NATIONAL OPEN UNIVERSITY OF NIGERIA

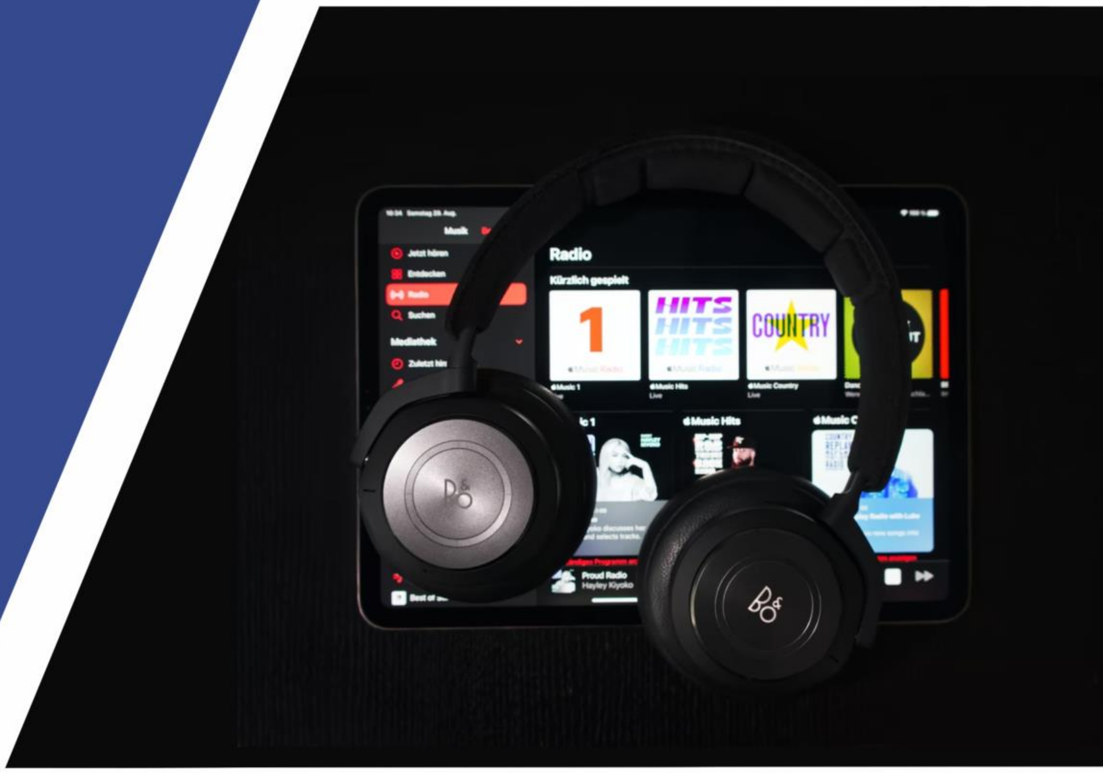
CIT 891

Advanced Multimedia Technology

Module 2

NATIONAL OPEN UNIVERSITY OF NIGERIA

CIT 891



Advanced Multimedia Technology

Module 1

CIT 891
Advanced Multimedia Technology
Module 2

Course Developer/Writer
Dr. Ikhu-Omoregbe, Nicholas, National Open University of Nigeria

Course Coordinator
A. A. Afolorunso, National Open University of Nigeria

Programme Leader
Prof. MoniOluwa Olaniyi, National Open University of Nigeria

Credits of cover-photo: Henry Ude, National Open University of Nigeria

National Open University of Nigeria - University Village, Plot 91, Cadastral Zone, Nnamdi Azikiwe Expressway, Jabi, Abuja-Nigeria

www.nou.edu.ng centralinfo@nou.edu.ng
oer.nou.edu.ng oerunit@noun.edu.ng OER repository
Published 2013, 2021 by the National Open University of Nigeria

Unit 1 Overview of Current Techniques in Image/Video Compression

1.0 Introduction

Compression is the process of transforming information from one representation to another, smaller representation from which the original, or a close approximation to it, can be recovered. Compression and decompression processes are often referred to as encoding and decoding. Data compression has important applications in the areas of data storage and data transmission. The data compression process is said to be lossless if the recovered data are certain to be like to the source; otherwise the compression process is said to be lossy. Lossless compression techniques are essential for applications involving textual data. Other applications, such as those involving voice and image data, may be sufficiently flexible to allow controlled degradation in the data.

2.0 Objectives

At the end of this unit, you should be able to:

- define multimedia data compression
- explain some compression techniques
- explain how a compression systems work
- list some advantages and disadvantage of data compression.

3.0 Main Content

3.1 Foundation for Multimedia Compression

In laying the foundation for multimedia compression, we would first examine two reasons that necessitate multimedia compression.

1.Perceptual Redundancy

2.Demand on Computing Resources

Perceptual Redundancy
In digital representation of multimedia (still image, video, voice, animation) data, significant amount of unnecessary or redundant information are used as far as the human perceptual system is concerned. By human perceptual system, we are referring to our eyes and ears. It is of interest to note that when observing a natural image, the human perceptual system does not easily notice the variation in values of the neighboring pixels in the smooth region of the object and at times sees them as very similar. Likewise, the successive frames in a motionless or slowly changing science in a video are very similar and redundant to the eyes of a viewer. In the same light, of discussion, there are some audio data that are beyond the human audible frequency range and therefore considered to be irrelevant for all practical purposes. These are all indications that not all data should be allowed to go through transmission channels or stored on memory devices at all since they are not perceived by human beings. Thus, there are data in audio-visual signals that cannot be perceived by the human perceptual system. This is what is meant by perceptual redundancy.

Therefore, in the context of data compression, the aim is usually to reduce the redundancies in data representations in order to decrease data storage requirements and hence communications cost. Any efforts to reduce the storage requirements is equivalent to increasing the capacity of the storage medium and hence communication bandwidth. This is one of the reasons why the development of efficient compression will continue to be a design challenge for future communication systems and advanced multimedia applications.

Demand on Computing Resources
Data compression is best achieved by first converting analog signals in which multimedia data may exist to digital signals. Though, there are quite a lot of advantages in converting analog signals to their digital equivalents, the need for large bits for storage, high bandwidth and time for data transmission are issues to be tackled. For example:
a high-quality audio signal requires approximately 1.5 megabits per second for digital representation and storage.

- a television-quality low-resolution colour video of 30 frames per second with each frame containing 640 x 480 pixels (24 bits per color pixel) would need more than 210 megabits per second of storage.
- a digitised one-hour colour movie would require approximately 95 gigabyte of storage.
- the storage requirement for high-definition television (HDTV) of resolution 1280 x 720 at 60 frames per second would certainly be far greater.
- a digitised one-hour colour movie of HDTV-quality video would require about 560 gigabyte of storage.
- a small document collection in electronic form in a digital library system may easily require to store several billion characters.
- the total amount of information spread over the internet is mind bogging .etc.

Even when analog signal are successfully converted to digital signals, the transmission of these digital signals through limited bandwidth communication channel poses greater challenge. Though, the cost of storage has decreased drastically over the past few decades as a result of advances in microelectronics and storage technology the requirement of data storage and transmission for many multimedia applications has grown so explosively in recent times to outpace this achievement.

The values in table 3.1 shows a comparative demand on resources (disk space, transmission bandwidth, and transmission time needed to store and transmit) some multimedia data. The prefix kilo- denotes a factor of 1000 rather than 1024.

Table 3.1: Comparative Resource Demand of Multimedia Data

Multimedia Data	Size/ Duration	Bits/ Pixel or Bits/Sample	Uncompressed Size (B for bytes)	Transmission Bandwidth	Transmission Time (Using a 28.8K Modem
A page of text	11" x 8.5"	Varying resolution	4-8 KB	32-64 Kb/page	1.1 – 2.1 sec
Telephone quality	10 sec	8 bps	80 KB	64 Kb/sec	22.2 sec

speech					
Grayscale Image	512 x 512	8 bpp	262 KB	2.1 Mb/image	73 sec
Color Image	512 x 512	24 bpp	786 KB	6.29 Mb/image	219 Sec
Full-motion Video	640 x 480, 1 min (30 frames/sec)	24 bpp	1.66 GB	221 Mb/sec	5 days 8 Hrs

It is obvious from the examples above that adequate computing resource must be available for the storage and transmission of multi-media data. This leaves us with no other option than to compress multimedia data before storing or transmitting it, and decompressing it at the receiver for play back. For example, with a compression ratio of 64:1, the space, bandwidth, and transmission time requirements can be reduced by a factor of 64, with acceptable quality.

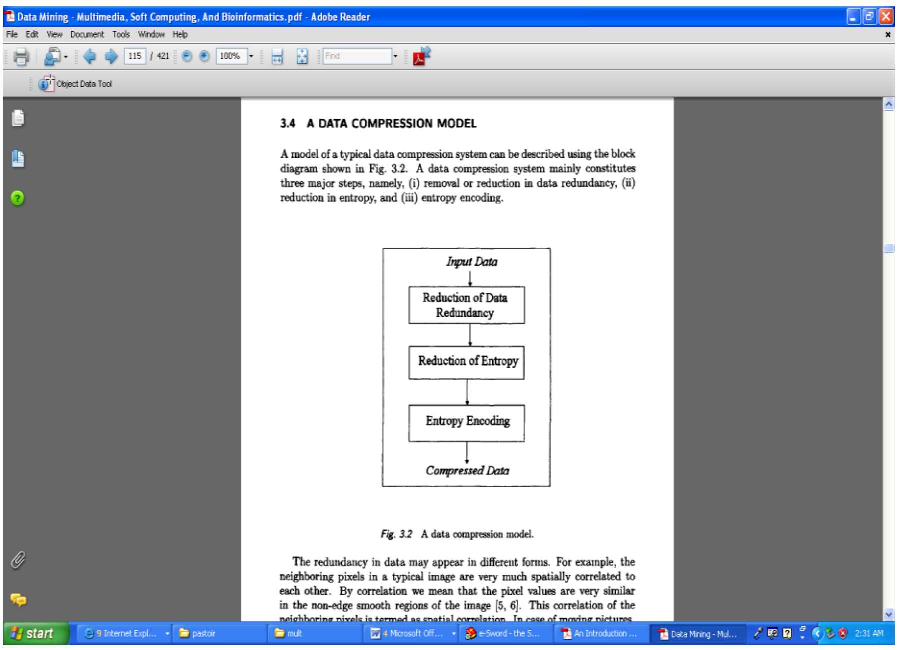
3.2 A Data Compression Model

Figure 1.1 represents a model for typical data compression system. The diagram models a three step process for data compression. These steps are:

1 removal or reduction in data redundancy

2 reduction in entropy, and

3 entropy encoding.


3.4 A DATA COMPRESSION MODEL

A model of a typical data compression system can be described using the block diagram shown in Fig. 3.2. A data compression system mainly constitutes three major steps, namely, (i) removal or reduction in data redundancy, (ii) reduction in entropy, and (iii) entropy encoding.

Fig. 3.2 A data compression model.

The redundancy in data may appear in different forms. For example, the neighboring pixels in a typical image are very much spatially correlated to each other. By correlation we mean that the pixel values are very similar in the non-edge smooth regions of the image [5, 6]. This correlation of the

Fig 1.1: A Compression Model

The redundancy in data may appear when the neighbouring pixels in a typical image are very much spatially correlated to each other. By correlation we mean that the pixel values are very identical in the non-edge smooth regions of the image. This correlation of the neighbouring pixels is termed as spatial correlation. In case of video or animation the consecutive frames could be almost similar, with or without minor displacement, if the motion is slow. The composition of words or sentences in a natural text follows some context model, based on the grammar being used. Similarly, the records in a typical numeric database may have some sort of relationship amongst the atomic entities which comprise each record in the database. For natural audio or speech data, there are usually rhythms and pauses in regular intervals. All these redundancies in data representation can be reduced in order to realise compression.

Removal or reduction in data redundancy is usually achieved by a transformation process that converts the source data from one form of representation to another, in order to decorrelate the spatial information redundancies present in the data. Amongst the popular techniques used for spatial redundancy reduction is prediction of data samples using some model, transformation of the original data from spatial to frequency domain using methods such as Discrete Cosine Transform (DCT), decomposition of the original dataset into different sub bands as in Discrete Wavelet Transformation (DWT), etc. The fact remains that, this spatial redundancy reduction potentially yields more compact representation of the information in the original dataset, in terms of fewer transformed coefficients or equivalent, and hence makes it amenable to represent the data with less number of bits in order to achieve compression.

The next major stage in a lossy data compression system is "quantisation." This technique is applied on the decorrelated data, in order to further reduce the number of symbols or coefficients, by masking irrelevant parts and retaining only the significant details in the data. The outcome of this, is a reduction in entropy of the data, and hence makes it further open to compression by allocating less number of bits for data transmission or storage. The reduction in entropy is realised by dropping irrelevant details in the transformed data and preserving fewer significant symbols only. Taking a look at the human visual system for example, when an image is transformed in frequency domain, the high-frequency transformed coefficients can be actually dropped because the human vision system is not sensitive to these. By retaining a smaller number of transformed coefficients in the useful low-frequency range, we can ascertain the fidelity of the reconstructed image. In actual fact, the quality of the reconstructed data is majorly determined by the nature and amount of quantisation. The quantised coefficients are then lossless encoded, using some entropy encoding scheme to compactly represent the quantised data for storage or transmission. Since the entropy of the quantised data is less than that of the source, it can be represented by a fewer number of bits relative to the source data set and hence we realise compression. The decompression system is just an inverse process to reconstruct the data.

In the next two sections we shall describe the principles of compression of still images and video.

3.3 Principles of Still Image Compression

. The general model of still image compression framework can be represented by the block diagram depicted in figure 1.2 The statistical analysis of a typical image indicates that a strong correlation usually exist among the neighboring pixels. This leads to redundancy of information in the digital representation of the image. The redundancy can be significantly removed by transforming the image with some sort of preprocessing in order to achieve the desired compression. On a general note, still image compression techniques rely on two fundamental redundancy reduction principles. These are:

- Spatial redundancy
- Statistical redundancy reduction.

By spatial redundancy, we mean the similarity of neighbouring pixels in an image. It can be reduced by applying decorellation or transformation techniques such as predictive coding, transform coding, sub band coding, etc.

The statistical redundancy reduction is also known as entropy encoding. The essence is to further reduce, the redundancy in the decorrelated data by using variable-length coding techniques such as Huffman Coding, Arithmetic Coding, etc.

These entropy encoding technique, some of which are discussed in later sections of this study material allocate the bits in the codeword in such a way that more probably appearing symbols are represented with a smaller number of bits compared to the less probably appearing pixels, which helps to achieve the desired compression.

Taking a look at the figure once more, the decorellation or preprocessing block is the step required to reduce the spatial redundancy of the image pixels due to strong correlation among the neighboring pixels. In lossless coding mode, this decorrelated image is directly processed by the entropy encoder to encode the decorrelated pixel using a variable-length coding technique.

On the other hand, if the lossy compression mode is what you desire, the decorrelated image is subjected to further preprocessing in order to mask or remove irrelevant details depending on the nature of the application of the image and its reconstructed quality requirements. This process of masking is known as quantisation process.

Finally the decorrelated and quantised image pixel is subjected through the entropy encoding process to further compact its representation using variable-length codes to produce the compressed image

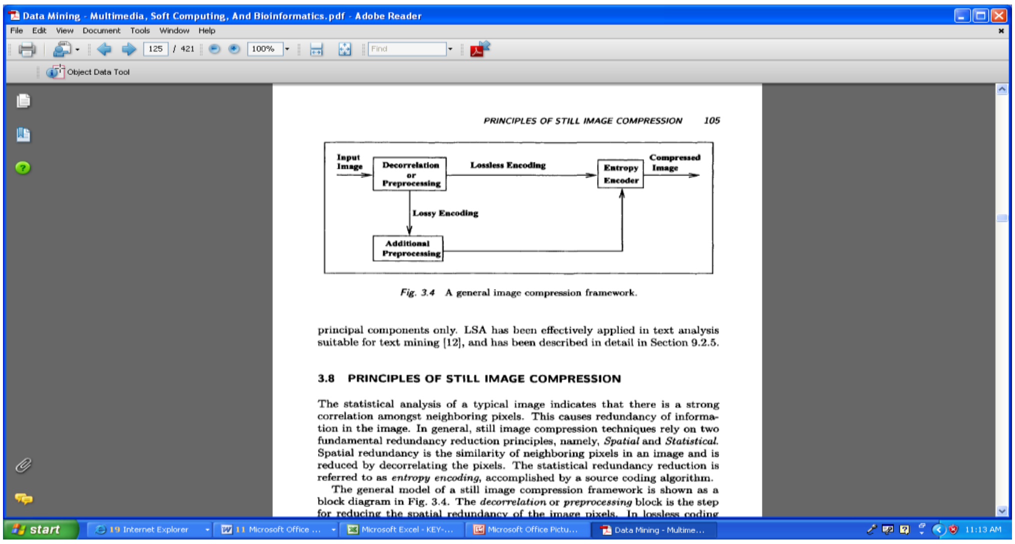


Fig 1.2: Still Image Compression Model

3.4 Principles of Video Compression

The principle of still image compression is very similar to that of video compression. Video is simply sequence of digitised picture. Video can also be referred to as moving picture. The terms “frame” and “pictures” are used interchangeably in relation to video. However, we shall use the term frame in relation to videos except where particular standard uses the term picture.

In principle, one way to compress video source is to apply any of the common algorithms such as JPEG algorithm independently to each frame that makes up a video. This approach is also known as moving JPEG or MPEG. For now typical compression ratios of about 29:1 obtained with JPEG are not large enough to produce the compression ratio needed for multimedia applications.

In practice, in addition to the spatial redundancy present in each frame considerable redundancy is often present between a set of frame since, in general, only a small portion of each frame is involved with any motion that is generally, only a small portion of each frame is involved with any motion that is taking place. For an example, consider the movement of a person’s lip or eye in a video telephony application.

To exploit the high correlation between successive frames, we adopt a technique which predicts the content of many of the frames. As we shall describe, this is based on a combination of a preceding – and in some instances succeeding – frame. Instead of sending the original video as a set of individually compressed frames, just a selection is sent in the form and for the remaining frames, only the difference between the actual frame content and the predicted frame content is sent.

How well any movement between successive frames is estimated will go a long way in determining the accuracy of the prediction operation. The operation is known as motion estimation, and since the estimation process is not exact (just an estimation), therefore, more information must also be sent to indicate any small differences between the predicted and actual positions of the moving segment involved. The latter is known as motion compensation. We shall further discuss this in Module four of the course.

3.5 Classification of Compression Algorithms

Data compression can be accomplished by applying one or more algorithms to multimedia source data. In this section we shall briefly describe the most common ones.

3.5.1.Run Length Encoding

This is a very simple form of data compression in which runs of data (that is, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as the original run. The idea behind Run length encoding (RLE) is to encode strings of zeros and ones by the number of repetitions in each string. RLE has become a standard in facsimile transmission. For a binary image, there are many different implementations of RLE; one method is to encode each line separately, starting with the number of 0's. So the following binary image:

0 1 1 0 0 0

0 0 1 1 1 0

1 1 1 0 0 1

0 1 1 1 1 0

0 0 0 1 1 1

1 0 0 0 1 1

would be encoded as (123) (231)(0321)(141)(33)(132)

Another method is to encode each row as a list of pairs of numbers; the first number in each pairs given the starting position of a run of 1's and the second number its length. So the above binary image would have the encoding

(22) (33) (1361)(24)(43)(1152)

Furthermore, another implementation would be to encode a sequence or run of consecutive pixels of the same color (such as black or white) as a single codeword. For example, the sequence of pixels

88 88 88 88 88 88 88 88

could be coded as

8 88 (for eight 88's)

Run length encoding can work well for bi-level images (e.g. black and white text or graphics) and for 8 bit images. Run length encoding does not work well for 24 bit natural images in general. Runs of the same color are not that common. Run-length encoding performs lossless data compression and is well suited to palette-based iconic images. It does not work well at all on continuous-tone images such as photographs, although JPEG uses it quite effectively on the coefficients that remain after transforming and quantizing image blocks. Run-length encoding is used in fax machines (combined with other techniques into Modified Huffman coding. It is relatively efficient because most faxed documents are mostly white space, with occasional interruptions of black.

3.5.2 Huffman Coding

Run length coding and Huffman coding are referred to as source coding. From the information theoretic perspective, source coding can mean both lossless and lossy compression. However, researchers often use it to indicate lossless coding only. In the signal processing community, source coding is used to mean source model based coding.

The Huffman coding was invented in 1952 by D. A. Huffman as a technique to produce the shortest possible average code length, given the source symbol set and the associated probability of occurrence of the symbols. The Huffman coding technique is based on the following two observations regarding optimum prefix codes.

1. The more frequently occurring symbols can be allocated with shorter codewords than the less frequently occurring symbols.
2. The two least frequently occurring symbols will have codewords of the same length, and they differ only in the least significant bit.

The mean of the length of these codes is closed to the entropy of the source. For example, if we have *m* source symbols $\{S_i, S_{2...}S_m\}$ with associated probabilities of occurrence $\{P_i, P_{2,...,}P_m\}$. Using these probability values, we can generate a set of Huffman codes of the source symbols. The Huffman codes can be mapped into a binary tree, popularly known as the Huffman tree depicted in figure1.3

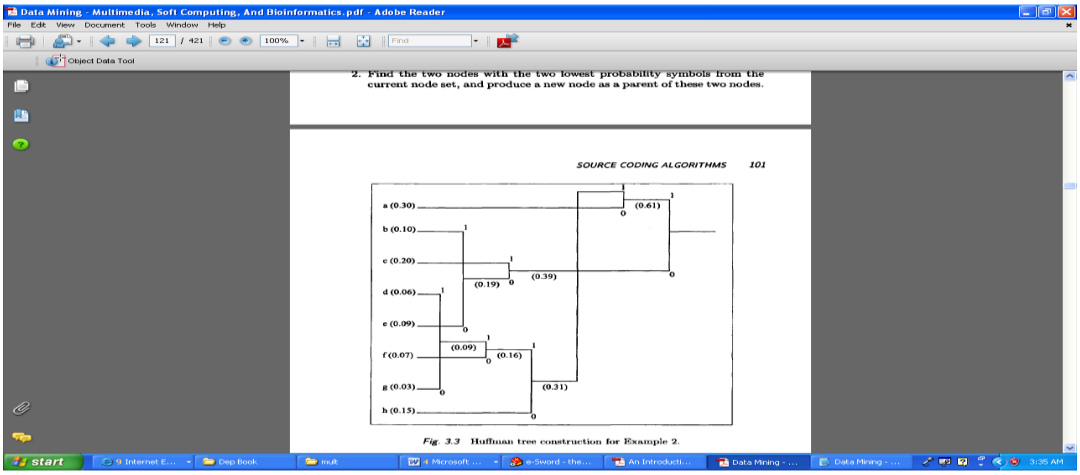


Fig 1.3: An example of Huffman Tree form by Appling Huffman Algorithm

Source [Mitra,S.& Tinkuacharya. (2003)]

3.5.3 Predictive Coding

This coding technique was originally proposed by Cutler in 1952. It should be clear to you from our preceding discussions, that the adjacent pixels in a typical image are highly correlated. This makes it possible to extract a great deal of information about a pixel from its neighboring pixel values. In predictive coding, a pixel value is predicted by a set of previously encoded neighbouring pixels. The difference between the pel and its prediction forms the signal to be coded. It is obvious that, the better the prediction, the smaller the error signal and the more efficient the coding system. The difference between the pel and its prediction forms the signal to be coded. Invariably, the better the prediction, the smaller the error signal and the more efficient the coding system. We represent this coding scheme with a block diagram in Figure 1.4

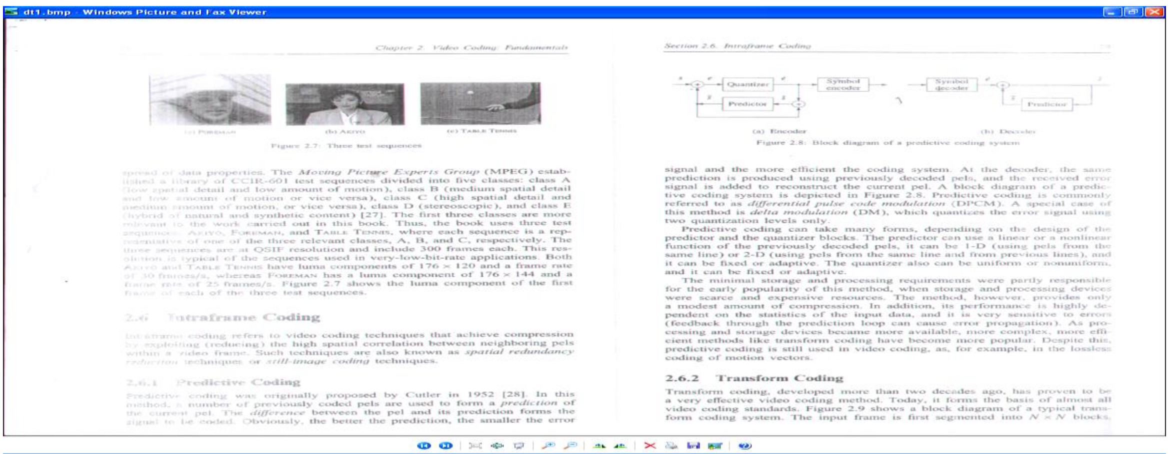


Fig. 1.4 :A Predictive Coding System

At the decoder, the same prediction is created using previously decoded pixels and the received errors signal is added to reconstruct the current pixels. Predictive coding is also known as differential pulse code modulation (DPCM). A specialised case of this technique is delay modulation (DM), which quantises the error signal using two quantisation level only.

Predictive coding can be realised in different ways depending on the design of the predictor and the quantiser blocks. For example, the predictor can use a linear or nonlinear function of the previously decoded pel, it can be 1-dimensional, that is, using pixels from the same line or 2-dimensional using pixels from the same line and from previous lines, and it can be fixed or adaptive. For the quantiser, it can be made to be uniform or nonuniform, and it can be fixed or adaptive.

The predicting coding technique requires a small amount of storage and minimum processing to achieve the desired compression on multimedia data. These were partly responsible for the early popularity of this method, when storage and processing devices were scarce and expensive resources. This approach provides only a modest amount of compression. In addition, its performance relies on the statistics of the source data, and it is very sensitive to errors as feedback through the prediction loop can cause error propagation. With the advancement in computing and memory technology that has led to the availability of cheap storage and faster processor, more complex, more efficient methods, like transform coding have become more popular. Despite this, there are instances where predictive coding is still used in video coding.

3.5.4 Transform Coding

In predictive coding, the coding process takes place pixel by pixel. Transform coding is an effective way of coding a group of spatially correlated pixels. This technique exploits the fact that the energy of most natural images is largely intense in the low-frequency regions. An appropriate transformation technique produces fewer number of correlated transformed coefficients as compared to the source image, and a large amount of image information is concentrated in these fewer correlated transformed coefficients. As a result, we can get rid of or mask the insignificant transformed coefficients, mainly consisting of the high-frequency components, using a suitable quantisation technique without affecting the

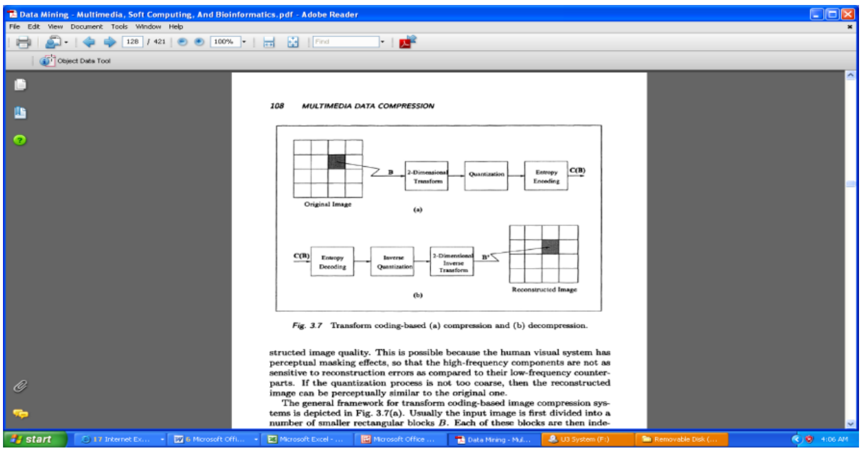


Fig 1.5 : Transform Coding

reconstructed image quality. This is possible because the human visual system has perceptual masking effects. As a result, the high-frequency components are not as perceptive to reconstruction errors as compared to their low-frequency counterparts. If the quantisation process is not too coarse, then the reconstructed image can be perceptually similar to the source.

The general framework for transform coding-based image compression systems is captured by Figure1.5. A close observation shows that the input image is first divided into a number of smaller rectangular blocks *B*. Each of these blocks is then independently transformed, using a choice linear transformation method. The transformed coefficients are quantised and entropy encoded into bit-stream *c(B)* in order to achieve compression. In fact, the process of decompression entails that the compressed bit stream *c(B)* is first entropy-decoded to generate the quantised coefficients. (See figure 1.5 above). This is followed by inverse quantisation in order to generate an approximation of the transformed coefficients. The inverse transformation is applied on these coefficients to reconstruct the image block *B'*. The composition of the reconstructed image blocks forms the reconstructed image, as shown in the diagram. The choice of a transformation technique is a major decision in this method. The desire is to transform from the spatial domain to another domain (usually frequency domain) is to represent the data in a more compact form in the transformed domain., using the right transformation technique helps to minimise the mean squared Also error of the reconstructed image. The Karhunen-Loeve Transform (KLT) is a ready choice since it has been proven to be optimal in terms of the compaction efficiency, by being able to represent images using few principal components containing a significant portion of the image information. However, there are no fast algorithms for practical implementation of KLT, because of its dependency on the input source signal. As a result, other less efficient transform such as the Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Discrete Sine Transform (DST), and Discrete Hadamard Transform (DHT), etc, are readily used in digital image compression. Amongst these examples, the DCT is the most popular block-based transform, because its performance is very close to that of KLT and a number of fast algorithms exist for DCT. Discrete cosine transform is a lossy compression algorithm that samples an image at regular intervals, analyzes the frequency components present in the sample, and discards those frequencies which do not affect the image as the human eye perceives it. Finally, DCT is the basis for most of the image and video compression algorithms, especially the still image compression standard JPEG in lossy mode and the video compression standards MPEG-1, MPEG-2, MPEG-4, H.263, etc.

3.5.5 Vector Quantisation

Vector quantisation (VQ) is a blocked-based spatial domain method that has become very accepted ever since the early 1980s. The way it works is that the input image data is first decomposed into k-dimensional input vectors. The generation of the input vector can be done in a number of ways; they could be the pel values themselves or to some suitable transformation of them. For example a k = M x M block of pels can be ordered to form a k-dimensional input vector $s = [s_1 \ldots s_k]^T$. Then the k-dimensional space $\Re^K$ is decomposed into N region of cells, Ri. Any input vector that falls into the cell Ri is then represented by a representative codevector $ri=[r_1,\ldots rk]^T$. The set of codevectors €={r1, ….rN} is called the codebook. Therefore, the aim of the encoder is to locate the codevector r_i that best matches the input vector according to some distortion measure $d(s,r_i)$. The index i of this code vector is then transmitted to the decoder using at most $I=\log_2 N$ bits. At the decoder, this index is used to lookup the codevector from a matching codebook. See figure 1.6 below.

Input
Reconstructed Image

image

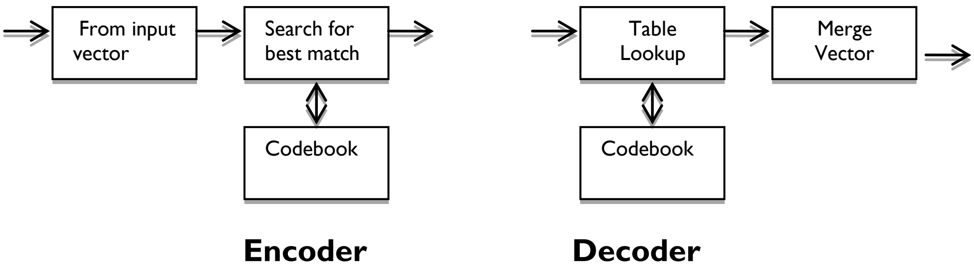


Fig.1.6: Vector Quantisation System

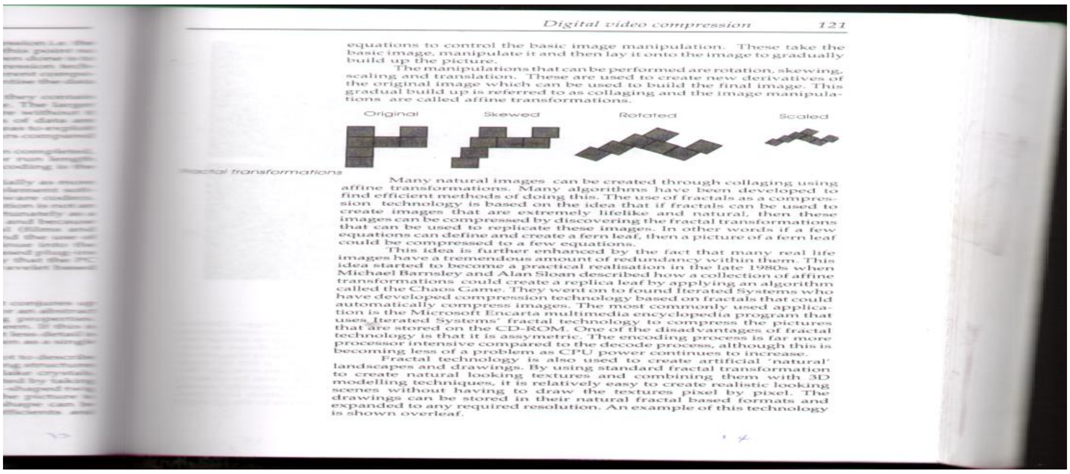
Source [Jankerson,D. *et al*. (2003)]

VQ realises compression by using a code with relatively few codevectors compared to the number of possible input vector. The resulting bit rate of a VQ is denoted by I/k bits/pel. In theory, as k tends to infinity, the performance of VQ tends towards the rate-distortion bound. Values of K used in typical systems are k= 4 x 4 and N=1024. When k becomes large, it becomes difficult to store and search the codebook.

A very important problem in VQ is the design of the codebook. The Linde-Buzo-Gray (LBG) algorithm helps to address this. The LBG algorithm computes a codebook with a locally minimum average distortion for a given training set and given codebook size. There are many variants of VQ. It has a performance that rivals that of transform coding discussed earlier in section 3.5.4 of this unit. Although the decoder complexity is insignificant in VQ, the high complexity of the encoder and the high storage requirements of the method limit its use in practice. Like transform coding, VQ suffers from blocking artifacts at very low bit rates.

3.5.6 Fractal Compression

Bernoit Mandelbroth was the first to use the word fractal to describe a fractured structure which is made up of many similar looking structures and forms. This often occurs in nature in snowflakes crystals trees and river deltas. These kinds of images can be created by taking a simple structure and using it as a building block. By this same principle a basic shape can be reused to create a new image using a set of coefficient and equations to control the processing of the basic image. This process takes the basic image, manipulate it and then lay it onto the image to gradually build up a desired picture.


Digital video compression 121

equations to control the basic image manipulation. These take the basic image, manipulate it and then lay it onto the image to gradually build up the picture.

The manipulations that can be performed are rotation, skewing, scaling and translation. These are used to create new derivatives of the original image which can be used to build the final image. This gradual build up is referred to as collaging and the image manipulations are called affine transformations.

Many natural images can be created through collaging using affine transformations. Many algorithms have been developed to find efficient methods of doing this. The use of fractals as a compression technology is based on the idea that if fractals can be used to create images that are extremely lifelike and natural, then these images can be compressed by discovering the fractal transformations that can be used to replicate these images. In other words if a few equations can define and create a fern leaf, then a picture of a fern leaf could be compressed to a few equations.

This idea is further enhanced by the fact that many real life images have a tremendous amount of redundancy within them. This idea started to become a practical realisation in the late 1980s when Michael Barnsley and Alan Sloan described how a collection of affine transformations could create a replica leaf by applying an algorithm called the Chaos Game. They went on to found Iterated Systems who have developed compression technology based on fractals that could automatically compress images. The most commonly used application is the Microsoft Encarta multimedia encyclopedia program that uses Iterated Systems' fractal technology to compress the pictures that are stored on the CD-ROM. One of the disadvantages of fractal technology is that it is assymetric. The encoding process is far more processor intensive compared to the decode process, although this is becoming less of a problem as CPU power continues to increase.

Fractal technology is also used to create artificial 'natural' landscapes and drawings. By using standard fractal transformation to create natural looking textures and combining them with 3D modelling techniques, it is relatively easy to create realistic looking scenes without having to draw the textures pixel by pixel. The drawings can be stored in their natural fractal based formats and expanded to any required resolution. An example of this technology is shown overleaf.

Fig.1.7:Fractal Transformation

Source Heath, S.(1996).

Using figure 1.7 as our example, the manipulations that can be performed are rotation, skewing, scaling and translation. These are used to create new derivatives of the original image which can be used to build the final image. This gradual build up is referred to as collaging and the image manipulation are called affine transformation.

Many images that appear in nature can be created through collaging using affine transformation. The use of fractals as a compression technology is principally based on the idea that if fractals can be used to create images that are extremely lifelike and natural, then these images can be compressed by discovering the fractal transformations that can be used to replicate these. Invariably, if a few equations can be defined and create a fern leaf, then a picture of a fern leaf could be compressed to a few equations. This idea which is behind fractal compression is further enhanced by the fact that many real life images have tremendous amount of redundancy within them.

The most popularly used application of fractal compression is in the compression of images of Microsoft Encarta multimedia encyclopedia programs. One of the disadvantages of fractal technology is that it is asymmetric. The encoding process is far more process intensive compared to the decoding process, although this is fast becoming a non issue as a result of emergence of high speed processing system.

3.5.7. Wavelet Coding

Signal representation using Fourier series in terms of the sinusoids has been well known for more than a century as an effective means of processing stationary as well as non stationary signals. Image compression techniques using Discrete Wavelet Transform (DWT) have received wide attention in recent years. Wavelet coding is a transform coding technique that is not limited to the block-based implementation only. Usually the wavelet transform is performed on the whole image. Wavelet transform decomposes the input signal into low-frequency and high-frequency sub bands. Using a separable two dimensional filtering function, two-dimensional DWT can be computed by applying one-dimensional DWT row-wise and column-wise independently. The technique for wavelet coding is depicted in figure 1.8 below

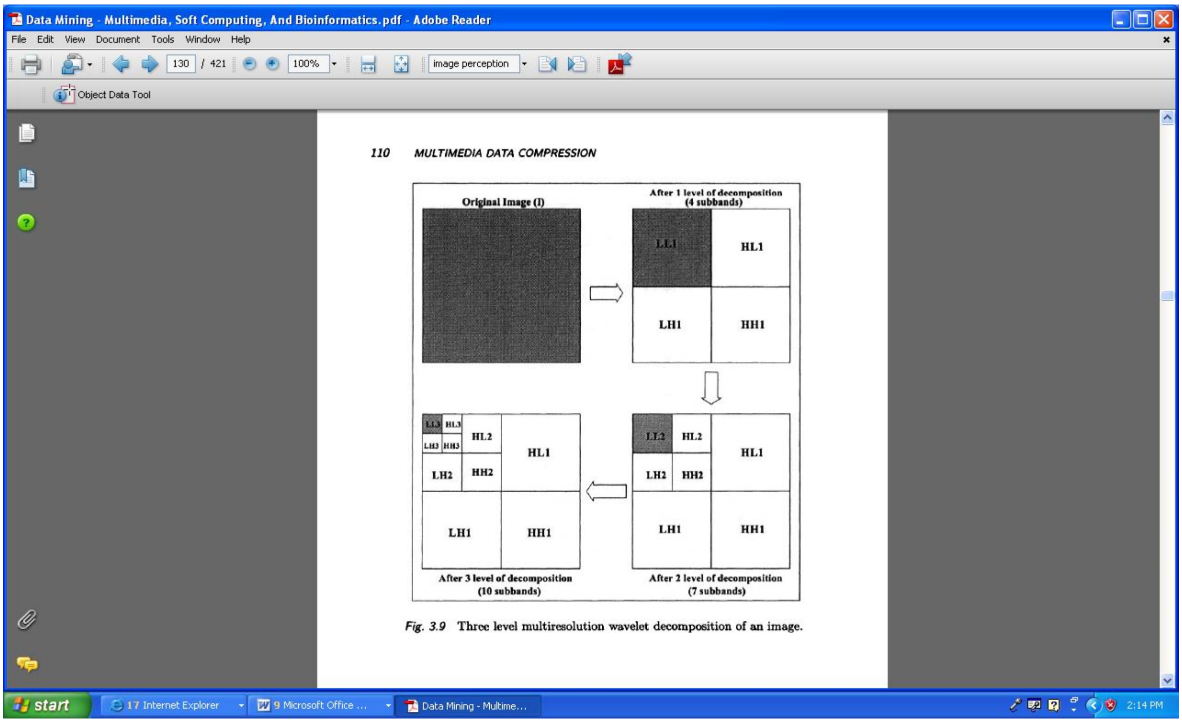


Fig.1.8 : Wavelet Coding

The diagram shows a hierarchical wavelet decomposition of an image into ten subbands which is achieved after three levels of decomposition. As depicted in figure 1.8, after the first level of decomposition, the original image is decomposed into four subbands *LI/1*, *HL1, LHl,* and *HHl.* The *LLl* subband is the low-frequency subband which can be considered as a 2:1 subsampled (horizontally and vertically) version of the original image, and its statistical characteristic is similar to the original image which is shown by the shaded regions in the diagram. In this case, *HL1,,LH1,* and *HHl* are called the high-frequency subbands, where *HLl* and *LH 1* correspond to the horizontal and vertical high frequencies, respectively. *HHl* constitutes the high frequencies that are not in either horizontal or vertical orientations. Each of these spatially oriented (horizontal, vertical, or diagonal) subbands mostly contain information of local discontinuities in the image, and the massive energy in each of the high-frequency subbands are more in the neighborhood of areas which correspond to edge activity in the source image. Since the low-frequency subband *LLl* has comparable spatial and statistical characteristics as the source image, it can be further decomposed into four subbands LI/2, *HL2, LH2,* and *HH2.* By repeating the same process for decomposition in I/L2, the original image is decomposed into 10 subbands I/L3, *HL3, LH3, HH3, HL2, LH2, HH2, HLl, LHl,* and *HHl* after three levels of pyramidal multiresolution subband decomposition, as shown in the diagram. The same approach is taken to decompose LL3 into higher levels. Once the right wavelet filters and quantisation strategy is chosen for subband we can guarantee that good compression performance will be achieved. Each decomposed subband may be encoded separately using a suitable coding scheme. We can allocate different bit-rates to different subbands. Because of the hierarchical nature of the subbands in wavelet decomposition, a smaller number of bits need to be allocated to the high-frequency subbands in a lower level as compared to the high-frequency subbands in upper levels. This helps to ascertain a proper reliability of the reconstructed image and thereby achieves good compression. Experimental results show that we can even allocate zero bits to the *HHl* subband and still guarantee quality of reconstructed image from natural images.

3.6 Advantages of Data Compression

The main advantage of compression is that it reduces the data storage requirements. It also offers an attractive approach to reduce the communication cost in transmitting high volume of data over long distances. The reduction in data rates can make the quality of multimedia presentation through limited-bandwidth communication channels to be increased by compression. Also because of the reduced data rates offered by compression techniques, computer network and Internet usage are becoming more and more image and graphics friendly, rather than being just data – and text-centric phenomena. In fact, many of the creative applications such as digital library, digital achieving, video teleconferencing, telemedicine, and digital entertainment we see today are as a result of high-performance compression. Other secondary advantages of compression are:

Data compression may enhance the database performance because more compressed records can be stored in memory at any time. This potentially increases the probability that a record being searched will be found in the main memory.

Data security is another area, in which compression is useful. The encryption parameters can be compressed before transmitting them separately from the compressed database files to restrict access of propriety information. An extra level of security can thus be accomplished by making the compression and decompression processes totally transparent to unauthorised users.

The rate of input-output operations in a computing device can be greatly increased due to shorter representation of data.

Data compression no doubt can reduce the cost of backup and recovery of data in computer systems by storing the enabling large database files in compressed form to be stored easily.

Self-Assessment Exercise

i. Discuss the Run Length Encoding algorithm.
ii. Explain the term lossy compression**.**

3.7 Disadvantages of Data Compression

Data compression offers quite a good number of advantages and has opened diverse areas of opportunities for multimedia industries. It however, also offers some disadvantages which are as follows:

Data compression usually leads to reduced reliability of the data records. For example a single bit error in compressed code will cause the decoder to miss-interpret all subsequent bits producing incorrect data. You may also want to consider the transmission of very sensitive compressed data (e.g. medical information) through a noisy communication channel (such as wireless media). This could become risky because the burst error introduced by the noisy channel can destroy the transmitted data.

Similar to above problems associated with compression is the disruption of data properties, since the compressed data is different from the original data. For example, sorting and searching schemes into the compressed data may be inapplicable as the lexical ordering of the original data is no longer preserved in the compressed data.

Furthermore, the extra overhead incurred by encoding and decoding processes is one of the most serious limitations of data compression, which discourages its usage in some areas (e.g., in many large database applications). The extra overhead is usually required in order to uniquely identify or interpret the compressed data.

In many hardware and systems implementations, the extra complexity added by data compression can increase the system's cost and reduce the systems' efficiency, especially in the areas of applications that requires very low-power VLSI implementations.

4.0 Conclusion

The data compression process is said to be lossless if the recovered data are assured to be identical to the source; otherwise the compression process is said to be lossy. The multimedia data type determines the kind of technique to be used. For example compressing an image is significantly different than compressing raw binary data. Of course, general purpose compression programs can be used to compress images, but the result is less than optimal. This is because images have certain statistical properties which can be exploited by encoders specifically designed for them. Also, some of the finer details in the image can be sacrificed for the sake of saving a little more bandwidth or storage space. This also means that lossy compression techniques can be used in this area.

Lossless compression techniques are recommended when compressing data which, when decompressed, are expected to be an exact replica of the original data. This is the case when binary data such as executables, documents etc. are compressed. They need to be exactly reproduced when decompressed. On the other hand, images (and music too) need not be reproduced 'exactly'. An approximation of the original image is enough for most purposes, as long as the error between the original and the compressed image is acceptable.

Further more, in lossless compression schemes, the reconstructed image, after compression, is numerically similar to the original image. However lossless compression can only be achieved at modest amount of compression. An image reconstructed following lossy compression contains degradation compared to the original. Often this is because the compression scheme completely discards redundant information. However, lossy schemes are capable of achieving much higher compression. Under normal viewing conditions, no visible loss is perceived (visually lossless).

5.0 Summary

A common characteristic of most images is that the neighbouring pixels are interrelated and therefore contain irrelevant information. The foremost task in compression is usually to find the less correlated representation of the image. Two fundamental components of compression are redundancy and irrelevancy reduction. Redundancy reduction aims at removing duplication from the signal source (image / video). Irrelevancy reduction aims at omitting parts of the signal that will not be noticed by the signal receiver say the Human Visual System (HVS). In general, three types of redundancy can easily be identified:

- Spatial Redundancy or correlation between neighbouring pixel values.

- Temporal Redundancy or correlation between adjacent frames in a sequence of images (in video applications).

The aim of image compression research is to reduce the number of bits needed to represent an image by removing the spatial and spectral redundancies as much as possible. In this unit we examined different algorithms for achieving image and video compression.

6.0 Self-Assessment Exercise

1. What are the major difference between Vector Quantisation and transform coding?
2. With the aid of a well labeled diagram describe **three** (3) compression algorithms
3. Differentiate between lossy and lossless compression?
4. What are the advantages and disadvantages of compression?

7.0 References / Further Reading

Gersho,A.& Gray,R. (1992). *Vector Quantisation and Signal Compression*. Boston. MA

Heath, S. (1996). *Multimedia and Communications Technology*.Oxford: An imprint of *Butterworth-Heinemann. Jordan Hill.*

Jankerson, D., Harris, G.A.& Johnson,P.D. (2003)..USA: *Introduction to Information Theory and Data Compression*. ,(2nd ed.). Chapman and Hall / CRC , Florida.

Mitra,S.& Tinkuacharya, (2003). *Data Mining Multimedia, Soft Computing, and Bioinformatics*. Canada: John Wiley & Sons. Inc, Hoboken. New Jersey.

http://encyclopedia.jrank.org/articles/pages/6922/Video-Coding-Techniques-and-Standards.html

S. Image Compression - from DCT to Wavelets : A Review,

http://www.acm.org/crossroads/xrds6-3/sahaimgcoding.html

Unit 2 Image Processing and Human Visual System

1.0 Introduction

A digital image is a representation of a two-dimensional image using ones and zeros i.e. binary digits. Depending on whether or not the image resolution is fixed, it may be of vector or raster type. Whenever the term "digital image" is used, it usually refers to raster images or bitmap images. These contain a fixed number of rows and columns of pixels. Pixels are the smallest individual element in an image.

Digital images can be created by a variety of input devices and techniques, such as digital cameras, scanners, coordinate-measuring machines, seismographic profiling, airborne radar, etc. They can also be synthesised from arbitrary non-image data, such as mathematical functions or three-dimensional geometric models. Lots of researches are going on that aim at providing algorithms for their transformation. Some major topics within the field of image processing include image restoration, image enhancement, and image segmentation.

2.0 Objectives

At the end of this unit, you should be able to:

- explain the meaning of image processing
- identify the different types of images
- explain the human visual system

3.0 Main Content

3.1 What is Image Processing?

Image processing entails analysing and manipulating an image in order to either improve its pictorial information for human interpretation or render it more suitable for an independent machine perception. In this unit, we shall focus on **digital** image processing, which involves using a computer to change the nature of a digital image. Humans like their images to be sharp, clear and detailed while machines prefer their images to be simple and uncluttered. The process of improving the pictorial information of images for human interpretation requires one or more of the following processes:

- Enhancing the edges of an image to make it appear sharper
- Removing noise from an image. A simple way to look at noise is to consider it as random errors in the image. Figure 2.1a shows an image with noise while figure 2.1b shows an image without noise

Fig.2.1a: Image with Noise Fig 2.1b: Image with Noise Removal

- Removing motion blur from an image.

To render images suitable for an independent machine perception may entail:

- Obtaining the edges of an image. See figure 2.1c and figure 2.1d

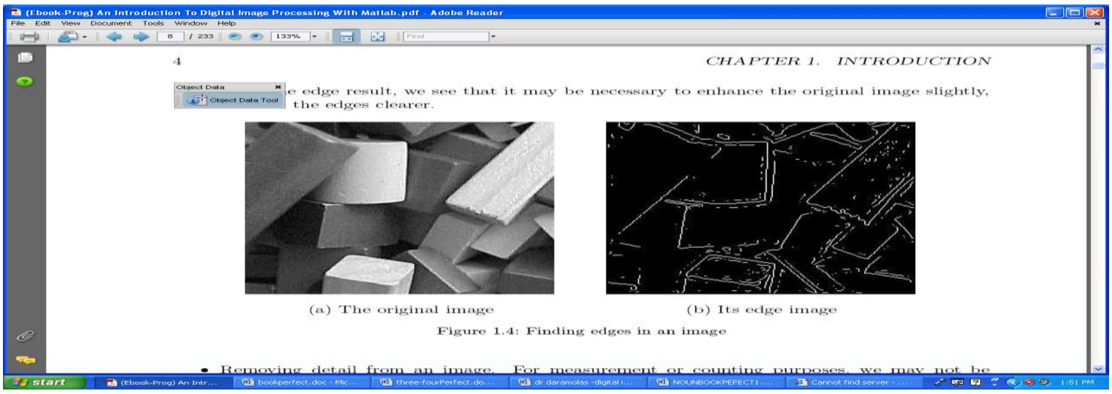


Fig.2.1c: Original Image **Fig. 2.1d:TheEdged Image**

Source McAndrew,A. (2004)

- Removing details from an image.

3.1.1 Subdivision of Image Processing

Image processing is a broad field and generally entails the following broad subclasses which include image restoration, image enhancement, and image segmentation to mention a few. In the following section we shall briefly explain the meaning of each of these subclasses with a view of revisiting some of them in subsequent modules.

Image Enhancement:
This is the process of improving the quality of a digitally stored image by manipulating the image with software or some techniques. Advanced image enhancement software may support many filters for altering images in various ways in making them more suitable for a particular application. Image enhancement tasks include for example:

- making an image lighter or darker,
- sharpening or de-blurring an out-of-focus image,
- highlighting edges,
- improving image contrast, or brightening an image
- removing noise

Image Restoration

This is the reversing of the damage done to an image by a known cause, for example:

- removing of blur caused by linear motion,
- removal of optical distortions,
- removal of periodic interference.

Image Segmentation

This involves subdividing an image into constituent parts, or isolating certain aspects of an image. For example:

- finding lines, circles, or particular shapes in an image,
- in an aerial photograph, one may be interested in identifying cars, trees, rivers, human beings, or roads networks.

3.1.2 An Image Processing Task

The tasks involved in image processing include image acquisition, image preprocessing, image segmentation, image description and representation and image recognition and interpretation. Let us consider a typical real world problem where we are required to obtain the postcodes from envelopes. The image processing will involve the following:

Acquire the Image.
The first task in processing an image is to acquire it from source via an input device. This can be done using either a charge-coupled device (CCD) camera or a scanner. So the first process in the problem above would be to scan the envelops.

Preprocess the Image
This is the step taken before the major image processing task. The problem here is to perform some basic tasks in order to render the resulting image more suitable for the job to follow. This may entail enhancing the contrast, removing noise, or identifying regions likely to contain the postcode from the scanned document.

Segment the Image
Here is where we actually get the postcode; in other words we extract from the image that part of it which contains just what we want. Example is the postcode from the entire scanned document.

Represent and Describe the Image
These terms refer to extracting the particular features which allow us to differentiate between objects that make up the image. In the case of a post code, we will be looking for curves, holes and corners which allow us to distinguish the different digits which constitute a postcode.

Recognise and Interpret the Image

This means assigning labels to objects based on their descriptors (from the previous step), and assigning meanings to those labels. Still with the example of post code under consideration, we identify particular digits, and we interpret a string of four digits at the end of the address as the postcode.

3.1.3 Types of Digital Images

When images are captured in digital forms, they are usually stored in different ways. We consider the following types of digital images

Binary

In this kind of images, each pixel is just black or white. Since there are only two possible values for each pixel, we only need one bit per pixel. Such images can therefore be very efficient in terms of storage. Images for which a binary representation may be suitable include text (printed or handwriting), fingerprints or architectural plans. Binary images are also called bi-level or two-level. This means that each pixel is stored as a single bit (0 or 1). The names black-and-white, B&W, monochrome or monochromatic are often used for this concept, but may also designate any images that have only one sample per pixel, such as grayscale images. Binary images often arise in digital image processing as masks or as the result of certain operations such as segmentation, thresholding, and dithering. Some input / output devices, such as laser printers, fax machines, and bi level computer displays, can only handle bi level images.

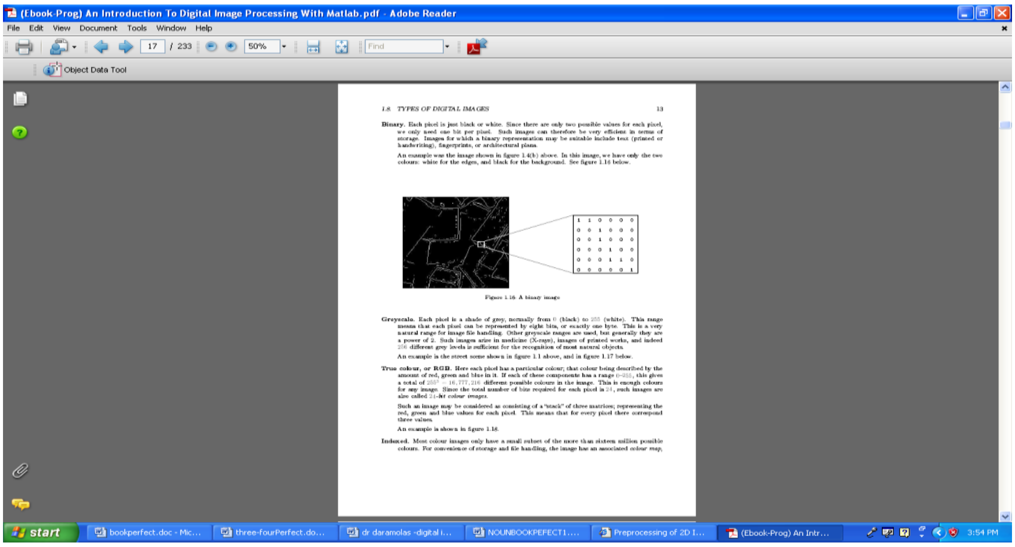


Fig.2.2: Binary Image

Grayscale

In photography and computing, a grayscale digital image is an image in which the value of each pixel is a single sample, that is, it carries only intensity information. Images of this sort, also known as black-and-white, are composed exclusively of shades of gray, varying from black at the weakest intensity to white at the strongest. Grayscale images are distinct from one-bit black-and-white images, which in the context of computer imaging are images with only the two colors, black, and white (also called bi level or binary images). Grayscale images have many shades of gray in between. Grayscale images are also called monochromatic, denoting the absence of any chromatic variation. Grayscale images are often the result of measuring the intensity of light at each pixel in a single band of the electromagnetic spectrum (e.g. infrared, visible light, ultraviolet, etc.), and in such cases they are monochromatic proper when only a given frequency is captured. But also they can be synthesised from a full colour image.

True Colour, or RGB

There are in fact a number of different methods for describing colours, but for image display and storage the standard model is RGB. RGB is the standard for the display of colours: on computer monitors; on TV sets. But it is not a very good way of describing colours. Here each pixel has a particular colour; that colour being described by the amount of red, green and blue in it. If each of these components has a range 0f 0-255, this gives a total of 256^3 =16,777,216 different possible colours in the image. This is enough colours for any image. Since the total number of bits required for each pixel is 24, such images are also called 24-bit colour image.

For most digital colours, image pixel is just a RGB data value (Red, Green, and Blue). Each pixel's colour sample has three numerical RGB components (Red, Green, and Blue) to represent the colour. These three RGB components are three 8-bit numbers for each pixel. Three 8-bit bytes (one byte for each of RGB) are called 24 bit colour. Each 8-bit RGB component can have 256 possible values, ranging from 0 to 255. For example, three values like (250, 165, 0), meaning (Red=250, Green=165, Blue=0) to denote one Orange pixel.

The composite of the three RGB values creates the final colour for that one pixel area. In the RGB system, we know Red and Green make Yellow. So, (255, 255, 0) means Red and Green, each fully saturated (255 is as bright as 8 bits can be), with no Blue (zero), with the resulting colour being Yellow.

Black is a RGB value of (0, 0, 0) and White is (255, 255, 255). Gray is interesting too, because it has the property of having equal RGB values. So (220, 220, 220) is a light gray (near white), and (40,40,40) is a dark gray (near black). Gray has no unbalanced colour cast.

Since gray has equal values in RGB, Black & White grayscale images only use one byte of 8 bit data per pixel instead of three. The byte still holds values 0 to 255, to represent 256 shades of gray.

Self -Assessment Exercise

1. What is image processing?
2. List the steps in image processing task

3.2 Human Visual System

The visual system is the part of the central nervous system which enables organisms to see. It interprets the information from visible light to build a representation of the world surrounding the body. The visual system accomplishes a number of complex tasks, including the reception of light, and the formation of monocular representations; the construction of a binocular perception from a pair of two dimensional projections; the identification and categorisation of visual objects; assessing distances to and between objects; and guiding body movements to visual objects. The psychological manifestation of visual information is known as visual perception.

The eye is a complex biological device. The functioning of a camera is often compared with the workings of the eye, mostly since both focus light from external objects in the visual field onto a light-sensitive medium. In the case of the camera, this medium is film or an electronic sensor; in the case of the eye, it is an array of visual receptors. With this simple geometrical similarity, based on the laws of optics, the eye functions as a transducer, as does a CCD camera.

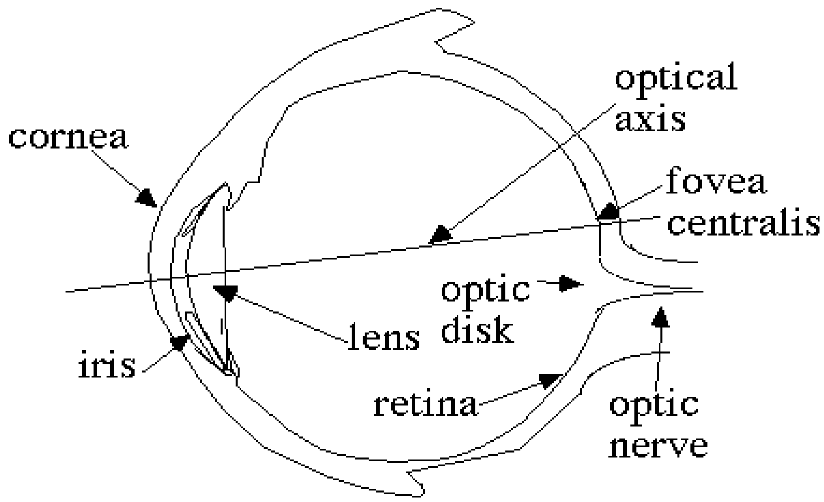


Fig.2.3: Human Visual System

Light entering the eye is refracted as it passes through the cornea. It then passes through the pupil (controlled by the iris) and is further refracted by the lens. The cornea and lens act together as a compound lens to project an inverted image onto the retina. The retina contains two types of photo sensor cells: rods and cones. There are 75 to 150 million rod cells in the retina. The rods contain a blue-green absorbing pigment called rhodopsin. Rods are used primarily for night vision (also called the scotopic range) and typically have no role in colour vision. Cones are used for daylight vision (called the photopic range). The tristimulus theory of colour perception is based upon the existence of three types of cones: red, green and blue. The pigment in the cones is unknown. We do know that the phenomenon called adaptation (a process that permits eyes to alter their sensitivity) occurs because of a change in the pigments in the cones. The retina cells may also inhibit each another from creating a high-pass filter for image sharpening. This phenomenon is known as lateral inhibition. Finally, the eye-brain interface enables integration between the sensors' polar coordinate scans, focus, iris adjustments and the interpretation engine. These interactions are not typical of most artificial image processing systems.

3.3 The Human Visual Systems Verse Computer Screen

One big advantage that the human eye has over a computer screen or electronic camera is its spectral responses to colour. Unlike the PC screen, which typically uses the same number of bits for red, green and blue (RGB) images, the eye's sensitivity is such that the weighting is different, and as a result, it becomes difficult to differentiate between colours that the PC is capable of displaying. If the eye cannot discriminate between all the colours, data can be saved by artificially restricting the data that the PC displays. In other words, if the eye cannot discriminate between 16 or 24 bits per pixel, why waste bandwidth transmitting all the data? This data reduction to match the eye's response is normally done by converting RGB data into other formats.

4.0 Conclusion

Any image from a scanner, or from a digital camera, or in a computer is represented in digital form for processing. Digitising is the process of capturing and converting an analog signal in digital form. Computers process only digitised images. The fundamental thing to understand about digital images is that they consist of pixels. The size of the image is dimensioned in pixels, X columns wide and Y rows tall. The number of pixels in width and height is the digital image's spatial resolution. The ratio of width to height in inches is known as images aspect ratio. The bit depth of an image is the number of bits used to store the value in each pixel. Since the images are to be examined and acted upon by people, the understanding of how the human visual system operates is necessary. The major topics within the field of image processing include image restoration, image enhancement, and image segmentation.

5.0 Summary

In this unit we covered image processing and its subfields, examined the different types of image representation within the computer and the human visual system. We also made a comparison between the human visual system and the computer screen

6.0 Self-Assessment Exercise

1. With the aid of a labeled diagram explain how the human visual system works.
2. Explain the different types of image representation within the computer. Computer.

7.0 References/Further Reading

McAndrew, A. (2004). *An Introduction to Digital Image Processing with Matlab*.

School of Computer Science and Mathematics, Victoria University of Technology.

Onka,M.S., Hlavac,V.& Boyle, R. (1999). *Image Processing, Analysis and Machine Vision*. (2nd ed). PWS Publishing.

Umbaugh,S.E. (1998). *Computer Vision and Image Processing*: A Practical Approach Using CVIPTools. Prentice-Hall.

http://en.wikipedia.org/wiki/YIQ

Unit 3 2d Data Transform With Dtft, Dft, Dct

1.0 Introduction

From the knowledge of engineering mathematics and signal processing we know that periodic function can be viewed in either the time domain or the frequency domain. It is very important to realise that both viewpoints are valid and either can be used for any signal. A transform is a mathematical tool used to change one representation into another and to make calculation easier. Specifically, the Fourier transform's utility lies in its ability to analyse a signal in the time domain for its frequency content. The transform works by first translating a function in the time domain into a function in the frequency domain. The signal can then be analysed for its frequency content because the Fourier coefficients of the transformed function represent the contribution of each sine and cosine function at each frequency. An inverse Fourier transform does just the reverse, that is, it transforms data from the frequency domain into the time domain.

2.0 Objectives

At the end of this unit, you should be able to:

- describe how signals are transformed from one domain to the other
- explain the different classes of signals
- list and define the different types of transforms
- discuss applications of Fourier transforms in multimedia processing.

3.0 Main Content

3.1 Background

The Discrete Fourier Transform (DFT) is a specific form of Fourier analysis to convert one function (often in the time or spatial domain) into another (frequency domain). DFT is widely employed in signal processing and related fields to analyse frequencies contained in a sample signal, to solve partial differential equations, and to perform other operations such as convolutions. Fast Fourier Transform (FFT) is an efficient implementation of DFT and is used, apart from other fields, in digital image processing. Fast Fourier Transform is applied to convert an image from the image (spatial) domain to the frequency domain. Applying filters to images in frequency domain is computationally faster than to do the same in the image domain.

Mathematically, Suppose

f = [$f_{0,} f_{1,} f_{2,} \dots, f_{N-1}$]

3.0

is a sequence of length N. Then the discrete Fourier transform can be defined as:

f = [$F_{0,} F_{1,} F_{2,} \dots, F_{N-1}$]

3.1

Where

$$Fu = \frac{1}{N}\sum_{x=0}^{N-1}\exp\left[-2\pi i\frac{xu}{N}\right]f_x$$

3.2

The formula for the inverse DFT is very similar to the forward transform:

$$x_u = \sum_{x=0}^{N-1}\exp\left[2\pi i\frac{xu}{N}\right]f_u$$

3.3

When you try to compare equations 3.2 and 3.3., you will notice that there are really only two differences:

1 there is no scaling factor 1/N

2 the sign inside the exponential function is now positive instead of negative

3.2 The Fast Fourier Transform

One of the many aspects which make the DFT so attractive for image processing is the existence of very fast algorithm to compute it. There are a number of extremely fast and efficient algorithms for computing a DFT; any of such algorithms is called a fast Fourier transform, or FFT. When an FFT is used, it reduces vastly the time needed to compute a DFT.

A particular FFT approach works recursively by separating the original vector into two halves as represented in equation 3.4 and 3.5, computing the FFT of each half, and then putting the result together. This means that the FFT is most different when the vector length is a power of 2.

$$F(u) = \sum_{x=0}^{M-1} f(x)\exp\left[-2\pi i\frac{xu}{M}\right]$$

3.4

$$f(x) = \frac{1}{M}\sum_{v=0}^{M-1} F(u)\exp\left[2\pi\frac{xu}{M}\right]$$

3.5

Table 3.1 is used to depict the benefits of using the FFT algorithm as opposed to the direct arithmetic definition of equation 3.4 and 3.5 by comparing the number of multiplication required for each method. For a vector of length 2^n, the direct method takes $(2^n)^2 = 2^{2n}$ multiplications; while the FFT takes only $n2^n$. Here the saving with respect to time is of an order of 2^n /n. Obviously, it becomes more attractive to use FFT algorithm as the size of the vector increases.

Because of this computational advantage, it is advisable for any implementation of the DFT to use an FFT algorithm.

Table 3.1: Comparison of FFT and Direct Arithmetic

2^n	Direct Arithmetic	FFT	Increase in speed
4	16	8	2.0
8	64	24	2.67
16	256	64	4.0
32	1024	160	6.4
64	4096	384	10.67
128	16384	896	18.3
256	65536	2048	32.0
512	262144	2406	56.9
1024	1048576	10240	102.4

3.3 The Two-Dimensional DFT

In two dimensions, the DFT takes a matrix as input, and returns another matrix, of the same size as output. If the original matrix values are $f(x,y)$, where x and y are the indices, then the output matrix values are $F(u,v)$. We call the matrix F the Fourier transform f and write

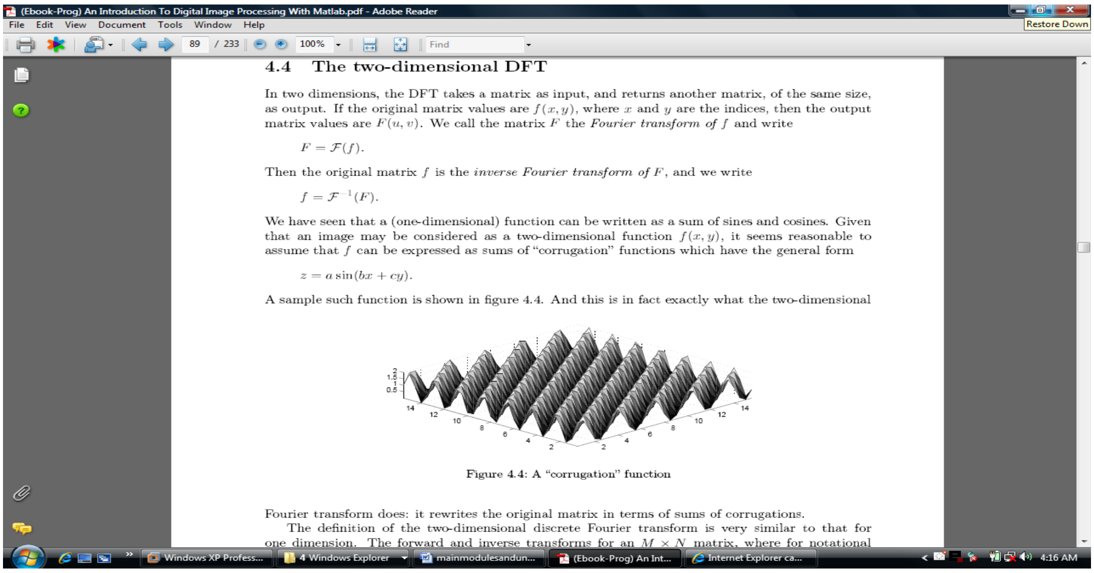
$$F = \mathcal{F}(f).$$

Then the original matrix f is the inverse Fourier transforms of F, and we write

$$f = \mathcal{F}^{-1}(F)$$

We have seen that a (one-dimensional) function can be written as a sum of sines and cosines. Given that an image may be considered as a two dimensional function, it seems reasonable to assume that F can be expressed as sums of “corrugations” functions which have the general form

$$z = a \sin(bx+cy)$$



(Ebook-Prog) An Introduction To Digital Image Processing With Matlab.pdf - Adobe Reader

4.4 The two-dimensional DFT

In two dimensions, the DFT takes a matrix as input, and returns another matrix, of the same size, as output. If the original matrix values are $f(x, y)$, where x and y are the indices, then the output matrix values are $F(u, v)$. We call the matrix F the *Fourier transform of* f and write

$$F = \mathcal{F}(f).$$

Then the original matrix f is the *inverse Fourier transform of* F, and we write

$$f = \mathcal{F}^{-1}(F).$$

We have seen that a (one-dimensional) function can be written as a sum of sines and cosines. Given that an image may be considered as a two-dimensional function $f(x, y)$, it seems reasonable to assume that f can be expressed as sums of "corrugation" functions which have the general form

$$z = a\sin(bx + cy).$$

A sample such function is shown in figure 4.4. And this is in fact exactly what the two-dimensional

Figure 4.4: A "corrugation" function

Fourier transform does: it rewrites the original matrix in terms of sums of corrugations.

The definition of the two-dimensional discrete Fourier transform is very similar to that for one dimension. The forward and inverse transforms for an $M \times N$ matrix, where for notational

Fig. 3.1: Corrugate Function

A sample of such function is depicted in figure 3.1 And this is in fact exactly what the two dimensional Fourier transforms does: it rewrites the original matrix in terms of sums of corrugation.

The definition of the two-dimensional discrete Fourier transform is very similar to that for one dimension. The forward and inverse transforms for an *M x N* matrix where for notational convenience we assume that the x indices are from 0 to *M-1* and the *y* indices are from 0 to *N-1* are:

$$F(u,v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\exp\left[-2\pi i\left(\frac{xu}{M}+\frac{yv}{N}\right)\right] \qquad 3.6$$

$$F(x,y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} F(u,v)\exp\left[-2\pi i\left(\frac{xu}{M}+\frac{yv}{N}\right)\right] \qquad 3.7$$

You may need to revise your mathematics to fully comprehend the formulas. However, they are not as difficult as they look.

Self- Assessment Exercise

Study Table 3.1 and see if you can reproduce the values in columns 2, 3, and 4 based on the theory provided

3.4 Some Properties of the Two Dimensional Fourier Transform

All the properties of the one-dimensional DFT transfer into two dimensions. We shall briefly consider some which are of particular use for image processing.

Similarity. A close study of the formulae for the forward and inverse transforms reveals some similarity except for the scale factor *1/M N* in the inverse transform and the negative sign in the exponent of the forward transform. This means that the same algorithm, only very slightly adjusted, can be used for both the forward and inverse transform. The DFT can thus be used as a spatial Filter

Linearity - An important property of the DFT is its linearity; the DFT of a sum is equal to the sum of the individual DFT's, and the same goes for scalar multiplication:

Thus

F (f+g) = F(f) + F (g)

F *(kf)* = k F*(f)*

Where *k* ☐ is a scalar product and *f* and *g* are matrices. This follows directly from the definition given in equation 3.6

This property is of great use in dealing with image degradation such as noise which can be modeled as a sum: *d*=*f* + *n* where *f* is the original image, *n* is the noise, and *d* is the degraded image. Since

F (*d*) = F(*f*) + F (*n*)

We may be able to remove or reduce *n* by modifying the transform. And we shall see some noise appear on the DFT in a way which makes it particularly easy to remove

3.5 DTFT

The **Discrete-Time Fourier Transform (DTFT)** is one of the specific forms of Fourier analysis. As such, it transforms one function into another, which is called the frequency domain representation, or simply the "DTFT", of the original function (which is often a function in the time-domain). But the DTFT requires an input function that is discrete. Such inputs are often created by sampling a continuous function, like a person's voice.

The DTFT frequency-domain representation is always a periodic function. Since one period of the function contains all of the unique information, it is sometimes convenient to say that the DTFT is a transform to a "finite" frequency-domain (the length of one period), rather than to the entire real line. It is Pontryagin dual to the Fourier series, which transforms from a periodic domain to a discrete domain.

Given a discrete set of real or complex numbers : x[n], n$\in \mathbb{Z}$(integer), the discrete-time Fourier transform (DTFT) is written as:

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n] e^{-i\omega n}$$

The following inverse transforms recovers the discrete-time sequence

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\omega).e^{i\omega n} d\omega$$

$$=T\int_{-\frac{1}{2T}}^{\frac{1}{2T}} X_T(f).e^{i2\pi fnT}df$$

Since the DTFT involves infinite summations and integrals, it cannot be calculated with a digital computer. Its main use is in theoretical problems as an alternative to the DFT. For instance, suppose you want to find the frequency response of a system from its impulse response. If the impulse response is known as an array of numbers, such as might be obtained from an experimental measurement or computer simulation, a DFT programme is run on a computer. This provides the frequency spectrum as another array of numbers, equally spaced between, for example, 0 and 0.6 of the sampling rate. In other cases, the impulse response might be given as an equation, such as a sine function or an exponentially decaying sinusoid. The DTFT is used here to mathematically calculate the frequency domain as another equation, specifying the entire continuous curve between 0 and 0.6. While the DFT could also be used for this calculation, it would only provide an equation for samples of the frequency response, not the entire curve.

3.6. Discrete Cosine Transform (DCT)

A discrete cosine transform (DCT) expresses a sequence of finitely many data points in terms of a sum of cosine functions oscillating at different frequencies. In particular, a DCT is a Fourier-related transform similar to the discrete Fourier transform (DFT), but using only real numbers. DCTs are equivalent to DFTs or roughly twice the length, operating on real data with even symmetry (since the Fourier transform of a real and even function is real and even), where in some variants the input and/or output data are shifted by half a sample. The cosine transform, like Fourier Transform, uses sinusoidal basis functions. The difference is that the cosine transform basis functions are not complex; they use only cosine functions, and not sine functions. The two-dimensional discrete cosine transform (DCT) equation for an N x N image for an example is as given by:

$$F(u,v) = C(u)C(v)\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f[m.n]\cos\left(\frac{(2m+1)u\pi}{2N}\right)\cos\left(\frac{(2n+1)v\pi}{2N}\right)$$

for $0 \le u,v < N$

with $C(u)= \begin{cases} \sqrt{1/N} \; for u=0 \\ \sqrt{2/N} \; for u\neq 0 \end{cases}$

We can interpret this as the projection of *f* [*m*,*n*] onto basis functions of the form:

$$e_{u,v}[m,n] = C(u,v)\cos\left(\frac{(2m+1)u\pi}{2N}\right)\cos\left(\frac{(2n+1)v\pi}{2N}\right)$$

Since this transform uses only the cosine function it can be calculated using only real arithmetic, instead of complex arithmetic as the DFT requires. The cosine transform can be derived from the Fourier transform by assuming that the function (the image) is mirrored about the origin, thus making it an even function. Thus, it is symmetric about the origin. This has the effect of canceling the odd terms, which correspond to the sine term (imaginary term) in Fourier transform. This also affects the implied symmetry of the transform, where we now have a function that is implied to be 2N x 2N.

Some sample basis functions are shown in Figure 3.2, for a value of *N*=8. It can be shown that this basis is orthonormal.

Based on the preceding discussions, we can represent an image as a superposition of weighted basis functions (using the inverse DCT):

$$f[m,n] = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} C(u)C(v)F[u,v]\cos\left(\frac{(2m+1)u\pi}{2N}\right)\cos\left(\frac{(2n+1)v\pi}{2N}\right)$$

for $0 \le m,n < N$

with $C(u) = \begin{cases} \sqrt{1/N} \; for u=0 \\ \sqrt{2/N} \; for u \neq 0 \end{cases}$

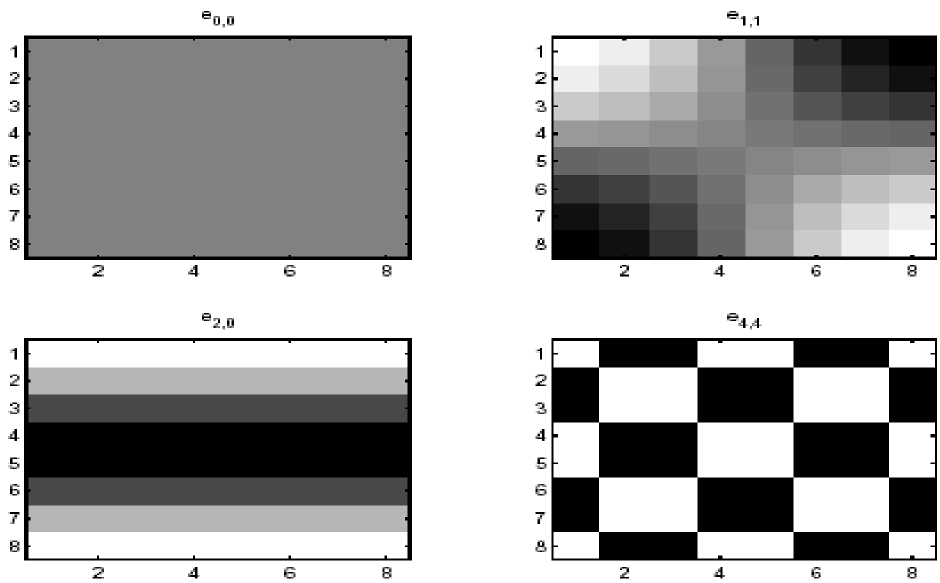


Fig. 3.2: Sample Basis Functions for an 8x8 Block of Pixels

The above four have been chosen out of a possible set of 64 basis functions.

This goes to show that DCT coefficients are similar to Fourier series coefficients in that they provide a mechanism for reconstructing the target function from the given set of basis functions. In itself, this is not particularly useful, since there are as many DCT coefficients as there were pixels in the original block. However, it turns out that most real images (natural images) have most of their energy concentrated in the lowest DCT coefficients. This is explained graphically in Figure 3.3 where a 32 x 32 pixel version of the test image and its DCT coefficients are shown. It can be shown that most of the energy is around the (0,0) point in the DCT coefficient plot. This is the motivation for compression – since the components for high values of *u* and *v* are small compared to the others, why not drop them, and simply transmit a subset of DCT coefficients, and reconstruct the image based on these. This is further illustrated in Figure 3.4, where we give the reconstructed 32 x 32 image using a small 10x10 subset of DCT coefficients. As you can see there is little difference between the overall picture of Figure 3.3(a) and Figure 3.7, so little information has been lost. However, instead of transmitting 32x32=1024 pixels, we only transmitted 10x10 =100 coefficients, which is a compression ratio of 1024 to 1.

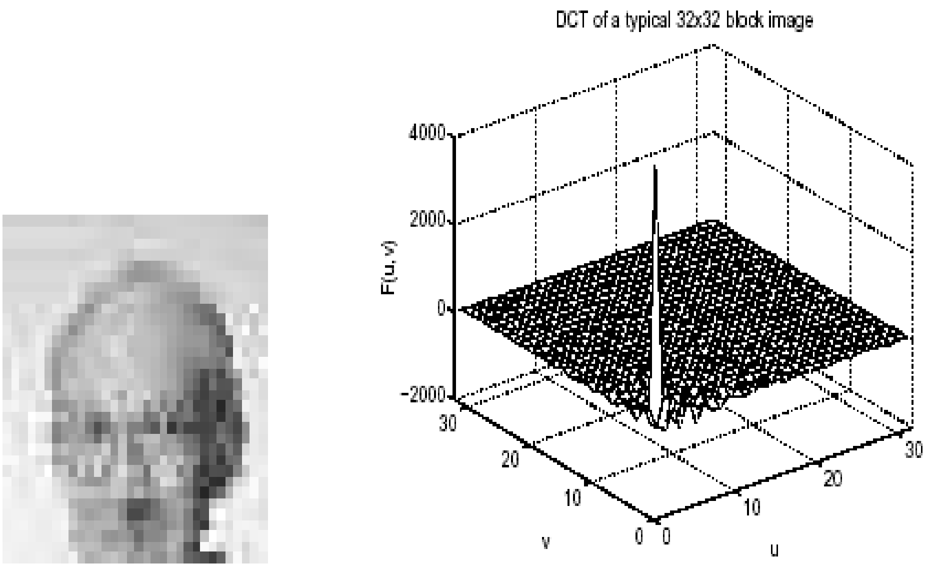


Fig. 3.3: (a) 32 x 32 Pixel Version of our Standard Test Image. (b) The DCT of this Image

Fig. 3.4: 32 x 32 Pixel Image Reconstructed from 10x10 Subset of DCT Coefficients. Overall information has been retained, but some detail has been lost

An optimal transform for compression would maximise the "energy-compressing" feature of the transform; that is the transform of the image would have most of its energy in the fewest number of coefficients. The DCT is not the optimal transform from this perspective; it can be shown mathematically that a Karhunen-Loeve transform (Hotelling transform) will provide the best basis for compression. However, this optimal basis is image-dependent and computationally intensive to find, so it is not commonly used in image compression systems.

DCTs are important to numerous applications in science and engineering, from lossy compression of audio and images (where small high-frequency components can be discarded), to spectral methods for the numerical solution of partial differential equations. The use of cosine rather than sine functions is critical in these applications: for compression, it turns out that cosine functions are much more efficient whereas for differential equations the cosines express a particular choice of boundary conditions. The DCT is the basis of many widespread image-coding standards: specifically, JPEG, MPEG, and H.26X which are respectively still image, video-playback, and video telephony standards.

4.0 Conclusion

The Fourier Transform is of fundamental importance to image processing. It allows us to perform tasks which would be impossible to perform by any other way; its efficiency allows us to perform most compression tasks more quickly. The Fourier transform is a very useful mathematical tool for multimedia processing. The Fourier Transform and the inverse Fourier transforms are the mathematical tools that can be used to switch from one domain to the other.

5.0 Summary

In this unit, we covered the definition of Fourier transforms, types of Fourier transform and its application in digital image processing

6.0 Self-Assessment Exercise

1. The DCT is the basis of many widespread image-coding standards". Explain why this is possible
2. Explain two (2) properties of the two dimensional Fourier transform.

7.0 References/Further Reading

Gibson, J.D. (Ed.). (2001). *Multimedia Communications: Directions and Innovations*. USA: Academic Press.

Lim, J.S.(1990). *Two-Dimensional Signal and Image Processing*. Prentice Hall.

McAndrew,A. (2004). *An Introduction to Digital Image Processing with Matlab*. School of Computer Science and Mathematics, Victoria University of Technology.

Schuler,C., Chugani,M. (2005). *Digital Signal Processing, A Hands-on*

Approach. USA: McGraw Hills.

William, K., Pratt, (1991). *Digital Image Processing*. (2nd ed.). John Wiley and Sons.

S. Image Compression - from DCT to Wavelets : A Review.

http://www.acm.org/crossroads/xrds6-3/sahaimgcoding.html