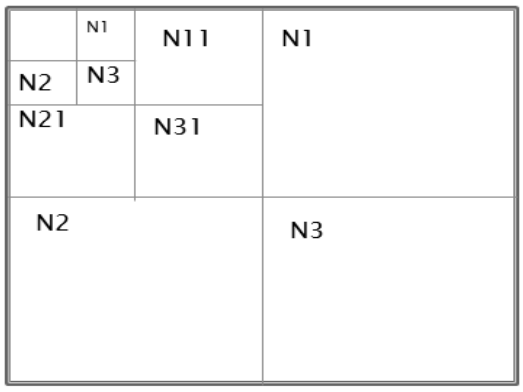

Figure-4: 6-FIFOs distribution to various sub-bands

By observing it is easy to find out that all coefficients in the HL3, LH3, and HH3 have to be scanned. Assignment of these coefficients to processing elements is important because of the scanning order that the dominant pass has to follow. If HL3, LH3, and HH3 have R rows each and assuming PEs are numbered from 1 to N, then first PE calculates codes (P, N, Z, T) of first R/N row's coefficients and stores them in the corresponding FIFO. Similarly next PE (PE-2) calculates for next R/N row's coefficients. Distributions of coefficients of the HL3 are shown in the figure-5. In the next higher scale sub-bands only those coefficients are scanned whose parents are not Zero tree. After scanning the high scale sub-bands (HL3, LH3, and HH3). Each PE's N1, N2, and N3 are populated. While passing through these FIFOs if we encounter a coefficient which is not zerotree then their child coefficients are add to the FIFO of the corresponding child's sub-band. For example, if there is a Non-Zerotree coefficient in N1, then its child coefficients are added to N11. While passing through a FIFO all the significant coefficients will be added to the LIST associated with the PE. After scanning a sub-band, all the codes and the LISTs with each PE have to be merged. LISTs are appended to the main List associated with the subordinate pass.

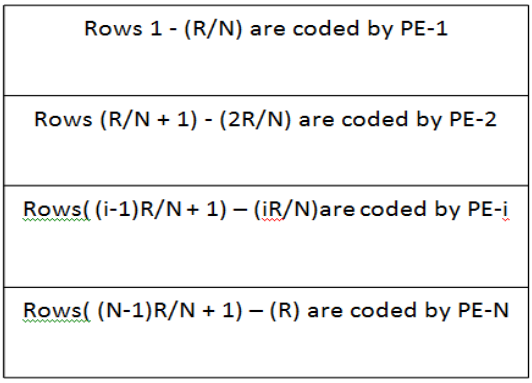

Figure-5: Distributions of coefficients of the HL3 sub-band

4.3 Merging the Codes

Significance map (binary representation of codes) associated with PEs are merged and written to the file, i.e. first PE's significance map is flushed to the file followed by the second PE's significance map then followed by third PE. Similarly we flush the entire significance map to the file. Unit length of file input/output is byte length but our significance map's unit length is bit. So merging the significance map induces extra work which has to be done sequentially.

For example if PE-1's significance map is:

1 1 0 1 1 0 0 0 1 1 - - - - - -

And PE-2 's significance map is:

0 0 0 0 1 1 0 1 1 0 1 0 1 1 - -

So after merging both PEs significance map we should write to the file in following order:

1 1 0 1 1 0 0 0 1 1 0 0 0 0 1 1
0 1 1 0 1 0 1 1

5. RESULT

Implementation is done using OPENMP and two test images. Results are shown below.Table-1 and Figure-6 show Speedups of the parallel EZW encoding when used with different number of PEs. Table-2 lists the time taken by parallel EZW encoding when used with different number of PEs.

Table-1: Speedup Vs #PE

#PE	Image-1 Speedup	Image-2 Speedup
2-PE	1.255	1.22
3-PE	1.42	1.43
4-PE	1.53	1.57

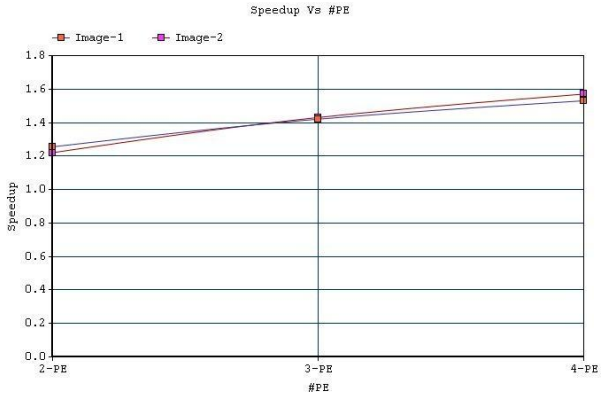

Figure-6: Speedup Vs #PE Plot

Table-2: Time taken Vs #PE

#PE	Image-1 (seconds)	Image-2 (seconds)
sequential	20.593	24.5
2-PE	17.383	20
3- PE	15.358	17.2
4- PE	14.17	15.8

Image specifications and system configurations are given below in Table-3 and Table-4.

Table-3: Test images specification

Attributes	Image-1	Image-2
Dimensions	3264 X 2448	3072 X 2304
Size	7.62MB	6.7MB
Color	RGB	RGB
Bit depth	24 bits	24 bits

Table-4: System configuration

Processor	Intel i3-CPU
Speed	2.4GHz
Ram	4GB
Word Size	64-Bit
OS	UBUNTU

6. CONCLUSION

This work presents a new way to improve the encoding time of Embedded Zerotree Wavelet (EZW) Coding by reducing the execution time of the dominant pass in the algorithm through parallelization. Since no changes in the subordinate pass have been made, the contribution of subordinate passes remains the same. The speed up observed is not linear. This is because of two reasons. First, because of a little extra work induced in dominant pass for merging the significance map associated with each PE, and second, due to sequential execution of subordinate pass. But still reasonable speedups are possible with higher number of processing elements. It is evident from this work, how parallelizing the fraction of the algorithm can also produce good result. The future scope of the proposed work lies in the improvement of video codecs such as MPEG-4 based on EZW and other wavelet transforms based video codecs. Reduced execution time of EZW will directly have an effect on video codecs and video streaming.

7. REFERENCES

[1] Shapiro, J. M. R. B., 1993, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients", IEEE Transactions on Signal Processing, vol. 41, no. 12.

[2] Fowler, J. E. (5/2003). Embedded Wavelet-Based Image Compression: State of the Art. Retrieved October 2, 2004 from the World Wide Web: http://www.extenzaeps.com/extenza/loadPDFInit?objectIDvalue:22708

[3] M. Antonini, M. Barlaud, P. Mathieu, I.Daubechies, Image coding using wavelet transform, IEEE Trans. Image Process. 1 (2) (April 1992) 205–220.

[4] D. Le Gall, A. Tabatabai, Subband coding of digital images using symmetric kernel filters and arithmetic coding techniques, in: Proceedings of the International Conference on Acoustics, Speech Signal Processing, New York, USA, April 1988, pp. 761–764.

[5] R.C. Calderbank, I. Daubechies, W. Sweldens, B.-L. Yeo, Wavelet transforms that map integers to integers, Appl. Comput. Harmon. Anal. 5 (3) (1998) 332–369.

[6] M.D. Adams, F. Kossentini, Reversible integer-to-integer wavelet transforms for image compression: performance evaluation and analysis, IEEE Trans. Image Process. 9 (6) (June 2000) 1010–1024.

[7] Akansu, Ali N.; Haddad, Richard A. (1992), Multiresolution signal decomposition: transforms, subbands, and wavelets, Boston, MA: Academic Press, ISBN 978-0-12-047141-6

[8] M. Albanesi, S. Bertoluzza, Human vision model and wavelets for high-quality image compression, in: Proceedings of the Fifth International Conference in Image Processing and its Applications, Edinburgh, UK, July 1995, Vol. 410, pp. 311–315.

[9] Algazi, V. R. and R.R. Estes. Analysis based coding of image transform and sub-band coefficients. Proceedings of the SPIE, Vol. 2564 (1995), p. 11-21.

[10] Creusere, C. D. A new method of robust image compression based on the embedded zerotree wavelet algorithm. IEEE Transactions on Image Processing, Vol. 6, No. 10 (1997),p. 1436-1442.