WAVELET BASED IMAGE COMPRESSION

P- 2327

A PROJECT REPORT

Submitted by

SRIHARI P	**71204106052**
ASHWANTH C.D	**71204106006**
SATHISHKUMAR S	**71204106050**
KATHIRAVAN R	**71204106024**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE-641006

ANNA UNIVERSITY: CHENNAI 600 025

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

ertified that this project report **"WAVELET BASED IMAGE COMPRESSION"** the bonafide work of

SRIHARI P	**71204106052**
ASHWANTH C.D	**71204106006**
SATHISHKUMAR S	**71204106050**
KATHIRAVAN R	**71204106024**

who carried out the project work under my supervision.

IGNATURE
Dr. RAJESWARI MARIAPPAN Ph.D.,
IEAD OF THE DEPARTMENT
lectronics and Communication Engineering,
umaraguru College of Technology
oimbatore-641006

SIGNATURE
Prof. S.Govindaraju ,
PROJECT GUIDE
Professor
Electronics and Communication Engineering,
Kumaraguru College of Technology
Coimbatore-641006

Submitted for the University Examination held on 21-4-08

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

Image compression is an important aspect of multimedia data transmission, especially through bandlimited, time-varying channels. The cost and limitation in bandwidth of wireless channels has made data compression a necessity. Due to the constraints of wireless channels, progressive image transmission has gained widespread acceptance. Wavelet-based image coding lends itself naturally for progressive transmission, where the important low frequency information in the image is transmitted first, followed by the less important high frequency details. In this project, the coding of images for progressive transmission is addressed, by applying a lifting wavelet, thresholding,quantization of the transform coefficients.Finally is encoded using SPIHT encoding scheme and hence compression is obtained. Lifting wavelet transform has its advantages over the ordinary wavelet transform by way of reduction in memory required for its implementation. This is made possible because lifting transform uses in-place computation. The lifting coefficients replace the image samples present in the respective memory locations. The Haar wavelet has been used here because of its simplicity, but it can be extended to other wavelets that may give better results. The performance of the system has been evaluated based on the compression ratio and the peak signal-to-noise ratio.

ACKNOWLEDGEMENT

We express our sincere thanks to our almighty of God, the guiding light of our life for giving us the potency and courage to complete this project successfully.

We would like to express out gratitude to the **Dr. Joseph.V.Thanikal, Ph.D.,** for being out the project successfully and for strengthening the ray of hope.

We are highly indebted to **Dr. Rajeshwari Mariappan, Ph.D.,Head of the Department** for her support that have been invaluable for the project development and improvement.

We are highly grateful to our beloved Project coordinator **Ms. R.Latha,M.E.,** and our Project guide **Prof S.Govindaraju**,Professor ECE Department,for their valuable guidance,timely helps,constant encouragement and advice rendered throughout the project period for the successful completion of the project.

We also thank all other Staff Members and other technicians for their co-operation and valuable guidance.

Finally we thank our College Library for providing us with many informative books that help us to enrich our knowledge to bring out the project successfully.

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS, ABBREVIATIONS AND NOMENCLATURE

DIP	Digital Image Processing
JPEG	Joint Photographic Experts group
DWT	Discrete wavelet Transform
VQ	Vector Quantization
SPIHT	Set Partitioning in Hierarchical Trees
DCT	Discrete Cosine Transform
PCRD	Post Compression ate Distortion
RLC	Run Length Coding
ZC	Zero Coding
MSE-	Mean Square Error
PSNR	Peak Signal to Noise Ratio
TIFF	Tagged Image File Format
PNG	Portable Network Graphics
GIF	Graphics Interchange Format
BPP	Bits per pixel
CR	Compression Ratio

Chapter 1
AN INTRODUCTION

1.1 IMAGE PROCESSING

Image processing is any form of signal processing for which the input is an image, such as photographs or frames of video; the output of image processing can be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal-processing techniques to it.Image processing usually refers to digital image processing, but optical and analog image processing are also possible.

Among many other image processing operations are:

- Image Compression
- Geometric transformations such as enlargement, reduction, and rotation
- Color corrections such as brightness and contrast adjustments, quantization, or conversion to a different color space
- Digital compositing or optical compositing (combination of two or more images). Used in filmmaking to make a "matte"
- Interpolation, de mosaicing, and recovery of a full image from a raw image format using a Bayer filter pattern

- Image editing (e.g., to increase the quality of a digital image)
- Image differencing
- Image registration (alignment of two or more images)
- Image stabilization
- Image segmentation
- Extending dynamic range by combining differently exposed images

Applications of Image processing

- Computer vision
- Face detection,
- Feature detection
- Lane departure warning system
- Medical image processing
- Microscope image processing
- Morphological image processing
- Remote sensing

1.2 DIGITAL IMAGE PROCESSING

Digital image processing is the use of computer algorithms to perform image processing on digital images. As a subfield digital signal processing, digital image processing has many advantages over analog image processing; it can avoid problems such as the build-up of noise and signal distortion during processing.The most common kind of digital image processing is digital image editing.

1.2.1 HISTORY

Many of the techniques of digital image processing, or digital picture processing as it was often called, were developed in the 1960s at the Jet Propulsion Laboratory, MIT, Bell Labs, University of Maryland, and a few other places, with application to satellite imagery, wirephoto standards conversion, medical imaging, videophone, character recognition, and photo enhancement. But the cost of processing was fairly high with the computing equipment of that era. In the 1970s, digital image processing proliferated, when cheaper computers and dedicated hardware became available. Images could then be processed in real time, for some dedicated problems such as television standards conversion. As general-purpose computers became faster, they started to take over the role of dedicated hardware for all but the most specialized and compute-intensive operations.

With the fast computers and signal processors available in the 2000s, digital image processing has become the most common form of image processing, and is generally used because it is not only the most versatile method, but also the cheapest.

1.2.2 TASKS

Digital image processing allows the use of much more complex algorithms for image processing, and hence can offer both more sophisticated performance at simple tasks, and the implementation of methods which would be impossible by analog means.

In particular, digital image processing is the only practical technology for:

- Classification
- Feature extraction
- Pattern recognition
- Projection
- Multi-scale signal analysis
- Some techniques which are used in digital image processing include:
- Principal components analysis
- Independent component analysis
- Self-organizing maps
- Hidden Markov models
- Neural networks

1.3 IMAGE COMPRESSION

One of the major challenges in enabling mobile multimedia data services will be the need to process and wirelessly transmit a very large volume of data. While significant improvements in achievable bandwidth are expected with future wireless access technologies, improvements in battery technology will lag the rapidly growing energy requirements of future wireless data services. One approach to mitigate to this problem is to reduce the volume of multimedia data transmitted over the wireless channel via data compression techniques.

This has motivated active research on multimedia data compression techniques such as JPEG [1,2], JPEG 2000 [3,4] and MPEG [5]. These approaches concentrate on achieving higher compression ratio without sacrificing the quality of the image. However these efforts ignore the energy consumption during compression and RF transmission.

Since images will constitute a large part of future wireless data, we focus on developing energy efficient and adaptive image compression and communication techniques. Based on wavelet image compression, energy efficient multiwavelet image transform is a technique developed to eliminate computation of certain high-pass coefficients of an image.

1.4 PRINCIPLE OF COMPRESSION

Image compression addresses the problem of reducing the amount of data required to represent a digital image. The underlying basis of the reduction process is the removal of redundant data. From a mathematical viewpoint, this amounts to transforming a 2-D pixel array into a statistically

uncorrelated data set. The transformation is applied prior to storage and transmission of the image. The compressed image is decompressed at some later time, to reconstruct the original image or an approximation to it.

1.4.1 DIFFERENT TYPES OF DATA REDUNDANCIES:

Interpixel redundancy: Neighboring pixels have similar values. This property is exploited in the wavelet transform stage.

Psychovisual redundancy: Human visual system cannot simultaneously distinguish all colors. This property is exploited in the lossy quantization stage.

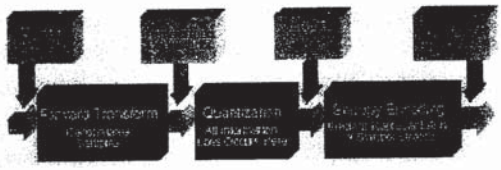
Coding redundancy: Fewer bits represent frequent symbols.

1.4.2 COMPRESSION ALGORITHMS

There are various algorithms for image transformation:

- Discrete cosine transform (DCT)
- JPEG
- Sub-band coding
- Embedded zero wavelet transform (EZW)
- Adaptive multiwavelet transform

1.5 IMAGE COMPRESSION PROCESS



1.1 The Image Compression Process

Fig.1.1 illustrates the main block diagram of the image compression process. The image sample first goes through a transform, which generates a set of frequency coefficients. The transformed coefficients are then quantized to reduce the volume of data. The output of this step is a stream of integers, each of which corresponds to an index of particular quantized binary. Encoding is the final step, where the stream of quantized data is converted into a sequence of binary symbols in which shorter binary symbols are used to encode integers that occur with relatively high probability. This reduces the number of bits transmitted.

The best image quality at a given bit-rate (or compression rate) is the main goal of image compression. However, there are other important properties of image compression schemes are :

Scalability generally refers to a quality reduction achieved by manipulation of the bitstream or file (without decompression and re-compression). Other names for scalability are *progressive coding* or *embedded bitstreams*. Despite its contrary nature, scalability can also be found in lossless codecs, usually in form of coarse-to-fine pixel scans. Scalability is especially useful for previewing images while downloading them (e.g. in a web browser) or for providing variable quality access to e.g. databases.
There are several types of scalability:

Quality progressive or layer progressive: The bitstream successively refines the reconstructed image.

Resolution progressive: First encode a lower image resolution; then encode the difference to higher resolutions.

Component progressive: First encode grey; then color.

Region of interest coding. Certain parts of the image are encoded with higher quality than others. This can be combined with scalability (encode these parts first, others later).

Meta information. Compressed data can contain information about the image which can be used to categorize, search or browse images. Such information can include color and texture statistics, small preview images and author/copyright information.

Processing power. Compression algorithms require different amounts of processing power to encode and decode. Some high compression algorithms require high processing power.

The quality of a compression method is often measured by the Peak signal-to-noise ratio. It measures the amount of noise introduced through a lossy compression of the image. However, the subjective judgement of the viewer is also regarded as an important, perhaps the most important, measure.

Table1.1 shows the qualitative transition from simple text to full-motion video data and the disk space and transmission time needed to store and transmit such uncompressed data.

TABLE 1.1 Multimedia data types and uncompressed storage space and transmission time required.

Multimedia Data	Size/Duration	Bits/pixel or Bits/Sample	Uncompressed Size (B for bytes)	Transmission Time(Using a 28.8K Modem)
A page of text	11"X 8.5"	Varying resolution	4-8KB	1.1-2.2sec
Telephone Quality	10 sec	8bps	80KB	22.2sec
Grayscale Image	512 X 512	8bpp	262KB	1 min 13 sec
Color Image	512 X 512	24bpp	786KB	3 min 39 sec
Medical Image	2048 X 1680	12bpp	5.16MB	23 min 54 sec
SHD Image	2048 X 2048	24bpp	12.58MB	58 min 15 sec
Full-motion Video	640 X 480, 1 min (30 frames/sec)	24bpp	1.66GB	5 days 8 hrs

2.1 WAVELET TRANSFORM OVERVIEW

Wavelets are mathematical functions defined over a finite interval and having an average value of zero that transform data into different frequency components, representing each component with a resolution matched to its scale.

The basic idea of the wavelet transform is to represent any arbitrary function as a superposition of a set of such wavelets or basis functions. These basis functions or baby wavelets are obtained from a single prototype wavelet called the mother wavelet, by dilations or contractions (scaling) and translations (shifts). They have advantages over traditional Fourier methods in analyzing physical situations where the signal contains discontinuities and sharp spikes. Many new wavelet applications such as image compression, turbulence, human vision, radar, and earthquake prediction are developed in recent years. In wavelet transform the basis functions are wavelets. Wavelets tend to be irregular and symmetric. All wavelet functions, $w(2kt - m)$, are derived from a single mother wavelet, $w(t)$. This wavelet is a small wave or pulse like the one shown in Fig. 2.1

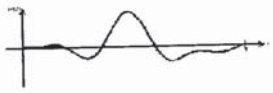
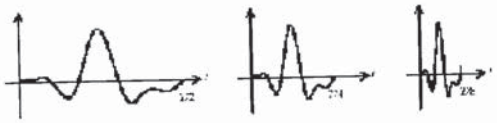


Fig. 2.1 Mother wavelet w(t)

Normally it starts at time $t = 0$ and ends at $t = T$. The shifted wavelet $w(t - m)$ starts at $t = m$ and ends at $t = m + T$. The scaled wavelets $w(2kt)$ start at $t = 0$ and end at $t = T/2k$. Their graphs are $w(t)$ compressed by the factor of $2k$ as shown in Fig. 3.3. For example, when $k = 1$, the wavelet is shown in Fig 2.2 (a). If $k = 2$ and 3, they are shown in (b) and (c), respectively.



(a)w(2t) **(b)w(4t)** **(c)w(8t)**

Fig. 2.2 Scaled wavelets

The wavelets are called orthogonal when their inner products are zero. The smaller the scaling factor is, the wider the wavelet is. Wide wavelets are comparable to low-frequency sinusoids and narrow wavelets are comparable to high-frequency sinusoids.

2.1.1 SCALING

Wavelet analysis produces a time-scale view of a signal. Scaling a wavelet simply means stretching (or compressing) it. The scale factor is used to express the compression of wavelets and often denoted by the letter a. The smaller the scale factor, the more "compressed" the wavelet. The scale is inversely related to the frequency of the signal in wavelet analysis.

2.1.2 SHIFTING

Shifting a wavelet simply means delaying (or hastening) its onset. Mathematically, delaying a function f(t) by k is represented by: f(t-k) and the schematic is shown in fig. 2.3

(a) Wavelet function $\Psi(t)$ (b) Shifted wavelet function $\Psi(t-k)$

Fig. 2.3 Shifted wavelets

2.1.3 SCALE AND FREQUENCY

The higher scales correspond to the most "stretched" wavelets. The more stretched the wavelet, the longer the portion of the signal with which it is being compared, and thus the coarser the signal features being measured by the wavelet coefficients. The relation between the scale and the frequency is shown in Fig. 2.4.

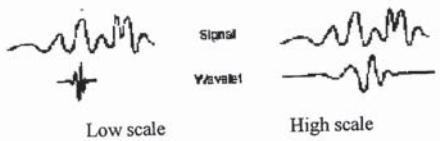


Fig. 2.4 Scale and frequency

Thus, there is a correspondence between wavelet scales and frequency as revealed by wavelet analysis.

2.2 DISCRETE WAVELET TRANSFORM

Calculating wavelet coefficients at every possible scale is a fair amount of work, and it generates an awful lot of data. If the scales and positions are chosen based on powers of two, the so-called dyadic scales and positions, then calculating wavelet coefficients are efficient and just as accurate. This is obtained from discrete wavelet transform (DWT).

2.2.1 ONE-STAGE FILTERING

For many signals, the low-frequency content is the most important part. It is the identity of the signal. The high-frequency content, on the other hand, imparts details to the signal. In wavelet analysis, the approximations and details are obtained after filtering. The approximations are the high-scale, low frequency components of the signal. The details are the low-scale, high frequency components. The filtering process is schematically represented as in Fig. 2.5.

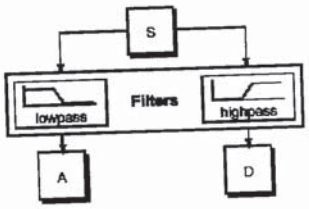


Fig. 2.5 Single stage filtering

The original signal, S, passes through two complementary filters and emerges as two signals. Unfortunately, it may result in doubling of samples and hence to avoid this, downsampling is introduced. The process on the right, which includes downsampling, produces DWT coefficients. The schematic diagram with real signals inserted is as shown in Fig. 2.6.

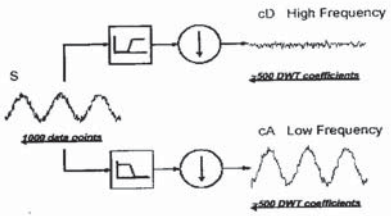


Fig. 2.6 Decomposition and decimation

2.2.2 MULTIPLE-LEVEL DECOMPOSITION

The decomposition process can be iterated, with successive approximations being decomposed in turn, so that one signal is broken down into many lower resolution components. This is called the wavelet decomposition tree and is depicted as in Fig. 2.7.

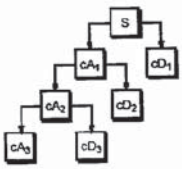


Fig. 2.7 Multilevel decomposition

2.2.3 WAVELET RECONSTRUCTION

The reconstruction of the image is achieved by the inverse discrete wavelet transform (IDWT). The values are first upsampled and then passed to the filters. This is represented as shown in Fig. 2.8.

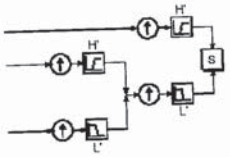


Fig. 2.8 Wavelet Reconstruction

The wavelet analysis involves filtering and downsampling, whereas the wavelet reconstruction process consists of upsampling and filtering. Upsampling is the process of lengthening a signal component by inserting zeros between samples as shown in Fig. 2.9.

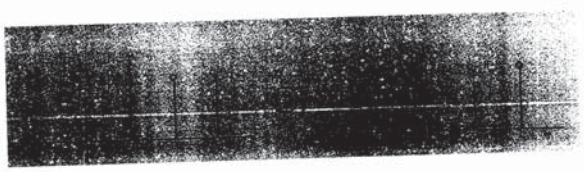


Fig. 2.9 Reconstruction using upsampling

2.2.4 RECONSTRUCTING APPROXIMATIONS AND DETAILS

It is possible to reconstruct the original signal from the coefficients of the approximations and details. The process yields a reconstructed approximation which has the same length as the original signal and which is a real approximation of it.

The reconstructed details and approximations are true constituents of the original signal. Since details and approximations are produced by downsampling and are only half the length of the original signal they cannot be directly combined to reproduce the signal. It is necessary to reconstruct the approximations and details before combining them. The reconstructed signal is schematically represented as in Fig. 2.10.

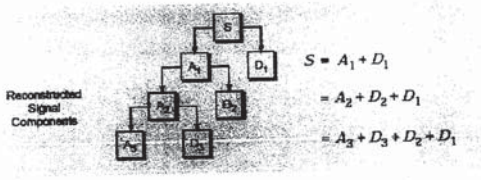


Fig. 2.10 Reconstructed signal components

$$l_i = \sum_{j=-n_l}^{n_L} s_j x_{2i+j} \quad \text{and} \quad h_i = \sum_{j=-n_H}^{n_H} t_j x_{2i+1+j}$$

Although l and h are two separate output streams, together they have the same total number of coefficients as the original data. The output stream l, which is commonly referred to as the low-pass data may then have the identical process applied again repeatedly. The other output stream, h (or high-pass data), generally remains untouched. The inverse process expands the two separate low- and high-pass data streams by inserting zeros between every other sample, convolves the resulting data streams with two new synthesis filters s' and t', and adds them together to regenerate the original double size data stream.

$$y_i = \sum_{j=-n_H}^{n_H} t'_j l'_{i+j} + \sum_{j=-n_H}^{n_l} s'_j h'_{i+j} \quad \text{where} \quad l'_{2i} = l_i, \; l'_{2i+1} = 0, \; h'_{2i+1} = h_i, \; h'_{2i} = 0$$

To meet the definition of a wavelet transform, the analysis and synthesis filters s, t, s' and t' must be chosen so that the inverse transform perfectly reconstructs the original data. Since the wavelet transform maintains the same number of coefficients as the original data, the transform itself does not provide any compression. However, the structure provided by the transform and the expected values of the coefficients give a form that is much more amenable to compression than the original data. Since the filters s, t, s' and t' are chosen to be perfectly invertible, the wavelet transform itself is lossless. Later application of the quantization step will cause some data loss and can be used to control the degree of compression. The forward wavelet-based transform uses a 1-D subband decomposition process; here a 1-D set of samples is converted into the low-pass subband (Li) and high-pass subband (Hi). The low-pass subband represents a down sampled low-resolution version of the original image. The high-pass subband represents residual information of the original image, needed for the perfect reconstruction of the original image from the low-pass subband

2.2.5 1-D WAVELET TRANSFORM

The generic form for a one-dimensional (1-D) wavelet transform is shown in Fig. 3.12. Here a signal is passed through a low pass and high pass filter, h and g, respectively, then down sampled by a factor of two, constituting one level of transform.

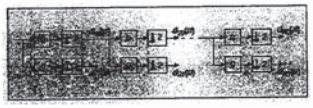


Fig. 2.11 1D Wavelet Decomposition.

Repeating the filtering and decimation process on the lowpass branch outputs make multiple levels or "scales" of the wavelet transform only. The process is typically carried out for a finite number of levels K, and the resulting coefficients are called wavelet coefficients.

The one-dimensional forward wavelet transform is defined by a pair of filters s and t that are convolved with the data at either the even or odd locations. The filters s and t used for the forward transform are called analysis filters.

2.2.6 2-D TRANSFORM HEIRARCHY

The 1-D wavelet transform can be extended to a two-dimensional (2-D) wavelet transform using separable wavelet filters. With separable filters the 2-D transform can be computed by applying a 1-D transform to all the rows of the input, and then repeating on all of the columns.

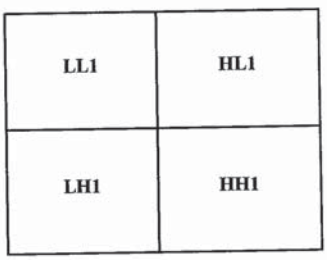


Fig. 2.12 Subband Labeling Scheme for a one level, 2-D Wavelet Transform

The original image of a one-level (K=1), 2-D wavelet transform, with corresponding notation is shown in Fig. 2.12. The example is repeated for a three-level (K =3) wavelet expansion in Fig. 2.13. In all of the discussion *K* represents the highest level of the decomposition of the wavelet transform.

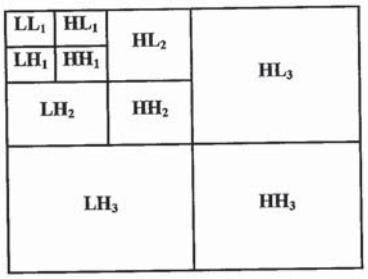


Fig. 2.13 Subband labeling Scheme for a Three Level, 2-D Wavelet Transform

The 2-D subband decomposition is just an extension of 1-D subband decomposition. The entire process is carried out by executing 1-D subband decomposition twice, first in one direction (horizontal), then in the orthogonal (vertical) direction. For example, the low-pass subbands (Li) resulting from the horizontal direction is further decomposed in the vertical direction, leading to LLi and LHi subbands.

Similarly, the high pass subband (Hi) is further decomposed into HLi and HHi. After one level of transform, the image can be further decomposed by applying the 2-D subband decomposition to the existing LLi subband. This iterative process results in multiple "transform levels". In Fig. 2.13 the first level of transform results in LH1, HL1, and HH1, in addition to LL1, which is further decomposed into LH2, HL2, HH2, LL2 at the second level, and the information of LL2 is used for the third level transform. The subband LLi is a low-resolution subband and high-pass subbands LHi, HLi, HHi are horizontal, vertical, and diagonal subband respectively since they represent the horizontal, vertical, and diagonal residual information of the original image. An example of three-level decomposition into subbands of the image CASTLE is illustrated in Fig. 2.14.

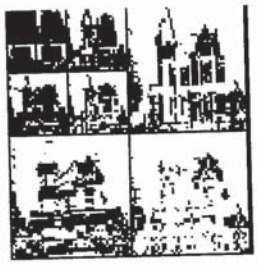


Fig. 2.14 The process of 2-D wavelet transform applied through three transform levels

To obtain a two-dimensional wavelet transform, the one-dimensional transform is applied first along the rows and then along the columns to produce four subbands: low-resolution, horizontal, vertical, and diagonal. (The vertical subband is created by applying a horizontal high-pass, which yields vertical edges.) At each level, the wavelet transform can be reapplied to the low-resolution subband to further decorrelate the image. Fig. 2.15 illustrates the image decomposition, defining level and subband conventions used in the AWIC algorithm. The final configuration contains a small low-resolution subband. In addition to the various transform levels, the phrase level 0 is used to refer to the original image data. When the user requests zero levels of transform, the original image data (level 0) is treated as a low-pass band and processing follows its natural flow.

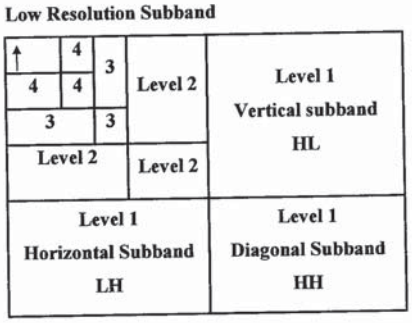


Fig. 2.15 Image Decomposition Using Wavelets

2.2.7 WAVELET COMPUTATION

In order to obtain an efficient wavelet computation, it is important to eliminate as many unnecessary computations as possible. A careful examination of the forward and reverse transforms shows that about half the operations either lead to data which are destroyed or are null operations (as in multiplication by 0).

The one-dimensional wavelet transform is computed by separately applying two analysis filters at alternating even and odd locations. The inverse process first doubles the length of each signal by inserting zeros in every other position, then applies the appropriate synthesis filter to each signal and adds the filtered signals to get the final reverse transform.

2.3 ENERGY EFFICIENT WAVELET IMAGE COMPRESSION

The proposed image codec works on the algorithm based on EEWITA (Dong-Gi Lee 2002). The EEWITA follows the classical paradigm of transformation, quantization and encoding but exploits the multiresolution property of wavelets. A modified wavelet transformation is employed during the image decomposition process.

A wavelet-based transform algorithm (EEWITA) that aims at minimizing computation energy (by reducing the number of arithmetic operations and memory accesses) and communication energy (by reducing the number of transmitted bits) is undertaken. Further, the algorithm aims at

effecting energy savings while minimally impacting the quality of the image. EEWITA exploits the numerical distribution of the high-pass coefficients to eliminate a large number of samples from consideration in the image compression process.

The high-pass coefficients are generally represented by small integer values. Because of the numerical distribution of the high-pass coefficients and the effect of the quantization step on small valued coefficients, we can estimate the high-pass coefficients to be zeros (and hence avoid computing them) and incur minimal image quality loss.

This approach has two main advantages. First, because the high-pass coefficients do not have to be computed, EEWITA helps to reduce the computation energy consumed during the wavelet image compression process by reducing the number of executed operations. Second, because the encoder and decoder are aware of the estimation technique, no information needs to be transmitted across the wireless channel, thereby reducing the communication energy required.

2.4 COMPRESSION

2.4.1 HH ELIMINATION METHOD

During the wavelet transform, each input image goes through the row and column transform decomposing the image into four subbands (LL, LH, HL, HH). The modified wavelet transformation is used for EEWITA to implement the HH elimination and H* elimination methods. To implement the HH elimination method after the row transform, the high pass coefficients are only fed into the low pass filter and not the high pass filter in the following column transform step as shown in Fig. 2.16. This avoids the generation of the diagonal subband (HH). This method saves computational energy as in accordance with the analysis of computation that denotes the two loads namely data access load and computation load that are associated with the subband generation.

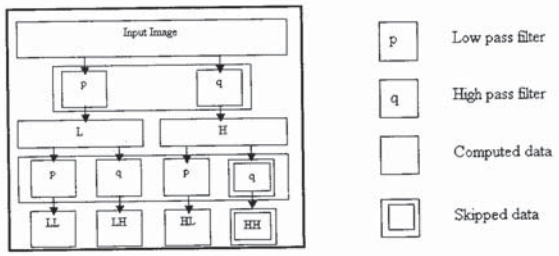


Figure 2.16 HH Elimination Method

The HH elimination method, which eliminates the insignificant high pass subband, can then be then be continued to process the image by a greater depth of transformation. The multiple level of decomposition is performed in the same way neglecting the high pass subband at the first level of decomposition and be subjected to a normal two-dimensional wavelet transform from the second level with the consideration of the image loss.

A number of useful information is lost when the elimination is continued after one or two levels of decomposition. To retain the image quality, the elimination level is performed only in the lower levels and to achieve high compression and to save computational energy, elimination level is performed to high levels of transformation.

2.4.2 H* ELIMINATION METHOD

The H* elimination method also employs the modified wavelet transformation and in this method it retains the most significant low pass subband and eliminates all the high pass subbands i.e., horizontal, vertical and diagonal subbands.

In this method, the input image is processed only through the low pass filter during both the row and column transform steps. The higher levels of image decomposition can be carried out in the same way, eliminating all high pass subbands only if the compression is of major concern and with image quality being of importance, the elimination process can be stopped at the lower levels of transform.

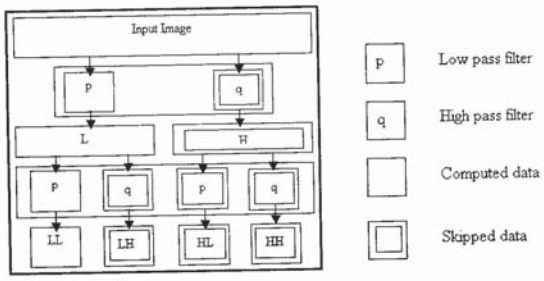


Figure 2.17 H* Elimination Method

2.5 WAVELET IMAGE COMPRESSION

The foremost goal is to attain the best compression performance possible for a wide range of image classes while minimizing the computational and implementation complexity of the algorithm. For a compression algorithm to be widely useful, it must perform well on a wide variety of image content while maintaining a practical compression/ decompression time on modest computers. In order to allow a broad range of implementation, an algorithm must be amenable to both software and hardware implementation.

COMPRESSION STEPS

The steps needed to compress an image are as follows:

1. Digitize the source image into a signal, which is a string of numbers.
2. Decompose the signal into a sequence of wavelet coefficients.
3. Use quantization to convert coefficients to a sequence of binary symbols.
4. Apply entropy coding to compress it into binary strings.

The first step in the wavelet compression process is to digitize the image. The digitized image can be characterized by its intensity levels, or scales of gray that range from 0 (black) to 255 (white), and its resolution, or how many pixels per square inch. The wavelets process the signal, but upto this point, compression has not yet occurred.

The next step is quantization which converts a sequence of floating numbers to a sequence of integers. The simplest form is to round to the nearest integer. Another option is to multiply each number by a constant and then round to the nearest integer. Quantization is called lossy because it introduces error into the process, since the conversion is not a one-to-one function.

The last step is encoding that is responsible for the actual compression. One method to compress the data is Huffman entropy coding. With this method, an integer sequence, is changed into a shorter sequence, with the numbers being 8 bit integers. The conversion is made by an entropy coding table.

TYPICAL IMAGE CODER

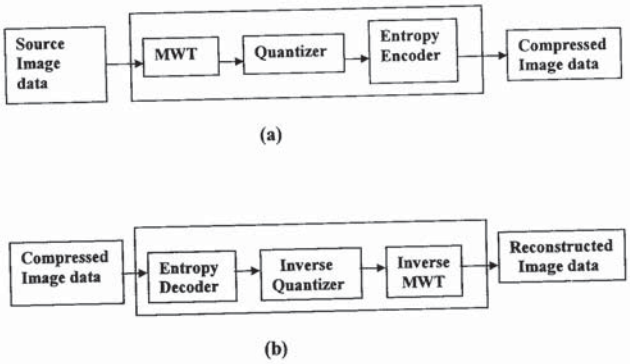


Fig. 2.18 (a) Wavelet Coder, (b) Wavelet Decoder

A typical image compression system consisting of three closely connected components namely (a) Source Encoder (b) Quantizer, and (c) Entropy Encoder is shown in Fig. 2.18. Compression is accomplished by applying a linear transform to decorrelate the image data, quantizing the resulting transform coefficients, and entropy coding the quantized values.

The source coder decorrelates the pixels. A variety of linear transforms have been developed which include Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) and many more, each with its own advantages and disadvantages.

A quantizer simply reduces the number of bits needed to store the transformed coefficients by reducing the precision of those values. Since this is a many-to-one mapping, it is a lossy process. Quantization can be performed on each individual coefficient, which is known as Scalar Quantization (SQ). Quantization can also be performed on a group of coefficients together, and this is known as Vector Quantization (VQ).

An entropy encoder further compresses the quantized values losslessly to give better overall compression. It uses a model to accurately determine the probabilities for each quantized value and produces an appropriate code based on these probabilities so that the resultant output code stream will be smaller than the input stream.

The most commonly used entropy encoders are the Huffman encoder and the arithmetic encoder, although for applications requiring fast execution, simple run-length encoding (RLE) has proven very effective. It is important to note that a properly designed quantizer and entropy encoder are absolutely necessary along with optimum signal transformation to get the best possible compression.

2.6 BASIC LIFTING SCHEME WAVELETS

Wavelet algorithms are recursive. The output of one step of the algorithm becomes the input for the next step. The initial input data set consists of 2^n elements. Each successive step operates on 2^{n-i} elements, were i = 1 ... n-1. For example, if the initial data set contains 128 elements, the wavelet tranform will consist of seven steps on 128, 64, 32, 16, 8, 4, and 2 elements.

On this web page $step_{j+1}$ follows $step_j$. If element i in step j is being updated, the notation is $step_{j,i}$. The forward lifting scheme wavelet transform divides the data set being processed into an even half and an odd half. In the notation below $even_i$ is the index of the i^{th} element in the even half and odd_i is the i^{th} element in the odd half. Viewed as a continuous array the even element would be a[i] and the odd element would be a[i+(n/2)].

Another way to refer to the recursive steps is by their power of two. This notation is used in *Ripples in Mathematics*. Here $step_{j-1}$ follows $step_j$, since each wavelet step operates on a decreasing power of two. This is a nice notation, since the references to the recursive step in a summation also correspond to the power of two being calculated.

2.6.1 PREDICT WAVELETS

Like all lifting scheme wavelets the predict wavelet transform starts with a split step, which divides the data set into odd and even elements. The predict step uses a function that approximates the data set. The difference between the approximation and the actual data replaces the odd elements of the data

set. The even elements are left unchanged and become the input for the next step in the transform. The predict step, where the odd value is "predicted" from the even value is described by the equation

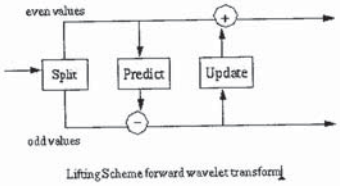
$odd_{j+1,i} = odd_{j,i} - P(even_{j,i})$

The inverse predict transform adds the prediction value to the odd element (reversing the prediction step of the forward transform). In the inverse transform the predict step is followed by a merge step which interleaves the odd and even elements back into a single data stream.

The simple predict wavelets are not useful for most wavelet applications. The even elements that are used to "predict" the odd elements result from sampling the original data set by powers of two (e.g., 2, 4, 8...). Viewed from a compression point of view this can result in large changes in the differences stored in the odd elements (and less compression). One of the most powerful applications for wavelets is in the construction of filters. The "down sampling" in the predict wavelets does not provide an approximation of the data at each step, which is one of the requirements for filter design.

2.6.2 THE UPDATE STEP

The update step replaces the even elements with an average. This results in a smoother input for the next step of the next step of the wavelet transform. The odd elements also represent an approximation of the original data set, which allows filters to be constructed. A simple lifting scheme forward transform is diagrammed in Figure 2.19.



Lifting Scheme forward wavelet transform

Figure 2.19 Lifting Scheme Forward Wavelet Transform

The update phase follows the predict phase. The original value of the odd elements has been overwritten by the difference between the odd element and its even "predictor". So in calculating an average the update phase must operate on the differences that are stored in the odd elements:

$even_{j+1,i} = even_{j,i} + U(odd_{j+1,i})$

2.6.3 LIFTING SCHEME HAAR TRANSFORM

In the lifting scheme version of the Haar transform, the prediction step predicts that the odd element will be equal to the even element. The difference between the predicted value (the even element) and the actual value of the odd element replaces the odd element. For the forward transform iteration j and element i, the new odd element, $j+1,i$ would be

$odd_{j+1,i} = odd_{j,i} - even_{j,i}$

In the lifting scheme version of the Haar transform the update step replaces an even element with the average of the even/odd pair (e.g., the even element s_i and its odd successor, s_{i+1}):

$$even_{j+1,i} = \frac{even_{j,i} + odd_{j,i}}{2}$$

The original value of the $odd_{j,i}$ element has been replaced by the difference between this element and its even predecessor. Simple algebra lets us recover the original value:

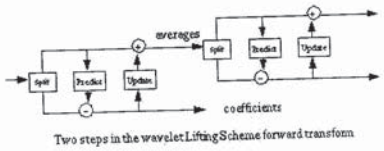
$odd_{j,i} = even_{j,i} + odd_{j+1,i}$

Substituting this into the average, we get

$$even_{j+1,i} = \frac{even_{j,i} + even_{j,i} + odd_{j+1,i}}{2}$$

$$even_{j+1,i} = even_{j,i} + \frac{odd_{j+1,i}}{2}$$

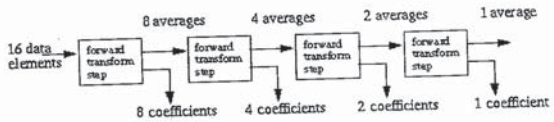
The averages (even elements) become the input for the next recursive step of the forward transform. This is shown in Figure 2.20, below.



Two steps in the wavelet Lifting Scheme forward transform

Figure 2.20 Two Steps in Wavelet Lifting Scheme Forward Transform

The number of data elements processed by the wavelet transform must be a power of two. If there are 2^n data elements, the first step of the forward transform will produce 2^{n-1} averages and 2^{n-1} differences (between the prediction and the actual odd element value). These differences are sometimes referred to as wavelet coefficients. Figure 2.21 shows a 4-steps forward wavelet transform on a 16-element data set.



4 steps of a 16 element wavelet transform

Figure 2.21 4 steps of a 16 element wavelet transform

The split phase that starts each forward transform step moves the odd elements to the second half of the array, leaving the even elements in the lower half. At the end of the transform step the odd elements are replaced by the differences and the even elements are replaced by the averages. The even elements become the input for the next step, which again starts with the split phase. The result of the forward transform is shown in Figure 2.22. The first element in the array contains the data average. The differences (coefficients) are ordered by increasing frequency.

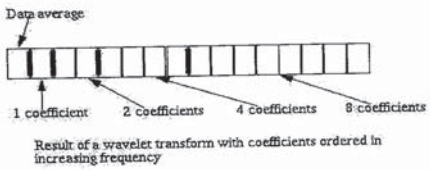


Figure 2.22 Result of a wavelet transform with co efficients ordered in increasing frequency.

One of the elegent features of the lifting scheme is that the inverse transform is a mirror of the forward transform. In the case of the Haar transform, additions are substituted for subtractions and subtractions for additions. The merge step replaces the split step.

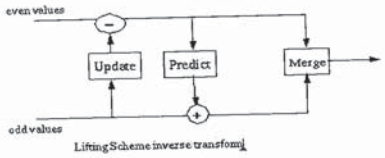


Figure 2.23 Lifting Scheme inverse transform

Chapter 3
THRESHOLDING

Thresholding is a process where coefficients which too small are suppressed, in the process retaining the more significant bits. There are two thresholding methods frequently used.

1. Soft thresholding
2. Hard thresholding

The soft-threshold function (also called the shrinkage function) takes the argument and shrinks it toward zero by the threshold.

The other popular alternative is the hard-threshold function which keeps the input if it is larger than the threshold ; otherwise,it is set to zero.

The wavelet thresholding procedure removes noise by thresholding only the wavelet coefficients of the detail subbands, while keeping the low resolution coefficients unaltered.The soft-thresholding rule is chosen over hard-thresholding for several reasons. First, soft-thresholding has been shown to achieve near-optimal minimax rate over a large range of Besov spaces.Second, for the generalized Gaussian prior assumed in this work, the optimal soft-thresholding estimator yields a smaller risk than the optimal hard-thresholding estimator (to be shown later in this section). Lastly, in practice, the soft-thresholding method yields more visually pleasant images over hard-thresholding because the latter is discontinuous and yields abrupt artifacts in the recovered images, especially when the noise energy is significant. In what follows, soft-thresholding will be the primary focus. While the idea of thresholding is simple and effective, finding a good threshold is not an easy task. For one-dimensional (1-D) deterministic signal of length , Donoho and Johnstone proposed for VisuShrink the universal threshold, which results in an estimate asymptotically optimal in the minimax sense (minimizing the maximum error over all possible -sample signals). One other notable threshold is the SURE threshold, derived from minimizing Stein's unbiased risk estimate when soft-thresholding is used. The SureShrink method is a hybrid of the universal and the SURE threshold, with the choice being dependent on the energy of the particular subband . The SURE threshold is data-driven, does not depend on explicitly,and SureShrink estimates it in a subband-adaptive manner. Moreover, SureShrink has yielded good image denoising performance and comes close to the true minimum MSE of the optimal soft-threshold estimator, and thus will be the main comparison to our proposed method.In the statistical Bayesian literature, many works have concentrated on deriving the best threshold (or shrinkage factor) based on priors such as the Laplacian and a mixture of Gaussians.With an integral approximation to the pixel-wise MSE distortion measure as discussed earlier, the formulation here is also Bayesian for finding the best soft-thresholding rule under the generalized Gaussian prior.

Chapter 4

QUANTIZATION

Quantization, involved in image processing, is a lossy compression technique achieved by compressing a range of values to a single quantum value. When the number of discrete symbols in a given stream is reduced, the stream becomes more compressible. For example, reducing the number of colors required to represent a digital image makes it possible to reduce its file size. Specific applications include DCT data quantization in JPEG and DWT data quantization in JPEG 2000.Here the transform coefficients are reduced in precision. This operation is lossy unless the quantization step is one and the coefficients are integer.

JPEG 2000 supports two types of quantization,

1. Scalar quantization
2. Vector quantization.

4.1 SCALAR QUANTIZATION

The wavelet coefficients are quantized using a uniform scalar quantizer with dead zone. For each subband *b*,a basic quantizer step size delta b is used to quantize all the coefficients in that subband according to

$$q = \text{sign}(y) \times \left\lfloor \frac{|y|}{\text{delta b}} \right\rfloor$$

where y is the input to the quantizer, sign(y) denotes the sign of the input coefficient y (it is +1 for positive and -1 for negative values), delta b is the step size, and q is the resulting quantizer index. Dead zone means that the quantization range around zero is (2*delta b). this ensures that more zeros result. This operation is lossy and is used for lossy compression. Scalar quantizer with dead zone is shown in fig 4.1

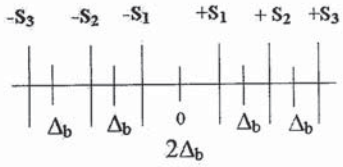


Fig 4.1 Dead Zone Quantizer

The reconstructed wavelet coefficients are dequantized to form the approximation of the original image. The formula used for this is given by,

$$Z = [q + r * \text{sign}(q)]\,\text{delta b} \quad \text{q not equal to 0}$$
$$Z = 0 \quad \text{otherwise}$$

Where q is the quantizer index, delta b is the step size and r is the reconstruction bias(r = 0.5 results in midpoint reconstruction i.e. no bias and r<0.5 biases the reconstruction towards zero). The quantization step size is same as that used in the encoder. Different step sizes are used for the various subbands.

4.2 VECTOR QUANTIZATION

A vector quantizer is composed of two operations. The first is the encoder, and the second is the decoder. The encoder takes an input vector and outputs the index of the codeword that offers the lowest distortion. In this case the lowest distortion is found by evaluating the Euclidean distance between the input vector and each codeword in the codebook. Once the closest codeword is found, the index of that codeword is sent through a channel (the channel could be a computer storage, communications channel, and so on). When the encoder receives the index of the codeword, it replaces the index with the associated codeword. Figure 2 shows a block diagram of the operation of the encoder and decoder.

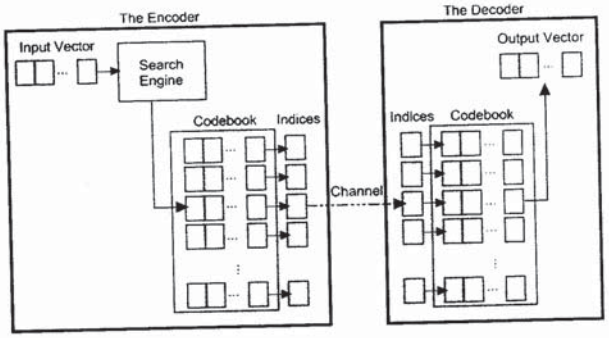


Figure 4.2: The Encoder and decoder in a vector quantizer. Given an input vector, the closest codeword is found and the index of the codeword is sent through the channel. The decoder receives the index of the codeword, and outputs the codeword.

Chapter 5
ENCODING-SPIHT

5.1 INTRODUCTION TO SPIHT

SPIHT is a powerful wavelet-based image compression method called *Set Partitioning in Hierarchical Trees* (SPIHT). This award-winning method has received worldwide acclaim and attention since its introduction in 1995. Thousands of people, researchers and consumers alike, have now tested and used SPIHT. It has become the benchmark state-of-the-art algorithm for image compression.

The SPIHT method is not a simple extension of traditional methods for image compression, and represents an important advance in the field. The method deserves special attention because it provides the following:

- Highest Image Quality
- Progressive image transmission
- Fully embedded coded file
- Simple quantization algorithm
- Fast coding/decoding
- Completely adaptive
- Lossless compression
- Exact bit rate coding
- Error protection

Each of these properties is discussed below. Note that different compression methods were developed specifically to achieve at least one of those objectives. What makes SPIHT really outstanding is that it yields all those qualities simultaneously.

5.1.1 IMAGE QUALITY

Extensive research has shown that the images obtained with wavelet-based methods yield very good visual quality. At first it was shown that even simple coding methods produced good results when combined with wavelets and is the basis for the most recently JPEG2000 standard. However, SPIHT belongs to the next generation of wavelet encoders, employing more sophisticated coding. In fact, SPIHT exploits the properties of the wavelet-transformed images to increase its efficiency.

Many researchers now believe that encoders that use wavelets are superior to those that use DCT or fractals. We will not discuss the matter of taste in the evaluation of low quality images, but we do want to say that SPIHT wins in the test of finding the minimum rate required to obtain a reproduction indistinguishable from the original. The SPIHT advantage is even more pronounced in encoding color images, because the bits are allocated automatically for local optimality among the color components, unlike other algorithms that encode the color components separately based on global statistics of the individual components.

5.1.2 PROGRESSIVE IMAGE TRANSMISSION

In some systems with progressive image transmission (like WWW browsers) the quality of the displayed images follows the sequence: (a) weird abstract art; (b) you begin to believe that it is an image of something; (c) CGA-like quality; (d) lossless recovery. With very fast links the transition from (a) to (d) can be so fast that you will never notice. With slow links (how "slow" depends on the image size, colors, etc.) the time from one stage to the next grows exponentially, and it may take hours to download a large image. Considering that it may be possible to recover an excellent-quality image using 10-20 times less bits, it is easy to see the inefficiency. Furthermore, the mentioned systems are not efficient even for lossless transmission.

The problem is that such widely used schemes employ a very primitive progressive image transmission method. On the other extreme, SPIHT is a state-of-the-art method that was *designed for* optimal progressive transmission (and still beats most non-progressive methods!). It does so by producing a fully embedded coded file (see below), in a manner that at any moment the quality of the displayed image is the best available for the number of bits received up to that moment.

So, SPIHT can be very useful for applications where the user can quickly inspect the image and decide if it should be really downloaded, or is good enough to be saved, or need refinement.

5.1.3 OPTIMIZED EMBEDDED CODING

A strict definition of the embedded coding scheme is: if two files produced by the encoder have size M and N bits, with $M > N$, then the file with size N is *identical* to the first N bits of the file with size M.

Let's see how this abstract definition is used in practice. Suppose you need to compress an image for three remote users. Each one have different needs of image reproduction quality, and you find that those qualities can be obtained with the image compressed to at least 8 Kb, 30 Kb, and 80 Kb, respectively. If you use a non-embedded encoder (like JPEG), to save in transmission costs (or time) you must prepare one file for each user. On the other hand, if you use an embedded encoder (like SPIHT) then you can compress the image to a single 80 Kb file, and then send the first 8 Kb of the file to the first user, the first 30 Kb to the second user, and the whole file to the third user.

SPIHT achieves this feat by optimizing the embedded coding process and always coding the most important information first.

An even more important application is for progressive image transmission, where the user can decide at which point the image quality satisfies his needs, or abort the transmission after a quick inspection, etc.

5.1.4 COMPRESSION ALGORITHM

The following is a comparison of image quality and artifacts at high compression ratios versus JPEG.

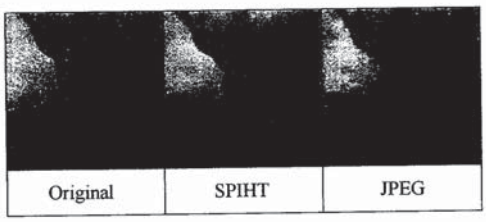


Fig.5.1 Image quality and artifacts at high compression ratios

SPIHT represents a small "revolution" in image compression because it broke the trend to more complex (in both the theoretical and the computational senses) compression schemes. While researchers had been trying to improve previous schemes for image coding using very sophisticated vector quantization, SPIHT achieved superior results using the *simplest* method: uniform scalar quantization. Thus, it is much easier to design fast SPIHT codecs.

5.1.5 ENCODING/DECODING SPEED

The SPIHT process represents a very effective form of entropy-coding. This is shown by the demo programs using two forms of coding: binary-uncoded (extremely simple) and context-based adaptive arithmetic coded (sophisticated). Surprisingly, the difference in compression is small, showing that it is not necessary to use slow methods (and also pay royalties for them!). A fast version using Huffman codes was also successfully tested, but it is not publicly available.

A straightforward consequence of the compression simplicity is the greater coding/decoding speed. The SPIHT algorithm is nearly symmetric, i.e., the time to encode is nearly equal to the time to decode. (Complex compression algorithms tend to have encoding times much larger than the decoding times.)

5.1.6 LOSSLESS COMPRESSION

SPIHT codes the individual bits of the image wavelet transform coefficients following a bit-plane sequence. Thus, it is capable of recovering the image perfectly (every single bit of it) by coding all bits of the transform. However, the wavelet transform yields perfect reconstruction only if its numbers are stored as infinite-precision numbers. In practice it is frequently possible to recover the image perfectly using rounding after recovery, but this is not the most efficient approach.

5.1.7 RATE OR DISTORTION SPECIFICATION

Almost all image compression methods developed so far do not have precise rate control. For some methods you specify a target rate, and the program tries to give something that is not too far from what you wanted. For others you specify a "quality factor" and wait to see if the size of the file fits your needs. (If not, just keep trying...)

The embedded coding property of SPIHT allows *exact* bit rate control, without any penalty in performance (no bits wasted with padding, or whatever). The same property also allows exact mean squared-error (MSE) distortion control. Even though the MSE is not the best measure of image quality, it is far superior to other criteria used for quality specification.

5.1.8 ERROR PROTECTION

Errors in the compressed file cause havoc for practically all important image compression methods. This is not exactly related to variable length entropy-coding, but to the necessity of using context generation for efficient compression. For instance, Huffman codes have the ability to quickly recover after an error. However, if it is used to code run-lengths, then that property is useless because all runs after an error would be shifted.

SPIHT is not an exception for this rule. One difference, however, is that due to SPIHT's embedded coding property, it is much easier to design efficient error-resilient schemes.

This happens because with embedded coding the information is sorted according to its importance, and the 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, the decoder can discard the data after that point and still display the image obtained with the bits received before the error. Also, with bit-plane coding the error effects are limited to below the previously coded planes.

Another reason is that SPIHT generates two types of data. The first is sorting information, which needs error protection as explained above. The second consists of uncompressed sign and refinement bits, which do not need special protection because they affect only one pixel.

While SPIHT can yield gains like 3 dB PSNR over methods like JPEG, its use in noisy channels, combined with error protection as explained above, leads to much larger gains, like 6-12 dB. (Such high coding gains are frequently viewed with skepticism, but they do make sense for combined source-channel coding schemes.)

5.2 APPLICATIONS

SPIHT exploits properties that are present in a wide variety of images. It had been successfully tested in natural (portraits, landscape, weddings, etc.) and medical (X-ray, CT, etc) images. Furthermore, its embedded coding process proved to be effective in a broad range of reconstruction qualities. For

instance, it can code fair-quality portraits and high-quality medical images equally well (as compared with other methods in the same conditions).

SPIHT has also been tested for some less usual purposes, like the compression of elevation maps, scientific data, and others.

5.3 SPIHT Optimized Algorithm

The the following are the suite of application specific SPIHT compression products:

D-SPIHT (Dynamic)

The D-SPIHT software is capable of the most efficient compression of monochrome, 1 and 2 byte per pel, and color images. It has the features of specifying bit rate or quality at encoding time. For bit rate specification, the compressed file is completely and finely rate-embedded. Rate-embedded means that any lower rate D-SPIHT file is a subset of the full file and can be decoded to the given smaller rate. The decoding has the special feature of producing smaller resolution image reconstructions (such as thumbnails) directly from the compressed file without offline processing.

S-SPIHT (Striped)

The S-SPIHT software shares many features of D-SPIHT, but it works with a small memory. The small memory leads to a slight degradation in performance compared to D-SPIHT. Therefore, this software is ideal for compression of very large images, such as those from GIS and remote sensing systems. It is both efficient and fast. A good strategy for compression of image of any size is to use a combination of D-SPIHT and S-SPIHT, with S-SPIHT taking over from D-SPIHT when an image dimension exceeds 1024 or 2048.

T-SPIHT (Tiled)

T-SPIHT is a specially designed form of SPIHT that compresses large images in tiles. Remote sensing, geographical, and meteorological data are often handled in tiles rather than all at once. Therefore, systems for processing such data would prefer to use a compression by tiles. T-SPIHT can efficiently compress, with constant quality and without tile boundary artifacts, large image or data sets, whether in tile format or not. In fact, the compression is significantly more efficient than JPEG2000 in its tiled compression mode.

P-SPIHT (Photo-ID)

P-SPIHT is especially designed to efficiently compress small grayscale and color images. It is more efficient in this application than JPEG2000 or any other competing software. It is ideal for storage of photo ID's on plastic cards with magnetic strips.

PROGRES (Progressive Resolution Decompression)

PROGRES is a progressive resolution, extremely fast version of SPIHT that has full capability of random access decoding. It chooses a quality factor for lossy compression of any size image and can decompress a given arbitrary region to any resolution very quickly. It comes either in a command line version for WINDOWS/DOS or UNIX (or Linux) or with additional region-of-interest (ROI) encoding and decoding in a UNIX (or Linux) GUI version. It is an excellent choice for remote sensing and GIS applications, where rapid browsing of large images is necessary.

PROGCODE (Lossless & Progressive)

PROGCODE is our pioneering progressive lossy to purely lossless (reversible) greyscale compression algorithm. It sets the standard for lossless compression. For larger images, the compression is still efficient, but the larger memory utilization slows down execution. This algorithm can efficiently compress medical images either lossless or with any desired rate. Moreover, the lossless file can act as an archive in a file server, from within which any desired lossy lower rate file can be extracted and decompressed directly without offline processing. This software is also ideal for multi-user storage or transmission systems, where users with different capabilities and requirements can receive the image to their desired accuracy from a single file or transmission.

SPM (Fastest Lossless!)

SPM software is especially designed for lossless and lossy compression of medical images. The method is not based on SPIHT, but on our patented quadrature splitting algorithm called AGP (for Amplitude and Group Partitioning). SPM calls for a quality factor, with 0 giving perfectly lossless compression. Because the intended application is medical images, the allowed lossy compression adheres to degrees of high quality. The outstanding characteristics of SPM, besides efficiency, are very fast compression and decompression, regardless of image size. It comes either in a command line version for WINDOWS/DOS or UNIX (or Linux) or in a WINDOWS GUI version. All versions allow reduced resolution decoding from a single compressed file.

Chapter 6
Project Description

Wavelet based image compression process involves two important processes

(i)Compression

(ii)Decompression

Lets first look at how the compression process is carried

6.1 COMPRESSION STAGE

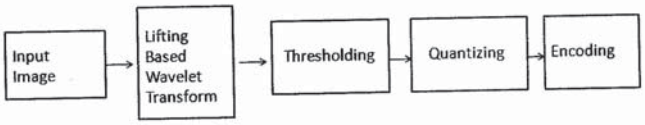


Fig.6.1 Foreward Compression block diagram.

6.1.1 READING AN IMAGE FILE:

The image file should be given as the input, the image is read in the form of pixels. An image file can be of any dimensions. Ideally, we take a 256 x 256 dimension image. An image that contains 256 rows and 256 columns .

FL(j,k)=even(j,k)+round(FH(j,k)/2);

2.The resultant frequency components FH and FL are taken now. Similar to the previous step, the frequency component FH is partitioned into odd and even entities which are HODD, HEVEN and FL is partitioned into LODD and LEVEN.

3.In order to obtain the second order of frequency components, we use the following computation .

HH

f2lhigh(j,k)=LODD(j,k)-LEVEN(j,k);

HL

f2llow(j,k)=LEVEN(j,k)+round(f2lhigh(j,k)/2);

LH

f2hhigh(j,k)=HODD(j,k)-HEVEN(j,k);

LL

f2hlow(j,k)= HEVEN (j,k)+round(f2hhigh(j,k)/2

This brings towards the end of forward lifting process as the image now contains 4 quadrants containing frequency components HH, HL, LH and LL.

6.1.2 FORWARD WAVELET LIFTING SCHEME:

Wavelets are widely used in data compression. Wavelet lifting is a process through which the whole image is partitioned into a set of odd and even entities and then deriving the first order (FL,FH) and second order of low and high frequency components(LL,HL,LH,HH).

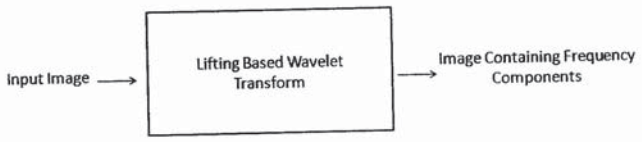


Fig.6.2 Forward Wavelet Lifting Scheme

Steps involved in Forward Lifting:

1. The first step in lifting is to split the odd and even entities of an image . From these odd and even entities we move further to find the set off low and high frequency components using the formula

FH(j,k)=odd(j,k)-even(j,k);

6.1.3 THRESHOLDING

Thresholding is a process where coefficients which is too small are suppressed, in the process retaining the more significant bits. This process is done in the high frequency quadrants HL,HH and LH containing the lesser significant lot.

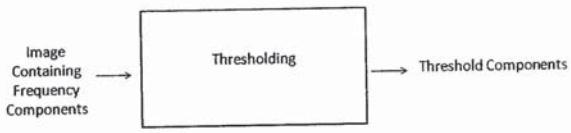


Fig.6.3 Thresholding

This process is done by selecting an optimum threshold value and comparing it with other pixel values and assigning zeros for the values below that particular value of threshold.At the end of this thresholding process we will get a lot of zeros and few significant values.

6.1.4 QUANTIZATION:

Quantization in image compression is a process in which the resulting threshold components are divided by a uniform scalar value.

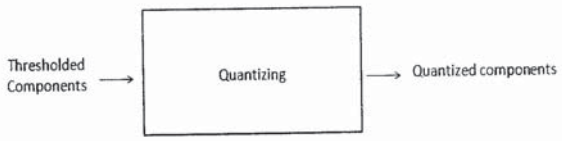


Fig.6.4 Quantization

By applying quantization,we ensure that the number of discrete symbols in a given stream is reduced,thereby reducing the number of bits to represent the discrete values.

6.1.5 ENCODING

The quantized components are now encoded using a very effective form of entropy encoding known as SPIHT(Set Partitioning In Hierarchial Trees).

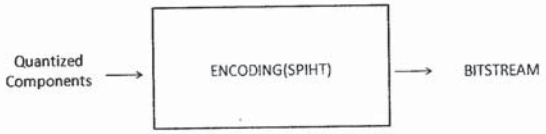


Fig.6.5 Encoding

SPIHT codes the individual bits of the image wavelet transform coefficients following a bit-plane sequence. Thus, it is capable of recovering the image perfectly (every single bit of it) by coding all bits of the transform. The output of this SPIHT encoding is a stream of bits.

6.1.6 TRANSMITTER SECTION

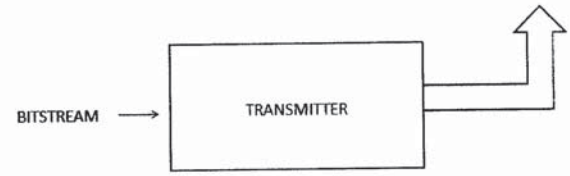


Fig.6.6 Transmitter Section

The bit streams are now ready to be transmitted to the receiving end. With this the image compressing stage comes to an end. Now, the processes at the receiver side can be explained in the sections to follow.

6.2 DECOMPRESSION STAGE

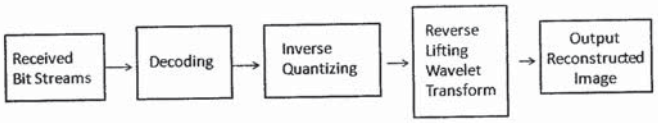


Fig.6.7 Decompression Stage block diagram

6.2.1 RECEIVER SECTION

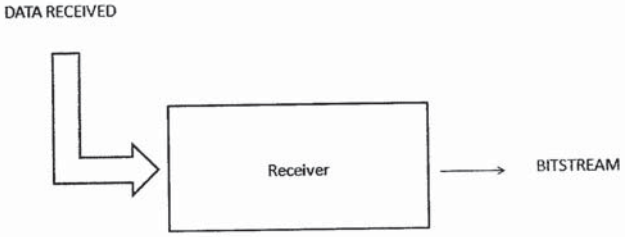


Fig.6.8 Receiver Section

The transmitted bits are received by a receiver at the opposite end. There will hardly be any error when a comparison is made between the transmitted and the received bits. Such is the efficiency of SPIHT encoding.

6.2.2 DECODING

Decoding is the reverse process of encoding where we use inverse-SPIHT in order to decode the bit streams to a quantized data. The original content is restored.

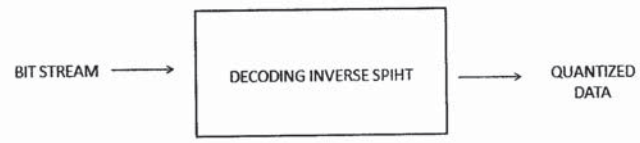


Fig.6.9 Decoding

SPIHT is therefore an error resilient form of coding .This happens because with embedded coding the information is sorted according to its importance, and the 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, the decoder can discard the data after that point and still display the image obtained with the bits received before the error.

6.2.3 INVERSE QUANTIZATION

In the case of inverse quantization, the quantized data is multiplied with a uniform scalar value , which was used in the forward quantization scheme.

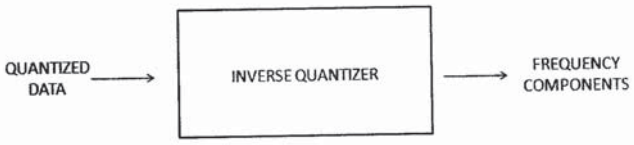


Fig.6.10 Inverse Quantization

The resulting function is applied to the four quadrants and this will lead to the restoration of the four different frequency components.

rf2hl (j,k)=LL(j,k)-round(LH(j,k)/2)

rf2hh (j,k)=LH(j,k)+rf2hl(j,k)

2. Having obtained the reverse components,we can now do odd and even separation as the second process in reconstruction. Thus, we obtain odd and even entities of the four frequency components odd and even entities of the four frequency components which are LEVEN,LODD,HEVEN,HODD...

3. Similarly, we recombine LEVEN,LODD as RL and HODD,HEVEN as RH.

4.The resulting RL and RH frequency components are computed to get the RLOW and RHIGH components as

RLOW(j,k)=rl(j,k)-round(rh(j,k)/2)

RHIGH(j,k)=r(j,k)+rh(j,k)

5. The final step is the interpolation step where we obtain the reconstructed image as the output.

6.2.4 REVERSE WAVELET LIFTING SCHEME

The reverse lifting process is done to the 4 frequency components LL, LH, HL, and HH.

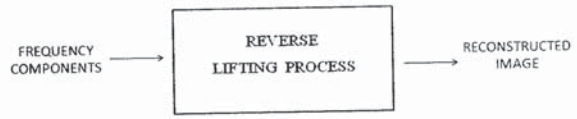


Fig.6.11 Reverse Wavelet Lifting Scheme

Steps involved in reverse lifting

1. At this juncture, we now have the four frequency components which are similar to the frequency components of the forward lifting process , using these four components we now focus our attention on calculating the reverse frequency components from which it is easier to do odd and even separation of the four frequency components.

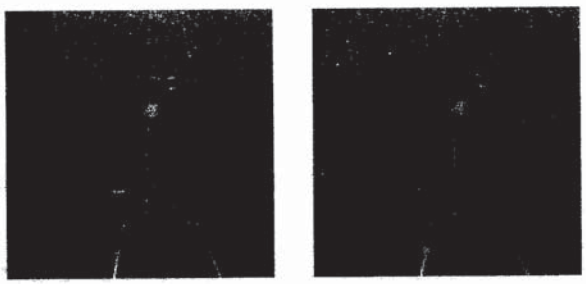
rf2ll (j,k)=HL(j,k)-round(HH(j,k)/2)

rf2lh (j,k)=rf2ll(j,k)+f2lhigh(j,k)

Chapter 7

RESULTS

Test Image 1



Input Image Reconstructed Output Image

Figure7.1

Table 7.1 Image1 Results

RESULTS	
PSNR	53.75 dB
CR	2.07

Test Image 2

Input Image Reconstructed Output Image

Figure7.2

Table 7.2 Image2 Results

RESULTS	
PSNR	62.10 dB
CR	3.5

Test Image 3

Input Image Reconstructed Output Image

Figure7.3

Table 7.3 Image3 Results

RESULTS	
PSNR	53.15 dB
CR	1.6

CONCLUSION

As a result, we obtain a very effective Compression ratio and a high Progressive Signal To Noise Ratio (PSNR) and consequently a very small Mean Square Error(MSE). The reconstructed image has a minimum difference compared to the original image.

Image Compression is done here using wavelet lifting process. The Wavelet lifting process is a very effective method prior to the thresholding and quantization process.

The Wavelet lifting process done over here is a lossy compression technique, since we are using thresholding and quantization methods.

Thresholding is effective because it makes computations simpler, efficient and occupies low memory space. Then, scalar quantization is done which reduces the number of bits to represent a discrete value and uses a minimum number of bits to encode the discrete values.

The encoding scheme used here is SPIHT encoding which is a very novel method of entropy-encoding. Since SPIHT encodes data in the form of bit steams we can achieve a high transmission efficiency. There is also a negligible utility of on-chip memory and memory access during the computation, so that it can achieve significant reduction in both die area and power dissipation.

In the future, Modifications to SPIHT including adding error protection to the bit-stream and region of interest coding will be considered.

REFERENCES:

1. Rafael C. Gonzalez and Richard E. Woods (2003), 'Digital Image Processing' Adison-Wesley edition.

2. JPEG2000: Image Compression Fundamentals, Standards and Practice by David Taubman (Editor), Michael Marcellin

3. Dong-Gi Lee and Sujit Dey (2002),' Adaptive and energy efficient wavelet image compression for mobile multimedia data services', IEEE Trans. on Information Theory and Signal Processing.

4. Rushanan. J.J (1997), 'AWIC-Adaptive Wavelet Image Compression for Still Image', MTR-97B0000041, The MITRE Corporation, Bedford, MA.

5. Sayood. K (2000), 'Introduction to Data Compression', San Mateo, CA-Mogan Kaufmann.

REFERENCES – THE WEB:

- http://www.jpeg.org/JPEG
- http://www.wikipedia.org/
- http://www.dsprelated.com/groups/imagedsp/1.php
- http://www.mathworks.com/products/matlab/
- http://www.siliconimaging.com/SPIHT.htm