DESIGN OF EFFICIENT BASELINE CODERS FOR IMAGE COMPRESSION

Oleh

Vicknatheeban A/L Tharmalingam

Disertasi ini dikemukakan kepada
UNIVERSITI SAINS MALAYSIA

Sebagai memenuhi sebahagian daripada syarat keperluan
untuk ijazah dengan kepujian

SARJANA MUDA KEJURUTERAAN (KEJURUTERAAN ELEKTRONIK)

Pusat Pengajian Kejuruteraan
Elektrik dan Elektronik
Universiti Sains Malaysia **Mac 2005**

ABSTRACT

Image compression is the process of reducing the number of bits required to represent an image. This can be achieved by reducing (or ideally, eliminating) various types of redundancy that exist in the imaging data. This research develops various types of down-sampling filters to compress the image and followed by up-sampling filters to decompress the image. The concept used to down-sample is by deleting either the odd or even numbered columns or rows. Besides that, the columns and rows are also deleted in two’s to further compress the image. The image is then up-sampled by using duplication or replacing the deleted row or columns with the average of other rows or columns. The compression ratio achieved through this research is 50% and 67%. The images are then compared by the Peak Signal-to-Noise Ratio and through observation to decide on the best type of filter. Through the research and results, it is proved that the filter that up-samples the image by replacing the deleted rows with the average of the two subsequent even numbered row has the best output.

ABSTRAK

Pemampatan imej merupakan satu proses untuk mengurangkan bilangan bit yang digunakan untuk mewakili sesuatu imej. Ini dapat dicapai dengan mengurangkan (atau secara lazimnya, menghapuskan) berbagai jenis pengulangan yang wujud dalam data sesuatu imej. Kajian ini menghasilkan berbagai jenis penuras pensampel ke bawah untuk memampatkan imej dan diikuti dengan penuras pensampel ke atas untuk menyahmampat imej tersebut. Konsep yang digunakan untuk pensampelan ke bawah adalah dengan menghapus baris atau lajur yang bernombor ganjil atau genap. Selain itu, baris dan lajur juga dihapus secara duaan untuk menghasilkan mampatan yang lebih tinggi. Imej tersebut kemudian dinyahmampat dengan kaedah penduaan atau dengan menggantikan baris atau lajur yang dihapus dengan purata baris atau lajur yang masih wujud. Kadar pemampatan yang diperolehi menerusi kajian ini adalah 50% dan 67%. Imej yang terhasil kemudiannya dibanding dengan pemerhatian dan juga PSNR. Menerusi kajian dan keputusan yang diperolehi, didapati penuras yang menyahmampat imej dengan menggantikan baris yang dihapus dengan purata baris sebelum dan selepasnya, menghasilkan keputusan yang terbaik.

ACKNOWLEDGEMENT

This project would not have been a success without the help and attention of many parties. I would like to take this opportunity to express my sincere gratitude to all those who have contributed in completing this project.

Firstly, I would like to express my appreciation to my final year project supervisor Prof. Farid Ghani for his expert supervision and constant attention throughout my final year project. It has been a privilege to work under such fine supervisory. His comments enabled me to produce a better quality performance in the project and the final report.

Furthermore, I would like to thank the School of Electrical & Electronic Engineering for providing me with this wonderful opportunity to carry out the final year project. This project has exposed me to the current technology and also too increases my knowledge and skills in MATLAB programming. I was also exposed to do a proper research in order to get a clear idea on the selected topic.

Finally, I wish to thank my parents and my friends who had supported and motivated me throughout the completion of this project.

Thank you,

VICKNATHEEBAN THARMALINGAM

(4th FEBRUARY 2005)

CONTENT

LIST OF FIGURES AND TABLES

CHAPTER 1
INTRODUCTION

1.1 Project Background

Image files, in an uncompressed form are very large. Uncompressed multimedia such as graphics, audio and video data require considerable storage capacity and transmission bandwidth. Despite the rapid progress in mass-storage density, processor speeds and digital communication system performance, demand for data storage capacity and data-transmission bandwidth continues to outstrip the capabilities of the available technologies. The recent growth of data intensive multimedia-based web applications have not only sustained the need for more efficient ways to encode signals and images but have made compression of such signals central to storage and communication technology.

An image is a picture, photograph, display or other form which gives a visual representation of an object or scene. In terms of Digital Image Processing, an image is a two dimensional array of numbers. Each number corresponds to one small area of the visual image and the number gives the level of darkness or lightness of the area. Each small area to which a number is assigned is called a pixel which is short for picture element. Each pixel has its value plus a line coordinate and a sample coordinate. This gives its location in the image array. We always assume that images are rectangular arrays.

In general, the purpose of image processing is to enhance or improve the image in some way or to extract information from it. Typical operations on image processing operations are to:

- Remove the blur from an image.
- Smooth-out the graininess, speckle or noise in an image.
- Improve the contrast or other visual properties of an image prior to displaying it.
- Segment an image into regions such as objects and backgrounds.
- Compress the image in some efficient way for storage or transmission.

This project is devoted to the last operation on image processing mentioned above which is to compress the image in some efficient way for storage or transmission.

Digital image processing has two main advantages. The first is precision. In each generation of photographic process, there is a loss of image quality, and electrical signals are degraded by the physical limitations of the electrical components, whereas digital image processing can essentially maintain exact precision. The second advantage is its extreme flexibility. Using an enlarger, an image may be magnified, but in digital image processing, one part maybe magnified but another reduced, another rotated and so on. The contrast and brightness of a television picture may be adjusted, but in digital image processing, literally dozens of adjustments can be made.

A common characteristic of most images is that the neighboring pixels are correlated and therefore contain redundant information. The foremost task then is to find 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 omits parts of the signal that will not be noticed by the signal receiver, namely the Human Visual System (HVS). In general, three types of redundancy can be identified:

- Spatial Redundancy or correlation between neighboring pixel values.
- Spectral Redundancy or correlation between different color planes or spectral bands.
- Temporal Redundancy or correlation between adjacent frames in a sequence of images (in video applications).

Image compression research aims at reducing the number of bits needed to represent an image by removing the spatial and spectral redundancies as much as possible. Since we will focus only on still gray scale image compression, we consider about temporal redundancy any further.

Many may ask why we have to compress an image that is to be stored or be transmitted. The figures in the table below explain this. It shows the qualitative transition from simple text to full-motion video data and the disk space, transmission bandwidth and the transmission time needed to store and transmit such uncompressed data.

Table 1.1: Types of multimedia data and its size, transmission bandwidth and time

Multimedia Data	**Size/Duration**	**Bits/Pixel or Bits/Sample**	**Uncompressed Size (B for bytes)**	**Transmission Bandwidth (B for bits)**	**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 – 2.2 sec
Telephone quality speech	10 sec	8 bps	80 KB	64 Kb/page	22.2 sec
Grayscale Image	512 x 512	8 bpp	262 KB	2.1 Mb/image	1 min 13 sec
Colour Image	512 x 512	24 bpp	786 KB	6.29 Mb/image	3 min 39 sec
Medical Image	2048 x 1680	12 bpp	5.16 MB	41.3 Mb/image	23 min 54 sec
SHD Image	2048 x 2048	24 bpp	12.58 MB	100 Mb/image	58 min 15 sec
Full-motion Video	640 x 480, 1 min (30 frames/sec)	24 bpp	1.66 GB	221 Mb/sec	5 days 8 hrs

The examples above clearly illustrate the need for sufficient storage space, large transmission bandwidth and long transmission time for image, audio and video data. At the present state of technology, the only solution is to compress multimedia data before storage and transmission and decompress it at the receiver for playback.

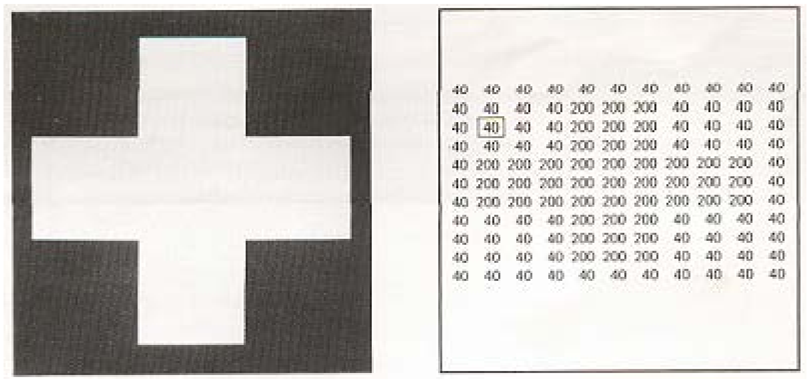
1.2 Scope of the Project

This project is concerned in designing new coding schemes to improve the compression that has been achieved in images. The coding schemes will involve the design of up-sampling and down-sampling filters in order to further compress the image without adversely effecting the quality of the image as compared to the original image.

CHAPTER 2
IMAGE REPRESENTATION

2.1 Definition of an Image

As it was mention earlier, an image is a picture, photograph, display or some other form which gives the viewer a visual representation of an object or scene. In the context of image processing, an image is a two dimensional array of numbers. These numbers are represented in a matrix array. The figure 2.1 below shows an image of a simple geometric pattern and the corresponding digital image.



40	40	40	40	40	40	40	40	40	40	40
40	40	40	40	200	200	200	40	40	40	40
40	$\boxed{40}$	40	40	200	200	200	40	40	40	40
40	40	40	40	200	200	200	40	40	40	40
40	200	200	200	200	200	200	200	200	200	40
40	200	200	200	200	200	200	200	200	200	40
40	200	200	200	200	200	200	200	200	200	40
40	40	40	40	200	200	200	40	40	40	40
40	40	40	40	200	200	200	40	40	40	40
40	40	40	40	200	200	200	40	40	40	40
40	40	40	40	40	40	40	40	40	40	40

Figure 2.1: (a) The image and (b) the corresponding array of numbers

The image above has eleven rows and eleven columns. Each number in (b) corresponds to one small area of the visual image, and the number gives the level of darkness or lightness of the area. It is assumed that the higher the number, the lighter the area, so zero is black and the maximum value are white. The intermediate values are shades of gray. This is actually arbitrary and could be done the other way round. Each small area to which a number is assigned is called a "pixel", which is short for a picture element. The size of the physical area represented by a pixel is called the spatial resolution of the pixel. This varies greatly, from a few nanometers in microscope images to tens of kilometers in satellite images. Each pixel has its value and a coordinate of row and column. These row and column gives the location of the pixel in the image array. For example, the pixel at row

3 and column 2 in figure 2.1(b) (which has the box around it) has the value 40.(Niblack W. 1986)

It is always assumed that images are rectangular arrays, where there are m rows and n columns in the image. Often images are square, and typical image sizes are 256x256, 512x512 and 1024x1024. An image has two possible representations, that is either as a gray-scale picture or as a colour picture.

2.2 Gray-Scale Images

A grayscale image or also know as a gray-level image is simply one in which the only colours are shades of gray. The reason for differentiating such images from any other type of colour image is that less information needs to be provided for each pixel. In fact a `gray' colour is one in which the red, green and blue components all have equal intensity in RGB space, and so it is only necessary to specify a single intensity value for each pixel, as opposed to the three intensities needed to specify each pixel in a full colour image.

Often, the grayscale intensity is stored as an 8-bit integer giving 256 possible different shades of gray from black to white. If the levels are evenly spaced then the difference between successive gray-levels is significantly better than the gray-level resolving power of the human eye.

Grayscale images are very common, because much of today's display and image capture hardware can only support 8-bit images. In addition, grayscale images are entirely sufficient for many tasks and so there is no need to use more complicated and harder-to-process colour images. Several different graphics file formats are used to save grayscale images, with the TIF format being the most prevalent.

2.3 Colour Model

Colour provides a significant portion of visual information to human beings and enhances their abilities of object detection. It is experimentally estimated that the human eye can distinguish about 350,000 different colours.

Colour images take advantage of the fact that each colour can be expressed as a combination of three primary colours which are red, green and blue (RGB) or yellow, magenta and cyan (YMC). Therefore a colour picture can be considered as the superimposition of three “simpler” pictures which are called colour planes, with each of them encoding the brightness of a primary colour. In other words, each colour plane of an image can be treated much like a gray-scale picture with a range of values based on the luminosity of that particular colour. This type of representation, called RGB or YMC according to the colour planes, is “hardware-oriented” where monitors, printers and photographic devices use these colour schemes and it is usually used when dealing with synthetic (computer-generated) images.

How we perceive colour image information is classified into three perceptual variables: hue, saturation and lightness. When we use the word colour, typically we are referring to hue. Hue distinguishes among colours such as green and yellow. Hues are the colour sensations reported by an observer exposed to various wavelengths. It has been shown that the predominant sensation of wavelengths between 430 and 480 nanometers is blue. Green characterizes a broad range of wavelengths from 500 to 550 nanometers. Yellow covers the range from 570 to 600 nanometers and wavelengths over 610 nanometers are categorized as red. Black, gray, and white may be considered colours but not hues.

CHAPTER 3
IMAGE COMPRESSION

3.1 Predictive and Transform Coding

In the process of compressing an image, there are two different types of coding techniques that could be used. The two different types of coding techniques are Predictive coding and transform coding. Each type of coding technique has its own advantage and disadvantages.

3.1.1 Predictive Coding

Predictive coding is an image compression technique which uses a compact model of an image to predict pixel values of an image based on the values of neighbouring pixels. A model of an image is a function model(x; y), which computes (predicts) the pixel value at coordinate (x; y) of an image, given the values of some neighbours of pixel (x; y), where neighbours are pixels whose values are known. Typically, when processing an image in raster scan order neighbours are selected from the pixels above and to the left of the current pixel.

Nonlinear models assign arbitrarily complex functions to the models. Suppose that we have a perfect model of an image, that is, one which can perfectly reconstruct an image given the pixel value of the border pixels (assuming we process the pixels in raster order). Then, the value of the border pixels and this compact model is all that needs to be transmitted in order to transmit the whole information content of the image. In general, it is not possible to generate a compact, perfect model of an image, and the model generates an error signal (the differences at each pixel between the value predicted by the model and the actual value of the pixel in the original image.

There are two expected sources of compression in predictive coding based image compression (assuming that the predictive model is accurate enough). First, the error signal for each pixel should have a smaller magnitude than the corresponding pixel in the original image (therefore requiring fewer bits to transmit the error signal). Second, the error signal should have less entropy than the original message, since the model should remove many of the “principal components” of the image signal.

Differential Pulse Code Modulation (DPCM) is one particular example of predictive coding.

3.1.2 Transform Coding

Transform coding is used to convert spatial image pixel values to transform coefficient values. Since this is a linear process and no information is lost, the number of coefficients produced is equal to the number of pixels transformed.

The desired effect is that most of the energy in the image will be contained in a few large transform coefficients. If it is generally the same few coefficients that contain most of the energy in most pictures, then the coefficients may be further coded by lossless entropy coding . In addition, it is likely that the smaller coefficients can be coarsely quantized or deleted (lossy coding) without doing visible damage to the reproduced image.

The energy of a pixel may be defined as the square of its value times some scaling factor. Similarly, the energy of a transform coefficient may be defined as the square of its value times some scaling factor. With the proper scaling factor, the total energy of the pixels in a picture will always equal the total energy in the transform coefficients. The transform process simply concentrates the energy into particular coefficients, generally the "low frequency" ones.

Many types of transforms have been tried for picture coding, including for example Fourier, Karhonen-Loeve, Walsh-Hadamard, lapped orthogonal, discrete cosine (DCT), and recently, wavelets. The various transforms differ among themselves in three basic ways that are of interest in picture coding:

- the degree of concentration of energy in a few coefficients;
- the region of influence of each coefficient in the reconstructed picture;
- the appearance and visibility of coding noise due to coarse quantization of the coefficients.

Karhunen-Loeve is a statistically based transform method that can be tailored to one image or group of images, and therefore has the optimum energy concentration. However, it generally will not have this optimum concentration for images not in the basis set.

Fourier transforms (discrete) have good energy concentration characteristics, but become unwieldy when dealing with large images requiring large numbers of coefficients. Block transforms, which work on a small portion of the image at a time, are therefore preferred. The discrete Fourier transform may be applied to a block of pixels. Other transforms which fall in this category are Walsh-Hadamard, and the DCT. The lapped orthogonal transforms are a special case in which the coefficients' influence is confined to a few adjacent blocks, with a tapering-off influence toward the edges.

The transform coding method provides greater data compression compared to predictive methods, although at the expense of greater computation.

3.2 Lossless Compression

Lossless compression guaranties that the decompressed image is absolutely identical to the image before compression. This is an important requirement for some application domains, for example medical imaging, where not only high quality is in demand, but unaltered archiving is a legal requirement. Lossless techniques can also be used for the compression of other data types where loss of information is not acceptable, for example text documents and program executables. Among the lossless compression methods are:

- Run length encoding
- Huffman encoding
- Area coding

3.2.1 Run Length Encoding

Run length encoding is a very simple method for compression of sequential data. It takes advantage of the fact that, in many data streams, consecutive single tokens are often identical. Run length encoding checks the stream for this fact and inserts a special token each time a chain of more than two equal input tokens are found. This special input advises the decoder to insert the following token *n* times into his output stream.

Run length coding is easily implemented, either in software or in hardware. It is fast and very well verifiable, but its compression ability is very limited.

3.2.2 Huffman Encoding

This algorithm, developed by D.A. Huffman, is based on the fact that in an input stream certain tokens occur more often than others. Based on this knowledge, the algorithm builds up a weighted binary tree according to their rate of occurrence. Each element of this tree is assigned a new code word, whereas the length of the code word is determined by its position in the tree. Therefore, the token which is most frequent and becomes the root of the tree is assigned the shortest code. Each less common element is assigned a longer code word. The least frequent element is assigned a code word which may have become twice as long as the input token.

The compression ratio achieved by Huffman encoding uncorrelated data becomes something like 1:2. On slightly correlated data, as on images, the compression rate may become much higher, the absolute maximum being defined by the size of a single input token and the size of the shortest possible output token

3.2.3 Area Coding

Area coding is an enhanced form of run length coding, reflecting the two dimensional character of images. This is a significant advance over the other lossless methods. For coding an image it does not make too much sense to interpret it as a sequential stream, as it is in fact an array of sequences, building up a two dimensional object. Therefore, as the two dimensions are independent and of same importance, it is obvious that a coding scheme aware of this has some advantages. The algorithms for area coding try to find rectangular regions with the same characteristics. These regions are coded in a descriptive form as an element with two points and a certain structure. The whole input image has to be described in this form to allow lossless decoding afterwards.
The possible performance of this coding method is limited mostly by the very high complexity of the task of finding largest areas with the same characteristics. Practical implementations use recursive algorithms for reducing the whole area to equal sized sub-rectangles until a rectangle does fulfill the criteria defined as having the same characteristic for every pixel.

This type of coding can be highly effective but it bears the problem of a nonlinear method, which cannot be implemented in hardware. Therefore, the performance in terms of compression time is not competitive, although the compression ratio is.

3.3 Lossy Compression

In most of applications we have no need in the exact restoration of stored image. This fact can help to make the storage more effective, and this way we get to lossy compression methods. Lossy image coding techniques normally have three components:

- *image modeling* which defines such things as the transformation to be applied to the image
- *parameter quantization* whereby the data generated by the transformation is quantized to reduce the amount of information
- *encoding*, where a code is generated by associating appropriate code words to the raw data produced by the quantifier.

Each of these operations is in some part responsible of the compression. Image modeling is aimed at the exploitation of statistical characteristics of the image (i.e. high correlation, redundancy). Typical examples are transform coding methods, in which the data is represented in a different domain (for example, frequency in the case of the Fourier Transform [FT], the Discrete Cosine Transform [DCT], the Kahrunen-Loewe Transform [KLT], and so on), where a reduced number of coefficients contains most of the original information. In many cases this first phase does not result in any loss of information.

The aim of quantization is to reduce the amount of data used to represent the information within the new domain. Quantization is in most cases not a reversible operation; therefore, it belongs to the so called 'lossy' methods. Encoding is usually error free. It optimizes the representation of the information (helping, sometimes, to further reduce the bit rate), and may introduce some error detection codes.
Among the lossy compression methods are:

- Vector quantization
- Fractal coding

3.3.1 Vector Quantization

Vector quantization, breaks an image into blocks or also know as vectors of *n* X *n* pixels. These blocks are then compared with a set of representative blocks. This collection of representative vectors is called a codebook. A summation of differences between the pixels in the source vector and the codebook vector is computed for each codebook entry. The codebook entry with the smallest difference summation is chosen as the representative vector. The index of that vector is then stored to a file or transmitted.

The toughest part of vector quantization is generating codebooks. Many people instinctively think that you can just count the frequency of all vectors in a large set of representative images. The codebook could then be composed of the most frequently occurring vectors. Although this seems like a great idea, it creates a lousy codebook. Vectors that contain much information (like edges) may not occur frequently in an image and may be left out of a codebook. This produces images of poor quality. There are many elaborate schemes for generating good codebooks. Most of them have great computational requirements.

Vector quantization comes in many flavors. One method, recursive VQ, repetitively encodes the image and the difference between the image and its approximation (the value from the codebook). Another method removes the mean of a vector before encoding.

3.3.2 Fractal Compression

Fractal compression is a radical departure from the conventional image compression techniques. The difference between it and the other techniques is much like the difference between bitmapped graphics and vector graphics. Rather than storing data for individual pixels, fractal compression stores instructions or formulas for creating the image. Because of that, images compressed with fractal compression are resolution independent. They can be scaled up to a resolution higher than the original image without the distracting artifacts associated with scaling. These scalable images are well suited for graphics systems that are typically composed of devices of differing resolutions (graphics cards, printers, etc.).

Fractals are images that are composed of smaller images. Fractals were first widely introduced (or reintroduced) in the book *The Fractal Geometry of Nature* by Benoit

Mandelbrot. Fractal compression does very well with natural scenes and claims to achieve compression ratios greater than 100.

Like vector quantization, fractal compression is asymmetrical. Although it takes a long time to compress an image, decompression is very fast. These asymmetrical methods are well suited to such applications as video on a CD-ROM where the user doesn't care about compression but does expect to see images quickly. Decompression simply reads the mathematical formulas and recreates the image.

3.4 Compression Standards

Currently there are various forms of image compression standards such as JPEG, GIF and PNG. Each of the compression techniques are explained as follows.

3.4.1 Joint Photographic Experts Group (JPEG)

JPEG compression is currently the best way to compress PHOTOGRAPHIC IMAGES for the web. JPEG (pronounced “jay-peg”) is a standardized image compression mechanism. JPEG stands for Joint Photographic Experts Group, the original name of the committee that wrote the standard. JPEG is designed for compressing full-colour or gray-scale images of natural, real-world scenes. It works well on photographs, naturalistic artwork and similar material but not so well on lettering, sample cartoons or line drawings. JPEG handles only still images but there is a related standard called MPEG for motion pictures.

JPEG is an example of the lossy image explained earlier. Although it is a lossy image compression technique, it achieves much greater compression than is possible with lossless methods. JPEG is designed to exploit known limitation of the human eye, notably the fact that small colour changes are perceived less accurately than small changes in brightness. Thus, JPEG is intended for compressing images that will be looked at by humans.

Another important aspect of JPEG is that decoders can trade off decoding speed against image quality, by using fast but inaccurate approximations to the required calculations.

3.4.2 Graphics Interchange Format (GIF)

GIF (Graphics Interchange Format) is best used for graphics that have a limited color pallet and large areas of flat tone, like cartoons or banners. GIF is a lossless method of compression. All that means is that when the program that creates a GIF squashes the original image down it takes care not to lose any data. It uses a simple substitution method of compression. If the algorithm comes across several parts of the image that are the same, say a sequence of digits like this, 1 2 3 4 5, 1 2 3 4 5, 1 2 3 4 5, it makes the number 1 stand for the sequence 1 2 3 4 5 so that you could render the same sequence 1 1 1, obviously saving a lot of space. It stores the key to this (1 = 1 2 3 4 5) in a hash table, which is attached to the image so that the decoding program can unscramble it.

The maximum compression available with a GIF therefore depends on the amount of repetition there is in an image. A flat colour is able to be compressed down to one tenth of the original file size - while a complex, non-repetitive image will fare worse, perhaps only saving 20% or so.

There are several problems when GIF compression technique is used. One is that they are limited to a palette of 256 colours or less. The group which created the GIF, did at one point say it would attempt to produce a 24-bit version of the GIF, but then along came problem number two: Unisys. Unisys discovered that it owned some patents to key parts of the GIF compression technology, and has started demanding fees from every company whose software uses the (freely available) GIF code. This has somewhat stifled development.

3.4.3 Portable Network Graphics (PNG)

PNG (Portable Network Graphics) is a relatively new format. The PNG format provides a portable, legally unencumbered, well-compressed, well-specified standard for lossless bitmapped image files. Although the initial motivation for developing PNG was to replace GIF, the design provides some useful new features not available in GIF, with minimal cost to developers.

Among the GIF features retained in PNG include:

- Indexed-color images of up to 256 colors.
- Streamability: files can be read and written serially, thus allowing the file format to be used as a communications protocol to display of images.
- Progressive display: a suitably prepared image file can be displayed as it is received over a communications link, yielding a low-resolution image very quickly followed by gradual improvement of detail.
- Effective, 100% lossless compression.

Among the new features of PNG that is not available in GIF, include:

- True-color images of up to 48 bits per pixel.
- Grayscale images of up to 16 bits per pixel.
- Reliable, straightforward detection of file corruption.
- Faster initial presentation in progressive display mode.

CHAPTER 4
DESIGN OF EFFICIENT BASELINE CODERS USING MATLAB

4.1 Introduction to MATLAB and Image Processing Toolbox

The software used in this project to design the efficient baseline coders is MATLAB. Among the toolboxes in MATLAB is the Image Processing Toolbox. This toolbox was very helpful in understanding image compression.

4.1.1 MATLAB

To design the Up-sampling and Down-sampling filters to compress and decompress the images, a very powerful software known as MATLAB is used.

MATLAB is a high-level technical computing language and interactive environment for algorithm development, data visualization, data analysis, and numerical computation. By using MATLAB, technical computing problems can be solved faster than solving it with traditional programming languages, such as C, C++, and Fortran.

MATLAB can be used in a wide range of applications, including signal and image processing, communications, control design, test and measurement, financial modeling and analysis, and computational biology. Add-on toolboxes (collections of special-purpose MATLAB functions, available separately) extend the MATLAB environment to solve particular classes of problems in these application areas. By minimizing human time, MATLAB is particularly useful in the initial investigation of real problems; even though they may eventually have to be solved using more computationally efficient ways on super computers.

Among the key features of the MATLAB software is:

- High-level language for technical computing
- Development environment for managing code, files, and data
- Interactive tools for iterative exploration, design, and problem solving
- Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, and numerical integration
- 2-D and 3-D graphics functions for visualizing data
- Tools for building custom graphical user interfaces
- Functions for integrating MATLAB based algorithms with external applications and languages, such as C, C++, Fortran, Java, COM, and Microsoft Excel

4.1.2 Image Processing Toolbox

MATLAB has various toolboxes to help in the usage of the software. One of the most useful toolbox in MATLAB for this project is the Image Processing toolbox. The Image Processing Toolbox is a collection of functions that extend the capability of the MATLAB numeric computing environment. The toolbox supports a wide range of image processing operations, including

- Spatial image transformations
- Morphological operations
- Neighborhood and block operations
- Linear filtering and filter design
- Transforms
- Image analysis and enhancement
- Image registration
- De-blurring
- Region of interest operations

Many of the toolbox functions are MATLAB M-files, a series of MATLAB statements that implement specialized image processing algorithms. The capabilities of the Image Processing Toolbox can be extended by writing M-files, or by using the toolbox in combination with other toolboxes, such as the Signal Processing Toolbox and the Wavelet Toolbox.

4.2 Test Images

Gray-scale stilt test images would be used throughout this project. Images that are compressed using the up-sampling and down-sampling filters that would be designed would be compared with the original image to determine the quality. Among the types of test images that would be used are Lena, Baboon and Cameraman which is shown below. These images are obtained from the USC database.

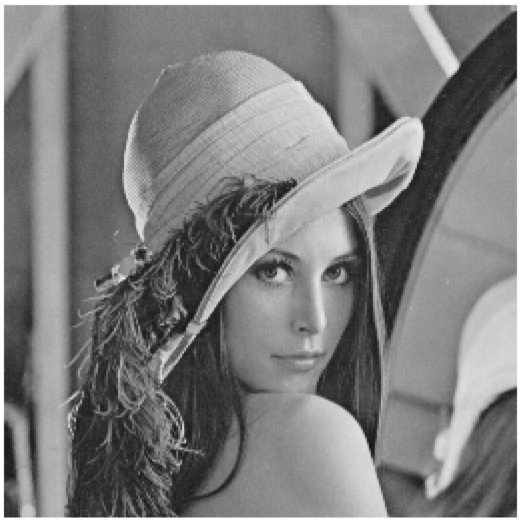


Figure 4.1: Lena.tif

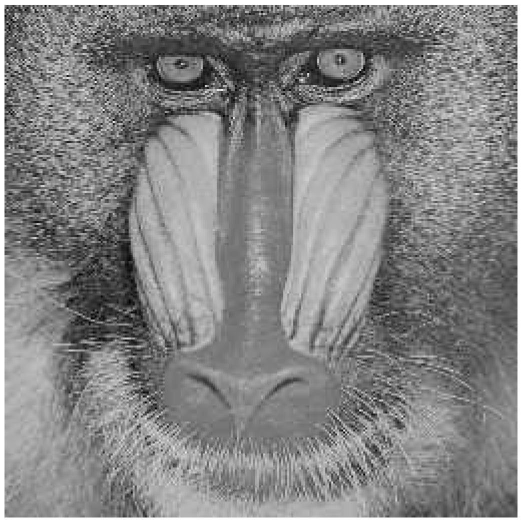


Figure 4.2: Baboon.tif

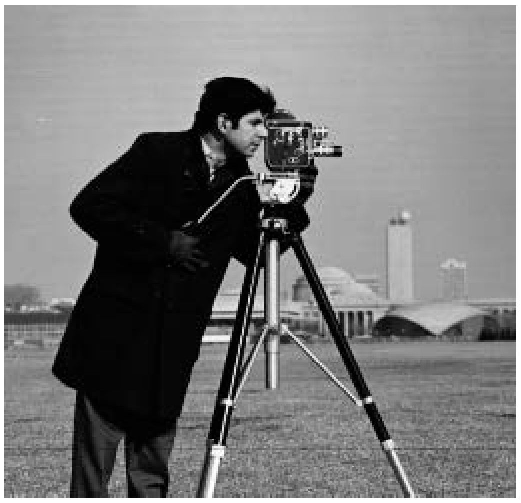


Figure 4.3: Cameraman.tif

4.3 Design Of Down-Sampling And Up-Sampling Filters

In MATLAB images can be represented and manipulated as N-dimensional arrays. In this project a 2-dimensional array is used as only gray-scale images are used. An image named X which is of size 256x256 can be represented as a 2-dimensional matrix as follows:

$$
X\,(m,n) = \begin{array}{cccccc}
m_1n_1 & m_1n_2 & m_1n_3 & m_1n_4 & m_1n_5 & m_1n_{256} \\
m_2n_1 & m_2n_2 & m_2n_3 & m_2n_4 & m_2n_5 & m_2n_{256} \\
m_3n_1 & m_3n_2 & m_3n_3 & m_3n_4 & m_3n_5 & m_3n_{256} \\
m_4n_1 & m_4n_2 & m_4n_3 & m_4n_4 & m_4n_5 & m_4n_{256} \\
 & & & & & \\
m_{256}n_1 & m_{256}n_2 & m_{256}n_3 & m_{256}n_4 & m_{256}n_5 & m_{256}n_{256}
\end{array}
$$

different types of filters but every filter has a different up-sampling filter to reconstruct the compressed image. In the explanation that follows, the test image used is Lena.tif where as three different types of test images were used to test the filters as described earlier.

4.3.1 Filter 1

In Filter 1, the down-sampling filter is designed to delete the even numbered rows from the actual image matrix array. This means rows m_2, m_4, m_6, m_8 and so on until m_{256} is deleted from the actual image matrix array. The compressed Lena'.tif image obtained from the down-sampling filter would be represented by the following matrix array:

$$\text{Lena' (m,n)} = \begin{array}{cccccc} m_1n_1 & m_1n_2 & m_1n_3 & m_1n_4 & m_1n_5 & m_1n_{256} \\ m_3n_1 & m_3n_2 & m_3n_3 & m_3n_4 & m_3n_5 & m_3n_{256} \\ m_5n_1 & m_5n_2 & m_5n_3 & m_5n_4 & m_5n_5 & m_5n_{256} \\ & & & & & \\ m_{253}n_1 & m_{253}n_2 & m_{253}n_3 & m_{253}n_4 & m_{253}n_5 & m_{253}n_{256} \\ m_{255}n_1 & m_{255}n_2 & m_{255}n_3 & m_{255}n_4 & m_{255}n_5 & m_{255}n_{256} \end{array}$$

The algorithm to develop the down-sampling filter, to delete the even rows is as follows:

- Load the original image (Lena.tif) into the MATLAB program and name it “I”.
- Delete the even numbered rows from the matrix array of the original image and rename the new matrix array as “del_even_row.tif”.
- Display the original image that is image “I”.
- Display the compressed image, that is image “del_even_row”.
- Save the compressed image as “del_even_row.tif”

The developed MATLAB program for the down-sampling filter to delete odd rows is as follows:

```
 I = imread('lena.tif');
del_even_row = I(1:2:end, : );
imagesc(I); colormap(gray)
imagesc(del_even_row); colormap(gray)
imwrite(del_even_row,'del_even_row.tif');
```

The flow chart in figure 4.4 below describes the steps in developing this down-sampling filter:

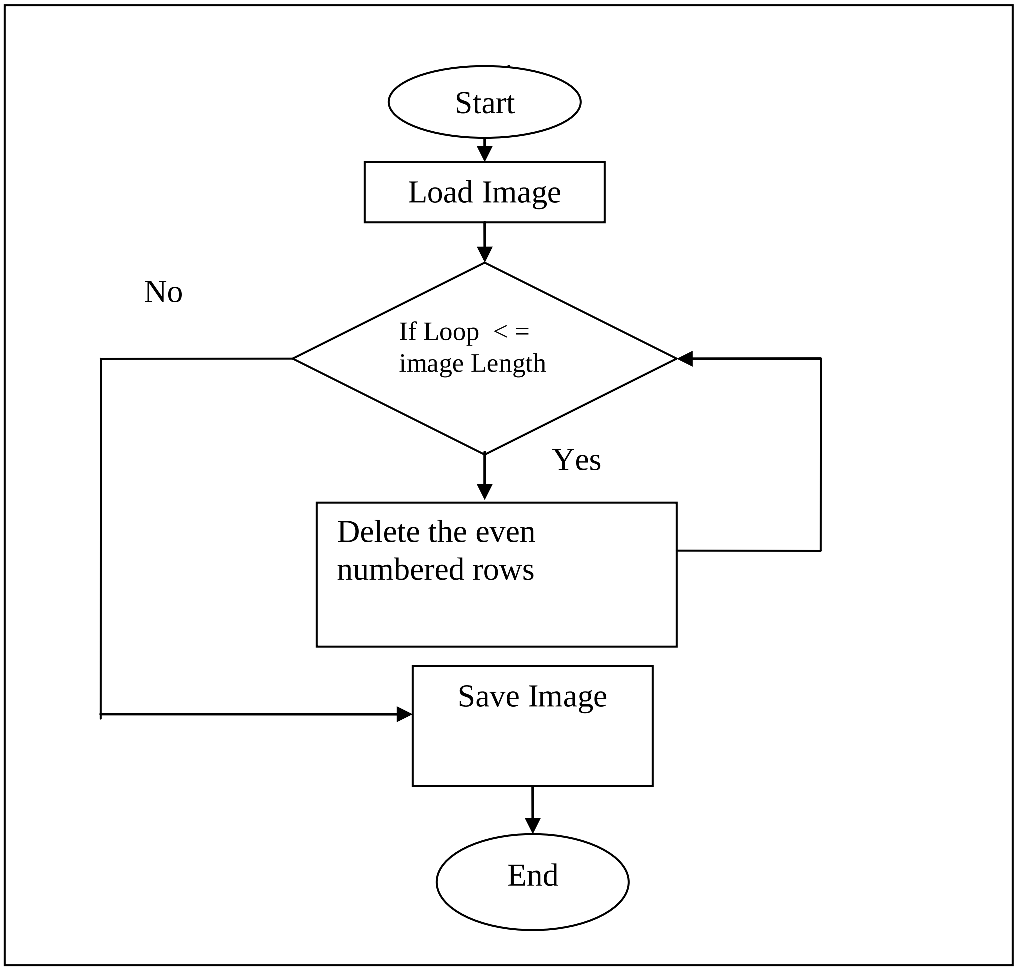


Figure 4.4: Flow chart for the down-sampling filter of Filter 1

Form the down-sampling filter developed previously, the pixels from the even numbered rows of the original matrix array were deleted. It is not possible to obtain the original pixel value to decompress the image. An up-sampling filter is developed to decompress the image so that the size of the image is the same as the original image.

The objective of the up-sampling filter is to construct an image Lena''.tif(256x256) which is of size 65536 bytes from the compressed image Lena'.tif(128x256) which is of size 32768 bytes.

In the design of Filter 1, the Up-sampling filter that was designed, functions to duplicate each of the odd numbered rows to replace the deleted even numbered rows. The matrix array below describes the process in detail.

Equation 4.3 below shows the matrix array of the compressed image Lena'.tif

$$\text{Lena' (m,n)} \quad = \quad \begin{matrix} m_1n_1 & m_1n_2 & m_1n_3 & m_1n_4 & m_1n_5 & & m_1n_{256} \\ m_3n_1 & m_3n_2 & m_3n_3 & m_3n_4 & m_3n_5 & & m_3n_{256} \\ m_5n_1 & m_5n_2 & m_5n_3 & m_5n_4 & m_5n_5 & & m_5n_{256} \\ & & & & & & \\ m_{253}n_1 & m_{253}n_2 & m_{253}n_3 & m_{253}n_4 & m_{253}n_5 & & m_{253}n_{256} \\ m_{255}n_1 & m_{255}n_2 & m_{255}n_3 & m_{255}n_4 & m_{255}n_5 & & m_{255}n_{256} \end{matrix}$$