Image Compression using Discrete Wavelet Transform

G.Madhuri

J.Hema Sandhya

K.Ujjwal Kumar Chowdary

CONTENTS

ABSTRACT

ABSTRACT

This Project presents an approach towards MATLAB implemention of the Discrete Wavelet Transform (DWT) for image compression. The design follows the JPEG2000 standard and can be used for both lossy and lossless compression.In order to reduce complexities of the design linear algebra view of DWT has been used in this concept.

With the use of more and more digital still and moving images, huge amount of disk space is required for storage and manipulation purpose. For example, a standard 35-mm photograph digitized at 12μm per pixel requires about 18 Mbytes of storage and one second of NTSC-quality color video requires 23 Mbytes of storage.

JPEG is the most commonly used image compression standard in today's world. But researchers have found that JPEG has many limitations. In order to overcome all those limitations and to add on new improved features, ISO and ITU-T has come up with new image compression standard, which is JPEG2000.

INTRODUCTION

The proliferation of digital technology has not only accelerated the pace of development of many image processing software and multimedia applications, but also motivated the need for better compression algorithms.

Image compression plays a critical role in telematics applications. It is desired that either single images or sequences of images be transmitted over computer networks at large distances so as that they could be used for a multitude of purposes. For instance, it is necessary that medical images be transmitted so as that reliable, improved and fast medical diagnosis performed by many centers could be facilitated. To this end, image compression is an important research issue. The difficulty, however, in several applications lies on the fact that, while high compression rates are desired, the applicability of the reconstructed images depends on whether some significant characteristics of the original images are preserved after the compression process has been finished.

Image compression is a fast paced and dynamically changing field with many different varieties of compression methods available. Images contain large amount of data hidden in them, which is highly correlated. A common characteristic of most images is that the neighboring pixels are correlated and therefore contain redundant information. Research advances in wavelet theory have created a surge of interest in applications like image compression. The investigation and design of computationally efficient and effective software algorithms for lossy image compression forms the primary objective of this thesis. In wavelet image compression, parts of an image is described with reference to other parts of the same image and by doing so, the

redundancy of piecewise self-similarity is exploited. There are a number of problems to be solved in image compression to make the process viable and more efficient. A lot of work has been done in the area of wavelet based lossy image compression. However, very little work has been done in lossless image compression using wavelets to improve image quality. So the Proposed methodology of this paper is to achieve high compression ratio in images using 2D-Haar Wavelet Transform by applying different compression thresholds for the wavelet coefficients. That is, different compression ratios are applied to the wavelet coefficients belonging in the different regions of interest, in which of either each wavelet domain band of the transformed image.

IMAGE:

An image may be defined as a two-dimensional function, $f(x, y)$, where x and y are spatial (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point. When x, y and the amplitude values of f are all finite, discrete quantities, we call the image a ‘digital image‘.

Image compression is used to minimize the amount of memory needed to represent an image. Images often require a large number of bits to represent them, and if the image needs to be transmitted or stored, it is impractical to do so without somehow reducing the number of bits. The problem of transmitting or storing an image affects all of us daily. TV and fax machines are both examples of image transmission, and digital video players and web pictures of Catherine Zeta-Jones are examples of image storage.

Image compression is the process of reducing the amount of data required to represent an image. How to achieve compression means by removing the redundant information, we can achieve it. We have three types of redundancies.

1. Coding redundancy
2. Inter pixel redundancy
3. Psycho visual redundancy

1. Coding redundancy:

Image is nothing but combination of pixels and each pixel is represented in binary bits. The number of bits used to represent each pixel is based on number of gray levels used to represent the image. If we use variable length code (number of bits used to represent each pixel) is different for different pixels in an image. If we use less number of bits for more frequent gray levels and more number of bits for less frequent gray levels in the image, then we represent the entire image by using least possible number of bits. In this way we can reduce the coding redundancy.

Which intensity value will appear after the pixel p is undefined. So we can treat the gray levels as random variables. Let the gray level r_k occurs with probability $p_r(r_k)$.

$$p_r(r_k) = \frac{n_k}{MN}$$

K=0, 1, 2, 3……..L-1

Where L=Number of gray levels present in the image

n_k=Number of times the k th gray level appears

MN = Number of pixels in the image

If l (r_k) is the number of bits used represent k th gray level then average number of bits used to represent each pixel is

$l_{avg}=\sum_{k=0}^{L-1} l(r_k)p_r(r_k)$.....average length of code

Total number of bits used to represent M×N image=M×N× l_{avg}

If we use fixed length (m) for each pixel then the total number of bits used to represent the M×N image=M × N × m

Generally M × N × m ≥ M×N× l_{avg}

2. Inter pixel redundancy:

In an image each pixel depends on its neighbors. Information is unnecessarily replicates in the representations of the correlated pixels. Correlation of pixels means completely dependent on one another.

Examples: In video, if frame rate is high then the successive frames contain almost same information.

In still images, if spatial resolutions is high then inter pixel redundancy is high.

3. Psycho visual redundancy:

Information that is ignored by human visual system or its extraneous to the intended use of an image are obvious candidates for omission.

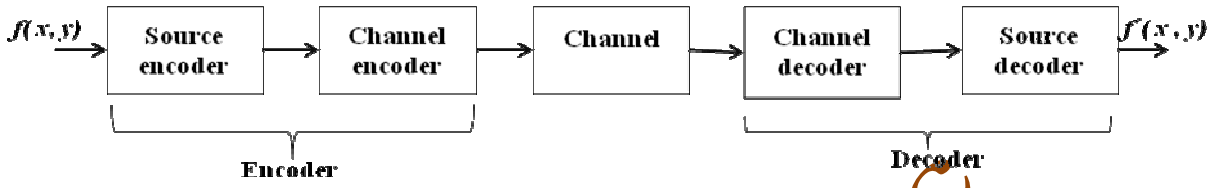
We can define compression ratio as ratio of original image size to the compressed image size.

►Need for image compression:

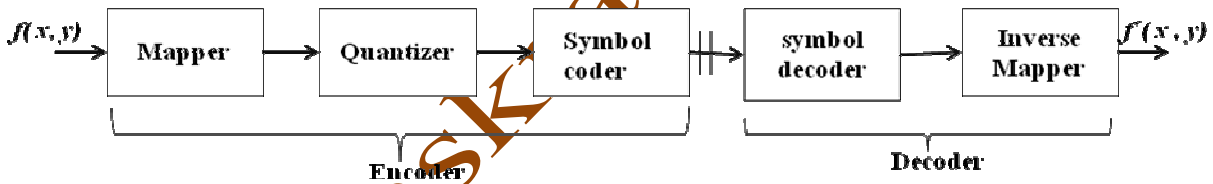
- To save memory required to store the image
- To save transmission band width
- To save transmission time

For example, Frame with 352x288 contains 202,752 bytes of information. Recoding of uncompressed version of this video at 15 frames per second would require 3 MB. One minute→180 MB storage. One 24-hour day→262 GB. Using compression, 15 frames/second for 24 hour→1.4 GB, 187 days of video could be stored using the same disk space that uncompressed video would use in one day. Transmission and storage of uncompressed video would be extremely costly and impractical.

General image compression model:



It consists of two blocks encoder and decoder. Compression is done in the encoder and decompression is done in the decoder. Again encoder consists of three blocks and decoder consists of two blocks.



In the first stage of encoding process, a Mapper transforms *f(x, y)* into a format to reduce interpixel redundancy. This operation is reversible and runlength coding is an example for it. In the second stage quantizer keep the irrelevant information out of the compressed image, this operation is irreversible and results in lossy compression.In the third and final stage of encoder the symbol encoder generates fixed or variable length code to represent the quantizer output, generally variable length code is used to reduce the coding redundancy, this is reversible.

As quantization is irreversible process decoder contains two stages symbol decoder and inverse mapper, which perform just reverse operation of symbol encoder and mapper

►DISCRETE WAVELET TRANSFORM (DWT):

We have basically four types of wavelets

- Haar wavelet transform
- Daubechies wavelet transform
- Symlet wavelet transform
- Biorthogonal wavelet transform

As computational complexity increases, compression ratio also increases. Haar wavelet transform is the simplest transform for image compression, the principle behind this is very simple as calculating averages and differences of adjacent pixels. The Haar DWT is more computationally efficient than the sinusoidal based discrete transforms, but this quality is a tradeoff with decreased energy compaction compared to the DCT.

"The Haar transform operates as a square matrix of length N = some integral power of 2. Implementing the discrete Haar transform consists of acting on a matrix row-wise finding the sums and differences of consecutive elements. If the matrix is split in half from top to bottom the sums are stored in one side and the differences in the other. Next operation occurs column-wise, splitting the image in half from left to right, and storing the sums on one half and the differences in the other. The process is repeated on the smaller square, power-of-two matrix resulting in sums of sums. The number of times this process occurs can be thought of as the depth of the transform. In our project, we worked with depth four transforms changing a 256x256 images to a new 256x256 image with a 16x16 purely sums region in the upper-left hand corner".

Properties of Haar wavelet transform:

1. Haar Transform is real and orthogonal. Therefore

Hr = Hr* (1)

Hr = Hr (2)

Haar Transform is a very fast transform.

2. The basis vectors of the Haar matrix are sequence ordered.

3. Haar Transform has poor energy compaction for images.

4. Orthogonality: The original signal is split into a low and a high frequency part and filters enabling the splitting without duplicating information are said to be orthogonal.

5. Linear Phase: To obtain linear phase, symmetric filters would have to be used.

6. Compact support: The magnitude response of the filter should be exactly zero outside the frequency range covered by the transform. If this property is algebra satisfied, the transform is energy invariant.

7. Perfect reconstruction: If the input signal is transformed and inversely transformed using a set of weighted basis functions and the reproduced sample values are identical to those of the input signal, the transform is said to have the perfect reconstruction property. If, in addition no information redundancy is present in the sampled signal, the wavelet transform is, as stated above, ortho normal.

The advantages of Haar Wavelet transform as follows:

1. Best performance in terms of computation time.

2. Computation speed is high. Called lossless compression.

3. Simplicity (e.g., if, = 0).

4. HWT is efficient compression method.

5. It is memory efficient, since it can be calculated in place without a temporary array.

In order to implement the image compression algorithm we chose, we divided the process into various steps:

- calculate the sums and differences of every row of the image
- calculate the sums and differences of every column of the resulting matrix
- repeat this process until we get down to squares of 16x16
- quantize the final matrix using different bit allocation schemes
- write the quantized matrix out to a binary file

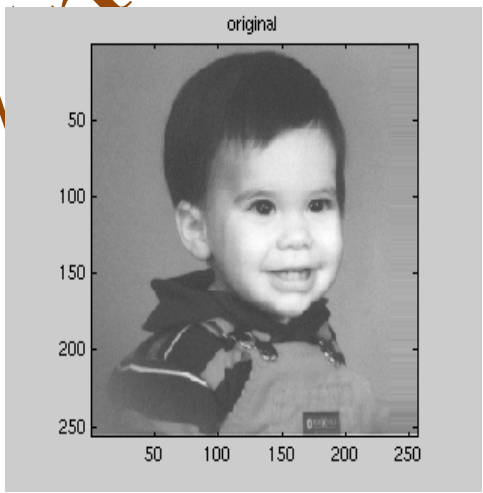


Figure 1 - Original image of boy.256

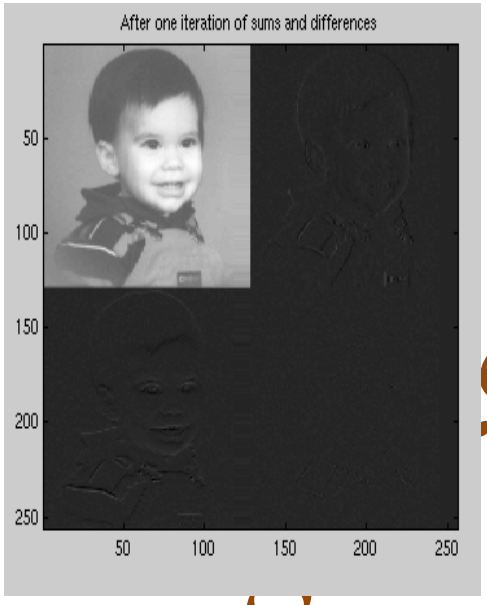


Figure 2 - Sums and differences after one iteration

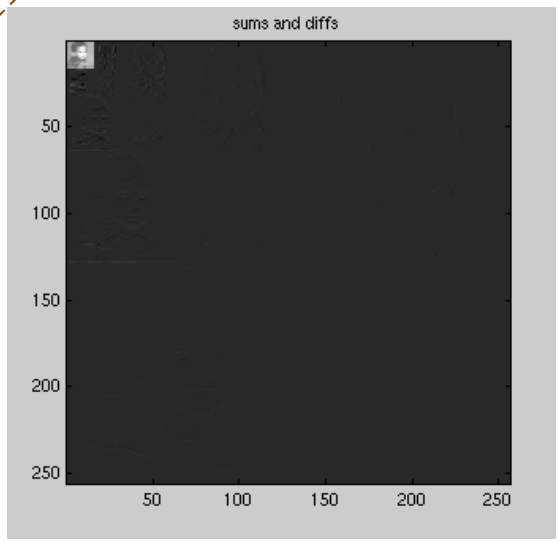


Figure 3 - Final sums and differences matrix

►CODE:

```
close all;

clear all;


a=imread('D:\Madhuri Photos\baby photos\DSC03311.jpg'); \\ read the input image

figure(1);

imshow(a);  \\ displaying the input image

title('color image');  \\ title of the figure


c=rgb2gray(a);  \\ converting color image to grayscale(binary) image

b=imresize(c, [512 512]);   \\ image size can be changed using this function & it creates the
                            \\ matrix with pixel values

figure(2);

imshow(b);

title('original gray scale image');

n=512;   \\  image is square image

k=1;   \\ start the calculation of averages and differences of all rows first

for i=1:1:n
   for j=1:2:n
      if (j<=n-1)
         s(i,k)=(b(i,j)+b(i,j+1))/2;
         d(i,k)=b(i,j)-b(i,j+1);
       end
```

```
    k=k+1;
  end
 k=1;
end

for i=1:1:n   \\ place the averages and differences in the original matrix
  for j=1:1:n/2
    b(i,j)=s(i,j);
    b(i,j+n/2)=d(i,j);
  end
end

for j=1:1:n/2  \\ calculate the averages and differences for the first half of  the matrix
  for i=1:2:n
    if i<=n-1
      s(k,j)=(b(i,j)+b(i+1,j))/2;
      d(k,j)=b(i,j)-b(i+1,j);
    end
    k=k+1;
  end
 k=1;
end
```

```
for i=1:1:n/2  \\ place the results in the original matrix
   for j=1:1:n/2
      b(i,j)=s(i,j);
      b(i+n/2,j)=d(i,j);
   end
end


figure(3);
imshow(b);  \\  image after doing one level
title('image after one level of compression')


for i=1:1:n/2  \\  do the same on the image obtained after one level compression
   for j=1:2:n/2
      if (j<=n/2-1)
         s(i,k)=(b(i,j)+b(i,j+1))/2;
         d(i,k)=b(i,j)-b(i,j+1);
       end
     k=k+1;
   end
  k=1;
end


for i=1:1:n/2
```

```
    for j=1:1:n/4
        b(i,j)=s(i,j);
        b(i,j+n/4)=d(i,j);
    end
end

for j=1:1:n/4
    for i=1:2:n/2
        if i<=n/2-1
            s(k,j)=(b(i,j)+b(i+1,j))/2;
            d(k,j)=b(i,j)-b(i+1,j);
        end
      k=k+1;
    end
   k=1;
end

for i=1:1:n/4
    for j=1:1:n/4
        b(i,j)=s(i,j);
        b(i+n/4,j)=d(i,j);
    end
end
```

```
figure(4);
imshow(b);
title('image after two level of compression')


for i=1:1:n/4  \\ perform the same on image obtained after second level of compression
   for j=1:2:n/4
      if (j<=n/4-1)
         s(i,k)=(b(i,j)+b(i,j+1))/2;
         d(i,k)=b(i,j)-b(i,j+1);
       end
     k=k+1;
   end
  k=1;
end

for i=1:1:n/4
   for j=1:1:n/8
      b(i,j)=s(i,j);
      b(i,j+n/8)=d(i,j);
   end
end
```

```
for j=1:1:n/8
   for i=1:2:n/4
      if i<=n/4-1
         s(k,j)=(b(i,j)+b(i+1,j)/2);
         d(k,j)=b(i,j)-b(i+1,j);
       end
     k=k+1;
   end
  k=1;
end

for i=1:1:n/8
   for j=1:1:n/8
      b (i, j)=s(i,j);
      b (i+n/8, j)=d(i,j);
   end
end

figure (5);
imshow (b);
title ('image after third level of compression')
```

►DWT DRAWBACKS:

- The cost of computing DWT as compared to DCT may be higher.
- The use of larger DWT basis functions or wavelet filters produces blurring and ringing noise near edge regions in images or video frames
- Longer compression time
- Lower quality than JPEG at low compression rates

►FUTURE IMAGE COMPRESSION:

- Improved low bit-rate compression performance
- Improved lossless and lossy compression
- Improved continuous-tone and bi-level compression
- Be able to compress large images
- Use single decompression architecture
- Transmission in noisy environments
- Robustness to bit-errors
- Progressive transmission by pixel accuracy and resolution
- Protective image security

REFERENCES:

- http://www.vlsilab.polito.it/Articles/mwscas00.pdf
- http://www.ee.vt.edu/~ha/research/publications/islped01.pdf
- http://www.vlsi.ee.upatras.gr/~sklavos/Papers02/DSP02_JPEG200.pdf
- http://www.etro.vub.ac.be/Members/munteanu.adrian/_private/Conferences/WaveletLosslessCompression_IWSSIP1998.pdf
- http://www.ii.metu.edu.tr/em2003/EM2003_presentations/DSD/benderli.pdf
- Digital Image Processing, Second Edition by Rafael C. Gonzalez and Richard E. Woods