ENTROPY CODING BASED SET PARTITIONING IN HIRARCHICAL TREES (SPIHT).

By

ALI IQBAL

A thesis submitted to the Faculty of Electrical Engineering Department, Military College of Signals, National University of Sciences and Technology, Rawalpindi is partial fulfillment of the requirements for the degree of MS in Electrical Engineering

June 2016

SUPERVISOR SERTIFICATE

It is to certify that the final copy of thesis has been evaluated by me, found as per the specified format and error free.

Date :____________ ____________________

Col Dr Imran Touqeer

ABSTRACT

As we know that images are the most important Digital data now a days. Image compression plays vital role in terms of saving storage space and reduction of transmission time. Wavelet transform is considered as landmark in the field of image compression due to the feature that it represents a signal in terms of functions those are localized in both frequency and time domain, as not in case of other Transformation techniques. Set Partitioning in Hierarchical Trees (SPIHT) is based on wavelet transform gives us the better image quality after the compression in a progressive manner. It works on the principal that partitioning of spatial orientation trees in such manner that insignificant and significant coefficients (with respect to some predefined threshold) are kept in the different sets. The output bit stream generated by SPIHT algorithm consists of large number of seriate '000' with probability nearly equal to ¼, and require further compression. This is achieve by the cascading Entropy encoding schemes with SPIHT algorithm.

The aim of this research is comparison between cascading of SPIHT algorithm with two entropy encoding schemes (Arithmetic coding and Huffman coding). For the cascading, the output bit stream of SPIHT is divided in sets of three bits to form $2^3 = 8$ symbols. These symbols are given to entropy encoding schemes (Arithmetic and Huffman). This cascading save lots of bits during transmission of data. Due to which it decreases the transmission time and requires less space on hard disk.

This research concludes that the concatenation of SPIHT and Arithmetic coding blocks provides better Bits saving capability as compared to SPIHT and Huffman coding concatenation. On the other hand, SPIHT combined with Huffman performs well in terms of algorithm efficiency, implementation and execution time by preserving PSNR.

ACKNOWLWDGEMENTS

Firstly I want to give gratefulness to Allah Rabb-ul-izzat for blessing me with entrance in the field of legendary research. Countless blessings be upon our Holy Prophet Hazrat Muhammad Mustafa ﷺ who introduce us with glorious teaching of Islam.

Secondly, I want to give thanks to Military College of Signals, NUST, for facilitating me with marvelous research atmosphere at the college. This meticulous achievement, I cannot accomplish without the technical and moral support of my supervisor Col Dr Imran Touqir who induced significance of this research and significant skills in me. I must oblige to him for his prestigious supervision, intense support and influential denigration. I am also very thankful to honorable committee members, Lt Col Dr Adil Masood Sidiqui and Lt Col Dr Abdul Ghafoor for their association in my research work.

Achievement of this work could not be possible without determined moral help and assistance of my siblings Anam Iqbal, Usman Iqbal and Sufyan Iqbal and particularly eldest sister Anam Iqbal as proof reader.

DEDICATION

Devoted to

Affectionate Parents, Hardworking Teachers

Loving Friends, Brothers and sister

Whose love, fondness and Help enabled me to attain yet another landmark in my professional career.

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1. Background

A lot of achievements have done in the field of organization of digital data especially digital images since last two decades. Organization of the data of digital images comprises on the storage of image's pixels, their processing and finally recovery. Image compression has a vital role in all steps of organization of digital data. Image compression has broad applications in rising areas of multimedia database, medical imaging diagnosis and worldwide web (www). As we know that web contents are principally comprises on images so in order to deal with it image compression is a counterpart. So image compression is a challenging field for all researchers.

Image compression is a technique in which we shrink the image size by, taking into account that there would be no consequence on the quality of the image. Image compression not only helps us to store the images in less disk space and save a lot of memory, rather than it provides us the facility in transmission and reception of the image in smaller time. To achieve this compression there are lot of techniques to implement. Internet utilize the famous technique i.e. JPEG. In many image compression systems Wavelet is employed.

Compression the digital image and compression of raw binary data are two separate things. If we compress digital image by using traditional image compression methods (normally used for raw binary data), this does not present good compression ratios and other compression measurement parameters. So there is need for those image compression techniques in which we could exploit the spatial features. If in some cases where quality is not big issue we can compress image by ignoring some details of the image, this technique known as Lossy compression.

At this stage we can observe compression in two different types, i.e. Lossy and lossless image compression techniques. In some situations where image quality and all details are key factors and can't be compromised, we use lossless image compression techniques. But the focus of researchers is on the Lossy image compression algorithms rather than lossless techniques, because most of the images are related to less sensitive human vision.

A tremendous progress is observed in the field of image compression in last three decades. Researchers are facilitated by the advent of Wavelet Transform. In which we get the details of the images, and get success in exploiting the spatial features and characteristics. There are lots of image compression algorithms which are based on Wavelet transform. In [4] Shapiro familiarize with the embedded-zero-tree wavelet (EZW). It is a progressive image compression technique, in which embedded bits stream comes out. This extensive work is extended in [5] by A. Said and W.A.Pearlman and brought a new idea of image compression, by employing the concept of spatial orientation trees, named Set Partitioning in Hierarchical Trees (SPIHT).

1.2. Research Inspiration

Dexterous Image compression is very obligatory in storing an image. For example to save an image of size 1024 X 1024 on the disk space requires nearly equal to 3MB. In addition time compulsory to transmit this image through ISDN network is 7 minutes. On the other hand by employing the proper compression techniques we can store the same image on 300KB disk space and reducing the transmission time up to 6sec. Delay enhances as we increase the size of the file to be transmitted. So compression becomes compulsory, when we are dealing with huge amount of data, without influencing the quality of digital image prior to storing/transmitting it. At receiving stage this compressed data is decompressed.

1.3. Principles of Image Compression

When we observe digital images, there exists correlation among the neighboring pixels. This correlation has least amount of information, which we have to remove. The quality of the image is affected whenever we try to remove any data from the image. Image compression also deals with quality of the image after reduction of the data. A statistically uncorrelated set of data is formed of the pixels of image before storing or transmitting the image. At receiving end decompression gives us real or approximated image. To remove redundancy is the key part of image compression techniques, this leads us to removing duplications of data. There are several types of redundancies.

1.3.1. Spatial Redundancy

Correlation that exists among the same object is called spatial redundancy.

1.3.2. Spectral Redundancy

This is the Redundancy that exists among numerous bands of spectrum or color planes.

1.3.3. Temporal Redundancy

This is the correlation between consecutive objects or frames of image.

If spectral and special redundancies are eliminated then the bits used to represent a digital image are decreased in numbers.

1.4. Image Compression

Image compression consists of different stages. Firstly the original image is uncorrelated (exploit self-similarity of pixels) by using a linear transformation method. Then the resulting coefficients of the previous stage are quantized using a quantization step, the yield of this feeds to entropy encoded stage and this results Lossy compression as shown in Fig. 1.1. Their inverses are cascaded in same manner at receiving stage to recover equivalent image.

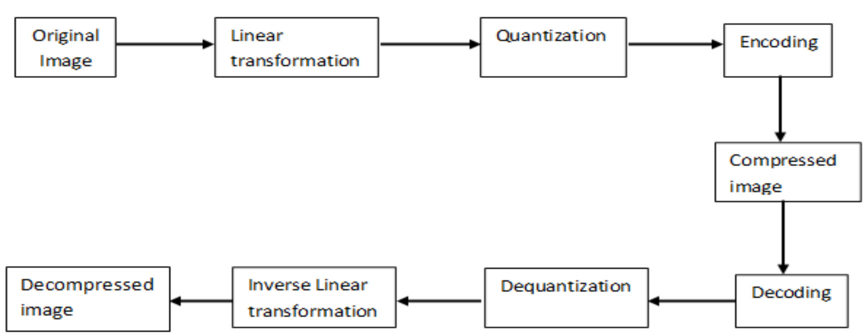


Figure 1 Error! No text of specified style in document.**-1 Compression and decompression**

Now these are discussed separately one by one.

1.4.1. Linear Transformation

Linear transformation is the procedure in which we exploit the self-similarity of the pixels to make them uncorrelated so that irrelevant data vanishes. For the sake of purpose there are lots of techniques like Discrete Cosine Transform (DCT), Discrete Fourier Transform (DFT) and Discrete Wavelet Transform (DWT). DWT considers the most differentiate than all other techniques. Later I will discuss DWT (Discrete Wavelet Transform) in detail.

1.4.2. Quantization

Quantization is the process in which discretization of intensity axis is done. Quantization minimizes the accuracy of coefficient values that obtained in the result of linear transformation. This also decreases the number of bits that are used to store an image [9].

1.4.3. Encoding

The encoder block is used further to compress the output of quantization block. In order to perform encoding first of all, probabilities of values are calculated with the help of a model. This model helps us for the generation of appropriate code for sake of purpose that resulting bit- stream is lesser than the input bit-stream [8]. As we know that Huffman and Arithmetic are those Lossless Encoding techniques which performs in much better way as compared to other techniques.

1.5. Objective of Research

The basic purpose of this research is to concatenate the Set Partitioning in Hierarchical Trees (SPIHT) with Lossless encoding schemes also known as Entropy encoding techniques like Huffman and Arithmetic encoding, in order to compare the performance parameters like Disk saving capability in terms of no. of bits saving, Compression Ratio (CR), PSNR performance and Execution time etc.

1.6. Thesis Organization

Literature Review comprises on chapter 2, 3, and 4. In Chapter 2 the discussion is on the basic concepts of wavelet Transform and Multi resolution Analysis (MRA). Chapter 3 explains in details the MRA based compression techniques like Set Partitioning in Hierarchal Trees (SPIHT) and Embedded Zero-tree wavelet (EZW) with examples. Chapter 4 throws some light on the Lossless Entropy encoding Techniques like Huffman and Arithmetic Encoding with Examples. Chapter 5 describes the Cascading of SPIHT with Entropy coding techniques Huffman and Arithmetic. In Chapter 6 MATLAB based simulation and results for the comparison of cascading of both Entropy encoding techniques with SPIHT.

Literature review

2 Wavelet Analysis

2.1. Introduction

From the last few decades, a transformation method, as replacement to sinusoidal transformation techniques like DFT and DCT come out for performing on low bit rates. In this transformation technique a new concepts of wavelets basis has introduced. Wavelets are the tiny waves of compact support i.e. limited time duration and of changing frequency. In order to examine the spatial frequency contents of pictorial data or image at multiple resolution values, these tiny wavelets can be scaled or shifted. In other words wavelets have the ability to observe the image at different resolutions and hence act as tremendous means in Multi-resolution Analysis (MRA). Additionally, Wavelet analysis is a tool for analyzing frequency contents of an image at varying spatial locations also called as space frequency localization. We can also associate the function of this technique with example that we are finding and analyzing the details of an image at particular special location by using a magnifying glass. We can zoom in or zoom out that magnifying glass to change its magnification in order to observe image details and also move slowly the magnifying glass horizontally over the surface of image to examine at different locations. This space localization property is absent in previous sinusoidal transformation techniques.

An image is nothing but an arrangement of pixels values that is stored two dimensionally; we can't get any information like spatial frequency from such images. Alternatively if we use sinusoidal transformation on image, spatial frequency information is achieved but one can't judge the location on image where these contents are present. Wavelet Transform caters for the solution to both defects and hence provides effective MRA and coding capability.

Wavelet transform is a powerful means for representing a signal. In conventional Fourier transform, a sinusoidal signal is used to represent a signal (this is also called basis function in Fourier transform). Spectral information is in detail at the cost of temporal information. On the other hand, in some of applications like music we do not need only frequencies (also called nodes) but we have to take into account of time information of a specific node i.e. we have to tradeoff between Spectral and temporal information both. In wavelet transform we use wavelets to represent a certain signal. Wavelets are the finite time function with average zero value [9, 10]. Some of famous wavelets are shown in fig. 2.1.

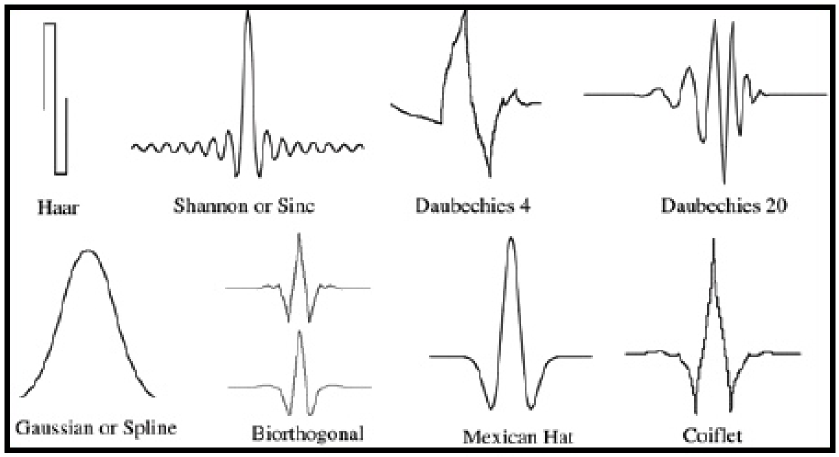


Figure 2-1 some important wavelets

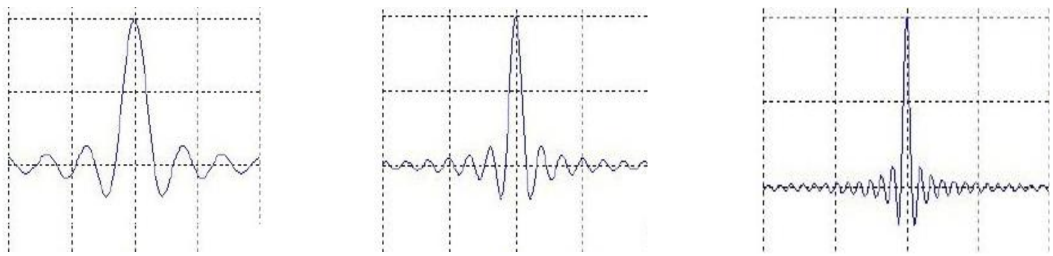
As we know that image processing is actually the matrix processing, and our eye is less sensitive to high frequency components in an image. So taking these things into consideration we can compress the high frequency components and can get a compressed image.

2.2. Background

An image is mathematically a two dimensional array (also called matrix) of color intensity values between “0” to “255” these intensity values also known as pixels of the image. An image comprises on both large and small objects and we need high resolution to see small size objects and low resolution for large size objects. This idea may lead to the employment of multi-resolution processing.

In wavelet transform a concept of mother wavelet W(t) is employed to represent any shape of waveform by translating and scaling of mother wavelet W(2kt-m). Where W(t) is wave from time t = 0 to t = T, and k is scaling factor and m is translating or shifting factor. So W(2kt-m) is that W(t) which exists from t = m to t = m+T and contract by factor of 2k.

Fig. 2.2 shows mother wavelet for different values of k. It is clear from the figure that as we enhance the value of scaling factor the mother wavelet gets narrower and narrower. Expended wavelets are comparable to sinusoids of lesser frequency, while compressed mother wavelet resembles to sinusoids of larger frequency. A wavelet is called orthogonal wavelet when the inner product of a wavelet is equal to zero.



(a) **(b)** **(c)**

Figure 2-2 Scaling the wavelet with (a) k=1, (b) k=2 and (c) k=3

2.3. Reason of Multi-Resolution Analysis

The analysis of an image concludes that different location in image has different detail levels. Locations with high level of details have large information as compared to low level of details. Better resolution is necessary for high detailed areas. That's the place where Multi-resolution analysis MRA plays its role to provide us location wise details knowledge, from which we can get our desired level of details for further processing. MRA plays a vital role in exploiting the self-similarities of image across the resolution. Wavelet analysis is one of the well-known Methods for Multi-resolution analysis.

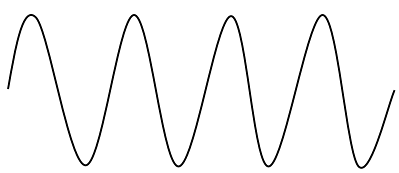
2.4. Importance of Wavelet Analysis

The preference is given to wavelet transform in many areas like Human vision, image compression, turbulence and earthquake prediction. Data compression is a key area where wavelet transform is employed on the data (image) at different resolution levels and divides the image data in different sub bands depending upon their frequencies as in [12], [13] and

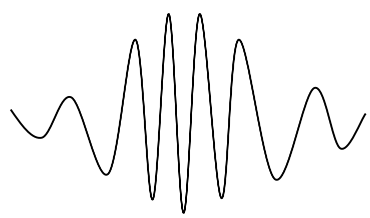
[14]. One of the most important feature of wavelet transform in performing local analysis of larger signal and using wavelet coefficient we can indicate the exact locations of the time domain discontinuities.

2.5. Benefits of Wavelet Transform

It is exhibited by the physical properties of wavelet transform that it is more advantageous than Fourier transform. As we know that in Fourier transform the bases functions are of infinite time interval i.e. from negative infinity to positive infinity, which is an impractical scenario. On the other hand, in wavelet transform the size of mother wavelet is of infinite time interval. Predictability is also one of the issues in Fourier transform due to of smoothness of the sinusoids, while in case of wavelet transform wavelets are irregular. So it is obvious from the figure 2.3 that irregularity of wavelets is a blessing to represent any kind of sharp changes in a signal. While in smooth sinusoids this goal is difficult to achieve.



(a)



(b)

Figure 2-3 Comparison of sine wave and wavelet (a) sine wave (b) wavelet

2.6. Wavelet Analysis

At this stage a new question arises in mind that what kind of mother wavelet W(t) would be useful to represent a signal. We know that an impulse function (Haar) provides us with the best resolution in time domain. Conversely the sinusoids, in Fourier transform, provide us best resolution in frequency domain. But we are interested in both best resolutions. This problem is solved by Daubechies in 1988, by presenting a new shaped wavelet also known as Daubechies Wavelet after his name as shown in figure 2.4. It is clear from figure that this wavelet is compactly supported wavelet.

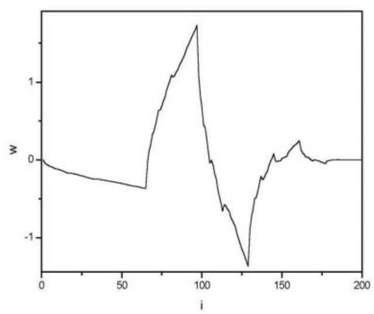


Figure 2-4 compactly supported Daubechies wavelet

A windowing notion is used in wavelet analysis. A window is an area of variable size rectangle as in [1].Different Interval sizes are used in wavelet analysis. Four different strategies are given depending upon the size of the window.

Shannon: A time domain view in which only high resolution of time scale is required and windows are adjusted only for time axis are also called Shannon as depicted in Fig. 2.5 (a).

Fourier: A frequency domain view in which only high resolution of frequency scale is required and windows are adjusted only for frequency axis are also known as Fourier transform as shown in Fig. 2.5 (b).

STFT: STFT stands for Short time Fourier Transform. In this case a fixed windows size is employed to represent a signal i.e. it is a tradeoff between Shannon and Fourier representation as shown in Fig. 2.5 (c).

Wavelet: In this representation a variable size windows is used to denote a signal. Fig. 2.5 (c).

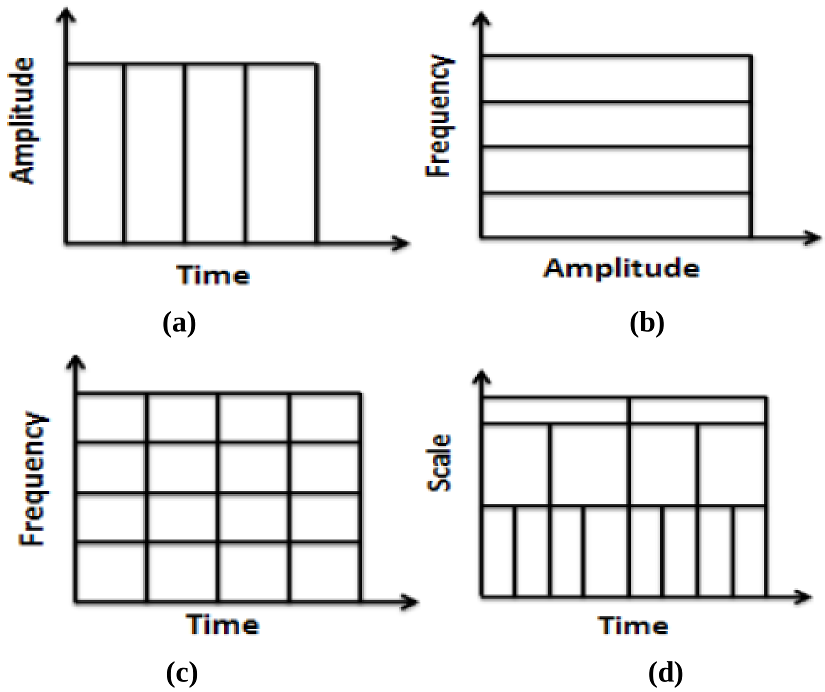


Figure 2-5 Different views of a signal (a) Time-domain (Shannon), (b) frequency-domain (Fourier), (c) STFT based (Gabor) (d) wavelet based

2.7. Perfect Reconstruction Filter Bank

A reconstruction filter bank is set of cascaded filters that are used to reconstruct the signal. After passing through this reconstruction filter bank, an input signal (matrix image) is split into two band limited sections called sub-bands. Sub-bands are formed after the passing of image through the set of band limited filters. Sub-band coding is a counter part that relates to the Multi-Resolution Analysis (MRA). On the receiving end synthesis filters are employed to rejoin the sub-bands to yield the reconstruct signal without introduction of errors. Fig. 2.6 depicts the block diagram of two channel perfect reconstruction filter bank. X(z) denotes the transmitted 1-D (one dimensional) signal. In this diagram $H_0(z)$ and $G_0(z)$ are notations of low-pass analysis filter and high-pass analysis filter respectively. Whereas $H_1(z)$ and $G_1(z)$ are the notations of low-pass synthesis filter and high-pass synthesis filter respectively.

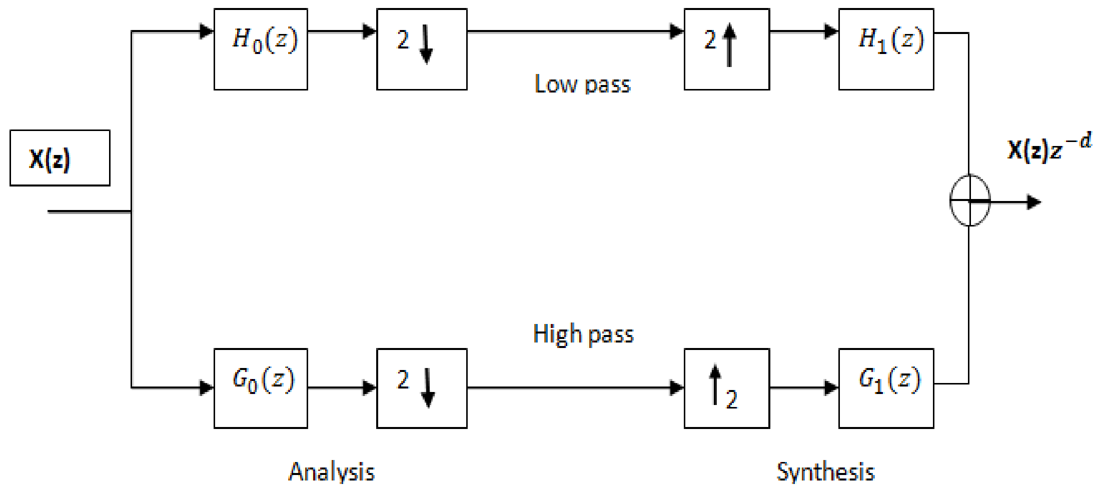


Figure 2-6 One dimensional, one level perfect reconstruction filter bank

Two sub-bands (high-pass and low-pass) are obtained as a result, when X(z) is input to analysis filters i.e. high-pass analysis $G_0(z)$ and low-pass analysis filter $H_0(z)$ respectively. The resulting sub-band signals from these analysis filters have smaller bandwidth than the original signal X(z). After that the output is down-sampled without skipping the information. This down-sampling completes the process of analysis. In reconstruction phase to recover the original signal are first up-sampled. The up-sampled signal is fed to the respective synthesis filters like low-pass synthesis filter $H_1(z)$ and high-pass synthesis filter $G_1(z)$. The resulting signal is then combine together to get back the original perfectly reconstructed signal. Here z^{-1} indicates the delaying factor of combining both sub-bands.

As the analysis filters does not capable of having model magnitude response so aliasing is familiarizes during the struggle for reserving the sampling rate after down-sampling. Analysis filters also introduced the phase and magnitude distortion to the signal. The purpose of the synthesis filters is to overcome these distortions. There should be a relation among analysis and synthesis filters as given below.

$$G_0(z)H_0(z) + G_1(z)H_1(z) = 2 \qquad \textbf{(2.1)}$$

$$G_0(z)H_0(-z) + G_1(z)H_1(-z) = 0 \qquad \textbf{(2.2)}$$

2.8. Classification of Wavelets

There is a little bit difference of relation among analysis and synthesis filters for orthogonal and bi-orthogonal bases. These wavelet bases are grouped among two classes.

(i) Orthogonal wavelet bases

(ii) Bi-orthogonal wavelet bases

2.9. Characteristics of Orthogonal Wavelet Bases

Some of real numbers constitute the coefficients of orthogonal filters. The length of these filter's coefficients are same and they are not symmetric. A time inverse relation is present among this synthesis and analysis filters.

$$H_1(z) = H_0(z^{-1}) \quad \textbf{(2.3)}$$

$$G_1(z) = G_0(z^{-1}) \quad \textbf{(2.4)}$$

If 'N' represents the length of the filter then relation can be written as under

$$G_o(z) = -z^{-N}H(-z^{-1}) \quad \textbf{(2.5)}$$

Now we can denote whole filter bank as a single filter i.e. low-pass analysis filter. The employment becomes easier by using set of orthogonal filters.

2.10. Characteristics of Bi-Orthogonal Wavelet Filter Banks

The coefficients of bi-orthogonal filters mostly comprises on integers or the real numbers. There exists the following relation as given below

$$G_0(z) = H_1(-z) \quad \textbf{(2.6)}$$

$$G_1(z) = -H_0(z^{-1}) \quad \textbf{(2.7)}$$

It is obvious from above equations that bi-orthogonal filter bank can be constructed or implemented using only two filters, which are low-pass analysis and synthesis filter. Bi-orthogonal wavelet bases can be achieved from filters with linear phase response. The length of both low-pass and high filters are different with each other. Low-pass filers possess symmetric properties while high-pass has asymmetric features.

2.11. Wavelet Transform

In wavelet transform we use wavelets to examine a signal, as we use sine and cosine basis to examine spectral components of signals in time domain. Here we discuss about the continuous wavelet transform (CWT) and discrete wavelet transforms (DFT).

2.12. Continuous Wavelet Transform (CWT)

Assuming that reader has knowledge about the Fourier transform and mathematical form of Fourier transform is given as under;

$$F(w) = \int_{-\infty}^{\infty} f(t)e^{-j\omega t}\,dt \tag{2.8}$$

In above equation exponential function denotes the combination of imaginary and real sinusoidal components. Mathematical form of CWT given as below,

$$C\,(\text{Scale}\,,\text{position}) = \int_{-\infty}^{\infty} f(t)\psi\,(\text{Scale}\,,\text{position}\,,t)\,dt \tag{2.9}$$

In order to achieve Continuous wavelet Transform (CWT) we have to take product of signal with mother wavelet. A signal is transformed in terms of scaled and translated versions of a short duration mother wavelet ψ(t) also known as compact support mother wavelet. The basis can be defined mathematically as given below,

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left[\frac{t-b}{a}\right]\;;\;\; a,b\epsilon R^1 \text{and} a > 0 \tag{2.10}$$

Where ‘a’ denotes scaling coefficient and ‘b’ denotes the translating coefficient in time domain. The mathematical form of Continuous wavelet Transform (CWT) is given as below,

$$W_f(a,b) = \int_{-\infty}^{\infty} x(t)\, \psi_{a,b}(t) dt \tag{2.11}$$

2.13. Inverse Wavelet Transform (IWT)

The inverse form of wavelet transform denoted mathematically as follows,

$$x(t) = \frac{1}{C}\int_0^{\infty}\int_{-\infty}^{\infty} W_f(a,b)\, \psi_{a,b}(t) db \frac{da}{a^2} \tag{2.12}$$

$$\text{While } C = \int_{-\infty}^{\infty} \frac{|\psi|^2}{\omega} d\omega < \infty \tag{2.13}$$

One should cater for two conditions. Firstly C should have zero mean and finite w.r.t x(t) to keep away from the singularity conditions. Mathematically,

$$\int_{-\infty}^{\infty} \psi(t) dt = 0 \tag{2.14}$$

Secondly, a finite amount of energy is associated with Mother Wavelet,

$$\int_{-\infty}^{\infty} |\psi(t)|^2 dt = \infty \tag{2.15}$$

2.14. Discrete Wavelet Transform (DWT)

Discrete wavelet transform (DWT) can be used for discrete time signals. It consists of two types, one dimensional DWT and two dimensional DWT.

2.14.1. One Dimensional DWT

The $L^2(\mathcal{R})$ associated analysis equation for orthogonal DWT given as below,

$$a_{j,k} = \int x(t) 2^{j/2} \phi(2^j t - k) dt \quad b_{j,k} = \int x(t) 2^{j/2} \psi(2^j t - k) dt \tag{2.16}$$

And synthesis equation belongs to orthogonal Inverse Discrete Wavelet Transform (IDWT) for any signal that relates to $L^2(\mathcal{R})$ as:

$$x(t) = 2^{N/2} \sum_k a_{N,k} \phi\,(2^N t - k) + \sum_{j=N}^{M-1} 2^{j/2} \sum_k b_{j,k} \psi\,(2^j t - k) \tag{2.17}$$

Where $\phi(t)$ is orthogonal scaling function, $a_{j,k}$ Are coefficients of scaling, $\psi(t)$ is orthogonal wavelet function, $b_{j,k}$ are wavelet coefficients.

The $L^2(\mathcal{R})$ associated analysis equation for orthogonal DWT given as below,

$$\tilde{a}_{j,k} = \int x(t) 2^{j/2} \tilde{\phi}\left(2^j t - k\right) dt \tilde{b}_{j,k} = \int x(t) 2^{j/2} \tilde{\psi}\left(2^j t - k\right) dt \tag{2.18}$$

And synthesis equation belongs to orthogonal Inverse Discrete Wavelet Transform (IDWT) for any signal that relates to $L^2(\mathcal{R})$ as:

$$x(t) = 2^{N/2} \sum_k \tilde{a}_{N,k} \phi\,(2^N t - k) + \sum_{j=N}^{M-1} 2^{j/2} \sum_k \tilde{b}_{j,k} \psi\,(2^j t - k) \tag{2.19}$$

Where $\phi(t)$ denotes function of scaling for Analysis Filter bank, $a_{j,k}$ are the scaling coefficient, $\psi(t)$ is wavelet function for analysis filter bank, $b_{j,k}$ are its coefficients, $\tilde{\phi}(t)$ is synthesis scaling function, $\tilde{a}_{j,k}$ are its coefficients, $\tilde{\psi}\,(t)$ denotes wavelet function for synthesis filters, $\tilde{b}_{j,k}$ are its coefficients.

2.14.2. Two Dimensional DWT

Images are nothing but 2D form of 1D signal. To examine an image cascade two stages of 1D wavelet transform stages in series. Input data of image is fed to 1D stage in row. The output of previous stage is given to the 1D stage in column. The figure given below represents the two dimensional DWT and IDWT in perfect reconstruction filter bank.

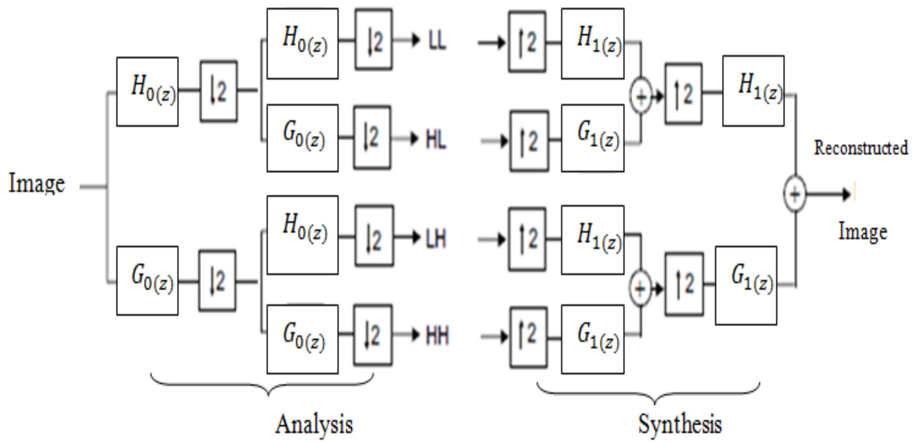


Figure 2-7 Reconstruction filter bank for 2D DWT and IDWT

Output is achieved in terms of transform coefficients, when image is fed to two dimensional bases functions. Equations for bases functions given below,

$$\phi\,(u,v) = \phi(u)\,\phi(v)$$
$$\psi_1(u,v) = \psi(u)\,\psi(v)$$
$$\psi_2(u,v) = \phi(u)\,\psi(v)$$
$$\psi_3(u,v) = \psi(u)\,\psi(v) \qquad \textbf{(2.20)}$$

Where $\phi\,(u,v)$ represents scaling function of image, $\psi_1(u,v), \psi_2(u,v)$ and $\psi_3(u,v)$ represents Wavelet functions.

As a consequence, a decomposed image is achieved as shown in figure having four sub-bands given below,

(i) LL is approximation sub-band

(ii) LH vertical details sub-band

(iii) HL horizontal details sub-band

(iv) HH diagonal details sub-band

3 SPIHT Coding

3.1. Introduction

This chapter comprises the study of embedded coding techniques based on wavelet transform, like Embedded Zero-tree Wavelet (EZW) and Set Partitioning in Hierarchical Trees (SPIHT).

3.2. Embedded Zero-Tree Wavelet (EZW)

Embedded Zero-tree Wavelet (EZW) is a technique in which self-similarity between the sub-bands of wavelet transformed image is exploited. The term Embedded refers to procedure that bit streams as a result of encoding are arranged according to their importance. In Embedded coding to achieve the desired bit rate limited by the channel EZW encoder is capable to finish the encoding at any level. To understand EZW, first of all we have to define the connection between different sub-bands in terms of spatial locations, and then establish a hierarchical tree structure which identifies the parent-offspring relationship between elements of sub-bands.

3.2.1. Relation Between Sub-bands

A Hierarchical relationship is here in sub-bands of image. In order to explain this hierarchy we must realize that a particular resolution coefficient relates to set of coefficients at next finer resolutions stages for same spatial locations. But this does not apply on highest frequency sub-band. Lower frequency sub-band coefficient can be associated as parent coefficients and that of higher frequency as offspring coefficients. Decedents are those all coefficient at finer resolution stages for particular parent at same spatial locations. In the same way Ancestors are set of those coefficients at lower resolution stages for particular offspring at same spatial locations.

This concept is further elaborated with the help of diagram given below. LL3 act as parent coefficient of all Descendent coefficients. HH3, HL3 and LH3 are three offspring of LL3. These three offspring are further related to HH2, HL2 and LH2 respectively which have four time resolution. These are further relates to HH1, HL1 and LH1 respectively which have Sixteen time resolution.

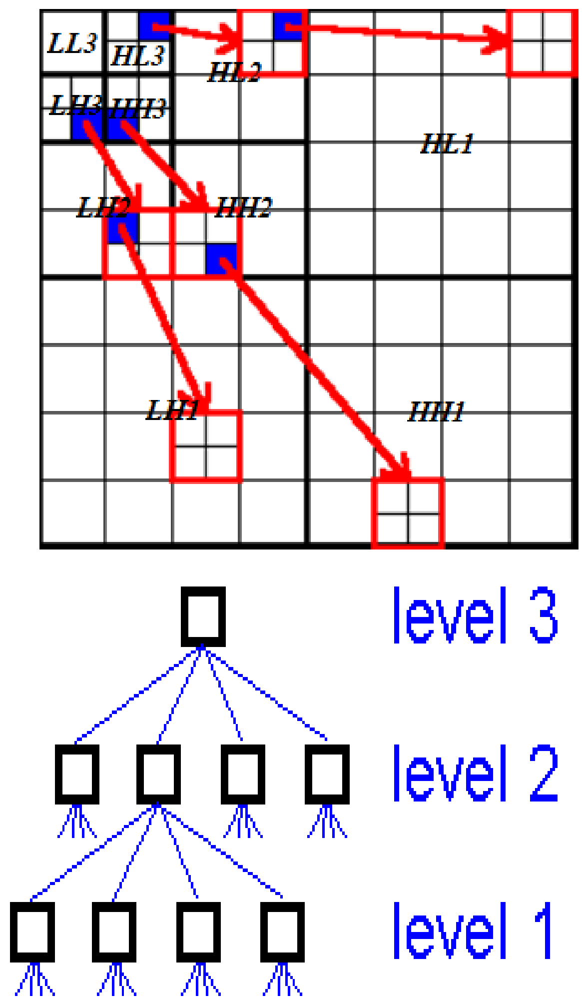


Figure 3-1 Parent-Offspring relationship b/w sub-bands

3.2.2. Significance of DWT Coefficients

The significance of DWT coefficient is very important in order to exploit the correlation between sub-bands of image to achieve encoding. A coefficient is said to be significant with respect to a certain threshold T_0 if $T_0 > |X|$ where $|X|$ is absolute value of DWT coefficient which is to be encoded, and insignificant otherwise. To define T_0 we have to take the upper bound in powers of two of the maximum value of coefficient in the DWT transformed Image. The First pass is completed when all the coefficients are encoded with given T_0 and then progressively decrease the T_0 with the factor of two and complete further passes. As well as we move towards the higher passes more and more coefficients get significant and encoded accordingly. To record the significance of coefficient a map is established at the end of each pass called significance map. If the coefficient is significant the entry in map is one otherwise it is zero. It should be noted that only magnitude of coefficient is responsible for check significance, the effect of sign (Positive/negative) is encountered later stage.

The scanning order of DWT coefficient is followed according to English alphabet 'Z' starting from the top left corner also called lowest frequency sub-band as depicted in given figure. In our given scenario it follows the path given as LL3, HL3, LH3, HH3, HL2, LH2, HH2, HL1, LH1 and finally HH1. It must be noted that no offspring should scanned before its parent coefficient.

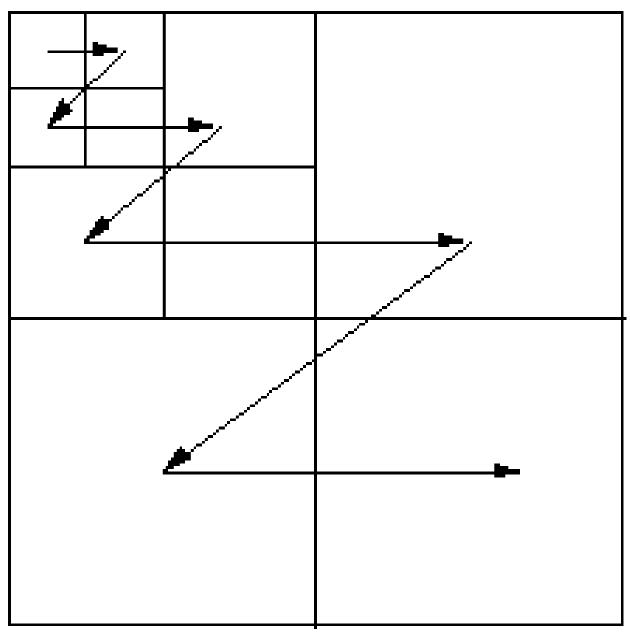


Figure 3-2 Sub-bands scanning procedure

3.2.3. Encoding in EZW

After the examination of significance of coefficients now they are ready for encoding. Before the encoding is done a hierarchical tree like data structure is need to be established called zero-tree. To define zero-tree a coefficient and their descendants must be insignificant with respect to provided threshold. Generally but not always, if a coefficient at the low frequency sub-band is insignificant then all their descendants be normally insignificant. So it is not true always for an insignificant coefficient to be zero-tree. In this case this is called isolated zero. So at this stage we define four symbols to encode the significant map given below.

(i) Zero-tree (ZTR)

(ii) Positive Significant (PS)

(iii) Negative Significant (NS)

(iv) Isolated Zero (IZ)

A coefficient is said to be positive significant if it is significant as well as positive sign, and negative significant if significant and have negative sign. In the figure given below a flow chart is used to elaborate the flow of algorithm.

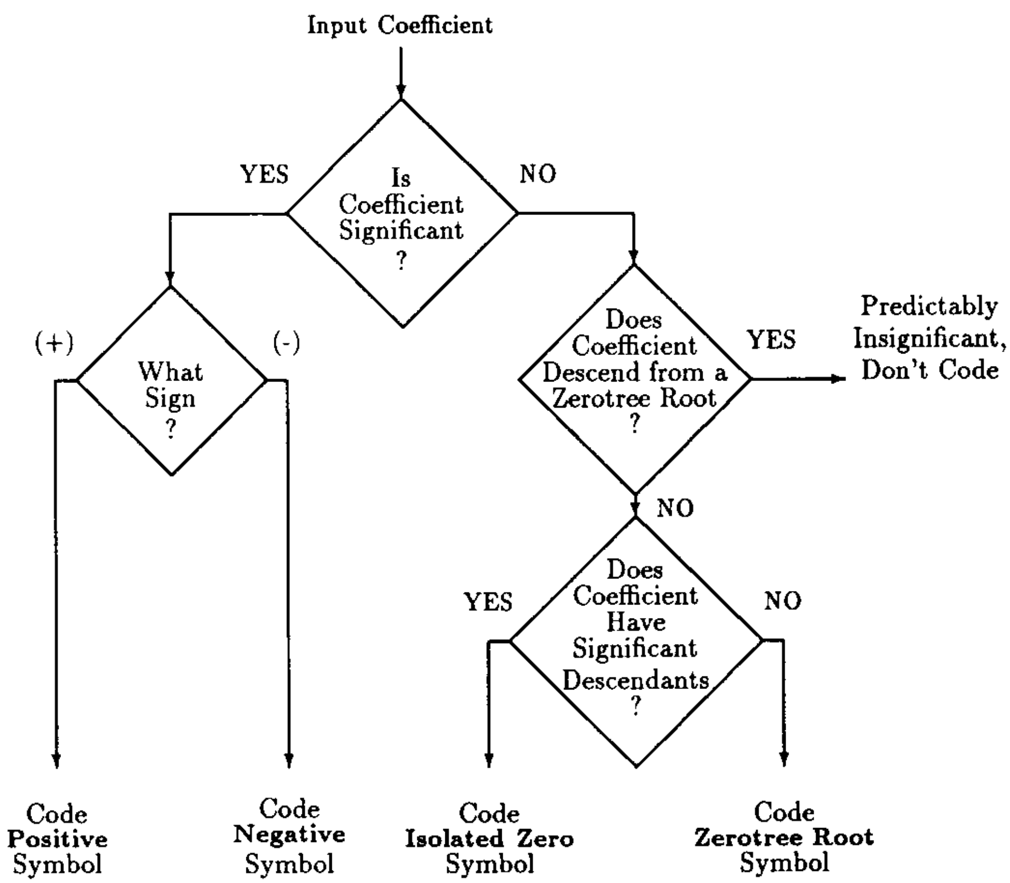


Figure 3-3 Flow chart denoting encoding of EZW

3.2.4. Successive Approximation Quantization

SAQ is a stage for quantizing the DWT coefficients with the help of a threshold. For the purpose a threshold T_0 is selected in order to attain quantization result. To define threshold we require X_{max} which is the maximum value among all DWT coefficient. T_0 is defined such as:

$$T_0 > |X_{max}| / 2 \tag{3.1}$$

After completing one stage of encoding the threshold is updated and repeats the whole procedure and compares significance with new threshold. The updating of threshold T_0 involves the making $T_0 = T_0 / 2$. So for N number of stages the threshold becomes:

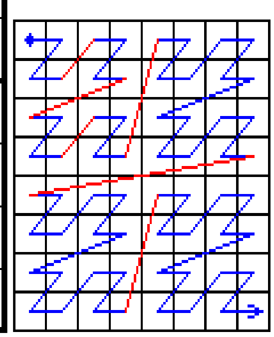
$$T_N = T_{N-1} / 2 \tag{3.2}$$

A stage include two passes one is called Dominant Pass and other is known as Subordinate Pass.

3.2.5. Example

I will exemplify the above declared algorithm with the help of following example. Data for the example is shown in figure 3.3.

63	-34	49	10	7	13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4



(a) **(b)**

Figure 3-4 (a) data set (b) scanning order (Morton scan)

First of all threshold $\mathbf{T_0}$=32 putting in equation 3.1. Following bit stream will be the output after one level of the two passes.

D1: pnztptttztttttttptt

S1: 1010

D2: ztnptttttttt

S2: 100110

D3: zzzzzppnppnttnnptpttnttttttttptttptttttttttpttttttttttt

S3: 10011101111011011000

D4: zzzzzzztztznzzzzpttptpptpnptntttttptpnpppptttttptptttpnp

S4: 11011111011001000001110110100010010101100

D5: zzzzztzzzzztpzzzttpttttnptppttptttnppnttttpnnpttpttppttt

S5: 10111100110100010111110101101100100000000110110110011000111

D6: zzzttztttztttttnnttt

We can skip the last level subordinate pass as the threshold at this level is reduced to minimum.

3.3. SPIHT Algorithm

Previously we have discussed about embedded zero-tree wavelet (EZW) encoding technique. Analysis of this algorithm exhibits two major strong points about the algorithm. Firstly, resulting bit stream is of embedded nature and DWT coefficients are arranged in accordance with their importance (significance) and precision. In this manner the output can be limited with respect to bit rate necessities of the channel. Secondly, it effectively exploits the self-similarity among sub-bands possessing similar location and it helps to trim down the data. In spite of its strengths, this algorithm is not optimal and it's a number of constraints like threshold needs to be optimized according to required bit rate. Another defect in EZW is that it does not have the capability to effectively encode the insignificant

DWT coefficients with respect to threshold and not provide confederacy of insignificant coefficients to enhance encoding efficiency.

In this chapter we discussed about the modified type of EZW algorithm, which continues the strong points of EZW like channel rate dependent transmission of data and the exploitation of correlation among sub-bands present at same locations. Additionally, it categorized the insignificant DWT coefficients. The Set Partitioning in Hierarchical Trees (SPIHT) is presented by Said and Pearlman. Experimentally, it is verified that SPIHT is optimized and well performance over EZW. Firstly we will discuss about fundamentals of progressive transmission of 2D data (image) and then throw some light on the basics of set partition terminology.

3.3.1. Coefficient Arrangement in Progressive Image Transmission

The Mathematical representation for the Hierarchical sub-band Transformation like Wavelet Transform is given below:

$$\boldsymbol{C} = \boldsymbol{\Omega}(\boldsymbol{S}) \tag{3.3}$$

Here $\boldsymbol{S}$ denotes matrix for image to be transformed $\boldsymbol{C}$ depicts the matrix of coefficients after transformation and $\boldsymbol{\Omega}$ is hierarchical sub-band transformation unitary matrix. The original image matrix and transformed coefficient matrix are dimensionally same. The function of the encoder is to transmit transformed coefficients in terms of bit stream and decoder receives the stream and estimated coefficients matrix $\hat{C}$ are generated which recovers the estimated image matrix $\hat{S}$ by inverse transformation:

$$\hat{S} = \boldsymbol{\Omega}^{-\mathbf{1}}(\hat{C}) \tag{3.4}$$

The MSE between original image and reconstructed image can be evaluated using this mathematical equation:

$$\boldsymbol{R}_{MSE}(\boldsymbol{S} - \hat{S}) = \frac{\left|\left|\boldsymbol{S} - \hat{S}\right|\right|^2}{M} = \frac{1}{M}\sum_i \sum_j \left(\boldsymbol{S}_{\boldsymbol{i},\boldsymbol{j}} - \hat{S}_{i,j}\right)^2 \tag{3.5}$$

Where M is total no. of pixels in the image and $\boldsymbol{S_{i,j}}$ is the pixel intensity value present at position(i, j). The MSE is independent to transformation and can also be represented given below mathematically:

$$\boldsymbol{R}_{MSE}(\boldsymbol{S} - \hat{\mathrm{S}}) = \boldsymbol{R}_{MSE}(\boldsymbol{C} - \hat{\mathrm{C}}) = \frac{1}{M}\sum_i \sum_j (\boldsymbol{C_{i,j}} - \hat{\mathrm{C}}_{i,j})^2 \qquad \textbf{(3.6)}$$

Here $\boldsymbol{C_{i,j}}$ denotes the pixel intensity value present at position(i, j). Initially, $\hat{\mathrm{C}}_{i,j} = 0$ for all coefficients and then encoder transmits the original value of the coefficient $\boldsymbol{C_{i,j}}$ the $\boldsymbol{R}_{MSE}(\boldsymbol{C} - \hat{\mathrm{C}})$ is lessen by $\frac{\boldsymbol{C_{i,j}^2}}{M}$**.** This depicts that MSE is largely dependent on the coefficient value. The larger the coefficient value the lesser will be the MSE. So this provides us a criterion for arrangement of coefficients. The arrangement of coefficient must satisfy the inequality$2^n \leq |\boldsymbol{C_{i,j}}| < 2^{n+1}$.

3.3.2. The Basic Purpose of Set Partitioning

In set partitioning procedure transmission of arranged and encoded coefficients is not performed explicitly. Alternatively, only those coefficients are inspected which fulfill the inequality$2^n \leq |\boldsymbol{C_{i,j}}| < 2^{n+1}$ for provided value of *n.* The significance of a coefficient is related to the condition:

$$|\boldsymbol{C_{i,j}}| > 2^n \qquad \textbf{(3.7)}$$

The coefficient to fall in the category of insignificant if it doesn't follow equation 3.7. A subset Z_m is defined, and is called significant if fulfill inequality given below and insignificant otherwise:

$$\max_{i,j \in Z_m} |\boldsymbol{C_{i,j}}| \geq 2^n \qquad \textbf{(3.8)}$$

After the declaring of significant subset it is further process unless and until separate out significant values.

3.3.3. Spatial Orientation Tree (SOT)

One major concept on which the base of SPIHT rests is association there in the sub-band structure of wavelet transformed image. This appearance a tree like structure in which the root of the tree is LL band denoted by * and the set of four children coefficients present at same spatial location are leaves as depicted in fig. given below:

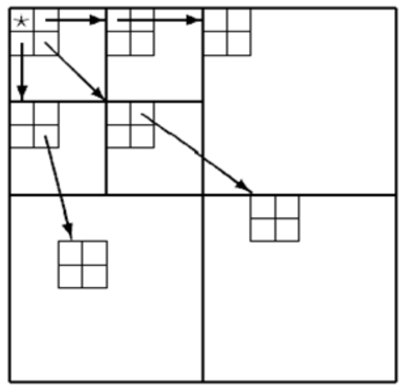


Figure 3-5 SOT diagram

The SOT for both SPIHT and EZW are resembles but with a difference that SPIHT SOT as discussed earlier, as not in the case of EZW.

3.3.4. Regulations for Set Partitioning

Before we proceed further in the algorithm, it is necessary implement set partitioning and declares some of Sets necessary for the running of algorithm.

- ❖ $\boldsymbol{O}(\boldsymbol{i},\boldsymbol{j})$ Is the set comprising on the coordinates of pixels those are children or offspring of the pixel at $(\boldsymbol{i},\boldsymbol{j})$ as '*b*' is in diagram. It might be possible that a node may have four children or it doesn't have any. So this set consists of four children or none depending on above statement. In below diagram b_1, b_2, b_3 and b_4 are offspring of *b*.
- ❖ $\boldsymbol{D}(\boldsymbol{i},\boldsymbol{j})$ Is the set comprising on the coordinates of pixels those are children and children of children also known as decedents. In diagram below b_1, b_2, b_3, b_4 and b_{11}, b_{12}, b_{13}, b_{14}, b_{21}, b_{22}, b_{23}, b_{24}, b_{31}, b_{32}, b_{33}, b_{34}, b_{41}, b_{42}, b_{43}, and b_{44} are decedents of pixel '*b*'.

❖ $\boldsymbol{L(i,j)}$ Is the set comprising on the coordinates of pixels those are difference of both decedents and offspring i.e. $\boldsymbol{L(i,j) = D(i,j) - O(i,j)}$. In diagram below $b_{11}, b_{12}, b_{13}, b_{14}, b_{21}, b_{22}, b_{23}, b_{24}, b_{31}, b_{32}, b_{33}, b_{34}, b_{41}, b_{42}, b_{43},$ and b_{44} are $\boldsymbol{L(i,j)}$ of pixel '*b*'.

❖ $\boldsymbol{H(i,j)}$ Is the set comprising on the coordinates of pixels those behaves as root of the tree. In diagram below *a, b, c,* and *d* are $\boldsymbol{H(i,j)}$.

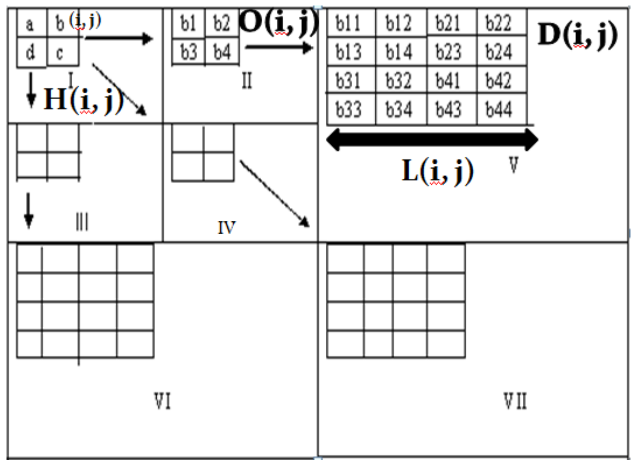


Figure 3-6 Set Partitioning scheme

3.3.5. SPIHT Encoding and Decoding Procedure

After the successful application of regulations for set partition now its turn to perform encoding and decoding operations on the data. As above stated both encoder and decoder does not support sending explicitly the arranged data as in case of SPIHT algorithm and hence enhance the coding efficiency of algorithm, so both encoder and decoder are inverse replica of each other. In order to preserve the changes in data three lists are devised to form:

❖ LIP (List of Insignificant Pixels)

❖ LSP (List of Significant Pixels)

❖ LIS (List of Insignificant sets)

Each entry in all these three sets is filled with respect to pixel$(\boldsymbol{i,j})$. LSP and LIP comprises on the pixels coordinates. While LIS comprises on set $\boldsymbol{D(i,j)}$ and $\boldsymbol{L(i,j)}$.

To carry on SPIHT encoding first of all initialization is performed in which some parameters are defined to run the algorithm properly, like 'n' a number of refinement passes. After this three passes are declared as follows:

- ✓ The sorting pass
- ✓ The magnitude refinement pass
- ✓ improvement of quantization step pass

The passes are recurred but for the transmission of least significant bits for refinement purpose. In the sorting pass the significance of entries of LIP with respect to certain threshold defined previously are checked. The significant entries are now transferred to LSP and insignificant entries remained in LIP. Furthermore, the scanning of LIP is performed and check the significance of sets and if set is significant then it significant entries are transferred to the LSP and that of insignificant entries to LIP. In refinement pass previously encoded LSP entries are dealt and transmit their nth most significant bit. After the above discussion, we can summarize the encoding algorithm as follows:

3.3.5.1. Initialization

- Output $n = \lfloor \log_2(\max_{(i,j)} \{|\boldsymbol{C_{i,j}}|\}) \rfloor$
- Set the LSP = { }
- Set the LIP = $\{(i, j) \in \boldsymbol{H}\}$and LIS = $\{\boldsymbol{D}(\boldsymbol{i}, \boldsymbol{j}), (\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{H}\}$

3.3.5.2. Sorting Pass

- In this pass examine entries of the list LIP for significance with respect to T_0 and transmit '1'transmit sign of significance and move that entry to LSP otherwise transmit '0'.
- Examine sets present in LIS for significance if set is significant transmit '1' for significance and '0' otherwise, further examine entries of that respective significant set and follow step 4. Similarly updating LIP LIS and LSP.

3.3.5.3. Refinement Pass

- In this pass examine previous entries of LSP and transmit the n^{th} most significant bit of that entry of LSP.

3.3.5.4. Renewing Quantization Step Pass

- Lessen the value of n by one and recur all previous steps unless and until n become zero.
- Now for the purpose to design decoder perform all steps in a reverse manner and out bit stream of encoder becomes the input of decoder. Further an entropy encoding is attached with this to make it more efficient.

3.3.6. Example

I exemplify the idea of this algorithm using this example.

26	6	13	10
-7	7	6	4
4	-4	4	-3
2	-2	-2	0

Figure 3-7 Data set for example

The encoder sends bit stream using three different passes then decoder recovers and decodes this bit stream.

3.3.6.1. First Pass

$T_0 = 2$ ^ n and n = 4 (as stated earlier) so threshold = 16. At this level encoder generates following three lists.

LIP: {(0,0) → 26 , (0,1) → 6, (1,0) → −7, (1,1) → 7}

LIS: {(0,1)D , (1,0)D , (1,1)D}

LSP: {}

Let's begin with the entries of LIP. As we know that first entry of the set LIP placed at (0,0) location is larger than 16 i.e. 26 is significant. So '1' is send to receiver for the coefficient to be significant chased by a '0' to indicate its positive sign and enter coordinate to the LSP. The subsequent coefficients of the LIP are insignificant so 0 is transmitted for all these remaining coefficients. These coefficients are not shifted to any other list. The elements of LIS are examined in the next step. None of descendants of coefficient at position (0,1) i.e. (13, 10, 6, and 4) are significant so one 0 is transmitted for these descendants. Similarly 0 is transmitted for the rest of coefficients whose descendants are found to be insignificant. In refinement pass we do not need to do anything as we don't have any element in LSP from previous pass. So 8 bits are transmitted after this pass i.e. 1 0 0 0 0 0 0 0 . After first pass the three lists are as follows:

LIP: $\{(0,1) \rightarrow 6\ ,\ (1, 0) \rightarrow -7\ ,\ (1,1) \rightarrow 7\}$
LIS: {(0,1)D , (1,0)D , (1,1)D }
LSP: $\{(0,0) \rightarrow 26\}$

3.3.6.2. Second Pass

For this pass n is reduced to 3. So threshold becomes 8. First, the elements of LIP are examined in this pass as well. All the elements are insignificant for this value of threshold so three 0s are transmitted. Now elements of LIS are examined. The first set of LIS has two of its descendants i.e. 13 and 10 significant. So the whole set is significant. So 1 is transmitted for this set. Now the offspring of these sets are checked. Out of these offspring, coefficient 13 is significant so 1 is transmitted for its significance and 0 for its positive sign. Same is the case with offspring 10. 1 and 0 are transmitted for 10. The coordinates of the two offspring are shifted to LSP. Further two offspring are insignificant so they are shifted to LIP transmitting 0 for both.

The elements of LSP came from previous pass are examined in refinement pass. Only 26 is the element that is from the previous pass. As n = 3, looking at 3^{rd} MSB of element

26 i.e. 1, we send a 1 in refinement pass. So after second pass 13 bits are transmitted. These are 0001101000001. After second pass the three lists are as follows:

LIP: {(0,1) → 6, (1, 0) → −7 , (1,1) → 7 , (1,2) → 6 , (1,3) → 4}
LIS: {(1,0)D , (1,1)D}
LSP: {(0,0) → 26 , (0,2) → 13 , (0,3) → 10}

3.3.6.3. Third pass

For this pass n is reduced to 2. So threshold is now 4. Since the threshold gets smaller, the chance of more number of coefficients considered significant is increased. After this pass the bit stream transmitted is 10111010101101100110000010 and the three lists are as follows:

LIP: {3,0) → 2 , (3,1) → −2 , (2,3) → −3 , (3,2) → −2 , (3,3) → 0}
LIS: {}
LSP: {(0,0) → 26 , (0,2) → 13 , (0,3) → 10 , (0,1) → 6 , (1,0) → −7 , (1,1) → 7 , (1,2) → 6 , (1,3) → 4 , (2,0) → 4 , (2,1) → −4 , (2,2) → 4}

3.3.7. Features of SPIHT

Set Partitioning in Hierarchical Trees (SPIHT) is an embedded coding technique which provides number of good characteristics given below:

- Better image quality with good PSNR values
- This algorithm is optimized
- Fully Embedded coded output
- Encoding decoding is fast
- An Adaptive Algorithm
- Can also be used for lossless compression
- Coding is done to any specified bit rate

It is obvious from widespread research that all the wavelet based algorithms provides very good image quality for the purpose of visualization. Initially simple wavelet based methods are used as the foundation for the JPEG2000 standards because of its good enough image quality. On the other hand, Set Partitioning in Hierarchal Trees (SPIHT) is the latest generation encoders which are based on Wavelet transform and have classier coding capabilities. SPIHT achieves this place by exploitation of self-similarity of sub-bands of the wavelet transformed images.

SPIHT, on the other end, is the most up-to-date algorithm that's provide the optimal results in progressive transmission of wavelet based images and it outclass all other non-progressive techniques. So it yields a fully embedded coded results based on the fact that best results are obtained for any given resources (i.e. on all required bit rates).

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. But what is the price to pay for this "amenity"? Surprisingly, with SPIHT all three users would get (for the same file size) an image quality comparable or superior to the most sophisticated non-embedded encoders available today. SPIHT achieves this feat by optimizing the embedded coding process and always coding the most important information first.

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.

4 Huffman & Arithmetic

In the previous chapter I have discussed about the Lossy image compression techniques those are wavelet based like EZW and SPIHT. In this chapter a discussion on Lossless Entropy Encoding schemes like Huffman and Arithmetic coding.

4.1. Huffman Coding

In this technique a symbol probabilities based coding of bit stream is done and in spite of assigning same no. of bits different no. of bits assigned to symbols depending on their probabilities. That's why this is also known as Variable Code Length (VLC).

4.1.1. Key Principles of Huffman Coding

Here are some principles upon which the base of Huffman coding rests:

- Larger probability symbols are represented by least number of bits while lesser probability symbols are represented by number bits as compared to larger probable symbols and assign variable length Code-word to fixed group of symbols.
- To make Huffman coding a distinctively decodable technique, in assigning Code-word to next symbol no previous Code-word is present as it is in current symbol's Code-word.
- Each Code-word of a symbol must be unique.

This coding is always used in cascading with a Run Length Code (RLC) like discussed in previous chapter. Our aim is to concatenate this coding technique with SPIHT and record results for sake of comparison.

4.1.2. Flowchart for Huffman Coding

In figure given below the sequence of algorithm is mentioned in terms of flowchart. In this first of all arrange the symbols in accordance with their decreasing probabilities values. Then make a subgroup by joining the two least probable symbols and then assigning both symbols a bit '1' to upper symbol and '0' to lower symbol. After that check for more unmerged groups if yes, then repeat previous steps otherwise start generating the Code-word for symbols.

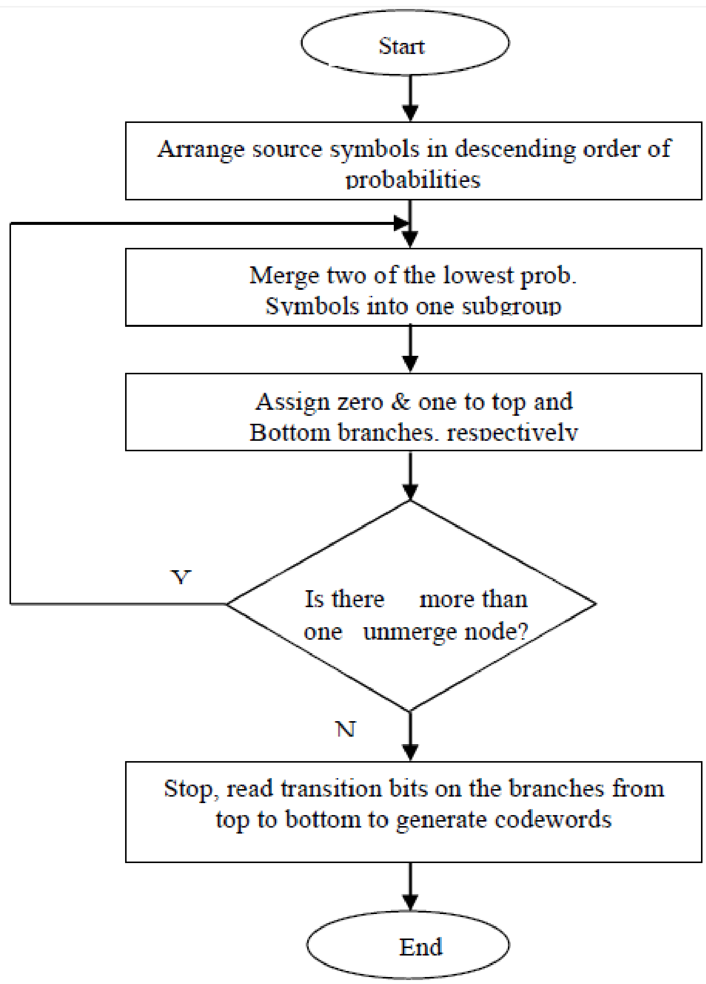


Figure 4-1 Flowcharts for Huffman Coding

4.1.3. Example

In order to understand the above stated procedure here is given an example to clarify the concept. In table below a set of symbols is given with their frequency of occurrence.

Symbols	**Frequency**
222	5
136	7
14	9
2	10
0	100

Step-1: Arrange symbols with respect to their decreasing frequency of occurrence.

Symbols	**Frequency**
0	100
2	10
14	9
136	7
222	5

Table 4-1 Tables of Arranged Symbols

Step-2: Merge two least frequent symbols to make subgroups and add their occurrences to get a total value of all subgroups.

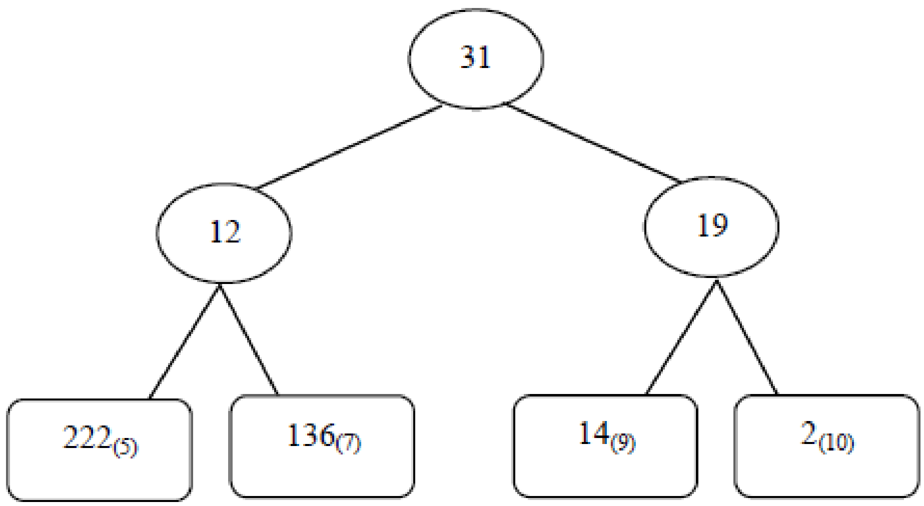


Figure 4-2 formation of Subgroups

Step-3: Check for the presence of one unmerge node

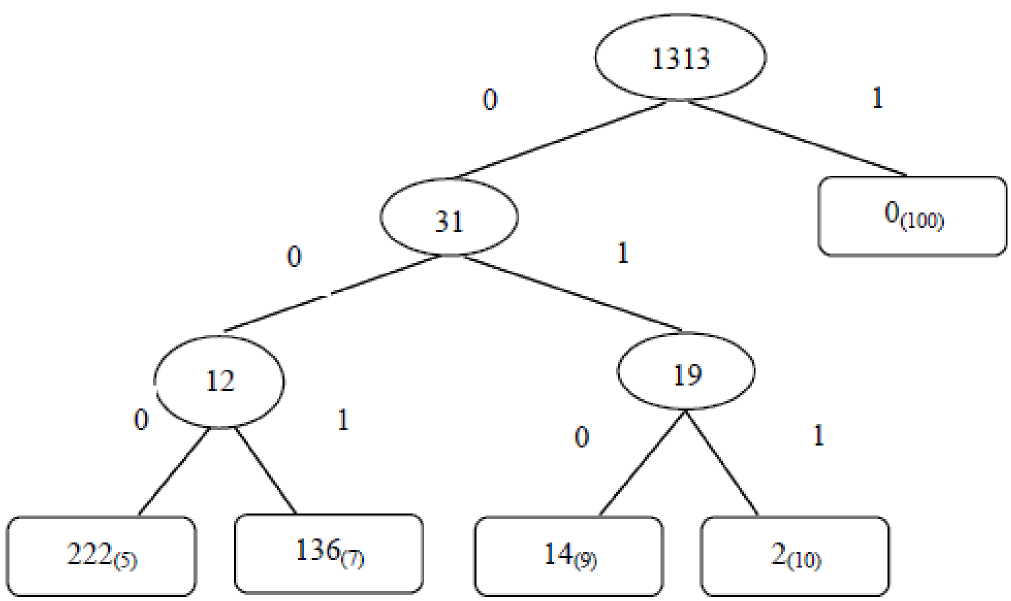


Figure 4-3 Huffman tree processing

Step-4: Assign Code-word to symbols.

Symbols	Code-word	Frequency
0	1	100
2	011	10
14	010	9
136	001	7
222	000	5

Table 4-2 Assigning Code-word

4.2. Arithmetic Coding

It is also a Variable Code Length (VLC) and Lossless coding technique like Huffman. This technique is also necessitating the information of priori of frequency of occurrence. The basic principles of Arithmetic coding are given as under:

4.2.1. Key Principles of Arithmetic Coding

Here are some principles upon which the base of Arithmetic coding rests:

- In this technique Variable length Code-word allocate to variable length of symbols, opposite to Huffman coding where Variable length Code-word allocate to fixed length of symbols.
- All the symbols in a message are regarded as jointly to assign a single arithmetic Code-word.
- The symbols and Code-word not corresponding one to one.
- The Code-word are depends upon a real number range from [0,1), which is a half-open interval. This interval is further sub-divided into smaller and smaller intervals as the coding progresses.

4.2.2. Flowchart for Arithmetic Coding

The Algorithm for Arithmetic coding is given as under:

```
   START
Low-limit = 0.0; High-limit = 1.0; Interval-size = 1.0;
while (symbol != terminator)
{ get (symbol);
low = low + range * Range_low(symbol);
high = low + range * Range_high(symbol);
range = high - low; }
output a code so that low <= code < high;
    END.
```

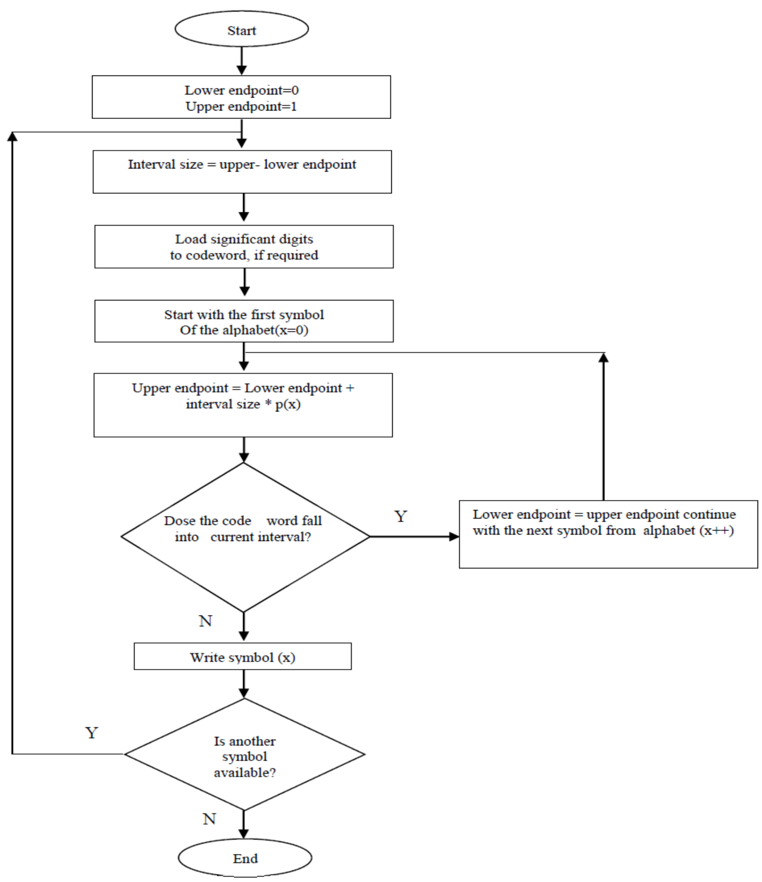


Figure 4-4 Flowchart of Arithmetic coding

4.2.3. Example

Here given a table of symbols with their probabilities and ranges.

Symbols	Probability	Range
0	0.63	[0 , 0.63)
2	0.11	[0.63 , 0.74)
14	0.1	[0.74 , 0.84)
136	0.1	[0.84 , 0.94)
222	0.06	[0.94 , 1.0)

Table 4-3 Symbols with Probabilities and Range

Symbol Array: 2 0 0 136 0

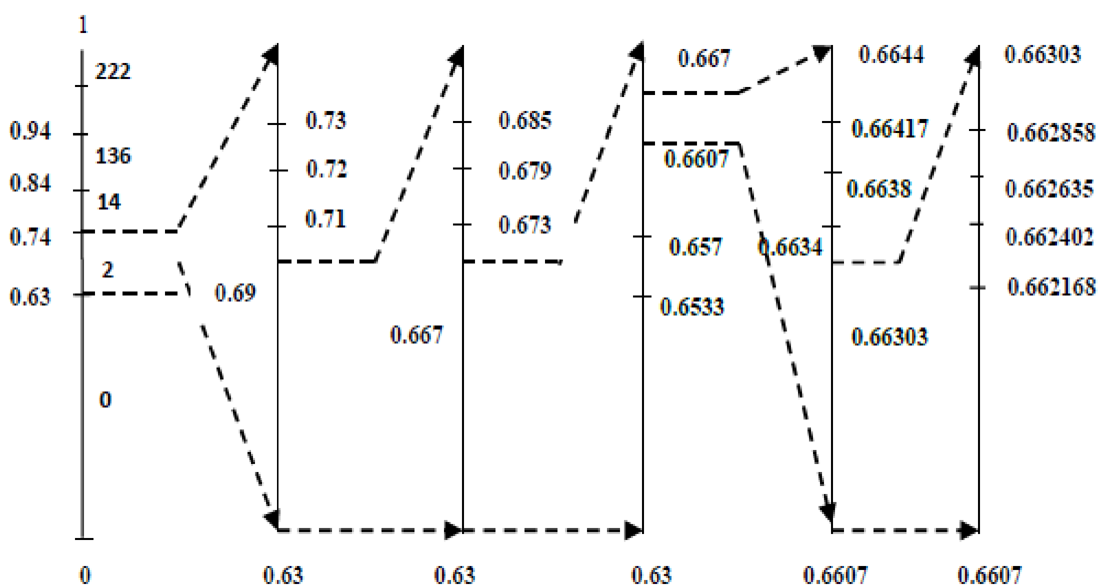


Figure 4-5 Shrinkage of Ranges procedure

Output: [0.6607 , 0.66303)

Entropy Encoding with SPIHT

5 Huffman and Arithmetic with SPIHT

5.1. Introduction

In this chapter I will discuss about concatenation procedure in detail of SPIHT and Entropy coding schemes (Arithmetic and Huffman), and then compare performance parameters of both cascading.

5.2. Concatenation of SPIHT and Huffman

As we observed in previous chapter that output bit-stream of SPIHT consists of seriate ‘0’. Statistical study of images has revealed that the symbol ‘000’ is highly probable symbol in most of images, normally$P('000') \geq 0.25$. Therefore the binary output bit-stream of SPIHT algorithm is grouped into the three bits and this group constitutes a symbol. This grouping of three bits provide us with eight possible symbols i.e. ‘000’, ‘001’, ‘010’, ‘011’, ‘100’, ‘101’, ‘110’and ‘111’. These symbols are then fed to the Huffman coder block which is a variable length coding scheme base of the symbol probabilities as discussed in previous chapter.

After the grouping and symbolization of binary data, there is a possibility that either zero, one or two last bits remain as ungrouped and hence cannot take part in symbolization. So the information of these reaming bits is added to the start of output of Huffman coder block as header bits. This header consists of two bits denoting either zero one or two remaining bits following the Huffman generated bit-stream. At the end of this resulting bit-stream the reaming bits are added to the output as shown in figure:

Header consisting of two bits denoting number of reaming bits.	Huffman generated bit stream	Reaming bits

Figure 5-1 Huffman coder block output

The probabilities of the symbols are calculated for Lena512 image which is recorded in table given below and their generated Code-word as a result of Huffman coder block.

Symbols	Probabilities	Code-word
'000'	0.2410	00
'001'	0.1430	100
'010'	0.1247	101
'011'	0.1011	110
'100'	0.1456	111
'101'	0.0785	0011
'110'	0.0982	010
'111'	0.0679	0110

Table 5-1 probabilities and Code-word for symbols for Lena512 at 0.5 Bpp

Now we can observe clearly from Table above that highly probable symbols are represented by least number of bits. So this cascading result the decrease the number of bits. This helps to save lot of disk space and transmission enhancement. At the Decoder stage all the steps are performed in reverse order to get back the reconstructed image.

To calculate the entropy or Average code length of Huffman coding uses the equation below:

$$Avg\ Code\ Length = \sum_{i=0}^{8} P(i) * L_i \tag{5.1}$$

Here $L_i$denote the length of corresponding Code-word and $P(i)$is probability of that Code-word.

5.3. Concatenation of SPIHT and Arithmetic

To concatenate Arithmetic coder block with SPIHT algorithm repeat the same procedure as done in previous cascading i.e. grouping of three bits and then convert the SPIHT bit-stream output to decimal symbols-stream consisting of eight symbols ‘0’, ‘1’, ‘2’, ‘3’, ‘4’, ‘5’, ‘6’ and ‘7’. Then apply the Arithmetic coding on resulting Decimal symbol-stream and get a decimal number.

5.4. Conclusion

After the successful cascading for both entropy coding scheme using MATLAB I reconstruct the images and calculate important measures to judge and compare both techniques and drawn important results as discussed in next chapter.

Simulation & Results

6 Simulation and Results

6.1. Simulation

This chapter consists of the details of simulation and deduced results from the simulation. So in order to compare the results of cascading of entropy encoding schemes (Arithmetic and Huffman) with SPIHT algorithm I use MATLAB 7.11.0 (R2010b), and calculate some performance measures like Peak Signal to Noise Ratio (PSNR) value, number of saved bits, Compression Ratio (CR) and Elapsed time or Execution time for algorithm. Here are provided the tables below which exhibit the results Using Lena image of different sizes like 64x64, 128x128, 256x256 and 512x512 and calculate performance measures like Number of bits saving by both cascading, PSNR performance which tells about the quality of reconstructed image, Compression ratio (CR) another measure for compression efficiency of algorithm and algorithm execution time denotes the speediness of algorithm with cascading of entropy encoding schemes (Huffman and Arithmetic).

Peak signal to noise ratio (PSNR) is measure the quality of image which represents mathematically as:

$$\mathrm{PSNR} = 10\ \log 10_{10}\left(\frac{\left(\max\left(\mathrm{h(i,j)}\right)\right)^2}{\mathrm{MSE}}\right) \tag{6.1}$$

Here $\mathrm{h(i, j)}$are the image pixel values. For grayscale image it is given as below:

$$\left(\max\left(\mathrm{h(i, j)}\right)\right) = 255 \tag{6.2}$$

Mean Square Error (MSE) denotes a term which is use to compare original image and reconstructed image, equation given below:

$$\mathrm{MSE} = \sum_{\mathrm{MN}} \frac{\mathrm{h(i,j)} - \tilde{\mathrm{h}}\mathrm{(i,j)}}{\mathrm{M \times N}} \tag{6.3}$$

Further in order to concatenate the Entropy encoded blocks (Huffman and Arithmetic) with SPIHT encoder block, make combination of three bits of the resultant bit-stream of SPIHT to form eight symbols set and calculate the probability of each symbol and then fed this to the Huffman and Arithmetic coding block and calculate performance measures and compare results of both techniques.

6.2. SPIHT + Huffman Vs. SPIHT + Arithmetic

After Successful cascading of both techniques I calculate some performance measurements and results. In Table 5-1 I have calculated the difference of bits for Only SPIHT with that of both cascading (i.e. Huffman and Arithmetic) and find Number of Bits saving and for Different sizes of Lena image, and Draw a histogram in Figure 5-1 for better visualization result only for Lena 512x512. So we can conclude that SPIHT and Arithmetic performs in much better way as compared to SPIHT and Huffman in terms of Bits Saving Capability.

Rate	Number of Bits Saving							
	Lena 64x64		Lena 128x128		Lena 256x256		Lena 512x512	
	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic
0.1	34	29	71	74	296	332	1149	1290
0.2	56	53	132	131	558	648	1888	2154
0.3	45	47	222	232	700	819	2908	3284
0.4	62	66	297	309	993	1186	3358	3838
0.5	91	94	311	331	1233	1437	4153	4810
0.6	122	129	379	420	1282	1527	4359	5150
0.7	124	136	433	487	1522	1816	3716	4851
0.8	143	159	524	582	1658	1989	4167	5570
0.9	154	175	541	613	1820	2156	5093	6676
1.0	168	189	570	653	1849	2218	5520	7224

Table 6-1 for comparison of bits saving for two schemes

Table 5-2 shows PSNR performance providing by both Cascading for Different sizes of Lena image like 64x64, 128x128, 256x256 and 512x512 and for different bit rates. We conclude from table that PSNR performance for both cascading is same. We can get image compression with SPIHT and Arithmetic by preserving PSNR value i.e. Quality for image is preserved. A graph is shown in Figure 5-2 PSNR for both on different image resolutions like 64x64, 128x128, 256x256 and 512x512, from which we can conclude that PSNR get increased as we continue to enhance bit rate and image size.

Rate	PSNR Performance							
	Lena 64x64		Lena 128x128		Lena 256x256		Lena 512x512	
	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic
0.1	18.41	18.41	21.25	21.25	24.45	24.45	28.13	28.13
0.2	20.36	20.35	23.55	23.55	27.11	27.11	31.02	31.02
0.3	21.85	21.85	25.22	25.22	28.73	28.73	32.95	32.95
0.4	23.10	23.10	26.33	26.33	30.44	30.44	34.30	34.30
0.5	24.35	24.35	27.47	27.47	31.71	31.71	35.56	35.56
0.6	25.14	25.14	28.64	28.64	32.66	32.66	36.53	36.53
0.7	25.89	25.89	29.74	29.74	33.76	33.76	37.36	37.36
0.8	26.67	26.67	30.59	30.59	34.86	34.86	38.15	38.15
0.9	27.36	27.36	31.33	31.33	35.74	35.74	38.83	38.83
1.0	28.21	28.21	32.22	32.22	36.47	36.47	39.65	39.65

Table 6-2 for comparison of PSNR for two schemes

Rate	Compression Ratio							
	Lena 64x64		Lena 128x128		Lena 256x256		Lena 512x512	
	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic
0.1	1.0907	1.0763	1.0453	1.0473	1.0473	1.0534	1.0458	1.0518
0.2	1.0734	1.0692	1.0420	1.0417	1.0445	1.0520	1.0374	1.0428
0.3	1.0380	1.0398	1.0473	1.0495	1.0369	1.0435	1.0384	1.0436
0.4	1.0393	1.0420	1.0475	1.0495	1.0394	1.0474	1.0331	1.0380
0.5	1.0465	1.0481	1.0395	1.0421	1.0391	1.0459	1.0327	1.0381
0.6	1.0522	1.0554	1.0401	1.0446	1.0337	1.0404	1.0285	1.0339
0.7	1.0452	1.0498	1.0392	1.0443	1.0343	1.0412	1.0207	1.0272
0.8	1.0456	1.0510	1.0416	1.0465	1.0327	1.0394	1.0203	1.0273
0.9	1.0436	1.0498	1.0381	1.0434	1.0318	1.0379	1.0221	1.0291
1.0	1.0428	1.0484	1.0360	1.0415	1.0290	1.0350	1.0215	1.0283

Table 6-3 for comparison of Compression Ratio (CR) for two schemes

Rate	Execution time (sec)							
	Lena 64x64		Lena 128x128		Lena 256x256		Lena 512x512	
	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic	SPIHT + Huffman	SPIHT + Arithmetic
0.1	0.4707	1.1237	1.4715	1.5116	1.7089	3.3623	7.7691	9.0000
0.2	0.1963	0.2192	1.4278	1.8258	4.0136	8.4732	14.9828	21.2069
0.3	0.2519	0.2806	0.7903	1.0740	5.1116	6.5069	22.7861	31.4346
0.4	0.3020	0.3918	1.0538	1.3373	6.2057	7.4415	36.1099	41.3791
0.5	0.6145	0.4376	2.6571	3.7305	7.2749	10.7266	46.4404	51.4574
0.6	1.0001	0.5163	2.2728	2.0527	10.3345	12.1818	64.1089	74.2677
0.7	1.0924	1.3293	2.3928	4.3114	12.0107	14.2440	72.5772	89.4216
0.8	0.6628	1.8093	3.1464	3.9456	13.6070	16.9060	107.5733	106.6014
0.9	0.5880	0.8097	3.7925	3.6699	16.6915	18.8235	124.0462	120.7071
1.0	0.6782	0.8633	2.7453	5.1543	18.7386	24.2221	148.4652	151.2966

Table 6-4 for comparison of Execution time for two schemes

Table 5-3 provides us a comparison for both in terms of Compression Ratio (CR) which gave same conclusion as that of Bits Saving capability. Table 5-4 provides the information of Execution times for both cascading in terms of Algorithm Efficiency. Here we can get an idea that SPIHT with Huffman is more efficient and have easy implementation as compared to SPIHT with Arithmetic. To make the idea more clear a histogram is drawn in Figure 5-3 which shows that SPIHT with Arithmetic take much time as compared to other.

Figure 5-4 shows the Original Lena Image at different resolutions and Reconstructed Lena Images with same resolution at bit rate 1.0 Bpp. As the size of image is increased the Quality gets better.

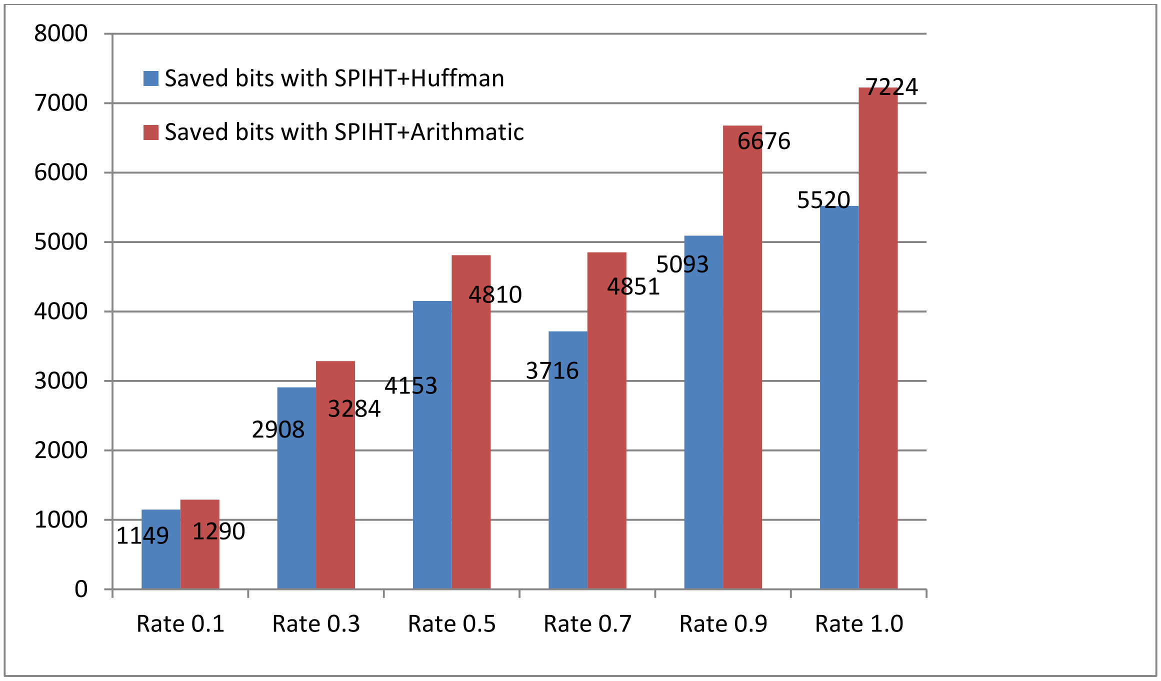


Figure 6-1 Histogram representing Bits saving using Lena 512x512

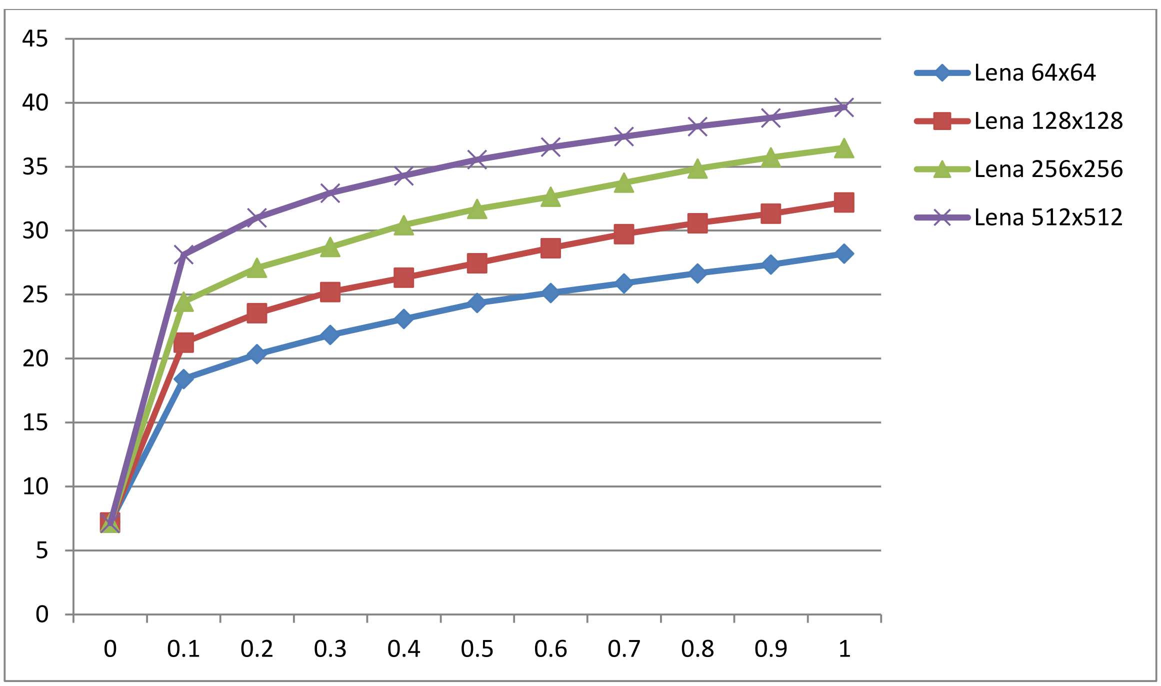


Figure 6-2 Graph representing PSNR performance

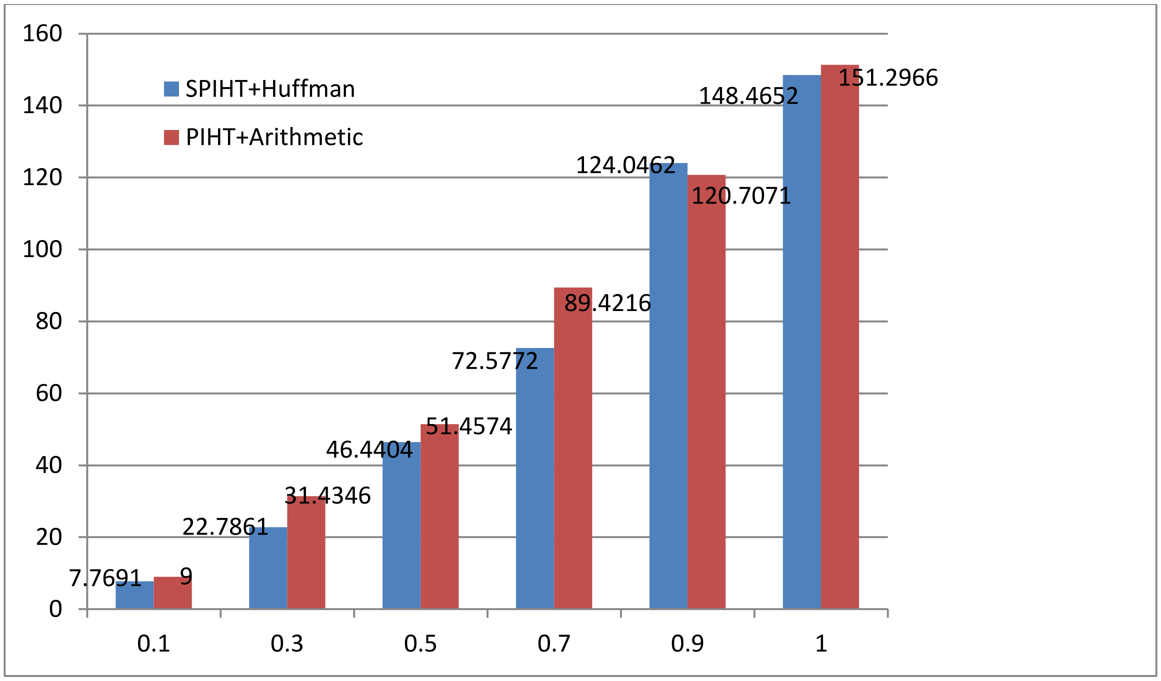
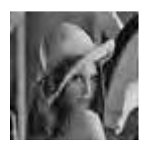


Figure 6-3 Histogram representing Execution time (sec) using Lena 512x512



Original Lena 64x64 **Reconstructed Lena 64x64**

Original Lena 128x128 **Reconstructed Lena 128x128**

Original Lena 256x256

Reconstructed Lena 256x256

Original Lena 512x512

Reconstructed Lena 512x512

Figure 6-4 Original and Reconstructed Image at different sizes

Some other images are tested for the sake of verification, the below images are consisted on bird, couple and Eliana images of 256x256. Left side original image and on right side reconstructed images.

More images tested for both Cascaded Algorithms

6.3. Discussion

From the above results we conclude that SPIHT combined with Arithmetic coding yields better Compression Ratio (CR) and less Disk storage capability as compared to SPIHT combined with Huffman coding. But in terms of Efficiency and implementation point of view SPIHT with Huffman performs much better as compared to SPIHT with Arithmetic coding. It is also evident that PSNR performance preserves for both techniques.

6.4. Future Work

I have done my research using standards Huffman and Arithmetic encoding blocks, which can be optimized further and can refine and enhance the results in terms of compression ratio (CR), Disk saving capability, PSNR and execution time.

References

1. BRAHIMI, T. (2006). Improvements to SPIHT for Lossless Image Coding. *supported by the Algerian project "Biomedical Signal Processing " at the University of Jijel.* (pp. 1445-1450). NORTHERN IRELAND: IEEE.

2. Cheng Li-chi, W. H.-x. (2004). *Wavelet theory and applications.* China: Beijing: Science Press.

3. Compression Efficiency for Combining Different Embedded Image Compression Techniques with Huffman Encoding. (2013). *International conference on Communication and Signal Processing, April 3-5, 2013* (pp. 816-820). India: IEEE.

4. Eddins, M. S. (1990). Analysis/synthesis techniques for subband image coding. *IEEE Trans* , 1446-1456.

5. Kaur, C. (1, January 2013). Improvements of SPIHT in Image Compression- Survey. *International Journal of Emerging Technology and Advanced Engineering* (pp. 652-656, Volume 3). Chandigarh: IJETAE.

6. Lightstone, E. M. (1997). *Lowbit-rate design considerations for waveletbased waveletbased.* india: Multidimensional Syst. Signal Processing.

7. MATHIEU, M. A. (1992). Image coding using wavelet transform. *IEEE Trans.* , 205-220.

8. Nguyen, G. S. (1996). *Wavelets and Filter Banks.* Cambridge: Wellesley-Cambridge.

9. Pandian, 1. (2012). Hybrid Algorithm for Lossless Image Compression using Simple Selective Scan order with Bit Plane Slicing. *Journal of Computer Science 8* (pp. 1338-1345). India: 2012 Science Publications.

10. Qi-bin, F. (2008). *Wavelet analysis.* Wuhan:: Wuhan University Press.

11. Rehman, M. (2015). *IMAGE COMPRESSION USING MODIFIED SET PARTITIONING IN HIERARCHICAL TREES ALGORITHM REDUCING MEMORY SPACE.* Rawalpindi: HEC.

12. Said, A. (June 1996). *IEEE Transactions on Circuits and Systems for Video Technology* , Vol. 6,.

13. Salomon, D. (2001). *Data Compression 2e.* Springer.

14. Sayood, K. (2006). *Introduction to Data Compression.* United States of America: by Elsevier Inc.

15. Shahbahrami, A. *Evaluation of Huffman and Arithmetic Algorithms for Multimedia Compression Standards.* Rasht: Department of Computer Engineering, Faculty of Engineering, University of Guilan.

16. Shapiro, J. (1993). Embedded image coding using zerotrees of wavelets coeffcients. *IEEE Trans. Signal Processing* , 3445-3462.

17. Thomas, T. M. (2006). *Elements of Information Theory 2e.* John Wiley and Sons.

18. Wavelet transform characteristics and compression coding of remote sensing images. (2004). *Optics and Precision Engineering*, (pp. 205-210).

19. Wei Li, Z. P. (2010). SPIHT algorithm combined with Huffman encoding. *Third International Symposium on Intelligent Information Technology and Security Informatics* (pp. 341-343). China: IEEE Computer Society.

20. WOODS, R. C. (2002). *Digital image processing 2e.* Beijing: Beijing Publishing House of Electronics Industry.

21. Zhu, H. (2010). An Improved SPIHT Algorithm Based on Wavelet Coefficient Blocks for Image. *International Conference on Computer Application and System Modeling* (pp. 646-649, Vol 2). china: ICCASM 2010.