Compression Efficiency for Combining Different Embedded Image Compression Techniques with Huffman Encoding

Sure. Srikanth, Sukadev Meher

***Abstract*— This paper proposes a technique for image compression which uses the different embedded Wavelet based image coding in combination with Huffman- encoder for further compression. There are different types of algorithms available for lossy image compression out of which EZW, SPIHT and Modified SPIHT algorithms are the some of the important compression techniques. EZW algorithm is based on progressive encoding to compress an image into a bit stream with increasing accuracy. SPIHT is a very efficient image compression algorithm that is based on the idea of coding groups of wavelet coefficients as zero trees. Modified SPIHT algorithm and the preprocessing techniques provide significant quality (both subjectively and objectively) reconstruction at the decoder with little additional computational complexity as compared to the previous techniques. Simulation results show that these hybrid algorithms yield quite promising PSNR values at low bitrates.**

***Index Terms*—** *EZW, SPIHT, progressive encoding, Huffman encoder , PSNR and bit rates.*

I. Introduction

Recently, a lot of work has been dedicated to efficient transmission of images in multimedia communications over wireless channels. Image compression algorithms based on discrete wavelet transform (DWT)[1], such as embedded zero wavelet(EZW)[2] and the set partitioning in hierarchical trees(SPIHT)[5] provide excellent rate distortion performance with low encoding complexity. However, it is quite fragile against bit errors in wireless channels because a single bit error propagation and possible loss of synchronization. The method proposed in[7] divides the SPIHT bit stream into three sub-streams according to their importance and use of UEP(Unequal error protection) scheme to protect the divided bit stream.

Sure. Srikanth, P.G student in ECE Department,
NIT Rourkela, Orissa, India
E-mail: suresrikanth101@gmail.com

Dr.Sukadev Meher. Professor & HOD of ECE Department
NIT Rourkela, Orissa, India.
E-mail: smeher@nitrkl.ac.in

This paper contributes to the implementation of the combining of lossy image compression techniques(EZW, SPIHT, Modified SPIHT algorithms) with Huffman encoding, and also discussing their advantages and disadvantages of lossy compression techniques.

II. Compression techniques

A. *EZW Algorithm:*

The Ezw encoder was originally designed to operate on images(2D signals) but it can also be used on other dimensional signals. It is based on progressive encoding to compress an image into a bit stream with increasing accuracy[3 4]. This means that when more bits are added to the stream, the decoded image will contain more detail, a property similar to JPEG encoded images. Using an embedded coding algorithm, an encoder can terminate the encoding at any point there by allowing a target rate or target accuracy to be met at exactly[8 9]. The EZW algorithm is base on 4 key concepts: 1)a discrete wavelet transform or hierarchical sub band decomposition, 2)prediction of the absence of significant formation across scales by exploiting the self similarity inherent in images, 3) entropy-encoded successive approximation quantization, and 4) universal lossless data compression which is achieved via adaptive Huffman encoding[11].

Features of EZW algorithm:

- ✓ Zero tree coding of significant wavelet coefficients providing compact binary maps.
- ✓ Successive approximation quantization of wavelet coefficients.
- ✓ Adaptive multilevel arithmetic coding.

B. *SPIHT Algorithm:*

SPIHT is having embedded coding property it sorts the information on demand and requirement for powerful error correction codes decreases from the beginning to the end of the compressed file. If an error is detected but not corrected then decoder reject that data after point and still displays the image obtained with the bits received before the error. It generates two types of data the first sorting the information which needs error protection and second consist of uncompressed sign and refinement bits which don't need special protection because they effect only one pixel[5]. The wavelets pixel sets are divided into trees originating from the lowest resolution band. The coefficients are grouped into 2 by 2 arrays, except for the coefficient in band 1, which are offspring of a coefficient of a lower resolution band. Again the coefficients in the smallest band are also divided into 2*2 arrays. The coefficient at the top left corner does not have any offspring and is known as the root nodes[10].

SPIHT algorithm is based on following concepts[5]:

- Ordered bit plane progressive transmission.
- Set partitioning sorting algorithm.
- Spatial orientation trees.

SPIHT keeps three lists: LIP, LSP and LIS. LIP stores insignificant pixels, LSP stores significant pixels and LIS stores insignificant sets. At the beginning, LSP is empty, LIP keeps all coefficients in the lowest sub band, and LIS keeps all tree roots which are at the lowest sub band. SPIHT starts coding by running two passes[6]. The first pass is the sorting pass. It first browses the LIP and moves all significant coefficients to LSP and outputs its sign. Then it browses LIS executing the significance information and following the partitioning sorting algorithms.

The second pass is the refining pass. It browses the coefficients in LSP and outputs a single bit alone based on the current threshold. After the two passes are finished, the threshold is divided by 2 and the encoder executes the two passes again. This procedure is recursively applied until the number of output bits reaches the desired number.

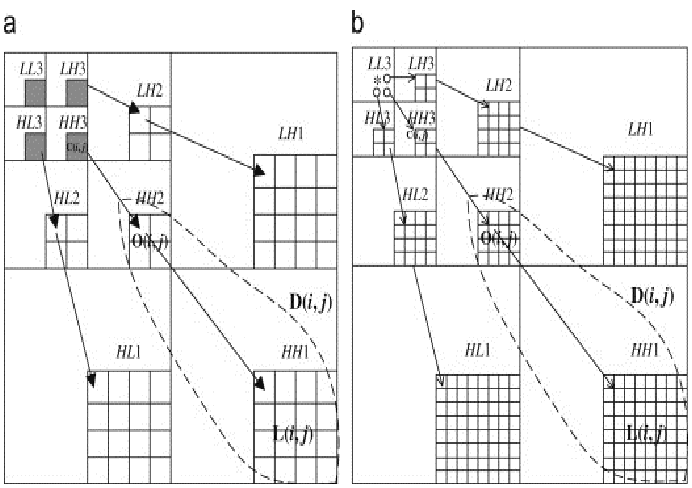

FIGURE 1. Spatial Orientation tree in 2D SPIHT

Advantages:

- We can interrupt the decoding at any time and result of maximum possible details can be reused with 1-bit precision.
- This is very useful in transmitting the files over the net since users with slower connection speed can download only a small part of the file, obtaining much more usable results when compared to the other codec such as Progressive JPEG.
- It is very compact output bit stream with large bit variation. No additional entropy coding has to be applied.

Disadvantages:

- It is vulnerable to bit corruption as a single bit error can introduce significant image distortion depending of its location.
- It consists of bit synchronization property because a leak in bit transmission can lead to complete misinterpretation from the side of decoder

C. *Huffman Encoding :*

Huffman coding is an entropy encoding algorithm used for lossless data compression. The term refers to the use of a variable –length code table for encoding a source symbol(such as a character in a life)where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It uses a specific method for choosing the representation for each symbol, resulting in a prefix code that expresses the most common source symbols using shorter strings of bits than are used for less common source symbols.

The Huffman algorithm is based on statistical coding, which means that the probability of a symbol has a direct bearing on the length of its representation. The more probable the occurrence of a symbol is, the shorter will be its bit-size representation. In any file, certain characters are used more than others. Using binary representation, the number of bits required to represent each character depends upon the number of characters that have to be represented. Using one bit we can represent two characters, i.e., 0 represents the first character and 1 represent the second character. Using two bits we can represent four characters, and so on[11].

Unlike ASCII code, which is fixed-length code using seven bits per character, Huffman compression is a variable-length code system that assigns smaller codes for more frequently used characters and larger codes for less frequently used characters in order to reduce the size of files being compressed and transferred[12].

D. *Modified SPIHT Algorithm:*

The parent offspring dependency and corresponding spatial orientation trees for the SPIHT algorithms is as shown in fig.1 for three levels of decomposition. Each node of the tree is represented by its coordinates in the algorithm. The tree is defined in such a way that each node has no offspring(leaves) or four offspring's at the same spatial location in the next finer level[5].

In original SPIHT, the pixels in the coarsest level e.g., LL sub band of the 3rd level for fig.1, are the tree roots and they are grouped into blocks of 2x2 adjacent pixels. In each block one(the left top corner) pixel has no descendant finally.

Implementation:

On the basis of above mentioned ideas for algorithm improvement, we propose a modified algorithm and briefly describe it in the following paragraphs.

In order to comprehend conveniently, symbols are given firstly.

B(i,j) which represents a wavelet coefficient block with coordinate *(i, j)* includes four coefficients *(i, j), (i+1, j), (i, j+1), (i+1,j+1),* like SPIHT described in detail in [3] and will be divided into its four offspring's with coordinates *(2i, 2j), (2i+2, 2j), (2i, 2j+2) and (2i+2, 2j+2).*

O(i, j): set of coordinates of all offsprings of *B(i, j).*

D(i, j): set of coordinates of all descendants of *B(i, j).*

L(i, j)=D(i, j)-O(i, j).

LSP=*{(i, j)| (i, j) ∈H}* and LIS have the same definitions as in [5]. But the set in LIS represents either *D(i, j)* or *L(i, j).* to distinguish them, we say that type D represents *D(i, j)* and type *L(i, j).* Define list of insignificant block as LIB= *{B(i, j)|(i, j) ∈H}* instead of LIP. It stores the first coordinate of a group of 2x2 adjacent pixels which are regarded as a block. H stands for the wavelet coefficient matrix.

Our algorithm encodes the sub band pixels by performing initialization and a sequence of sorting pass, refinement pass and quantization-step updating. However, differences of initialization and sorting pass still exist between the modified SPIHT and traditional SPIHT.

Sequentially, we will describe the initialization and sorting pass of the modified SPIHT.

1) Initialization:

LIB={B(0, 0), B(0,2), B(2,0), B(2,2)}, LIS={D(0, 2), D(2,0), D(2,2)}, T=2n, Ci,j is wavelet matrix coefficient and LSP is empty. N is expressed in below..

$$n = \left[\log_2 \left(\max_{(i,j)} \left| c_{i,j} \right| \right) \right]$$

2) Sorting pass:

The sorting pass consists of two tests: the LIB test (LIBT) and LIS test (LIST). The LIBT will code the block or coefficients in blocks, while the LIST mainly disposes the sets in LIS.

In each LIBT, if the maximum value of the coefficient block is smaller than the current threshold, the block is insignificant and '0' is the coded bit. Otherwise, '1' will be output and represented the significance of the block. Then, four coefficients will be respectively compared to the current threshold. When the coefficients have not been put into LSP, '0' is emitted if it is insignificant. Otherwise, 10 or 11 represent significant negative sign or significant positive sign, respectively. After that, it will be removed from the block and added to the tail of LSP. While the test is finished, the block will be removed from LIB if all the four coefficients have been put into LSP. Otherwise, the block will be tested again in next LIBT.

While in LIST, the set in LIS will be tested and coded according to its type.

For type D, if the maximum coefficients in *D(i, j)* is smaller than the current threshold, the set is insignificant and 0 is emitted. Otherwise, the significant bit 1 will be coded and *D(i, j)* will be divided into its children tree and four blocks with coordinate *(m, n) ∈Q(i, j)* rather than four adjacent coefficients.

The four blocks will be coded with the style as in LIBT, but we should add them to the tail of LIB corresponding to their significances. After coding the four blocks, our algorithm will later *D(i, j) to L(i, j)* and add *L(i, j)* to the tail of *LIS if D(i, j)* has grandson coefficients. Then, set *D(i, j)* will be removed from LIS.

For type L, if the maximum coefficient in *L(i, j)* is smaller than the current threshold, 0 will be output and represented the insignificance of the set. Otherwise, the significant bit 1 will be coded and L(i, j) will be divided into four sets *D(m, n), (m, n) ∈Q(i, j)* which will be added to the tail of LIS. Then, set *L(i, j)* will be removed from LIS.

After completing LIPT and LIBT tests, we perform the same refinement pass and updating the threshold as in traditional SPIHT.

For the Modified SPIHT, when the maximum value of a coefficient block put into LIB is small enough, only one bit will used to represent it and the four coefficients of it

Will not be coded until the current threshold is smaller than the maximum value. Therefore, this algorithm can better avoid repeat coding and early coding for non-important coefficients better. Moreover, this algorithm has the same scanning order and method to determine importance of wavelet coefficients as SPIHT. Consequently, it will inherit many advantages of SPIHT, such as embedded bit stream, excellent rate-distortion performance and so on.

III. EXPERIMENTAL RESULTS

A. PSNR comparison:

Computational formula of PSNR and mean square error(MSE).

$$PSNR = 10 * \lg \left[\frac{(2^n - 1)^2}{MSE} \right]$$

$$MSE = \sum_{i=0}^{K-1} \sum_{j=0}^{M-1} \frac{\left(f(i,j) - f1(i,j) \right)^2}{M * K}$$

Here below tabular forms shows the comparison of PNSRs of SPIHT and EZW algorithms in different wavelet families at different bit rates, and the input image is **boat.png.**

Wherein, n is the number of every pixel bit, M and K are the length and width of the matrix.

The PSNR at different bit rates for the traditional SPIHT and EZW algorithm are shown in Fig 2 and Fig 3. From Fig 2 and Fig 3, in the case of low bit rate, SPIHT algorithm can provide a higher PSNR than EZW. Furthermore, at high bit rate, the two algorithms has the similar PSNR.

Figures and Tables:

TABLE I

TABULAR FORM FOR PSNRS OF VARIOUS WAVELET FAMILIES APPLIED TO SPIHT IMAGE COMPRESSION ALGORITHM.

Bit rate (bpp)	**PSNR's (dB) of different wavelet families**											
	Db1	***Db2***	***Db4***	***Db8***	***Db10***	***Bior1.1***	***Bior2.2***	***Bior4.4***	***Bior6.8***	***Coif1***	***Coif 4***	***Coif 5***
0.1	24.88	25.57	25.92	25.76	25.76	24.88	30.67	30.67	30.72	25.64	26.01	25.99
0.2	27.01	27.89	28.27	28.11	28.07	27.01	30.88	30.87	30.93	27.95	28.40	28.41
0.3	28.34	29.37	29.81	29.63	29.57	28.34	31.09	31.08	31.16	29.43	29.96	29.96
0.4	29.64	30.72	31.24	31.11	31.03	29.64	31.30	31.34	31.45	30.77	31.44	31.42
0.5	30.66	31.71	32.19	32.14	32.08	30.66	31.55	31.67	31.78	31.76	32.42	32.39
0.6	31.43	32.55	33.06	33	32.94	31.43	31.79	31.96	32.06	32.62	33.29	33.26
0.7	32.19	33.29	33.92	33.88	33.82	32.19	32.05	32.24	32.34	33.36	34.17	34.16
0.8	32.86	34.08	34.65	34.63	34.58	32.86	32.32	32.52	32.62	34.15	34.89	34.90
0.9	33.63	34.71	35.20	35.19	35.17	33.63	32.58	32.82	32.93	34.77	35.41	35.42
1	34.23	35.20	35.69	35.68	35.67	34.23	32.89	33.15	3.27	35.26	35.90	35.91

TABLE II

TABULAR FORM FOR PSNRS OF VARIOUS WAVELET FAMILIES APPLIED TO EZW IMAGE COMPRESSION ALGORITHM.

Threshold	**Bior4.4**		**Bior6.8**		**Db4**		**Db10**		**Coif4**		**Coif5**	
	Bitrate	***PSNR***	***Bitrate***	***PSNR***	***Bitrate***	***PSNR***	***Bitrate***	***PSNR***	***Bitrate***	***PSNR***	***Bitrate***	***PSNR***
Th=100	0.08	24.38	0.09	24.73	0.09	24.41	0.09	24.26	0.09	24.49	0.09	24.59
Th=80	0.08	24.38	0.09	24.73	0.09	24.41	0.09	24.26	0.09	24.49	0.09	24.59
Th=50	0.22	26.56	0.24	26.89	0.24	26.48	0.25	27.07	0.24	26.70	0.24	27.03
Th=30	0.52	27.83	0.56	28.05	0.58	27.71	0.61	28.71	0.58	27.92	0.57	28.45
Th=20	0.52	27.83	0.56	28.05	0.58	27.71	0.61	28.71	0.58	27.92	0.57	28.45
Th=10	1.1	28.25	1.16	28.40	1.22	28.14	1.27	29.26	1.19	28.35	1.20	28.96

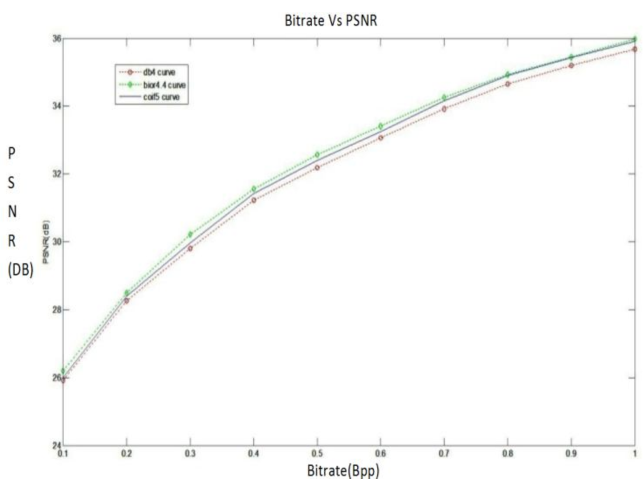

Fig.2: Comparative evolution of different wavelet families using SPIHT algorithm with boat.png image

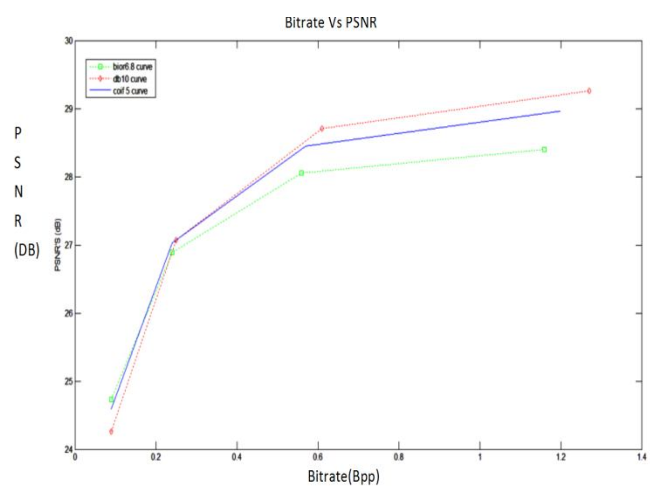

Fig.3: Comparative evolution of different wavelet families using EZW algorithm with boat.png image

Original Image

Db 4 Reconstructed image

PSNR=35.69 at bitrare=1

Bior6.8 Reconstructed image

PSNR=36.07 at bitrare=1

Coif 5 Reconstructed image

PSNR=35.91 at bitrare=1

IV. CONCLUSION

In this paper we implemented the SPIHT and EZW algorithms with Huffman encoding using different wavelet families and then compare the PSNRs and bitrates of these families. Among these different wavelet families, in the biorthogonal wavelet family '**bior4.4 & bior 6.8'** wavelet types, in the daubechies wavelet family '**db4 & db10'** wavelet types, and in the coiflet wavelet family '**coif5**' wavelet types having good PSNR at low bitrates. These algorithms were tested on different images, and it is seen that the results obtained by these algorithms have good quality and high compression ratio as compared to the previous lossless image compression techniques.

REFERENCES

[1] Surya Pemmaraju and Sunanda Mitra, "Efficient image coding using multiresolution wavelet transform and vector quantization" IEEE Southwest Symposium on Image Analysis and Interpretation, IEEE press, Santa Fe, New Mexico, April, 1996,pp:243-248.

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

[3] V.S.Shingate, T.R. Sotakke & S.N.Talbar, "Still Image Compression using Embedded Zerotree Wavelet Encoding", Inrenational Journal of Computer Science & Communication, Vol.1,No.1, January-June 2010, pp.21-24.

[4] Janaki.R, Dr.Tamilarasi.A, "Still Image Compression by Combining EZW Encoding with Huffman Encoder", International Journal of Computer Applications, Vol.13-No.7,January 2011.

[5] A.Said and W.A.Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees" IEEE Transactions on Circuits and systems for Video Technology, vol.6, no.3,pp.243-250,Jun.1996.

[6] Wei Li,Zhen Peng pang "SPIHT Algorithm with Huffman Encoding" Intelligent Information technology and Security Informatics(IITSI), 2010 Third International Symposium on. 22 april 2010.

[7] J.Hagenauer, "Rate-compatible punctured convolutional codes (RCPC Codes) and their applications" IEEE Transactions on Commun, vol.36, no.4, pp.389-400, Apr.1989.

[8] Creusere, C.D., "A New method of Robust Image Compression Based on the Embedded Zerotree Wavelet Algorithm, IEEE Transactions on Image processing, 6, No. 10(1997), pp.1436-1442.

[9] Kharate G.K., Ghatol A. A. and Rege P.P., "Image Compression Using Wavelet Packet Tree," ICGSTGVIP.

[10] Lu and W.A. Pearlman, Dec.1998, An Efficient Low- complexity Audio Cider Delivering Multiple Levels of Quality for Interactive Applications, IEEE Workshop on Multimedia Signal Processing.

[11] Tripatjot Singh, Sanjeev Chopra, Harmanpreet Kaur, Amandeep Kaur, "Image compression Using Wavelet and Wavelet packet Transformation" IJCST Vol.1, Issue 1, september 2010.

[12] Kharate G.K., Patil V.H., " Color Image Compression Based on Wavelet Packet Best Tree" IJCSI International Journal of Computer Science Issues, Vol.7,Issue 2, No 3, march 2010.