IMAGE COMPRESSION AND ANALYSIS IN WIRELESS ADHOC NETWORKS

Hassan Azwar, Murtaz Naqvi and Dr. Usman Akram
Department of Electrical and Computer Engineering,
National University of Science and Technology,
Islamabad, Pakistan.
Hassan.azwar16@ce.ceme.edu.pk

Abstract—In today’s technological world as our use of and reliance on computers continues to grow, so too does our need for efficient ways of storing large amounts of data and due to the bandwidth and storage limitations, images must be compressed before transmission and storage. In this paper, we will be implementing lossy techniques for image compression. The techniques used will be Discrete Fourier Transform (DFT), Cosine Transform (CT), Discrete Wavelet Transform (DWT) and Principal Component Analysis (PCA). These four techniques are very much used in the field of Digital Signal Processing for processing many Signals and have many other applications. The software used in our project is MATLAB 7.0. We use this software to implement all the techniques with our algorithms and compare the result.

Keywords- Image Compression, Lossy, Decomposition

I. INTRODUCTION

Wireless sensor networks (WSN) is a self-organized distributed intelligent system comprising low-cost, low power, multifunctional sensor nodes that are small and communicate with each other in short distances. The development of such networks was originally motivated by military applications such as battlefield surveillance. Recently, a lot of research related WSN have been conducted and people have been realizing theirs unlimited applicability. For example, WSN can be used for data collection purposes in situations such as environment and habitat monitoring, healthcare applications, home automation, structural monitoring, and equipment diagnostics. However, WSN face many challenges, mainly caused by communication failures, limited storage capability and computational constraints and limited power supply. Therefore, the technology of WSN is requiring more extensive research and development before it becomes practical.

Wireless Sensor Networks (WSN) are a new tool to capture data from the natural or built environment at a scale previously not possible. A WSN typically have the capability for sensing, processing and wireless communication all built into a tiny embedded device. This type of network has drawn increasing interest in the research community over the last few years, driven by theoretical and practical problems in embedded operating systems, network protocols, wireless communications and distributed signal processing. In addition, the increasing availability of low-cost CMOS or CCD cameras and digital signal processors (DSP) has meant that more capable multimedia nodes are now starting to emerge in WSN applications. This has led to recent research field of Wireless Multimedia Sensor Networks (WMSN) that will not only enhance existing WSN applications, but also enable several new application areas.

II. IMAGE COMPRESSION TECHNIQUES

The compression of an image at the sensor node includes several steps. The image is first captured from the camera. It is then transformed into a format suitable for image compression. Each component of the image is then split into 8x8 blocks and each block is compared to the corresponding block in the previous captured image, the reference image. The next step of encoding a block involves transformation of the block into the frequency plane. This is done by using a forward discrete cosine transform (FDCT). The reason for using this transform is to exploit spatial correlation between pixels. After the transformation, most of information is concentrated to a few low-frequency components. To reduce the number of bits needed to represent the image these components are then quantized. This step will lower the quality of the image by reducing the precision of the components. The tradeoff between quality and produced bits can be controlled by the quantization matrix, which will define the step size for each of the frequency component. The components will also be ZigZag scanned to put the most likely non-zero components first and the most likely zero components last in the bit-stream. The next step is entropy coding. We use a combination of variable length coding (VLC) and Huffman encoding.

The techniques used for image compression are:

1. Discrete Wavelet Transform (DWT)
2. Cosine Transform (CT)
3. Principal Component Analysis (PCA)
4. Discrete Fourier Transform (DFT)
III. EXPLANATION OF TECHNIQUES

1. DISCRETE WAVELET TRANSFORM (DWT)

In the discrete wavelet transform, an image signal can be analyzed by passing it through an analysis filter bank followed by a decimation operation. This analysis filter bank, which consists of a low pass and a high pass filter at each decomposition stage, is commonly used in image compression. When a signal passes through these filters, it is split into two bands. The low pass filter, which corresponds to an averaging operation, extracts the coarse information of the signal. The high pass filter, which corresponds to a differencing operation, extracts the detail information of the signal. The output of the filtering operations is then decimated by two. A two-dimensional transform can be accomplished by performing two separate one-dimensional transforms. First, the image is filtered along the x dimension and decimated by two. Then, it is followed by filtering the sub-image along the y-dimension and decimated by two. Finally, we have split the image into four bands denoted by LL, HL, LH and HH after one-level decomposition as shown in figure 1-1.

The reconstruction of the image can be carried out by the following procedure. First, we will up-sample by a factor of two on all the four sub-bands at the coarsest scale, and filter the sub-bands in each dimension. Then we sum the four filtered sub-bands to reach the low-low sub-band at the next finer scale. We repeat this process until the image is fully reconstructed as shown in Figure 1-3.

2. COSINE TRANSFORMATION (CT)

A cosine transform (CT) expresses a sequence of finitely many data points in terms of a sum of cosine functions oscillating at different frequencies. The JPEG process is a widely-used form of lossy image compression that centers on the Cosine Transform. CT and Fourier transforms convert images from time-domain to frequency-domain to decorrelate pixels. The CT transformation is reversible. The CT works by separating images into parts of differing frequencies. During a
step called quantization, where part of compression occurs, the less important frequencies are discarded, hence the use of the term “lossy”. Then, only the most important frequencies that remain are used retrieve the image in the decompression process. Thus, reconstructed images contain some distortion; but as we shall soon see, these levels of distortion can be adjusted during the compression stage. The JPEG method is used for both color and black and white images. The most common DCT definition of a 1-D sequence of length $N$ is:

$$C(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos \left( \frac{\pi (2x + 1) u}{2N} \right),$$

For $u = 0, 1, 2, ..., N-1$.

The Inverse Transform similarly:

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) C(u) \cos \left( \frac{\pi (2x + 1) u}{2N} \right)$$

For $x = 0, 1, 2, ..., N-1$. In both equations, as above, $\alpha(u)$ is defined as:

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u \neq 0. \end{cases}$$

Cosine Transform (CT) is one of many transforms that takes its input and transforms it into a linear combination of weighted basis functions. These basis functions are commonly the frequency. The 2-D Cosine Transform is just a one-dimensional CT applied twice, once in the $x$ direction, and again in the $y$ direction. One can imagine the computational complexity of doing so for a large image. Thus, many algorithms, such as the Fast Fourier Transform (FFT), have been created to speed the computation. The following equation computes the $i$th, $j$th entry of the transformed image from the pixel values of the original image matrix. For the standard 8x8 block that JPEG compression uses, $N$ equals 8 and $x$ and $y$ range from 0 to 7. Therefore $D(i, j)$ would be as in Equation (3).

$$D(i,j) = \frac{1}{4} C(i)C(j) \sum_{x=0}^{7} \sum_{y=0}^{7} p(x,y) \cos \left[ \frac{(2x + 1)\pi i}{16} \right] \cos \left[ \frac{(2y + 1)\pi j}{16} \right]$$

Because the DCT uses cosine functions, the resulting matrix depends on the horizontal and vertical frequencies. Therefore, an image black with a lot of change in frequency has a very random looking resulting matrix, while an image matrix of just one colour, has a resulting matrix of a large value for the first element and zeroes for the other elements.

3. PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal component analysis (PCA) is also known as the Karhunen-Lo‘eve transformation (KLT) or the Hotelling transformation. PCA is an orthogonal transformation that seeks the directions of maximum variance in the data and is commonly used to reduce the dimensionality of the data (Haykin, 1999). The transformation matrix is a rotation that aligns the coordinate system to the direction of the principal components. Principal component analysis has been proven to be the optimal transformation for transform coding under certain conditions. We will always refer to the mean squared error as the distortion measure and assume independent optimal variable-rate scalar quantizers and entropy codes. One important property that arises from the objective of seeking maximum variances in the data is that the PCA transform yields a minimum reconstruction error (over all orthogonal transforms and for all stationary sources) from a truncated decomposition. This is equivalent to the fact that the truncation keeps a subspace with maximum variance.

The major goal of principal components analysis is:

- a) To reveal hidden structure in a data set.
- b) We may be able to identify how different variables work together to create the dynamics of the system
- c) To reduce the dimensionality of the data
- d) To decrease redundancy in the data.
- e) To filter some of the noise in the data.
- f) To compress the data
- g) Prepare the data for further analysis using other techniques

The image compression approach considered here is based on the idea that sub-images of input image can be described using relatively few features. Instead of storing the original image, the values of the features in the sub-images are stored (and then used for reconstruction).

Therefore, you should first extract the image and extracts non-overlapping $m \times n = mxn$ pixel blocks. After extraction, each block is converted into a column vector and then extracted. Now we use cov and eig function to find out the covariance vector and the eigen vector and its eigen values. Now in PCA
analysis we have select N largest eigenvalues and the corresponding eigenvectors, in this case N-10. Now we calculate the mean value of the eigen values and eigen vector using function which is the mean value (mean image). In our procedures, we are showing three output compressed images. Now plot the eigenvectors (eigen images) which correspond to the N largest eigenvalues in the same window. Now using the mean vector and the eigenvectors corresponding to the N largest Eigen values to compress the sub-images. When all the above steps are done its time to reconstruct each of the sub-images in its low-dimensional representation including the mean value. Now using the MATLAB function reconstruct or using reshape and uint8 to convert the reconstructed sub-images into a n × m pixel image again. Below here are given input and output of the compressed images using PCA based Image Compression.

4. DISCRETE FOURIER TRANSFORM (DFT)

The Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a frequency contained in the spatial domain image. The MATLAB documentation provides the following equations for FFT and IDFT:

\[
X(k) = \sum_{j=1}^{N} x(j) \omega_N^{(j-1)(k-1)}
\]

\[
x(j) = \frac{1}{N} \sum_{k=1}^{N} X(k) \omega_N^{-(j-1)(k-1)}
\]

\[
\omega_N = e^{(-2\pi i)/N}
\]

The DFT is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image, i.e. the image in the spatial and Fourier domain is of the same size. We applied the following procedure in our implementation.

a. We apply the FFT which is a built-in function in MATLAB. FFT cannot be directly applied on the whole image. Hence, we apply FFT on the image by two methods.

b. We apply FFT on each row of the Image matrix formed in MATLAB.

c. We apply FFT on each row of the Image matrix formed in MATLAB.

d. In the end, we apply the IFFT with different number of samples. Thus, reducing the image size but the quality of the image becomes very lossy.

IV. IMPLEMENTATION OF TECHNIQUES

1. DISCRETE WAVELET TRANSFORM (DWT)

We created a simple GUI in MATLAB in for the implementation of image compression using DWT. Figure 1-4 shows the GUI interface for the image compression. The GUI is very user friendly. By clicking the “BROWSE” button, one can select the image to be compressed. After that, type the decomposition level in the Textbox. Currently, our software can support decomposition level up to 3 without any errors. We will be later increasing the level for more compression. By pressing the “COMPRESS” button, the following result can be obtained.
The example shown in (figure 1-5) is a 3-level decomposition using DWT. However, you can see, the resulting image is of very low quality. It’s nearly impossible to say how the quantizing affects the quality of the resulting image, because the quality of our IDWT routine is so bad to begin with. I would imagine however that because the detail coefficients of a wavelet transform are all close to zero, they could be easily represented using very few bits…in some cases one or two. Because of this we can conclude that the wavelet transform may be the most suitable candidate for image compression yet found.

2. COSINE TRANSFORMATION (CT)
Our implementation was again in MATLAB with a simple friendly GUI. The following processes were done in this software,

a) The image is broken into 8x8 blocks of pixels.
b) Working from left to right, top to bottom, the CT is applied to each block.
c) Each block is compressed through quantization.
d) The array of compressed blocks that constitute the image is stored in a drastically reduced amount of space.
e) When desired, the image is reconstructed through decompression, a process that uses the Inverse Cosine Transform (ICT).

Following image shows the GUI of our software.

We applied our algorithm on very famous LINA picture. Here are some results.
Due to time factor in our project duration, we were only able to apply PCA on grayscale images. Although the software can load RGB images but it converts the image into grayscale and implement the PCA algorithm. The following results were obtained from a bitmap.

![Figure 3-2: Results Obtained by PCA Implementation](image)

3. PRINCIPAL COMPONENT ANALYSIS (PCA)

Like our previous implementations, PCA software GUI has the following screen.

![Figure 3-1: GUI for Principal Component Analysis](image)

4. DISCRETE FOURIER TRANSFORM (DFT)

The DFT is the sampled Fourier Transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image, i.e. the image in the spatial and Fourier domain is of the same size. We applied the following procedure in our implementation.

a) We apply the FFT which is a built-in function in MATLAB. FFT cannot be directly applied on the whole image. Hence, we apply FFT on the image by two methods.

b) We apply FFT on each row of the Image matrix formed in MATLAB.

c) We apply FFT on each row of the Image matrix formed in MATLAB.

d) In the end, we apply the IFFT with different number of samples. Thus, reducing the image size but the quality of the image becomes very lossy.

We created the following GUI for the implementation.
The algorithm and steps use in this method are very much like the method used in COSINE TRANSFORM. However, the quality here is very much lost compared with the original image. The next image shows different results with different compression factors. These results are only for ROW COMPRESSION method.

2. COSINE TRANSFORMATION (CT)
Although this technique is very simple to implement but at the same time very effective one as well. We can see that that compression by a factor of 2 doesn’t affect the picture very much but still it halves the file size. There is bit difference in the quality when compared between ROW and COLUMN compression. Comparing the results between both, ROW compression was much better.

3. PRINCIPAL COMPONENT ANALYSIS (PCA)
Principal components analysis is a popular tool for studying high-dimensional data. It relies on four major assumptions:

a) Linearity. This means that the only interaction among different signal sources is that they add together. If the strength of a combined signal were the product of the strengths of contributing signals, for instance, this would be a non-linear interaction and PCA would not work.

b) The interesting dynamics have the largest variances.

c) Mean and variance are sufficient statistics. Since PCA is designed around the covariance matrix of mean-centered data, the only statistics it considers are the mean and variance. If the data cannot adequately be described by its mean and variance (e.g. it is not Gaussian or exponentially distributed), PCA will be inadequate.

d) Orthogonal components. This is a particularly strong assumption made by PCA. It is this assumption that allows PCA to be computationally straightforward, but it is not a realistic assumption in many data sets.

4. DISCRETE FOURIER TRANSFORM (DFT)
Although same algorithm was applied for COSINE TRANSFORM and FOURIER TRANSFORM, but it can have been seen from the results that cosine transform is very superior than Fourier transform. We lost most of the transformation provides a way where an image can be decorrelated without introducing any artifacts or distortions. In doing so, the more important information about the image is concentrated in fewer coefficients whereas the less important information about the image is spread over many coefficients. Quantization can be used to remove the spatial redundancies present in these coefficients. A simple encoder such as RLE has shown high compression ratios when used together with the uniform dead-zone quantizer. Higher compression ratios can be achieved at the expense of the quality of the image by quantizing the image coarsely or by using a more sophisticated entropy encoder such as Huffman encoder. However, the computation time will increase when Huffman encoder is used. One way of overcoming the long computation time is to use the RLE before Huffman. Such a combination has shown an improvement over the Huffman encoder for all quantizers implemented.
information in Fourier transform. However, we can build much better algorithms for keeping more data and only destroying the repeating and similar bits by interpolating them.

VI. REFERENCES


