An SPIHT Algorithm with Huffman Encoder for Image Compression and Quality Improvement

Geetha Madhuri (M.Tech)[1], V Vamsi Mohan Krishna[2]
[1]Department of ECE, Nimra College of Engineering and Technology, vijayawada,Ap,India.
[2] Assoc.proffesor &HOD, Department of ECE, Nimra College of Engineering and Technology, vijayawada,Ap,India..

***Abstract*— Traditional image coding technology mainly uses the statistical redundancy between pixels to reach the goal of compressing. The research on wavelet transform image coding technology has made a rapid progress. Because of its high speed, low memory requirements and complete reversibility, digital wavelet transform (IWT) has been adopted by new image coding standard, JPEG 2000. The embedded zero tree wavelet (EZW) algorithms have obtained not bad effect in low bit-rate image compression. Set Partitioning in Hierarchical Trees (SPIHT) is an improved version of EZW and has become the general standard of EZW So, In this paper we are proposing DWT and SPIHT Algorithm with Huffman encoder for further compression and Retinex Algorithm to get enhanced quality improved image.**

Keywords*— *Image Compression, JPEG, SPIHT, Wavelets, Encoding; DWT; SPIHT;Huffman

Introduction

SPIHT is computationally very fast and among the best image compression algorithms known today. According to statistic analysis of the output binary stream of SPIHT encoding, propose a simple and effective method combined with Huffman encode for further compression. Wavelet transform as a branch of mathematics developed rapidly, which has a good localization property in the time domain and frequency domain, can analyze the details of any scale and frequency. so, it superior to Fourier and DCT. It has been widely applied and developed in image processing and compression. EZW stands for „Embedded Zero tree Wavelet". "Embedded Image Coding Using Zero trees of Wavelet Coefficients". EZW is a simple and effective image compression algorithm, its output bit-stream ordered by importance. Encoding was able to end at any location, so it allowed achieving accurate rate or distortion. This algorithm does not need to train and require pre-stored codebook. In a word, it does not require any prior knowledge of original image. More improvements over EZW are achieved by SPIHT. SPIHT stands for "Set Partitioning In Hierarchical Trees". In this method, more (wide-sense) zero trees are efficiently found and represented by separating the tree root from the tree, so, making compression more efficient. The image through the wavelet transform, the wavelet coefficients "value in high frequency region are generally small, so it will appear seriate "0" situation in quantify. SPIHT does not adopt a special method to treat with it, but direct output. In this paper, focus on this point, propose a simple and effective method combined with Huffman encode for further compression.

The SPIHT Algorithm

One of the most efficient algorithms in the area of image compression is the Set Partitioning in Hierarchical Trees (SPIHT). In essence it uses a sub-band coder, to produce a pyramid structure where an image is decomposed sequentially by applying power complementary low pass and high pass filters and then decimating the resulting images. These are one-dimensional filters that are applied in cascade (row then column) to an image whereby creating a four-way decomposition: LL (low pass then another low pass), LH (low pass then high pass), HL (high and low pass) and finally HH (high pass then another high pass). The resulting LL version is again four-way decomposed, as shown in Figure 1. This process is repeated until the top of the pyramid is reached.

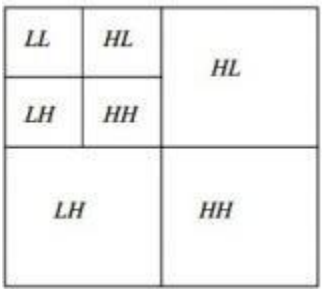

Figure.1.Image Decomposition Using Wavelets

There exists a spatial relationship among the 1-3,6 coefficients at different levels and frequency sub-bands in the pyramid structure. A wavelet coefficient at location (i,j) in the pyramid representation has four direct descendants (off-springs) at locations:

A wavelet-based still compresion and quality improve coding algorithm known as set partitioning in hierarchical trees (SPIHT) is developed that generates a continuously scalable bit stream. This means that a single encoded bit stream can be used to produce compresion and quality improves at various bit-rates and quality, without any drop in compression. The

decoder simply stops decoding when a target rate or reconstruction quality has been reached.

In the SPIHT algorithm, the compresion and quality improve is first decomposed into a number of sub bands using hierarchical wavelet decomposition. The sub bands obtained for a two-level decomposition are shown in fig. 1. The sub band coefficients are then grouped into sets known as spatial-orientation trees, which efficiently exploit the correlation between the frequency bands. The coefficients in each spatial orientation tree are then progressively coded bit-plane by bit-plane, starting with the coefficients with highest magnitude and at the lowest pyramid levels. Arithmetic coding can also be used to give further compression.

The results obtained without arithmetic coding, referred to as binary-encoded, for the grayscale lenna compresion and quality improve (256 by 256) are shown in fig. 2 for different levels of wavelet decomposition. In general, increasing the number of levels gives better compression although the improvement becomes negligible beyond 5 levels. In practice the number of possible levels can be limited by the compresion and quality improve dimensions since the wavelet decomposition can only be applied to compresion and quality improves with even dimensions. The use of arithmetic coding only results in a slight improvement as shown in fig. 3 for a 5 level decomposition.

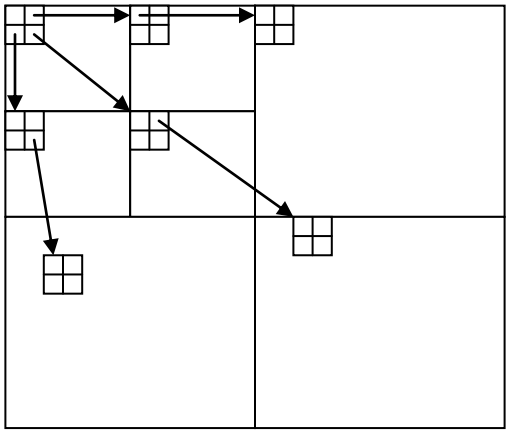

Figure 2: 2-level wavelet decomposition and spatial orientation tree.

The SPIHT algorithm sends the top coefficients in the pyramid structure using a progressive transmission scheme. This scheme is a method that allows obtaining a high quality version of the original image from the minimal amount of transmitted data. As illustrated in Figure 3, the pyramid wavelet coefficients are ordered by magnitude and then the most significant bits are transmitted first, followed by the next bit plane and so on until the lowest bit plane is reached. It has been shown that progressive transmission can significantly reduced the Mean Square Error (MSE) distortion for every bit-plane sent To take advantage of the spatial relationship among the coefficients at different levels and frequency bands, the SPIHT coder algorithm orders the wavelets coefficient according to the significance test defined as:

Algorithm: ***SPIHT coding algorithm***

Step 1: (Initialization)

Output $n = \lfloor \log_2 (\max_{(i,j)} \{| c_{i,j} |\} \rfloor$, set the LSP as an empty list,

add the coordinates $(i, j) \in H$ to the LIP,

add the coordinates $(i, j) \in H$ with descendants to the list LIS, as type A entries,

Step 2: (Sorting Pass)

2.1) for each entry (i, j) in the LIP do:

output Sn(i, j),

if Sn(i, j)=1

move (i, j) to the LSP,

output the sign of $c_{i,j}$,

2.2) for each entry (i, j) in the LIS do:

2.2.1) if the entry is of type A then

output Sn(D(i, j)),

if Sn(D(i, j)) = 1 then

* for each $(k,l) \in O(i, j)$ do:

output Sn(k, l),

if Sn(k, l) = 1 then

add (k, l) to the LSP,

output the sign of $c_{k,l}$,

if Sn(k, l)=0 then

add (k, l) to the end of the LIP,

*if $L(i, j) \neq 0$ then

move (i, j) to the end of the LIS, as an entry of type B,

go to Step 2.2.2).

otherwise

remove entry (i, j) from the LIS,

2.2.2) if the entry is of type B

output Sn(L(i, j)),

if Sn(L(i, j)) = 1 then

*add each $(k,l) \in O(i, j)$ to the end of the LIS as an entry of type A,

*remove (i,j) from the LIS,

Step 3: (Refinement Pass)

For each entry (i, j) in the LSP, except those included in the last sorting pass (i.e., with the same n), output the *n*th most significant bit of $| c_{i,j} |$,

Step 4: (Quantization-Step Update)

Decrement *n* by 1 and go to Step 2.

Pixels (LIP) and the list of significant pixels (LSP). In the decoder, the SPIHT algorithm replicates the same number of lists. It uses the basic principle that if the execution path of any algorithm is defined by the results on its branching points, and if the encoder and decoder have the same sorting algorithm then the decoder can recover the ordering information easily

Huffman Coding

The idea behind Huffman coding is to find a way to compress the storage of data using variable length codes. Our standard model of storing data uses fixed length codes. For example, each character in a text file is stored using 8 bits. There are certain advantages to this system. When reading a file, we know to ALWAYS read 8 bits at a time to read a single character. But as you might imagine, this coding scheme is inefficient. The reason for this is that some characters are more frequently used than other characters. Let's say that the character 'e' is used 10 times more frequently than the character 'q'. It would then be advantageous for us to use a 7 bit code for e and a 9 bit code for q instead because that could shorten our overall message length.

Huffman coding finds the optimal way to take advantage of varying character frequencies in a particular file. On average, using Huffman coding on standard files can shrink them anywhere from 10% to 30% depending to the character distribution. (The more skewed the distribution, the better Huffman coding will do.)

The idea behind the coding is to give less frequent characters and groups of characters longer codes. Also, the coding is constructed in such a way that no two constructed codes are prefixes of each other. This property about the code is crucial with respect to easily deciphering the code.

Encoding with Huffman Code

After reading the input file and creating Huffman code for it, we need to build dictionary -- some kind of data structure to hold for each character it's Huffman encoding (a: 0; b: 101; c: 100 etc). When you have the dictionary, encoding is simple: we just concatenate the codewords representing each character of the file. For example, with Huffman code from above figure, we'll encode the file containing "abc" as 0101100 (= 0 + 101 + 100).

Decoding with Huffman Code

For decoding we need to have the encoded file and Huffman code tree. Before you start reading bits from encoded file, define pointer pointing on the root of Huffman code tree. Then read one bit from encoded file; if it's 0, go to the left child of the node we are currently in; if it's 1, go to the right child of the node. When we reach leaf node, we got the decoded character. We write the character to decoded file, place pointer on the Huffman code tree root and continue reading from encoded file until you reach the file end.

As mentioned earlier, Huffman code is a prefix code. Since no codeword is prefix of any other, the codeword that begins an encoded file is unambiguous. We can simply identify the initial codeword, translate it back to the original character and repeat the decoding process on the reminder of the encoded file.

PERFORMANCE EVALUATION

Compression efficiency is measured by the compression ratio and is estimated by the ratio of the original compresion and quality improve size over the compressed data size. The *complexity* of an compresion and quality improve compression algorithm is calculated by the number of data operations required to perform both encoding and decoding processes. Practically, it is sometimes expressed by the number of operations. For a lossy compression scheme, a *distortion measurement* is a criterion for determining how much information has been lost when the reconstructed compresion and quality improve is produced from the compressed data. The most often used measurement is the mean square error (MSE

In the MSE measurement the total squared difference between the original signal and the reconstructed one is averaged over the entire signal. Mathematically,

$$\mathbf{MSE} = \frac{1}{N}\sum_{i=0}^{N-1}\left(\hat{x}_i - x_i\right)^2$$

where $\hat{x}_i$ is the reconstructed value of x_i . N is the number of pixels. The mean square error is commonly used

because of its convenience. A measurement of MSE in decibels on a logarithmic scale is the Peak Signal-to-Noise Ratio (PSNR), which is a popular standard objective measure of the lossy codec. We use the PSNR as the objective measurement for compression algorithms throughout this thesis. It is defined as follows

$$PSNR = 10\log\frac{MAX^2}{\frac{1}{w \times h}\sum_{i=1}^{w}\sum_{j=1}^{h}(o(i,j)-c(i,j))^2}$$

where *w* and *h* are the width and height of the compresion and quality improve respectively, o is the original compresion and quality improve data, and c is the compressed compresion and quality improve data. MAX is the maximum value that a pixel can have, 255.

Compressing an compresion and quality improve is significantly different than compressing raw binary data. Of course, general purpose compression programs can be used to compress compresion and quality improves, but the result is less than optimal. This is because compresion and quality improves have certain statistical properties which can be exploited by encoders specifically designed for them. Also, some of the finer details in the compresion and quality improve can be sacrificed for the sake of saving a little more bandwidth or storage space. This also means that lossy compression techniques can be used in this area.

Lossless compression involves with compressing data which, when decompressed, will be an exact replica of the original data. This is the case when binary data such as executables, documents etc. are compressed. They need to be exactly reproduced when decompressed. On the other hand, compresion and quality improves (and music too) need not be reproduced 'exactly'. An approximation of the original compresion and quality improve is enough for most purposes, as long as the error between the original and the compressed compresion and quality improve is tolerable.

Error Metrics

Two of the error metrics used to compare the various ecg compression techniques are the Mean Square Error (MSE) and the Peak Signal to Noise Ratio (PSNR). The MSE is the cumulative squared error between the compressed and the original ecg, whereas PSNR is a measure of the peak error. The mathematical formulae for the two are

$$MSE = \frac{1}{MN}\sum_{y=1}^{M}\sum_{x=1}^{N}[I(x,y) - I'(x,y)]^2$$

$$PSNR = 20 * \log10 (255 / sqrt(MSE))$$

where I(x,y) is the original ecg, I'(x,y) is the approximated version (which is actually the decompressed ecg) and M,N are the dimensions of the ecgs. A lower value for MSE means lesser error, and as seen from the inverse relation between the MSE and PSNR, this translates to a high value of PSNR. Logically, a higher value of PSNR is good because it means that the ratio of Signal to Noise is higher. Here, the 'signal' is the original ecg, and the 'noise' is the error in reconstruction. So, if you find a compression scheme having a lower MSE (and a high PSNR), we can recognise that it is a better one.

Results

In below figure shows first axes figure represent a original cameraman image. Second axes image represents a DWT and SPHIT Encoder techniques. Finally we are applying Huffman decoding, SPHIT decoding and IDWT retrieve original image. We are calculate CR, PSNR and MSE and displayed at below the third axes.

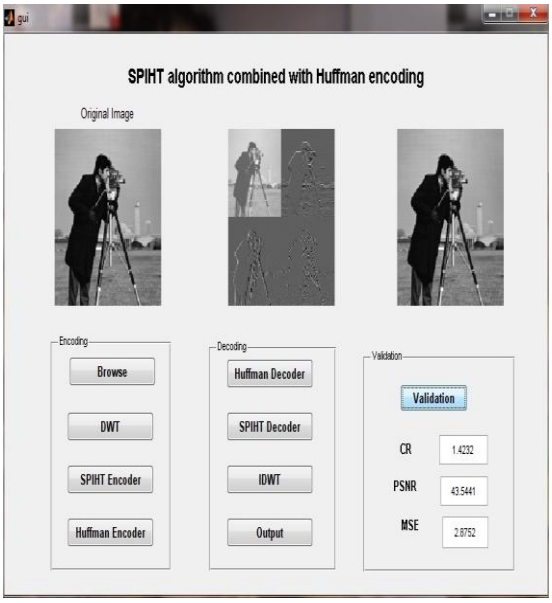

Figure 3:Simulated output of the SPHIT Algorithm with Huffman Encoding

CONCLUSION

Proposing a simple and effective method combined with Huffman encoding for further compression. In this paper a lot of bits are saves for the data transmission and storage purpose. In this decoding process uses the Retinex Algorithm for clear vision and quality improvement. So by using this large no. of image data‟s to be transmitted.

References

[1] 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.

[2] Ming Hao, Xingbo Sun "A modified Retinex Algorithm based on Wavelet Transformation", 2010 Second International Conference on Multi Media and Information Technology.

[3] FAN Qi-bin. "Wavelet analysis". Wuhan: Wuhan University Press, 2008.

[4] Cheng Li-chi, Wang Hong-xia, Luo Yong. "Wavelet theory and applications" Beijing: Science Press, 2004(Chinese).

[5] H.Wang, S.J Li, Y.Wang, "Face Recognition under Varying Lighting Conditions Using Self Quotient Image.Proc". IEEE Intl Conf. Automatic Face and Gesture Recognition, 2004.

[6] Rafael C. GONZALEZ Richard E. WOODS. "Digital image processing": second ed [M]. Beijing Publishing House of Electronics Industry 2002.

[7] Amir SAID William A.PEARLMAN . "A new fast and efficient image codec based on set partitioning in hierarchical trees [J]". IEEE Transactions On Circuits and Systems for Video Technology 1996.

[8] J. M. SHAPIRO. "Embedded image coding using zero tree of wavelets coefficients [J]". IEEE Trans. Signal Processing 1993.