Set Partitioning In Hierarchical Trees Image Compression With Local Self-Example Based Super Resolution

Nitin Are

Problem Report submitted to the

Benjamin M. Statler College of Engineering and Mineral Resources

at West Virginia University

in partial fulfillment of the requirements for the degree of

Master of Science in

Electrical Engineering

Dr. Xin Li, Ph.D., Chair

Dr. Daryl Reynolds, Ph.D.

Dr. Vinod Kulathumani, Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia

2014

Keywords: Super Resolution, Image Compression, Interpolation, Subband Coding

ABSTRACT

Set Partitioning In Hierarchical Trees Image Compression With Local Self-Example Based Super Resolution

Nitin Are

Image compression plays an important role in saving storage space and in reducing the time required to transmit image through internet or over telephone lines. In last few decades a number of algorithms were proposed to efficiently compress the image, one of which is Set Partitioning in Hierarchical Trees (SPIHT). SPIHT algorithm is capable of achieving higher PSNR and SSIM values when compared to other techniques like JPEG, Embedded Zero wavelet (EZW). This process is a Lossy Compression Technique which leads to loss of some information during compression and thus affects the image quality.

On the other hand High Resolution images have many applications in various fields like satellite imaging, military imaging, high definition television (HDTV). Super Resolution is a technique which can enhance the resolution of an image. Many sophisticated algorithms are available for Super Resolution and are used based on requirements of application. Local Self-Example based super resolution is a technique which can enhance the image resolution without external data set and with high computational speeds.

This paper combines the SPIHT compression with LSE super resolution for better performance. For this two different models were proposed, namely pre-processing and post-processing. Results from experiments conducted on two models shows that pre-processing model will give a better quality image for different compression rates.

Acknowledgements

First and foremost, I would like to thank my parents for their immense love and support.

I would like to express my deepest gratitude to my advisor, Dr. Xin Li, for his excellent guidance, and motivation. I would like to thank my committee members Dr. Daryl Reynolds and Dr. Vinod Kulathumani for their valuable time.

Table of Contents

List of Figures

List of Tables

Chapter 1

1. Introduction

1.1. Image compression

In this digital world, the data transmission and storage is very expensive. If we talk about an image, it is represented through a number of bits. Let us consider a grayscale image with 512×512 pixels and 8 bits/pixels; it requires around 2×10^6 bits to store. If this small grayscale image requires around 2MB space, then imagine about high resolution color images. Obviously, the storage of even a few images could pose a problem. On the other hand to transmit the same image over telephone lines using 9600 baud (bits/sec) modem, the transmission would take approximately 4 minutes, which is unacceptable for most applications [1] . Thus, the overall consideration is to save the transmission bandwidth for transmission of image and to save the space for storage of image. This can be achieved through several image compression techniques. Image data compression is of two types: lossy and lossless [2]. Lossless compression techniques, involve no loss of information during coding. The original image can be recovered exactly from the compressed data. Lossless compression is generally used for discrete data, such as text, and some kinds of image and video information. Arithmetic coding, Huffman coding etc. are the examples of lossless image data compression. Lossy compression techniques involve some loss of information, therefore data that have been compressed using lossy compression techniques cannot be reconstructed exactly. Lossy compression techniques allow us to save images in relatively smaller sizes, but at the cost of image quality. This paper examines the SPIHT, a lossy compression technique to compress an image, chapter 2 talks about the evolution of SPIHT compression technique.

1.2. Super Resolution

High Resolution images are required in many electronic imaging applications such as medical imaging, satellite imaging, military imaging, underwater imaging, remote sensing, and high definition television (HDTV) [2]. The main aim of Super Resolution is to obtain a high resolution image from either single or multiple low resolution images. There are a

number of Super-Resolution techniques available which are categorized as Multi-Frame Super-Resolution and Single-Frame Super-Resolution. Multi-Frame Super Resolution takes multiple Low Resolution images as input to generate a High Resolution image, whereas Single-Frame Super-Resolution considers only single Low-Resolution image to generate a High-Resolution image. This paper examines Local Self-Examples (LSE) based Single-Frame Super-Resolution technique, chapter 2 talks about the evolution of LSE based Super-Resolution technique.

1.3. Super-Resolution with Image compression

As we discussed in section 1.1, when we use lossy image compression techniques, significant amounts of information about the image are lost. One way to overcome this problem is to combine the Super-Resolution algorithm with Image Compression. *Figure 1-1* shows the process of image compression which requires K bits to represent the compressed image. To transmit the compressed image over web or telephone line requires K bits to be transmitted. If we combine the compression technique with super resolution technique we can obtain a better quality image using same number of bits (K). For this the image must be downsampled by some factor 'n' and then using the compression technique we can compress the image with more number of bits per pixel when compared to compressing the original image and then the downsampled compressed image can be upsampled by same factor 'n' using super resolution techniques. Thus without changing the cost (number of bits required to transmit image) we can obtain a better image through this process as explained in *Figure 1-2*.

Figure 1-1Block diagram of Image Compression

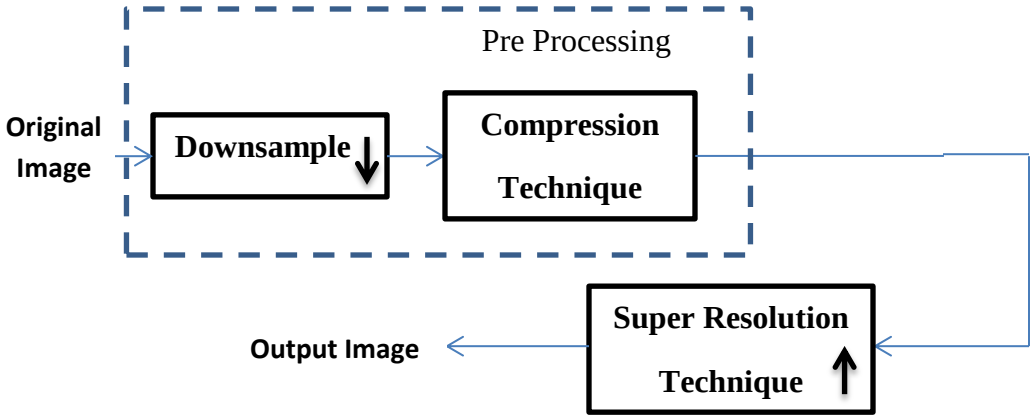

Figure 1-2 Block Diagram of Image Compression with Super Resolution (Model 1)

The other way to combine the compression technique with super resolution is through post processing, i.e. rather than downsampling the image and compressing it as we seen in model 1 (*Figure 1-2)*, the image is compressed first and then downsampled and then its resolution can be increased (upsampled) by using a super resolution technique as shown in *Figure 1-3* (Model 2).

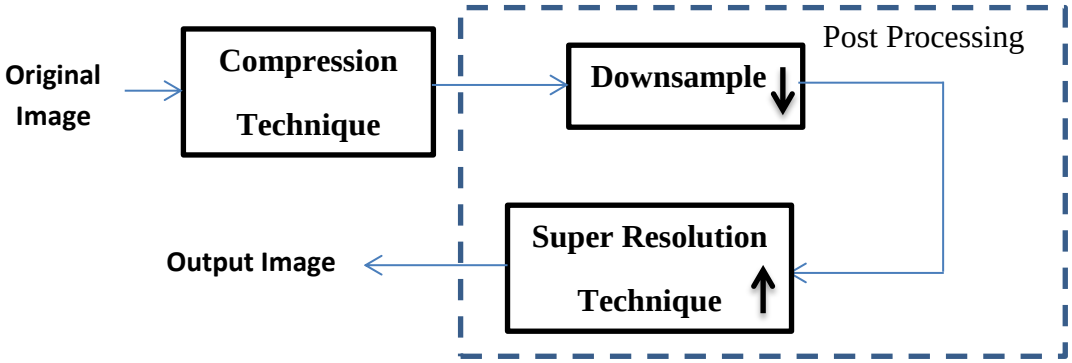

Figure 1-3 Block Diagram of Image Compression with Super Resolution (Model 2)

1.4. Performance measures

Image quality can be evaluated using performance measures such as Peak Signal to Noise Ratio (PSNR) and Structural Similarity (SSIM) [4].

1.4.1. Peak Signal to Noise Ratio (PSNR)

PSNR is one of the simplest and most widely used full reference quality metrics. Let us consider I(m,n) as input image and O(m,n) as output image obtained after applying some image processing technique. Then PSNR is calculated using below formula:

$$PSNR = 10 \times \log_{10} \frac{MAX^2}{MSE}$$

Where MAX is the maximum pixel value, if the image is represented by 8-bit then MAX value is 255 [5]. MSE is Mean Square Error and calculated as:

$$MSE = \frac{1}{M \times N} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \big(I(m,n) - O(m,n)\big)^2$$

M×N is the dimension of image.

1.4.2. Structural Similarity (SSIM)

PSNR is not very well matched to perceived visual quality. So, SSIM is used as a measure that compares local patterns of pixels intensities that have been normalized for luminance and contrast [4]. This measure gives better consistency with perceived quality. Let x and y be two image patches extracted from same spatial location from two images being compared [6], then SSIM is give as:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

Where $\mu_x \; and \; \mu_y$ are the mean of x and y respectively, σ_x^2 and σ_y^2 are the variance of x and y, and σ_{xy} is the covariance of x and y. C_1 and C_2 are small constants given as:

$$C_1 = (K_1 L)^2 \qquad C_2 = (K_2 L)^2$$

Where K_1 and K_2 are two scalar constants whose value are much smaller than 1 and L = 255 for 8 bits/pixel gray scale images [6].

1.5.Outline

Chapter 1 introduces Image compression and Super Resolution techniques and their importance. It briefly explains the evaluation metrics that can be used to measure image quality or evaluate performance of an algorithm.

Chapter 2 talks about evolution of SPIHT compression and LSE techniques, in which it explains compression and image enhancement techniques and lists its disadvantages and need for new techniques.

Chapter 3 explains the SPIHT compression and LSE base super resolution techniques.

Chapter 4 gives the experimental results with PSNR and SSIM values and also talks about how the image quality is enhanced w.r.t different images.

Chapter 5 provides conclusion for the work done.

Chapter 6 lists all the references used for this work.

Chapter 2

2. Evolution

2.1 Image Compression

2.1.1 Image compression through DCT

The main aim of image compression is to reduce the number of bits required to represent an image. This can be achieved by removing the redundancy present in it. The redundancy may be: spatial, spectral or temporal. Spatial redundancy is because of correlation between neighboring pixel values or spectral, spectral redundancy is due to correlation between color planes or spectral bands, and temporal redundancy is due to correlation between different frames in images [7] . As discussed in section 1.1, lossy compression techniques can achieve higher compression and are suitable for images where finer details in the image can be neglected for saving bandwidth or storage space. One of the basic and widely used lossy compression techniques is JPEG which uses Discrete Cosine Transform (DCT). *Figure 2-1* shows the process of image compression using DCT.

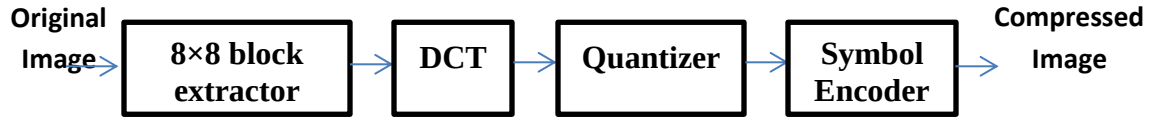

Figure 2-1 Block diagram of Image compression using DCT

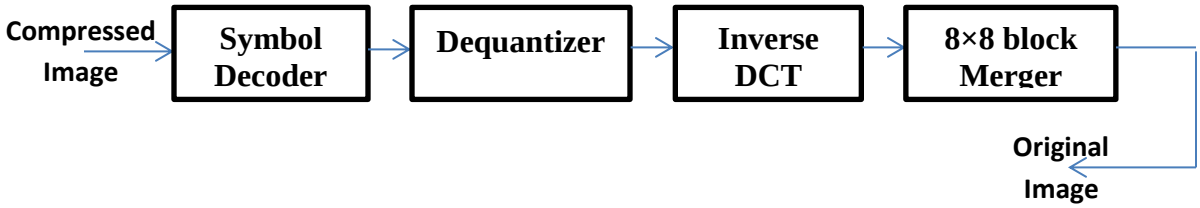

Figure 2-2 Block diagram of Image decompression using DCT

Given an image to compress, DCT follows the following procedure [7]:

1. First the original image is divided into 8×8 blocks.
2. For a black and white image the pixel values range from 0 to 255 whereas DCT is designed to work on values ranging from -128 to 127. So each 8×8 block is leveled off by subtracting 128 from each pixel value.

3. DCT is applied to each block from top-left to bottom-right.
4. Then each block is compressed through quantization.
5. Then the output quantized matrix is entropy encoded.

To reconstruct the image, reverse procedure is followed. As this is a lossy compression technique it leads to loss of some information depending on the compression ratio.

In the above process the transformer takes image and reduces the interpixel redundancies in it. First the 8×8 blocks are leveled off and then the matrix (say matrix M) is used to find the DCT. If we consider the p(x,y) as an element of the image, then the DCT is given by following equation:

$$D(i,j) = \frac{1}{\sqrt{2N}} C(i)C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} p(x,y) \cos\left[\frac{(2x+1)i\pi}{2N} \right] \cos\left[\frac{(2y+1)j\pi}{2N} \right]$$

$$C(u) = \left\{ \begin{array}{ll} \frac{1}{\sqrt{2}} & \text{if } u = 0 \\ 1 & \text{if } u > 0 \end{array} \right\}$$

Generally JPEG uses 8×8 blocks, so N equals to 8 and x and y range from 0 to 7. Using the above equation we can obtain a matrix T to compute the DCT as follows:

$$D = TMT'$$

The matrix D is the DCT of the block and consists of DCT coefficients. In DCT matrix the coefficients in top-left corner correspond to lower frequencies and the coefficients in bottom-right corner correspond to higher frequencies. Human eye is most sensitive to low frequencies than high frequencies. Now this matrix is compressed using quantization. JPEG defined several quantization matrices based on the quality level 1 to 100, where 1 gives lowest quality and 100 gives highest quality. Having the quantization matrix Q and the DCT matrix D from previous stage we can perform quantization by dividing the matrix D with corresponding element in quantization matrix Q as follows [8]:

$$C_{i,j} = round\left(\frac{D_{i,j}}{Q_{i,j}} \right)$$

In this matrix C most of the high frequency coefficients will zero due to division and these can be discarded to compress the image and can't be retrieved back. Now in the finals stage a linear sequence is extracted from the matrix C through zigzag scan an this sequence is compressed through symbol or entropy encoder using techniques like arithmetic encoder, Huffman encode and run-length encoder [8].

There are several disadvantages of using DCT such as, it only considers the spatial correlation of pixels inside the 8×8 block and neglects the correlation from the pixels of neighboring block and also produces some blocking artifacts which degrades the image quality. Beyond a certain bit rate it is not advisable to use DCT to compress an image due to blocking artifacts, blurring which degrades the image quality [9].

2.1.2 Image Compression through Wavelet Transforms

There are several disadvantages associated with former compression techniques such as one using DCT discussed in section 2.1.1. One most important drawback is that it does not consider about localization, i.e. if we take DCT where it divides the image to 8×8 blocks it cannot give some details like where it has high frequency or low frequency points inside the block as it assumes that signals are infinite in time [10]. Moreover the coefficients correspond to fixed size special area and a fixed frequency bandwidth. Edge information tends to disperse so that many non-zero coefficients are required to represent edges with good fidelity [11] . To overcome this, time-frequency analysis is used where the signal is represented both in time and frequency domain [10]. There are several ways to analyze a signal in time-frequency domain and the most basic one is Short-Time Fourier Transform [11]. The basic idea of Short-Time Fourier Transform is to determine the sinusoidal frequency and phase content of local sections of signal as it changes over time [12], but it suffers from resolution issues which lead to creation of Wavelet Transform [12].

The wavelet transform partitions the signal into set of functions, where each function is known as a wavelet. This partitioning is achieved through subband coding, where the signal is passed through a series of filters which gives the wavelets. In a simpler contest, given a signal it can be decomposed in to two wavelets using a sub band filter as shown below

Figure 2-3 Block diagram of Subband filtering

$$y_i = \frac{x_i + x_{i-1}}{2} \qquad \text{(average)}$$

$$z_i = \frac{x_i - x_{i-1}}{2} \qquad \text{(difference)}$$

$$\text{for } 1 \le i \le n \text{ and } x_0 = 0.$$

The original signal x can be obtained from y and z just by adding them. As we doubled the number of elements we can neglect the alternate elements by down sampling y and z by half. The averaging can be compared with low-pass filtering and difference can be compared with high-pass filtering. These can be further partitioned in to small wavelets by applying a series of filter banks. These filters can be implemented with many filters like Haar filter, Quadrature Mirror Filters etc. Quadrature Mirror Filters are widely used due to its advantages such as no aliasing distortion, reduced computation complexity etc.

This basic principle can be applied to Image to compute the wavelet transform. The image is divided into subbands along horizontal and vertical directions, subbands can be achieved using bank of filters (low pass and high pass) followed by 2:1 downsampling. In the first stage the image is divided in to four subbands using the filter bank as show in *Figure 2-4*. LL1, HL1, LH1, HH1 are the subbands obtained after firsts stage of subdivision.

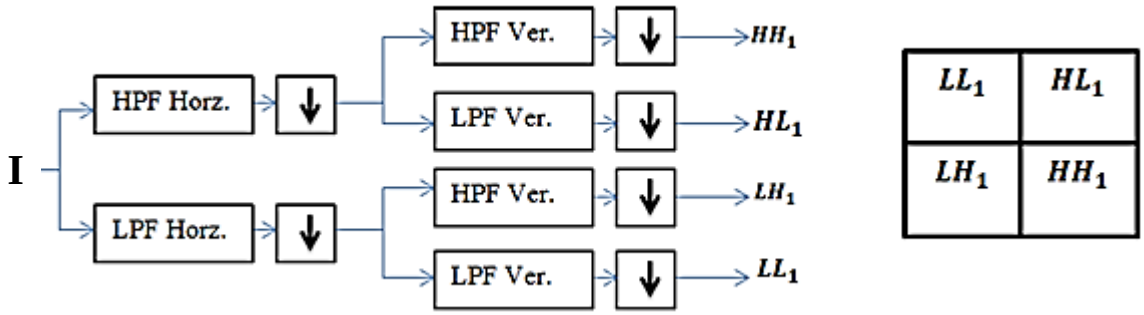

Figure 2-4 Analysis Filter Bank for Subband Coding for an image I

LL represents low pass filtering applied on horizontal and vertical directions respectively. One stage of subdivision will achieve a compression of factor 2 and the image can be further compressed by dividing the subbands further, as the low frequency coefficients have more priority in terms of human eye perception it is preferred to divide the LL band. Further division of LL band requires applying another level of filter bank to LL1 subbands which divides it to four subbands. *Figure 2-5* shows the filter banks for 2 level recursive lowpass filter analysis.

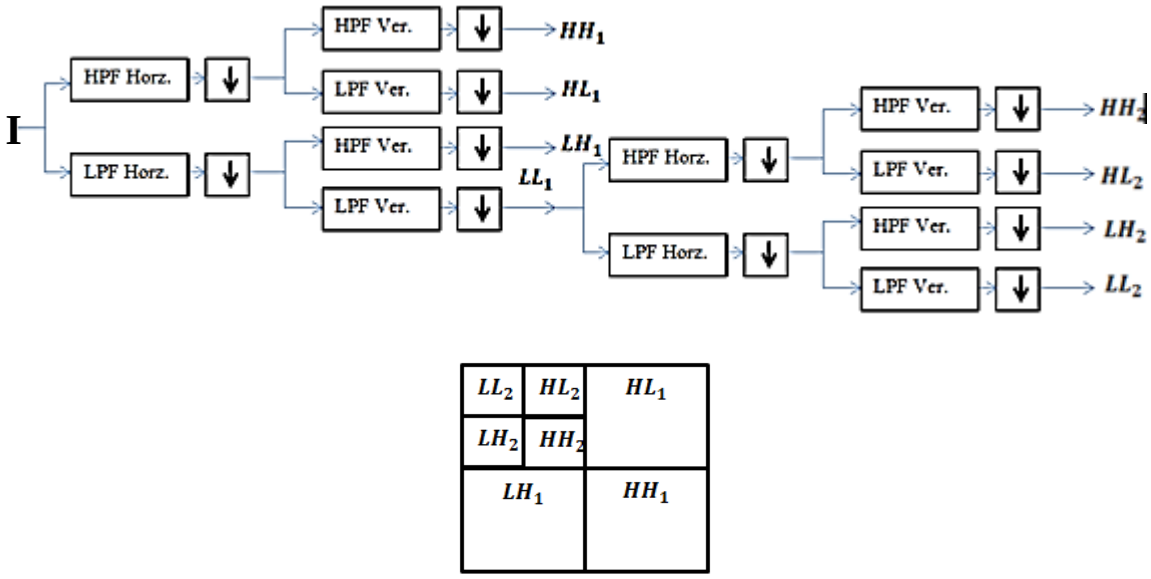

Figure 2-5 Two Level Analysis Filter Bank for Subband Coding for an image I

This process of subdividing the LL subbands further can be repeated until some final scale is reached and for each coarser scale, the wavelet coefficients represent a larger spatial area of the image but a narrowed band of frequencies. After first level of subdivision (LL1) the coefficients represent 2×2 area of original image with low frequency bandwidth $0<|w|<\pi/2$ and high frequency bandwidth $\pi/2<|w|<\pi$, after second level of subdivision (LL2) the coefficients represent 4×4 area of original image with low frequency bandwidth $0<|w|<\pi/4$ and high frequency bandwidth $\pi/4<|w|<\pi$ [11].

This overall process gives the transformation of image to wavelet coefficients from which the image can be reconstructed back almost without any loss [11]. In the image compression the transformation stage is followed by quantization where actual loss in information occurs to compress the image and is followed by an encoder. The wavelet coefficients in the subbands are two-dimensional and of we want to compress the image then we have to code the coefficient values and their position in space. A coefficient in a low subband (top left band) has four descendants in the lower level subband (which defines the relationship between subbands) as shown in *Figure 2-6(a).* This forms a tree like structure between the coefficients with a root node and descendants (*Figure 2-6 (b)).*

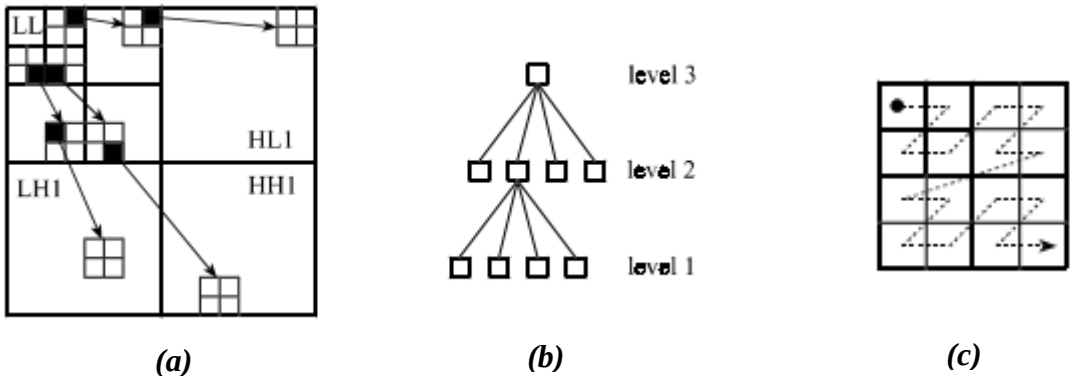

Figure 2-6 (a) Relationship between wavelet coefficients in different sub bands (b) Hierarchical tree structure with nodes representing pixels (c) Scanning Order [22]

As proposed by Shapiro [11], these coefficients are made significant or insignificant using a threshold and two pass algorithm. Initially a threshold is chosen, usually half of maximum coefficient's value and all the coefficients are scanned using the order shown in *Figure 2-6(c)* and coefficients below the threshold are said to be insignificant and above the threshold are significant with corresponding negative or positive sign. If a node (significant) in the tree has all of its descendants insignificant than it is said to be a zerotree root and this can be discarded without encoding, and if node is insignificant and some of its descendants are significant then it is said to be Isolated zero node. This process of encoding a coefficient is explained in *Figure 2-7.* The threshold value is halved after each stage which consists of two passes, dominant pass and subordinate pass. Dominant pass picks up the coefficients which are yet to be insignificant but becomes significant in this pass and Subordinate pass refines the coefficients which are already found to be significant by process of successive approximation [11].

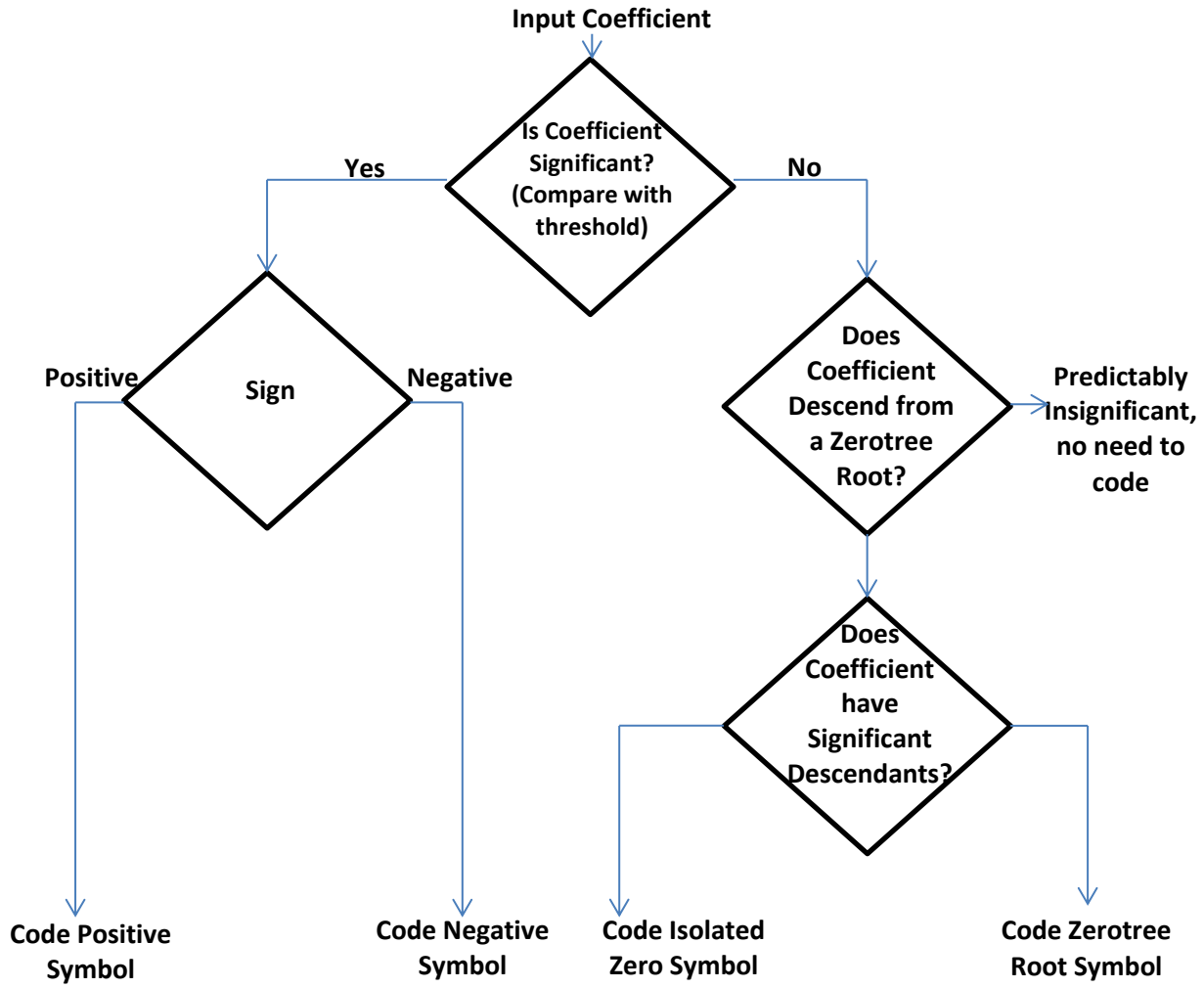

Figure 2-7 Flow Chart for encoding coefficients [11]

In this way the Dominant pass will encode coefficients as Positive, Negative, Isolated Zero or Zerotree root. The coefficients which are positive and negative (significant) are passed to subordinate pass by successive approximation it outputs the next most significant bit of all the coefficients on the subordinate list. So in a way or other, largest coefficients are transmitted first. In short this Embedded Zero Tree technique is based on three concepts: partially ordering image elements by magnitude, ordered bit plane transmission of refinement bits and exploration of the self-similarity of image wavelet transform across different scales [14]. This type of embedded coding has a lot of advantages especially in web applications, if a user have less bandwidth then it will send the most significant coefficients in some dominant and subordinate passes and if the user have more bandwidth it can perform more passes thus improving the received image quality. This also allows terminating the encoding at any point thereby allowing a target rate to

meet exactly [11]. Next to this arithmetic coding can be used to reduce the mean square error thus reducing the number of bits required further.

Even this technique suffers from limitations such as; it only assumes that LL subbands are being sub partitioned and not useful if we wish to split any subband other than LL. It exploits the redundancies between different subband scales but not within in the subband, i.e. neighborhood similarities.

2.2 Image Super Resolution

2.2.1 Interpolation

There are many applications for the techniques which could improve the image resolution as discussed in section 1.2. All these techniques try to reduce the blur or any type of noise which affect the image clarity, to get a better image [15]. Image interpolation is one of the techniques which are used to enhance resolution. Interpolation is a method of constructing new data points within the range of a discrete set of known points [16]. Image Interpolation is widely used for image enhancement, image zooming, image reduction etc. [15]. The image interpolation is the process which transforms the image from one resolution to another using known data to estimate values at unknown locations [17]. Interpolation techniques are generally divided in to two categories: Adaptive and Non-Adaptive.

The non-adaptive interpolation techniques are based on direct manipulation of pixels and they do not consider about any characteristics of the image. These techniques interpolate all pixels by fixed pattern and thus are easy to perform. Whereas the adaptive techniques consider several factors such as intensity value, edge information, texture etc. to interpolate and are capable of producing better quality images than non-adaptive algorithms but involved in more computational complexity. The general Non adaptive interpolation techniques are nearest neighbor interpolation, bilinear interpolation and bicubic interpolation which uses their own kernel to predict the missing pixel values. The general form of interpolation is given as:

$$g(x) = \Sigma_k \, c_k \, u(\text{distance}_k)$$

Where g is the interpolation function, u is the interpolation kernel and dk is the distance from the point under consideration x to grid point xk. Ck are the interpolation coefficients and are choosen such that g(xk)=f(xk) for all values of xk [18].

2.2.1.1. Nearest neighbor interpolation

This is the basic interpolation technique which requires very less computational time when compared to others [15]. Whenever the image is remapped from one pixel grid to another grid (greater size) it first places all the available pixel data and the missing pixels are filled by using the nearest available pixel value. It is nothing but copying the existing nearest values. It gives good result when the image has high resolution pixels [15]. But it can degrade the image quality to greater extent if there are number of unknown pixels to be mapped (if we wish to increase the size by greater factor). If assume a two dimensional image the nearest neighbor interpolation kernel is applied on each direction and is given as [15]:

$$u(s) = \begin{cases} 0 & |s| > 0.5 \\ 1 & |s| < 0.5 \end{cases}$$

Where s is the distance between the interpolated point and grid point.

2.2.1.2 Bilinear Interpolation

Bilinear interpolation considers the weighted average of four neighborhood pixel as the interpolated pixel value. This produces a smoother image and achieves better performance than nearest neighbor interpolation. It actually performs linear operation in one direction (horizontal) and again in other direction (vertical). The interpolation kernel is given as [15]:

$$u(s) = \begin{cases} 0 & |s| > 1 \\ 1 - |s| & |s| < 1 \end{cases}$$

2.2.1.3 Bicubic Interpolation

Bicubic interpolation is considered to be the pest method of all available non-adaptive methods [15]. This method determines the interpolated value by averaging the 16 closest pixel values. The closer pixels have more weight and this produces a sharper image when compared to bilinear interpolated image but at cost of time. The transformation kernel is given as [15]:

$$u(s) = \begin{cases} 3/2|s|^3 - 5/2|s|^2 + 1 & 0 <= |s| < 1 \\ -1/2|s|^3 + 5/2|s|^2 - 4|s| + 2 & 1 <= |s| < 2 \\ 0 & 2 < |s| \end{cases}$$

All these non-adaptive interpolation techniques suffer from some problems like blurring edges, artifacts around edges etc. Adaptive interpolation techniques such as new edge-directed interpolation, data dependent triangulation, iterative curvature-based interpolation etc. can be used some of the problems faced by non-adaptive techniques.

2.2.2 Super Resolution

The basic interpolation techniques involves only in upsampling the low-resolution image and also leads to some blurring, degradation and noise in the image as discussed in section 2.2.1. The super resolution techniques address these problems by combining three major processes upsampling, deblurring, and denoising. As discussed in section 1.2 the super resolution techniques are generally classified as multi frame/image super resolution and single frame/image super resolution. The multi frame super resolution techniques use a set of low resolution images from the same scene to generate a high resolution images. Such methods have several limitations such as it increases resolution only by small factor and every time it is not possible to obtain a sequence of images from same scene. Whereas single frame super resolution techniques which are also known as example-based resolution techniques produces high resolution image by learning correlation between low and high resolution patches from a database of low and high resolution image pairs [19]. Several techniques evolved during recent time extending the example based super resolution which do not relay on external database to find missing high resolution information, rather they rely on patches from localized regions in the input image. Such technique is known as Local Self–Examples based super resolution and is explained in section 3.2

Chapter 3

3. SPIHT compression & LSE Super Resolution

3.1 SPIHT image compression

The EZW coding discussed in section 2.1.2 explains the sub band theory and embedded coding which is an effective technique for image compression. Based on the same principle Amir Said and William A. Pearlman proposed a new algorithm known as Set Partitioning in Hierarchical Trees (SPIHT) which improves the overall performance and the results even with binary uncoded transmission are better than the EZW with arithmetic coding [14]. To improve the performance further, arithmetic coding can be performed for SPIHT. Moreover the EZW coding did not produce a single embedded file, and required for each rate, the optimization of a certain parameter [14]. The new algorithm SPIHT answers these by completely changing the transmission priority which provides a single embedded file. The overall algorithm is explained in next sub sections.

3.1.1 Progressive Image Transmission

This section explains how to prioritize the wavelet transform coefficients for a given image. Let us consider a two dimensional image with (i,j) as pixel coordinates with p as pixel array and c as coefficient array, then transformation is given as:

$$c = \Omega(p)$$

Where Ω represents unitary hierarchical subband transformation. For reconstruction the decoder initially sets the reconstruction vector $\hat{c}$ to zero and updates its array components according to the coded message [14]. From this reconstruction vector the pixel values can be reconstructed as:

$$\hat{p} = \Omega^{-1}(\hat{c})$$

Now to find which coefficients have more importance or more weight in terms of image quality let us consider the mean square error between the reconstructed pixel and the original pixel array with N elements:

$$D_{\text{msc}}(\mathbf{p}-\hat{\mathbf{p}}) = \frac{||\mathbf{p}-\hat{\mathbf{p}}||^2}{N} = \frac{1}{N}\sum_i\sum_j (p_{i,j}-\hat{p}_{i,j})^2 \quad = D_{\text{msc}}(\mathbf{c}-\hat{\mathbf{c}}) = \frac{1}{N}\sum_i\sum_j (c_{i,j}-\hat{c}_{i,j})^2.$$

If we reconstruct the exact pixel, i.e. if the exact coefficient value is transmitted then the mean square error decreases by$|c_{i,j}|^2/N$. Thus the coefficients with larger magnitude must be given priority as they have larger content of information [14] and must be transmitted first. As the higher magnitude coefficients have more priority, the coefficients are ordered in descending order of their magnitude as show in *Figure 3-1*.

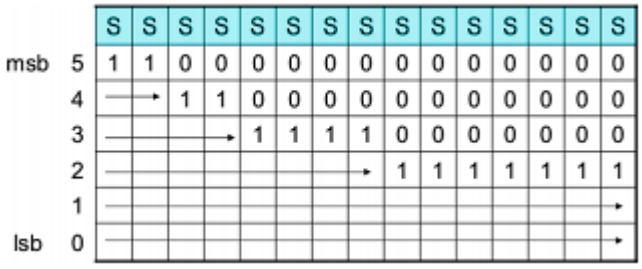

Figure 3-1 Binary Representation of magnitude-ordered Coefficients [20]

Each column in *Figure 3-1* corresponds to coefficient bits. The bits in the topmost row indicate the sign of the coefficient and remaining rows represents the bit plane [20], bottom row represent least significant bits and the top row represent most significant bits. Let us assume set of numbers μ_n , which represent the number of coefficient such that $2^n \leq |c_{i,j}| \leq 2^{n+1}$ [14]. So μ_n represent number of bits which are "1" in corresponding nth plane, e.g. in *Figure 3-1* μ_5 is 2 as 5th row has two bits which are "1"similarly μ_4 = 2 and μ_5 = 4. The progressive transmission can be implemented using the following algorithm 1 [14]:

Step 1: Initialization: Find the initial value of n and output its value to decoder, its value can be calculated as:

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

Step 2: Sorting Pass: Output μ_n along with pixel coordinates and sign of each of the μ_n coefficients.

Step 3: Refinement Pass: Output the nth most significant bit of all the coefficients whose coordinates are transmitted in previous sorting pass.

Step 4: Decrement n by 1 and go to Step-2 (repeating the process for lower bit planes)

As the coefficients are in descending order of their magnitude the first "1" bit and all the "0" bits (in *Figure 3-1*) at the end need not be transmitted. Depending on the required rate the above algorithm is stopped and as the coordinates which contains more information are being transmitted first, a good image can be obtained.

3.1.2 Set Partitioning Sorting Algorithm

In the algorithm 1 discussed in section 3.1.1, μ_n corresponds to coefficients such that $2^n \leq |c_{i,j}| \leq 2^{n+1}$, and if $|c_{i,j}| \geq 2^n$ then that coefficient is said to be significant. For sorting the coefficients the Set Partitioning Sorting Algorithm divides the entire set of coefficients in to subsets τ_m .

$$\max_{(i,j)\in T_m} \{|c_{i,j}|\} \geq 2^n \ ?$$

Now if the above condition is satisfied then the subset is said to be significant and it is further partitioned into new subsets and again the same rule is used to test the new subsets for significance. This sub partitioning of significant subsets is carried out until each significant coefficient is identified. The main aim of Set Partitioning Sorting Algorithm is to reduce the number of comparisons rather than sorting all the coefficients (max number of comparisons), this is achieved by partitioning such that subsets which are insignificant contain large number of elements and subsets which are significant contain only one element.

3.1.3 Set Partitioning Sorting Algorithm through Spatial Orientation Trees

Section 2.1.2 shows the procedure for transforming the wavelet transform coordinates in to subbands and then using the relationship between the coefficients in different subbands through hierarchical tree structure. The EZW algorithm uses the thresholds to distinguish between significant and insignificant coefficients. Using the same Hierarchical tree structure (or Spatial Orientation Trees) Set Partitioning Sorting Algorithm is used to sort the Coefficients in descending order of their magnitude reducing the number of comparisons. The partitioning

subsets are obtained from parts of spatial orientation trees. In order to achieve this, nodes in the tree are categorized as O set, D set and H set. For a particular node, all its direct descendants are grouped as Offspring set, these are the pixels of the same spatial orientation in the next finer level of the pyramid [14] and each node in the tree will have either four offspring or no offspring. The D set contains all the descendants of a particular node and H set contain all spatial orientation tree roots. The L set contains all descendants except their offspring. The classification of these sets is shown in *Figure 3-2*.

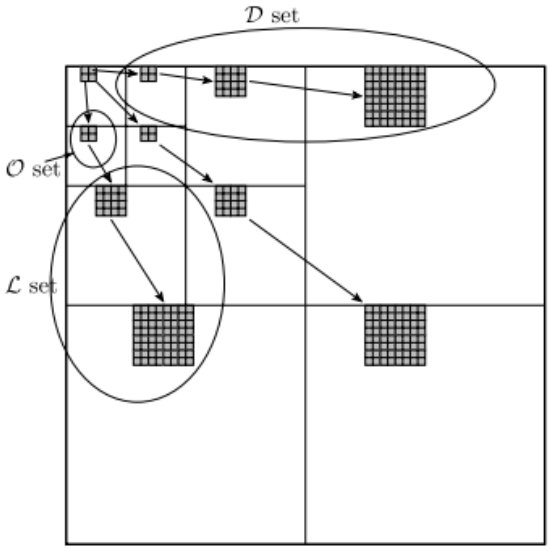

Figure 3-2 Different Set Types in a tree-block. Full Descendant D set, and Offspring O set, and a grand-descendant L set [23].

Using these sets the Set Partitioning Sorting Algorithm through Spatial Orientation Trees is achieved using following algorithm 2 [14]:

Step 1: Initial Partition is formed with sets {(i,j)} and D(I,j), for all (i,j) € H

Step 2: If D(i,j) is significant (significance rule is explained in section 3.1.2) then it is partitioned into L(i,j) along with four single element sets with (k,l) € O(i,j).

Step 3: If L(i,j) is significant then it is partitioned into four sets D(k,l), with (k,l) € O(i,j).

3.1.4 Coding Algorithm

The algorithm 1 in section 3.1.1 describes how to prioritize the coefficients and algorithm 2 in section 3.1.3 describes how to partition using sets in spatial orientation. In practical implementation, the significance information is stored in three ordered lists, List of Insignificant Sets (LIS), List of Insignificant Pixels (LIP), and List of Significant Pixels (LSP) where LSP and LIP represent individual pixels and LIS represent either the set D(i,j) or L(i,j) [14]. If LIS represent the D set then it of type A and if it represent the L set then it is of type B. Initially all the sets are placed in LIS and these are sequentially evaluated and if it is a significant set it is removed from the list and partitioned. If the new subsets contain more than one element it will be added back to LIS list while single-coordinate sets if they are significant they are added to the end of LIP if insignificant they are added to the end of LSP [14]. The LSP contains the coordinates of the pixels that are visited in the refinement pass [14]. Combining the algorithm 1 in section 3.1.1 and algorithm 2 in section 3.1.3 the detailed algorithm for SPIHT encoding is given below [14]:

Step 1: Initialization: Calculate the initial value of n and output its value

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

- LSP is initialized with an empty set
- All the coordinates (i,j) € H are added to the LIP
- The Descendants are added to LIS as Type A entries.

This step is executed only for the first time; this is followed by sorting and refinement pass as explained in algorithm 1.

Step 2: Sorting Pass:

2.1. For each entry (i,j) in the LIP do: (check for significant pixels in LIP)

2.1.1 Output $S_n(i,j)$

2.1.2 If $S_n(i,j)$ = 1 (If (i,j) is significant) then move (i,j) to the LSP and output the sign of $C_{i,j}$

2.2. For each entry (i,j) in the LIS do: (check for significant sets in LIS)

2.2.1 If the entry is of type A then

- Output $S_n(D(i,j))$
- If $S_n(D(i,j))$ = 1 then
  - For each (k,l) € O(i,j) do: (Check for significant pixels in O set)
    - Output $S_n(k,l)$
    - If $S_n(k,l)$=1 (if significant) then add (k,l) to the LSP and output sign of $C_{k.l}$
    - If $S_n(k,l)$=0 (if insignificant) then add (k,l) to the end of LIP
  - If L(i,j) ≠ ø (if (i,j) is node which have descendants with more than one level, as its already known that it is significant this can be further partitioned), move (i,j) to the end of the LIS, as an entry of type B, and go to step 2.2.2; else, remove entry (i,j) from the LIS

2.2.2 If the entry is of type B then

- Output $S_n(L(i,j))$
- If $S_n(L(i,j))$ =1 then
  - Add each (k,l) € O(i,j) to the end of the LIS as an entry of type A
  - Remove (i,j) from the LIS.

Step 3: Refinement Pass: For each (i,j) in LSP, except those included in last sorting pass, output the nth most significant bit of $|C_{i,j}|$.

Step 4: Quantization Step Update: Decrement n by 1 and go to step 2.

From the algorithm the transmitted information is formed of single bits [14]. The decoder exactly performs the reverse to reconstruct the image. For each value of n we already know that $2^n \leq$

$|c_{i,j}| \leq 2^{n+1}$ (explained in section 3.1.1) along with this information decode also gets the sign and estimates the coefficient as $\hat{c}_{i,j} = \pm 1.5 \times 2^n$ (where $\hat{c}_{i,j}$ is the reconstructed coefficient of $c_{i,j}$). Similarly, during the refinement pass the decoder adds or subtracts 2^{n-1} to $\hat{c}_{i,j}$ when it inputs the bits of binary representation of $|c_{i,j}|$. In this way the distortion gradually decreases during both the sorting and refinement pass [14].

The entropy-coding can be used for encoder output during refinement pass to improve the performance, but it is observed that this achieved only a little gain. From the subband coding discussed in section 2.1.2 there is a statistical dependence between the pixels in several subbands in next levels, i.e. dependence between $S_n(i,j)$ and $S_n(D(i,j))$ and also between the adjacent pixels [14]. This dependency can be considered and can be used to improve the gain further; this is achieved using adaptive arithmetic coding proposed by I.H. Witten [21]. Using the arithmetic coding the information is coded in group of four pixels, for which 2×2 coordinates were kept together in the lists (D, O, L lists) and their significance values were coded as single symbol [21]. As the decoder only needs to know the transition from insignificant to significant, the amount of information that needs to be coded changes according to the number of insignificant pixels (m) in the group [14]. Several adaptive models can be used for arithmetic coding [21], each with 2^m symbols, to code the information in a group of four pixels. Using different models for the different number of insignificant pixels, each adaptive model contains probabilities conditioned to the fact that a certain number of adjacent pixels are significant or insignificant [14] which exploits the dependency between magnitudes of adjacent pixels.

3.2 LSE based Super Resolution

Section 2.2 explains the need for super resolution techniques and their classification. The LSE base Super Resolution is a Single Frame Super Resolution technique which uses only one image (test image under consideration) to upsample which assumes “local self-similarity” property in natural images. A self-similar object is exactly or approximately similar to a part of itself, a simple example would be ‘Koch curve’ which has an infinitely repeating self-similarity when it is magnified [22].

Almost all the natural images contain several singularities, such as edges, high frequency textured regions and for such images if we apply normal upscaling techniques like interpolation

which predict the missing pixels in upscaled plane, it may lead to some artifacts such as ringing and stair casing. In order to overcome such effects Super Resolution techniques incorporate deblurring and denoising to provide much better quality. For a give low resolution image the LSE method extracts patches from extremely localized regions which reduce nearest-patch search time [23].i.e. this method uses patches within the image to predict the missing upper frequency band of upsampled image whereas some other algorithms use set of external patches for this prediction. This method performs well for small scaling factors using non-dyadic filter banks which are nearly biorthogonal [23].

3.2.1 Upscaling Scheme through Local Self Similarity

The Basic principle behind LSE is "there exists a similarity across the image and at multiple scales" this can be extended to natural images to observe various singular features in natural images that are similar to themselves under small scaling factors [23].

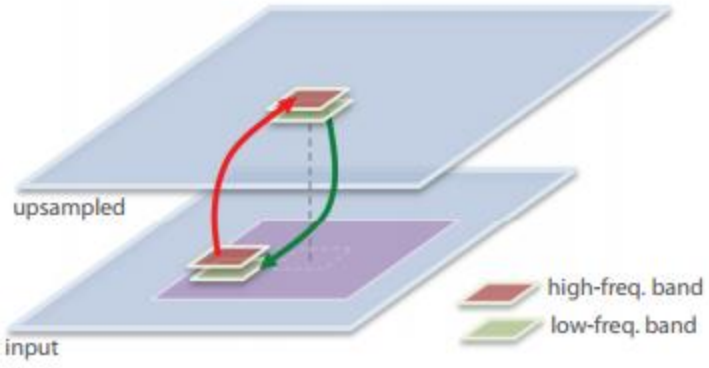

Figure 3-3 Upscaling scheme using Local Self Similarity Property

The upscaling mechanism is explained below:

- Let I_0 be the given image defined on a coarse grid of pixels g_0.
- The image I_0 is interpolated to a finer grid g_1 using linear interpolation operator U. g_l are rectangular pixels grids with resolution that increases by scaling factor and the operator U maps images from g_l to g_{l+1}.
- Now this initial upsampled image L_1 lacks high frequency information. From the basic principle of LSE, this missing high frequency band is predicted by exploiting the patches.

- For the high frequency band prediction the example patches are generated from the smoothed version of input image, say L_0 = U(D(I_0)) where D is the downsampling operator and U is the upsampling operator through which smoother version can be obtained.
- Every patch P (P€g_1 i.e. patches from upsampled grid) in the upsampled image is matched with the most similar patch q(P) € g_0 in the smoothed input L_0, i.e. a patch of low frequency band from the upsampled image is matched with its nearest patch with in a small window in low passed input image as shown in *Figure 3-3* .
- This searching for most similar patch is not performed against every patch in L_0, rather it searches for the patches with in a small rectangular window (purple shaded window in *Figure 3-3)* which is centered at the same relative coordinates (between g_0 and g_1).
- After finding the most similar patch, the complement high frequency content $H_0(q)$ = $I_0(q) - L_0(q)$ is used to predict the missing high frequency band as : $I_1(p)$ = $L_1(P) +$ $H_0(q(p))$. Different accounts for the same pixel, due to overlaps between nearby patches are averaged together [23].

In this way Local self-similarity between patches can be used to predict the higher frequency band and thus can be used to improve its resolution. But LSE method provides better results for small scaling factors, because when large scaling factors are used example patches do not contain any pure isolated singularities that obey scale invariance assumption [23]. The desired large magnification can be obtained by repeating the LSE method with small scaling (usually 1.25 is used, which does not have any errors) factors several times.

Chapter 4

4. Experimental Results

Using the model proposed in section 1.3 (*Figure 1-2)*, several experiments were conducted on different images. A set of ten grayscale images were used, most of which are standard images used in image processing. The main aim is to achieve better image quality by maintaining the maximum number of bits required to represent the image same. All the images are 512×512 grayscale images in raw format. To evaluate the performance, PSNR and SSIM metrics are used. PSNR may not be always a good metric to evaluate the performance and thus SSIM is also used which evaluates the performance based on human eye perception. For Model 1(refer *Figure 1-2)*, the entire process is divided into two stages, in the first stage the image is compressed using SPIHT with 0.1bpp, as the image size is 512×512 the compressed image requires 26214 bits for 0.1bpp compression. The PSNR and SSIM was calculated which tells how good the compressed image is. Now in the second stage the LSE super resolution technique is used along with SPIHT compression to obtain a better image by using the same 26214 bits. First the image is downsampled by a factor 0.8 and the downsampled image is compressed using SPIHT with 0.15625bpp. As the image is downsampled, its new size is 410×410 and when it is compressed with 0.15625bpp it also requires 26214 bits for transmission or to store, as there are more number of bits per pixel when compared to direct compression (without downsampling) it is possible to store more information for each pixel. The compressed downsampled image is now enhanced using LSE super resolution technique; PSNR1a and SSIM1a are evaluated for the output image. For the same model 1, the entire process is repeated with initial compression of 0.4bpp (compressing the image without downsampling using SPIHT with 0.4bpp) and the PSNR1b, SSIM 1b are calculated.

Now for Model 2 (refer *Figure 1-3*) which compresses the image first with given bpp and then uses post processing, i.e. downsamples the compressed image and then upsamples it using super resolution technique. Two experiments are conducted with model with different images, one with 0.1bpp SPIHT compression and another with 0.4bpp SPIHT compression.

The main aim of model 1 and model 2 is to increase the resolution by combining with SPIHT compression and LSE super resolution, but follows a different approach. Section 4.1 gives the results for different experiments conducted on model 1 and model 2 and then answers which model is best.

Figure 4-1 shows the set of images used to perform the experiment using MATLAB 7.9.0 (R2009b) on system with below configuration:

Operating System: Windows 7

Processor: Intel Core 2 Duo, 2.93 GHz

Installed Memory (RAM): 4.00 GB

System Type: 64-bit Operating System

(a) Barbara

(b) House

(c) Butterfly

(d) Lena

(e) Scene

(f) Bridge

(g) Boat

(h) Signal

(i)Goldhill

(j) University

Figure 4-1 Set of Images used to conduct Experiments

4.1 Results

Several experiments were conducted on model 1 and model 2 with different image and different compression rates, PSNR and SSIM are calculated for each experiment and are labeled as follows:

Experiment 1: PSNR, SSIM: Measures for compressing the image using SPIHT with 0.1bpp.

Experiment 2: PSNR 1a, SSIM 1a: Measures for Mode1, compressing with 0.1bpp (between input image and output image, input image is downsampled and compressed with 0.1bpp and then upsampled using LSE super resolution which gives output image)

Experiment 3: PSNR 1b, SSIM 1b: Measures for Model 1, compressing with 0.4bpp (between input image and output image, input image is downsampled and compressed with 0.4bpp and then upsampled using LSE super resolution which gives output image)

Experiment 4: PSNR 2a, SSIM 2a: Measures for Mode2, compressing with 0.1bpp (between input image and output image, input image is compressed with 0.1bpp and then downsampled which is upsampled back using LSE super resolution which gives output image)

Experiment 5: PSNR 2b, SSIM 2b: Measures for Model 2, compressing with 0.4bpp (between input image and output image, input image is compressed with 0.4bpp and then downsampled which is upsampled back using LSE super resolution which gives output image)

Table 1 lists the PSNR, SSIM values for SPIHT compression (0.1bpp) without LSE Super Resolution. *Table 2* and *Table 3* compares model 1 and model 2 with 0.1bpp and 0.4bpp compression rates respectively. From *Table 1 and 2* it clear that Model 1 gives better results than Model 2, in terms of image quality. In *Table 2* if we compare the PSNR values, Model 2 provides better PSNR, but PSNR is not always a good metric to decide the image quality or the performance of an algorithm, in this case even the PSNR for Model 2 is more there is an improvement in terms of visual appearance which is reflected in SSIM values. In *Table 2* which is with 0.4bpp the Model 1 performance is improved when compared with Model 2, and if we increase bpp further it is observed that Model 1 has high PSNR and SSIM when compared to Model 2. But if we increase bpp, it also increases the time for execution which is more in Model-

1. Further it is observed that, at higher bpp the actual PSNR and SSIM (without LSE) are better than PSNR and SSIM with LSE super resolution. So, the joint SPIHT compression and LSE super resolution model gives better results for low bpp when compared to high bpp.

Image	**PSNR**	**SSIM**
Barbara	24.301	0.77829
House	25.434	0.81333
Butterfly	27.219	0.85960
Lena	31.071	0.90763
Scene	25.757	0.78826
Bridge	32.274	0.89717
Boat	26.297	0.79763
Signal	24.606	0.75344
Goldhill	27.626	0.80724
University	27.581	0.77823

Table 4-1 Experimental results with 0.1bpp (SPIHT compressing original image, refer Figure 1-1 and Experiment 1 is section 4.1)

Image	Time for model 1 (sec)	Time for model 2 (sec)	PSNR1a Model 1 with 0.1bpp	PSNR2a Model 2 with 0.1bpp	SSIM1a Model 1 with 0.1bpp	SSIM2a Model 2 with 0.1bpp
Barbara	52.13	52.24	24.403	24.172	0.79486	0.7784
House	52.70	54.67	25.774	25.516	0.82453	0.81397
Butterfly	50.17	54.58	27.199	27.234	0.86415	0.86011
Lena	52.73	54.32	31.085	31.151	0.90398	0.90807
Scene	53.84	55.75	25.845	25.762	0.79632	0.78890
Bridge	52.23	54.50	32.372	32.411	0.90453	0.89827
Boat	57.12	52.31	26.279	26.277	0.80749	0.79796
Signal	52.80	52.86	24.592	24.535	0.76657	0.75389
Goldhill	53.08	52.00	27.562	27.639	0.81466	0.80775
University	52.80	58.87	27.531	27.607	0.78570	0.77914

Table 4-2 Comparison of experimental results for Model 1 and Model 2 with 0.1bpp compression rate; refer experiments 2 and 4 in section 4.1.

Image	Time for model 1 (sec)	Time for model 2 (sec)	PSNR1b Model 1 with 0.4bpp	PSNR2b Model 2 with 0.4bpp	SSIM1b Model 1 with 0.4bpp	SSIM2b Model 2 with 0.4bpp
Barbara	57.85	54.03	28.503	27.986	0.93856	0.91796
House	58.42	55.22	31.157	30.967	0.95053	0.94362
Butterfly	60.10	57.55	31.783	31.531	0.96578	0.95903
Lena	59.88	55.72	36.868	36.776	0.97612	0.97380
Scene	60.13	102.40	30.751	30.439	0.94113	0.92938
Bridge	59.70	54.06	36.792	36.495	0.96914	0.96281
Boat	60.19	54.3	31.172	30.990	0.94766	0.94119
Signal	60.90	56.75	29.318	29.218	0.9440	0.93276
Goldhill	69.39	54.28	31.682	31.321	0.94214	0.93586
University	66.20	54.29	31.802	31.620	0.93252	0.92657

Table 4-3 Comparison of experimental results for Model 1 and Model 2 with 0.4bpp compression rate; refer experiments 3 and 5 in section 4.

4.2 Observations from Model 1

(a) Original Image

(b) Compressed with 0.1bpp

(c) Downsampled Image (0.8)

(d) Compressed Downsampled Image (0.156bpp)

(e) Super Resolution Output

Figure 4-2 Complete set of images using Barbara image with initial bpp 0.1 (Compressing original image)

Figure 4-2 shows the detail process of the model and the images to be compared are compressed image with 0.1bpp and Super Resolution output image. If we look closer there are many improvements in terms of visual perception. The nose looks better in super resolution output image and some other areas like left eye, hand, stripes on scarf, and chair in the background are also improved. As the image is compressed with low bpp some information is completely lost and we can observe some blurred patches in both images.

(a) Original Image *(b) Compressed with 0.2bpp* *(c) Downsampled Image (0.8)*

(d) Compressed Downsampled Image (0.3125bpp) *(e) Super Resolution Output*

Figure 4-3 Complete set of images using Barbara image with initial bpp 0.2 (Compressing original image)

Figure 4-3 shows the obtained images for the same model but with 0.2bpp initial compression (before downsampling the image). The patches that are observed during 0.1bpp compression are no more and can increase bpp further to avoid such patches. But as the initial compression rate increases then we are using more number of bits per pixel which increases the overall number of bits required to store or to transmit the image and applying the LSE super resolution technique did not produce good results when compared to 0.1bpp for the Barbara image, but for some images it produces good results if we increase bpp where there is more self-similarity in neighborhood patches, this is observed in Barbara, House, Scene images.

Figure 4-4 Lena image compressed with 0.1 bpp (left) and super resolution output (right)

Figure 4-5 Signal image compressed with 0.2 bpp (left) and super resolution output (right)

Figure 4-6 Boat image compressed with 0.1 bpp (left) and super resolution output (right)

Figure 4-7 University image compressed with 0.2 bpp (left) and super resolution output (right)

Figure 4-8 Bridge image compressed with 0.1 bpp (left) and super resolution output (right)

Figure 4-9 House image compressed with 0.1 bpp (left) and super resolution output (right)

In *Figure 4-4* the Lena image to the right have some improvements in some regions like the left and the right eye, the shoulder edge, and the left part of the lip which is almost blurred in the compressed image to the left. In *Figure 4-5* signal image compression is with 0.2bpp (compressing the original image) and the super resolution output to the right avoids the blurring of edges in the cars and the right bottom corner pole is clearly visible. In *Figure 4-6* the boat image is compressed with 0.1bpp and improves the image to some extent, in areas around the poles and the edges. Similarly improvements in several areas are observed in other images like: roof in university image, rope in bridge image, bushes and roof edge in house image.

From the above experiments on different images it is clear that it is possible to improve the image quality if we join the SPIHT compression with LSE super resolution by keeping the max number of bits to represent the image same. So at same cost we can achieve a better image which can be used in applications like HDTV, satellite imaging, internet etc. But, if the image has fewer examples patches of greater relevance it may worsen the image in some areas but can improve in some other areas as observed in Lena image.

Chapter 5

5. Conclusion

This work explains the importance of using Image compression and Image super resolution techniques and also two different models were proposed (Section 1.3) which combines the image compression with super resolution. From the results in section 4.1 it is clear that Model 1 provide better image quality at different compression rates.

The disadvantages in common compression techniques are clearly stated in section 2.1, which also explains how to compress an image using DCT and Wavelet transforms and why a new method is required for better performance. To overcome all the disadvantages such as blocking artifacts in DCT and inability to exploit neighborhood similarities in DWT, a new compression technique SPIHT is used which uses the basic principles of subband coding from DWT and prioritizes the coefficients based on information content and set partitioning of hierarchical trees. This compression technique achieves very high PSNR and SSIM when compared to DWT and DCT compression and is capable of providing high compression ratio with reasonable image quality. This compression plays a very important role in saving storage space and also while transmitting image through telephone lines or over internet.

There are many advantages if we could improve the image quality of the compressed image and basic interpolation techniques cannot be used as they suffer from some problems like blurring edges, artifacts around edges etc. Super Resolution techniques can be used for such applications and especially the LSE method which does not rely on external database has many advantages. The LSE method magnifies the image in steps of small scaling factors and predicts the missing high frequency band information by exploiting extremely neighbor patches of image at lower scale. This provides very good results with less computational complexity and takes very less time when compared with algorithms which uses external database patches.

Thus when these two algorithms were combined, at same cost (maximum number of bits after compression) we can achieve better image quality. For same number of bits, the compressed image in Model 1 has more number of bits per pixel when compared to Model 2. Therefore more information is available for a pixel in Model 1 than Model 2. As a result, despite of less number

of pixels in Model 1 it gives better image when compared to Model 2.The experiments conducted in this work considered 512×512 grayscale images, this work can be extended to color images using 3D SPIHT compression and then using same LSE super resolution to improve the image quality.

Chapter 6

6. References

[1] P. W. J. Majid Rabbani, Digital Image Compression Techniques, SPIE-The International Society for Optical Engineering, 1995.

[2] M. P. K.K. Shukla, Lossy Image Compression- Domain Decomposition Based Algorithms, British Library Cataloguing in Publication Data, 2011.

[3] M. M. H. S. E. E.-K. Fathi E. Adb El-Samie, Image Super-Resolution and Aplpications, CRC Press, Taylor & Francis Group, 2013.

[4] R. K. D. S. K. P. Ashish Ghosh, "Pattern Recognition and Machine Intelligence," in *Second Internation Conference, PReMI 2007*, Kolkata, India., 2007.

[5] A. A. S. K. U. K. L. Chattrakul Sombattheera, "Multi-disciplinary Trends in Artificial Intelligence," in *5th International Workshop, MIWAI 2011*, Hyderabad, India, 2011.

[6] E. P. S. A. C. B. Zhuoy Wang, "Multi-Scale Structural Similarity for image quality assessment," in *37th IEEE Asilomar Conference*, Pacific Grove, GA, 2003.

[7] B. Gautam, "Image Compression using Discrete Cosine Transform & Discrete Wavelet Transform," May 2010. [Online]. Available: http://ethesis.nitrkl.ac.in/1731/1/project.pdf.

[8] K. C. a. P. Gent, "Image Compression and the Discrete Cosine Transform," [Online]. Available: http://www.lokminglui.com/dct.pdf.

[9] H. S. Yun Q. Shi, Image and Video Compression for Multimedia Engineering - Fundamentals, Algorithms, and Standards, CRC Press LLC, 2000.

[10] P. J. V. Fleet, Discrete Wavelet Transformations - An Elementary Approach with Applications, John Wiley & Sons, 2008.

[11] J. M. Shapiro, "Embedded Image Coding Using Zerotrees pf Wavelet Coefficients," *IEEE,* vol. 41, p. 3447, 1993.

[12] wikipedia, "Time Frequency Analysis," [Online]. Available: http://en.wikipedia.org/wiki/Time%E2%80%93frequency_analysis.

[13] Wikipedia, "Short-Time Fourier Transform," [Online]. Available: http://en.wikipedia.org/wiki/Short-time_Fourier_transform.

[14] W. A. P. Amir Said, "A new Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees," *IEEE,* 1996.

[15] P. K. M. Vaishali Patel, "A Review on Different Image Interpolation Techniques for Image Enchancements," *Internation Journal of Emerging Technology and Advanced Engineering,* vol. 3, no. 12, pp. 129-133, 2013.

[16] Wikipedia, "Interpolation," [Online]. Available: http://en.wikipedia.org/wiki/Interpolation.

[17] M. D. Patil, "Overview of Techniques used for Image Resolution Enhancement," *Internation Journal on Computer Sceince and Engineering,* vol. 4, no. 7, pp. 1345-1347, 2012.

[18] B. Smith. [Online]. Available: http://www.engr.mun.ca/~baxter/Publications/ImageZooming.pdf.

[19] P. Gaidhani, "Super Resolution," [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV1011/Super_Resolution_CVonline.pdf.

[20] A. P. a. A. S. Willia, "Image Wavelet coding systems: Part I of Set partition coding and image wavelet coding system," *Foundations and Trends in Signal Processing,* vol. 2, no. 3, pp. 181-246, 2008.

[21] R. N. a. J. C. I.H. Witten, "Arithmetic coding for data compression," *Communications of the*

ACM, vol. 30, no. 6, pp. 520-540, 1987.

[22] Wikipedia, "Self-Similarity," [Online]. Available: http://en.wikipedia.org/wiki/Self-similarity.

[23] G. F. a. R. Fattal, "Image and Video Upscaling from Local Self-Examples," *AMC Transactions on Graphics,* vol. 28, no. 4, 2009.

[24] C. Vaslens, "Embedded Zerotree Wavelet Encoding," 1999.

[25] W. A. P. a. A. Said, "Image Wavelet Coding Systems: Pat II of Set Partition Coding and Image Wavelet Coding Systems," *Foundation and Trends in Signal Processing,* vol. 2, no. 3, pp. 181-246, 2008.