International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-2 Number-4 Issue-6 December-2012

An Efficient Image Compression Technique Based on Arithmetic Coding

Rajendra Kumar Patel
Assistant Professor, Department of CSE, Patel College of Science &Technology, Bhopal

Abstract

The rapid growth of digital imaging applications, including desktop publishing, multimedia, teleconferencing, and high visual definition has increased the need for effective and standardized image compression techniques. Digital Images play a very important role for describing the detailed information. The key obstacle for many applications is the vast amount of data required to represent a digital image directly. The various processes of digitizing the images to obtain it in the best quality for the more clear and accurate information leads to the requirement of more storage space and better storage and accessing mechanism in the form of hardware or software. In this paper we concentrate mainly on the above flaw so that we reduce the space with best quality image compression. State-of-the-art techniques can compress typical images from 1/10 to 1/50 their uncompressed size without visibly affecting image quality. From our study I observe that there is a need of good image compression technique which provides better reduction technique in terms of storage and quality. Arithmetic coding is the best way to reducing encoding data. So in this paper we propose arithmetic coding with walsh transformation based image compression technique which is an efficient way of reduction.

Keywords

Walsh Transformation, image Compression, arithmetic Coding, digitization.

1. Introduction

Compression is a reversible conversion of data to a format that requires fewer bits, usually performed so that the data can be stored or transmitted more efficiently [1]. The size of the data in compressed form (C) relative to the original size (O) is known as the compression ratio (R=O/C). If the inverse of the process, decompression, produces an exact replica of the original data then the compression is lossless. Lossy compression, usually applied to image data, does not allow reproduction of an exact replica of the original image, but has a higher compression ratio. Thus lossy compression allows only an approximation of the original to be generated. For image compression, the fidelity of the approximation usually decreases as the compression ratio increases.

Compression is analogous to folding a letter before placing it in a small envelope so that it can be transported more easily and cheaply. Compressed data, like the folded letter, is not easily read and must first be decompressed, or unfolded, to restore it to its original form.

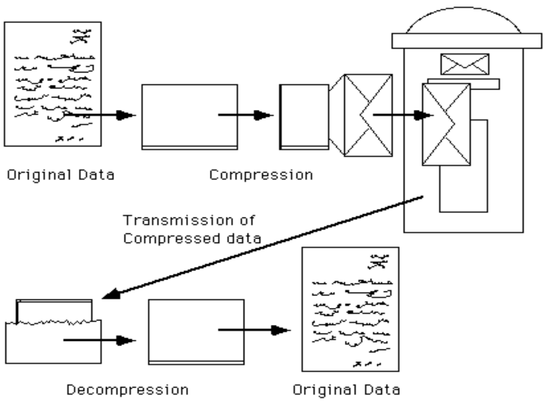

Figure 1: Image Compression

A device (software or hardware) that compresses data is often know as an encoder or coder, whereas a device that decompresses data is known as a decoder. A device that acts as both a coder and decoder is known as a codec.

A great number of compression techniques have been developed and some lossless techniques can be applied to any type of data. Development, in recent years, of lossy techniques specifically for image data has contributed a great deal to the realization of digital video applications.

Compression techniques used for digital video can be categorized into three main groups: [2]

- General purpose compression techniques can be used for any kind of data.
- Intraframe compression techniques work on images.
- Interframe compression techniques work on image sequences rather than individual images.

There are many popular general purpose lossless compression techniques that can be applied to any type of data [3].

Run Length Encoding
Run Length Encoding is a compression technique that replaces consecutive occurrences of a symbol with the symbol followed by the number of times it is repeated. For example, the string

111110000003355 could be represented by 15063252. Clearly this compression technique is most useful where symbols appear in long runs, and thus can sometimes be useful for images that have areas where the pixels all have the same value, cartoons for example.

Relative Encoding
Relative encoding is a transmission technique that attempts to improve efficiency by transmitting the difference between each value and its predecessor, in place of the value itself. Thus the values 15106433003 would be transmitted as 1+4-4-1+6-2-1+0-3+0+3. In effect the transmitter is predicting that each value is the same as its predecessor and the data transmitted is the difference between the predicted and actual values. Differential Pulse Code Modulation (DPCM) is an example of relative encoding. The signal above can have one of 7 possible values (-3 to +3) and so would require 3 bits per sample. Each sample can also be described by the difference between it and the previous sample. Each sample is either the same, one more, or one less than the previous sample. Only two bits are required to express the relationship between the samples. Coding the signal in this way results in a reduction of one third in the number of bits.

Huffman Coding
Huffman coding is a popular compression technique that assigns variable length codes (VLC) to symbols, so that the most frequently occurring symbols have the shortest codes [3]. On decompression the symbols are reassigned their original fixed length codes. When used to compress text, for example, variable length codes are used in place of ASCII codes, and the most common characters, usually space, e, and t are assigned the shortest codes. In this way the total number of bits required to transmit the data can be considerably less than the number required if the fixed length representation is used. Huffman coding is particularly effective where the data are dominated by a small number of symbols.

Arithmetic Coding
Although Huffman coding is very efficient, it is only optimal when the symbol probabilities are integral powers of two. Arithmetic coding [3] does not have this restriction and is usually more efficient than the more popular Huffman technique. Although more efficient than Huffman coding, arithmetic coding is more complex.

Lempel-Ziv Coding
Lempel-Ziv compressors use a dictionary of symbol sequences. When an occurrence of the sequence is repeated it is replaced by a reference to its position in the dictionary. There are several variations of this coding technique and they differ primarily in the manner in which they manage the dictionary. The most well-known of these techniques is the Lempel-Ziv-Welch variation.

We provide here an overview of Image Compression Technique. The rest of this paper is arranged as follows: Section 2 introduces Literature review; Section 3 describes problem domain; Section 4 shows the proposed approach; Section 5 describes Conclusion and outlook.

2. Recent Scenario

In 2006, Matthew J. Zukoski et al. [4] as medical/biological imaging facilities move towards complete film-less imaging, compression plays a key role. Although lossy compression techniques yield high compression rates, the medical community has been reluctant to adopt these methods, largely for legal reasons, and has instead relied on lossless compression techniques that yield low compression rates. The true goal is to maximize compression while maintaining clinical relevance and balancing legal risk. They proposes a novel model-based compression technique that makes use of clinically relevant regions as defined by radiologists. Lossless compression is used in these clinically relevant regions, and lossy compression is used everywhere else.

In 2007, R.Sukanesh et al. [5] a novel approach of information theory based Minimum Relative Entropy (MRE) and Entropy methods for image compression are discussed. A two stage compression process is performed through homogenous MRE method, and heterogeneous MRE. The compressed images are reconstructed through Region growing techniques. The performance of image compression and restoration is analyzed by the estimation of parametric values such as Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR). Higher the PSNR better the reconstruction process. Six radiographic medical images of various sizes are analyzed and Maximum PSNR of 33 is achieved.

In 2008, K. Veeraswamy,et al. [6] an adaptive image compression algorithm is proposed based on the prediction of AC coefficients in Discrete Cosine Transform (DCT) block during reconstruction of image. In the prediction phase, DC values of the nearest neighbour DCT blocks are utilized to predict the AC coefficients of centre block. Surrounding DC values of a DCT blocks are adaptively weighed for AC coefficients prediction. Linear programming is used to calculate the weights with respect to the image content. Results show that this method is good in terms of good Peak Signal to Noise Ratio (PSNR) and less blocking artifacts. The proposed scheme has been demonstrated through several experiments including Lena. Reconstructed image is of good quality with same compression ratio compared to the existing technique in the literature. In addition, an image watermarking algorithm is proposed using DCT AC coefficients obtained. The performance of the proposed watermarking scheme is measured in terms of PSNR and Normalized Cross Correlation (NCC). Further, this algorithm is robust for various attacks including JPEG compression on watermarked image.

In 2010, Jagadish al. [7] proposed the Lossless method of image compression and decompression using a simple coding technique called Huffman coding. This technique is simple in implementation and utilizes less memory. A software algorithm has been developed and implemented to compress and decompress the given image using Huffman coding techniques in a MATLAB platform.

In 2009, G.M.PADMAJA, et al. [8] analyzes various image compression techniques. In addition, specific methods are presented illustrating the application of such techniques to the real-world images. They have presented various steps involved in the general procedure for compressing images. They provided the basics of image coding with a discussion of vector quantization and one of the main technique of wavelet compression under vector quantization. This analysis of various compression techniques provides knowledge in identifying the advantageous features and helps in choosing correct method for compression.

In 2011, S.Parveen Banu et al. [9] a novel hybrid image compression technique for efficient storage and delivery of data is proposed. It is based on decomposing the data using daubechies-4 wavelet in combination with the lifting scheme and entropy encoding. This scheme is concerned with the compression ratio, bits per pixel and peak signal to noise ratio. Experimental results illustrate that the proposed scheme is efficient and feasible in terms of compression ratio, bits per pixel and peak signal to noise ratio.

In 2011, Yu Yanxin et al. [10] presented an image compression method suited to the space-borne application. To solve the problem of large-size RS images taking up large cache, the compression scheme based on overlap blocks was taken. The overlap blocks of the image were multi-levelly decomposed by lifting wavelet. According to human visual characteristics, the lossless encoding method was used for the low-frequency sub-band most sensitive to human vision, and the bit-plane coding method was take for the remaining high-frequency sub-bands. Simulation results show that the algorithm can remove the blocking artifacts and realize the high quality image compression.

In 2011, Baluram Nagaria et al. [11] discussed the comparative study of different wavelet-based image compression systems. When wavelet transform is applied to image compression, the chosen wavelet base affects the efficiency of signal strength, pixel values and the quality of the reconstructed image, because the property parameters of different wavelet bases are varied, it is very important to research the correlation between the wavelet base properties and image compression. They have discussed various statistical numerical measures and obtained results compared in terms of MSE, PSNR, Normalization and Compression Ratio with lower and higher pixel frames. The main objective of this research measure the quality of image with statistical numerical measures (PSNR, MSE respectively) using different wavelet families with suitable decomposition level. Different test image with 512 X 512 and 1024 X1024 pixel frames are used to evaluate the performance of image compression. The final choice of optimal wavelet in image compression application depends on image quality, minimum error, optimum PSNR and computational complexity. The experiments are performed using different wavelets at various levels of decomposition. This results show that Discrete Mayer wavelet with second and fourth level decomposition yields better quality for Lena (1024 X 124) and quality has been degrade as lower pixel frame with fruits(512 X 512)images. Our results provide a good reference for application developers to choose a go.

In 2011, Yu Shen et al. [12] addresses Compression is useful because it helps reduce the consumption of expensive resources, such as hard disk space or transmission bandwidth. Image compression may be lossless or lossey. Lossless image compression is a class of image compression algorithms that allows the exact original data to be reconstructed from the compressed data. The term lossless is in contrast to lossy image compression, which only allows an approximation of the original data to be reconstructed, in exchange for better compression rates. Lossless compression is preferred for archival purposes and often for medical imaging, satellite imaging, or technical drawings. For the urgent requirement of efficient lossless compression and high fidelity compression, more and more research of lossless image compression will be concerned. After the introduction of the lifting scheme and the integer to integer multiwavelets, they present the approach to build integer to integer multiwavelets. In addition, experimental results of applying these multiwavelets to lossless image compression are presented.

Intraframe compression is compression applied to still images, such as photographs and diagrams, and exploits the redundancy within the image, known as spatial redundancy. Intraframe compression techniques can be applied to individual frames of a video sequence.

Sub-sampling
Sub-sampling is the most basic of all image compression techniques and it reduces the amount of data by throwing some of it away. Sub-sampling reduces the number of bits required to describe an image, but the quality of the sub-sampled image is lower than the quality of the original. Sub-sampling of images usually takes place in one of two ways. In the first, the original image is copied but only a fraction of the pixels from the original are used, as illustrated below. Alternatively, sub-sampling can be implemented by calculating the average pixel value for each group of several pixels, and then substituting this average in the appropriate place in the approximated image. The latter technique is more complex, but generally produces better quality images.

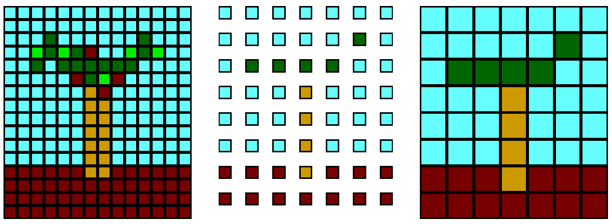

Figure 2: Sampling

In this example the pixels in every second row and every second column are ignored. To compensate for this, the size of the remaining pixels is doubled. The same procedure is illustrated below

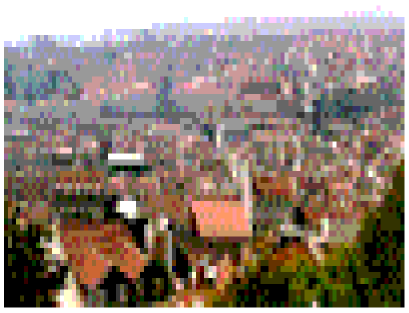

Figure 3: After Sub Sampling

For example, an image might be sub-sampled by 2 in both the x and y directions, thus every second line and every second column of the image is completely ignored. When the sub-sampled image is displayed at the same size as the original image the size of the pixels is doubled. This is known as pixel doubling.

When coding colour images, it is common to sub-sample the colour component of the image by 2 in both directions, while leaving the luminance component intact. This is useful because human vision is much less sensitive to chrominance than it is to luminance, and sub-sampling in this way reduces the number of bits required to specify the chrominance component by three quarters.

Sub-sampling is necessarily lossy but relies on the ability of human perception to fill in the gaps. The receiver itself can also attempt to fill in the gaps and try to restore the pixels that have been removed during sub-sampling. By comparing adjacent pixels of the sub-sampled image, the value of the missing in-between pixels can be approximated. This process is known as interpolation. Interpolation can be used to make a sub-sampled image appear to have higher resolution than it actually has and is usually more successful than pixel doubling. It can, however, result in edges becoming blurred.

Coarse quantization is similar to sub-sampling in that information is discarded, but the compression is accomplished by reducing the numbers of bits used to describe each pixel, rather than reducing the number of pixels. Each pixel is reassigned an alternative value and the number of alternate values is less than that in the original image. In a monochrome image, Figure 4 for example, the number of shades of grey that pixels can have is reduced. Quantization where the number of ranges is small is known as coarse quantization.

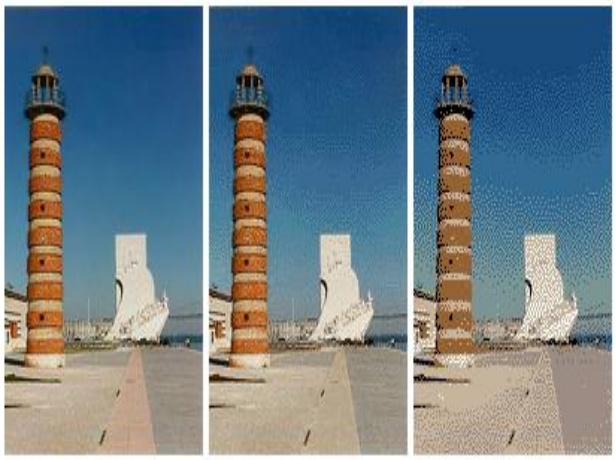

Figure 4: Coarse Quantization

These three images have different numbers of colors. The first has thousands of different colored pixels. The middle has 64 different colors and the rightmost uses only 8 different colors.
Photographic quality images typically require pixels of 24 bits, but can be reduced to 16 bits with acceptable loss. Images with 8 bits, however, are noticeably inferior to those with 16 bits per pixel. Coarse quantization of images, often called bit depth reduction, is a very common way of reducing the storage requirements of images.

3. Problem Domain

One of the paradoxes of technology evolution is that despite the development of the computer and the increased need for storing information, there is a lack of development in the area of data compression. As the evolution of computer systems progresses rapidly, the amount of stored information will increase at an unrelentingly fast rate. Hence, over the past two decades, several techniques have been developed for data compression and each of which has its own particular advantages and disadvantages. Each technique works best with the circumstances that it is designed to work for. Data compression is the process of observing regularities in the data and trying to eliminate them with the aim of reduce the total size of the data. Different types of data have different structures and different forms of regularities. For instance, a common pattern in English text files is ”th” or ”qu”, while a common pattern in images is the similarity of adjacent pixel values. In videos, a common pattern is the similarity of subsequent frames. Therefore, it is no surprise to observe poor performance in relation to the compression ratio when applying a compression algorithm in a different environment than the one it was designed to work for. This is because different compression algorithms are designed to capture different forms of redundancy.

Naturally, the process of compressing and decompressing files takes time and costs CPU cycles and, so, one always needs to balance these costs against the extra time and costs involved in the storage and transfer (to and from disk and/or via a network) of uncompressed files. While in some domains it may be more attractive to spend money on storage rather than on CPU cycles to compress and decompress, for most users of computers and other digital devices the balance is in favor of the use of compressed files, particularly considering how many CPU cycles are wasted every day in computers idling or running screen savers and the cost and difficulties in upgrading disks. For example, upgrading the hard disk of a laptop requires technical competence and a lengthy backup/restore procedure, which may even involve the complete re-installation of all software packages, including the operating system, from the original disks.

When a great deal is known about the regularities in the files to be compressed, e.g., in the compression of audio, photo and video data, it is possible to match different compression algorithms with different parts of the data in such a way to achieve the maximum compression ratio (i.e., compression ratio equal to the entropy of the data), thereby providing significant reductions in file sizes. These algorithms are difficult to derive and are often the result of many person years of effort (e.g., the JPEG and MPEG standards have been developed for approximately two decades). Nonetheless, these are so successful that effectively some types of data are regularly saved, kept and transferred, only in compressed form, to be decompressed automatically and transparently to the user only when loaded into applications that make use of them (e.g., an MP3 player). When the nature and regularities of the data to be compressed is less predictable or when the user has to deal with heterogeneous sets of data, then a general purpose compression system, while unavoidably less effective than the highly specialized ones mentioned above, is the only option. Heterogeneous data sets occur in a variety of situations. For example, a user may decide to use a compressed file system for his disk drive (which is offered as a standard option on many operating systems) to prolong its life, which implies that a mixture of file types (possibly some of which, such as music and videos, are already in compressed form) need to be kept in lossless compressed form. Perhaps even more frequently, a computer user may want to store multiple files in an archive file and then compress it to save space.

Several such algorithms have been in the past (or are still today) covered by patents. Also, the best current compression algorithms do capture regularities in heterogeneous data sets is to ensure files with each known extension are compressed with an algorithm which on average works well on files with that extension. Some systems (but not all) may even ensure each file in an archive is compressed with a supposedly good algorithm for that type of file. However, no algorithm attempts to capture the differences in the internal regularities within single files to any significant degree. Clearly, exploiting these differences could improve the compression ratios achievable very significantly. We noticed that it is very difficult to design a single generic universal compression algorithm that works effectively with any data type without having some knowledge or assumption of the data to be compressed. In this work we attempt to tackle the problem using an evolutionary approach.

Just as image compression has increased the efficiency of sharing and viewing personal images, it offers the same benefits to just about every industry in existence. Early evidence of image compression suggests that this technique was, in the beginning, most commonly used in the printing, data storage, and telecommunications industries. Today however, the digital form of image compression is also being put to work in industries such as fax transmission, satellite remote sensing, and high definition television, to name but a few.
In certain industries, the archiving of large numbers of images is required. A good example is the health industry, where the constant scanning and/or storage of medical images and documents take place. Image compression offers many benefits here, as information can be stored without placing large loads on system servers. Depending on the type of compression applied, images can be compressed to save storage space, or to send to multiple physicians for examination. And conveniently, these images can uncompress when they are ready to be viewed, retaining the original high quality and detail that medical imagery demands.
In the retail store example, the introduction and placement of new products or the removal of discontinued items can be much more easily completed when all employees receive, view and process images in the same way. Federal government agencies that standardize their image viewing, storage and transmitting processes can eliminate large amounts of time spent in explanation and problem solving. The time they save can then be applied to issues within the organization, such as the improvement of government and employee programs.
In the security industry, image compression can greatly increase the efficiency of recording, processing and storage. However, in this application it is imperative to determine whether one compression standard will benefit all areas. For example, in a video networking or closed-circuit television application, several images at different frame rates may be required. Time is also a consideration, as different areas may need to be recorded for various lengths of time. Image resolution and quality also become considerations, as does network bandwidth, and the overall security of the system.

Museums and galleries consider the quality of reproductions to be of the utmost importance. Image compression, therefore, can be very effectively applied in cases where accurate representations of museum or gallery items are required, such as on a Web site. Detailed images that offer short download times and easy viewing benefit all types of visitors, from the student to the discriminating collector. Compressed images can also be used in museum or gallery kiosks for the education of that establishment's visitors. In a library scenario, students and enthusiasts from around the world can view and enjoy a multitude of documents and texts without having to incur traveling or lodging costs to do so.

4. Proposed Approach

Our Proposed Compression algorithm consists of the following steps:

1. Two Levels Discrete Wavelet Transform
2. Apply 2D Walsh-Hadamard Transform on each 8x8 block of the low-frequency sub-band
3. Split all DC values form each transformed block 8x8
4. Compress each sub-band by using Arithmetic coding

For better understanding our proposed approach we explain the terminology and the proposed approach in the subsequent section.

Algorithm 1: Providing Wavelet Name and Property

```
Step 1: [Accept the Wavelet Name]
wavelet_name=get(handles.edit1,'String');
Step 2: qf1=str2num(get(handles.edit2,'string'));
Step 3 : qf2=str2num(get(handles.edit3,'string'));
Step 4: crf=get(handles.slider3,'value');
Step 5: [Parameter must be within the range of 0.02 and 0.5]
if(isempty(wavelet_name)||(qf1>0.5||qf1<0.02)||(qf2>0.5||qf2<0.02))
errordlg('Please Properly define wavelet_name and give parameter within the range(0.02 to 0.5));
```

else
Wrong parameter.

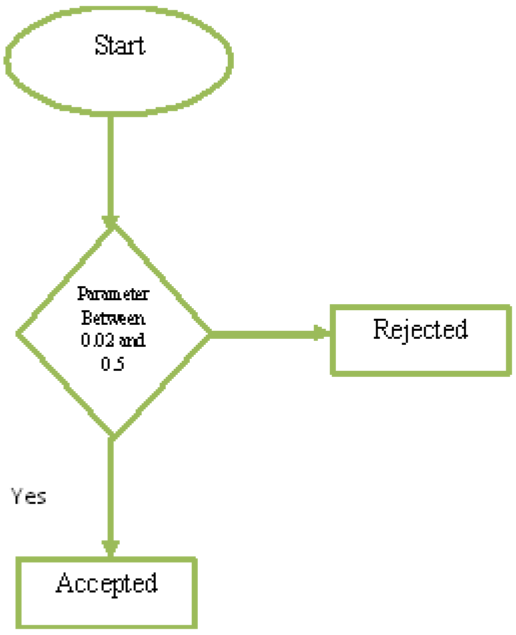

Figure 5: Flowchart 1

Algorithm 2: Discrete 2-D wavelet transforms
2D Discrete Wavelet Transform (2D DWT) is used in image processing as a powerful tool solving to image analysis, denoising, image segmentation and other. 2D DWT can be applied as a convolution of a selected wavelet function with an original image or it can be seen as a set of two matrices of filters, row and column one. Using a separability property of DWT, the first part of decomposition consists of an application of row filters to the original image. The column filter is used for further processing of image resulting from the first step. This image decomposition can by mathematically described by the below equation

C=X. I. Y
where C is the final matrix of wavelet coefficients, I represents an original image, X is a matrix of row filters and Y is a matrix of column filters.

In the first level of decomposition of 2D DWT, the image is separated into four parts. Each of them has a quarter size of the original image. They are called approximation coefficients (Low Low or LL), horizontal (Low High or LH), vertical (High-low or HL) and detail coefficients (High High or HH) . Approximation coefficients obtained in the first level can be used for the next decomposition level. Inverse 2D Discrete Wavelet Transform used in image reconstruction is defined by the below equation
$I_{rec = X}{}^{-1} . C . Y^{-1}$

For the orthogonal matrices this formula can be simplified into below equation
$I_{rec = X}{}^{T} . C . Y^{T}$

2D DWT decomposition separates an image into the four parts, each of them contains different information of the original image. Detail coefficients represent edges in the image, approximation coefficients are supposed to be a noise. A proper modification of approximation coefficients is the easiest way for edge detection.

Syntax :
[cA,cH,cV,cD] = dwt2(X,'wname')
[cA,cH,cV,cD] = dwt2(X,Lo_D,Hi_D)

The dwt command performs a single-level two-dimensional wavelet decomposition with respect to either a particular wavelet ('wname', see wfilters for more information) or particular wavelet decomposition filters (Lo_D and Hi_D) you specify.

[cA,cH,cV,cD] = dwt2(X,'wname') computes the approximation coefficients matrix cA and details coefficients matrices cH, cV, and cD (horizontal, vertical, and diagonal, respectively), obtained by wavelet decomposition of the input matrix X. The 'wname' string contains the wavelet name.

[cA,cH,cV,cD] = dwt2(X,Lo_D,Hi_D) computes the two-dimensional wavelet decomposition as above, based on wavelet decomposition filters that you specify.

Lo_D is the decomposition low-pass filter.
Hi_D is the decomposition high-pass filter.
Lo_D and Hi_D must be the same length.
Let sx = size(X) and lf = the length of filters; then size(cA) = size(cH) = size(cV) = size(cD) = sa where sa = ceil(sx/2), if the DWT extension mode is set to periodization. For the other extension modes, sa = floor((sx+lf-1)/2).
For information about the different Discrete Wavelet Transform extension modes, see dwtmode.
[cA,cH,cV,cD] = dwt2(...,'mode',MODE) computes the wavelet decomposition with the extension mode MODE that you specify.
MODE is a string containing the desired extension mode.
An example of valid use is
[cA,cH,cV,cD] = dwt2(x,'db1','mode','sym');
When X represents an indexed image, then X, as well as the output arrays cA,cH,cV,cD are m-by-n matrices. When X represents a truecolor image, it is an m-by-n-by-3 array, where each m-by-n

matrix represents a red, green, or blue color plane concatenated along the third dimension.

Algorithm 3:

```
Step 1: The current extension mode is zero-padding
Step 2: Load original image.
Step 3:  X contains the loaded image.
Step 4:  map contains the loaded colormap.
Step 5: nbcol = size(map,1);
Step 6:Perform single-level decomposition
 [cA1,cH1,cV1,cD1] = dwt2(X,'db1');
Step 7: Images coding.
cod_X = wcodemat(X,nbcol);
cod_cA1 = wcodemat(cA1,nbcol);
cod_cH1 = wcodemat(cH1,nbcol);
cod_cV1 = wcodemat(cV1,nbcol);
cod_cD1 = wcodemat(cD1,nbcol);
dec2d = [...     cod_cA1,    cod_cH1;    ...
cod_cV1,   cod_cD1    ...    ];
```

Each sub band is quantized by a factor which is given below:

HL2=HL2/Factor
LH2=LH2/Factor
HH2=HH2/Factor

Algorithm 4: Arithmetic Coding

```
Function:
function [Store_Byte,Counts,Table]=sym2bit(Data)
Input :Table=0; New_Data=0;
Output: Table(1)=Data(1);
Step 1: S_AC=size(Data);
   for jAC=1:S_AC(2)
      S_2AC=size(Table);Flag=0;
      for kAC=1:S_2AC(2)
         if (Table(kAC)==Data(jAC))
            Flag=1;
         end;
      end;
     Step 2: if (Flag==0)
Table(S_2AC(2)+1)=Data(jAC); end;
   end;
```

Arithmetic coding is similar to Huffman coding; they both achieve their compression by reducing the average number of bits required to represent a symbol.

Given:

An alphabet with symbols S0, S1, ... Sn, where each symbol has a probability of occurrence of p0, p1, ... pn such that ∑pi = 1.

From the fundamental theorem of information theory, it can be shown that the optimal coding for Si requires -(pi×log2(pi)) bits.

More often than not, the optimal number of bits is fractional. Unlike Huffman coding, arithmetic coding provides the ability to represent symbols with fractional bits.

Since, ∑pi = 1, we can represent each probability, pi, as a unique non-overlapping range of values between 0 and 1. There's no magic in this, we're just creating ranges on a probability line.

For example, suppose we have an alphabet 'a', 'b', 'c', 'd', and 'e' with probabilities of occurrence of 30%, 15%, 25%, 10%, and 20%. We can choose the following range assignments to each symbol based on its probability:

Table 4.1: Sample Symbol Ranges

Symbol	Probability	Range
a	30 %	0.00,0.30
b	15%	0.30, 0.45
c	25%	0.45, 0.70
d	10%	0.70, 0.80
e	20%	0.80, 1.00

Encoding Strings

By assigning each symbol its own unique probability range, it's possible to encode a single symbol by its range. Using this approach, we could encode a string as a series of probability ranges, but that doesn't compress anything. Instead additional symbols may be encoded by restricting the current probability range by the range of a new symbol being encoded. The pseudo code below illustrates how additional symbols may be added to an encoded string by restricting the string's range bounds.

```
lower bound = 0
upper bound = 1
while there are still symbols to encode
current range = upper bound - lower bound
upper bound = lower bound + (current range × upper bound of new symbol)
lower bound = lower bound + (current range × lower bound of new symbol)
end while
```

Any value between the computed lower and upper probability bounds now encodes the input string.

Example:

Encode the string "ace" using the probability ranges from Table 1.

Start with lower and upper probability bounds of 0 and 1.

Encode 'a'
current range = 1 - 0 = 1
upper bound = 0 + (1 × 0.3) = 0.3
lower bound = 0 + (1 × 0.0) = 0.0

Encode 'c'
current range = 0.3 - 0.0 = 0.3
upper bound = 0.0 + (0.3 × 0.70) = 0.210
lower bound = 0.0 + (0.3 × 0.45) = 0.135

Encode 'e'
current range = 0.210 - 0.135 = 0.075
upper bound = 0.135 + (0.075 × 1.00) = 0.210
lower bound = 0.135 + (0.075 × 0.80) = 0.195
The string "ace" may be encoded by any value within the probability range [0.195, 0.210).
It should become apparent from the example that precision requirements increase as additional symbols are encoded. Strings of unlimited length require infinite precision probability range bounds. The section on implementation discusses how the need for infinite precision is handled.

This program is depends on Wavelet and Walsh transform for transformation, and then using Arithmetic coding for compress an image.
This Compression algorithm consists of the following steps:

1. Two Levels Discrete Wavelet Transform
2. Apply 2D Walsh-Hadamard Transform on each 8x8 block of the low-frequency sub-band
3. Split all DC values form each transformed block 8x8
4. Compress each sub-band by using Arithmetic coding

Domains Factors-
For (Low Frequency) Qf1----->LL2
For Qf2(High Frequency) Qf2 ------->HH2 and HL2 and LH2 see above diagram.
and range values {0.02 - 0.5} for Qf1 and Qf2.
Parameter
Crf-range (1-10) it uses to generate quantization matrix. and what is quantization matrix and its importance in the image processing.

5. Conclusion

In this paper we survey several aspect of image compression technique. Different techniques are presented and compared. In this paper we also discuss our proposed approach which is the combination of walsh and arithmetic coding. This approach provides a better performance for compression technique. In future we also present a comparative study with our approach.

References

[1] H, Danny Lazar and Moshe Israeli, “Image Compression using WT and Multiresolution Decomposition”, IEEE Trans. On Image Proc.Vol. 5, No. 1, Jan 1996.
[2] Z. Xiong et.al, “A comparative study of DCT- and wavelet-based image coding,” IEEE Trans. on Circuits and Systems for Video Tech., vol.9, pp. 692-695, Aug. 1999.
[3] A. M. Eskicioglu and P. S. Fisher, “Image quality measures and their performance,” IEEE Signal processing letters, vol. 43, pp.2959-2965, Dec. 1995.
[4] Matthew J. Zukoski , Terrance Boult and Tunç Iyriboz , “A novel approach to medical image compression”, Int. J. Bioinformatics Research and Applications, Vol. 2, No. 1, 2006.
[5] Dr. R. Sukanesh, R. Harikumar,N.S. Balaji and S.R. Balasubramaniam , “Analysis of Image Compression by Minimum Relative Entropy (MRE) and Restoration through Weighted Region Growing Techniques for Medical Images” , Engineering Letters, 14:1, EL_14_1_16 , 2007.
[6] K. Veeraswamy, S. Srinivas Kumar , “Adaptive AC-Coefficient Prediction for Image Compression and Blind Watermarking”, Journal of Mutltimedia, VOL. 3, NO. 1, MAY 2008.
[7] Jagadish H. Oujar,Lohit M. Kadlaskar , “ A New Lossless Method of Image Compression and Decompression using Huffman Coding Techniques”, Journal of Theoretical and Applied Information Technology © 2005 - 2010 JATIT.
[8] G.M.Padmaja, P.Nirupama , “Analysis of Various Image Compression Techniques”, ARPN Journal of Science and Technology ©2011-2012.
[9] S.Parveen Banu Dr. Y. Venkataramani , “An Efficient Hybrid Image Compression Scheme based on Correlation of Pixels for Storage and Transmission of Images”, International Journal of Computer Applications (0975 – 8887) Volume 18– No.3, March 2011.
[10] Yu Yanxin , Song Xue , “A Remote Sensing Image Compression Method Suited to Space-borne Application”, 2011 International Conference on Computer Science and Network Technology.
[11] Baluram Nagaria , Mohammad Farukh Hashmi ,Imran Hussain ,Ravijeet Singh Chauhan ,” Comparative Analysis of an Optimal Image Compression using FDWT at Various Decomposition Level with Different Statistical Numerical Measures for Different Pixel Frame”, 2011 International Conference on Computational Intelligence and Communication Systems.
[12] Yu Shen, , Xieping Gao, Linlang Liu, Caixia Li, Qiying Cao , “Integer to Integer Multiwavelets for Lossless Image Compression”, Proceedings of IEEE IC-BNMT2011.

Prof. Rajendra Kumar Patel is an Assistant Professor in Department of Computer Science & Engineering, Patel College of Science &Technology, Bhopal. He did his Bachelor of Technology (2005-09) from M.G.C.G.V University Chitrakoot. Satna, M.Tech (2009-11) from M.A.N.I.T., Bhopal.