Image Compression Using Wavelets

Karen Lees

May 2002

Supervisor: Dr. Joab Winkler

This report is submitted in partial fulfilment of the requirement for the degree of Master of Computer Science with Honours in Computer Science by Karen Lees.

Declaration

All sentences or passages quoted in this dissertation from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations which are not the work of the author of this dissertation have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this dissertation and the degree examination as a whole.

Name: Karen Lees

Signature:

Date:

Abstract

Images require substantial storage and transmission resources, thus image compression is advantageous to reduce these requirements. The report covers some background of wavelet analysis, data compression and how wavelets have been and can be used for image compression. An investigation into the process and problems involved with image compression was made and the results of this investigation are discussed. It was discovered that thresholding was had an extremely important influence of compression results so suggested thresholding strategies are given along with further lines of research that could be undertaken.

Acknowledgments

I would like to thank:

Dr Joab Winkler for his help and support throughout the year.

My parents for proof reading my report, paying my tuition fees and buying me a computer.

Matthew Dootson for proof reading sections of this report.

Samantha Smith for the use of her colour printer.

Those already mentioned, Graham Richmond, Fai Ko, James Leigh, Viktoria Maier and Mandy Brown for cheering me up during the past 3 years of University.

Contents

Ch.

1. Introduction

Often signals we wish to process are in the time-domain, but in order to process them more easily other information, such as frequency, is required. Mathematical transforms translate the information of signals into different representations. For example, the Fourier transform converts a signal between the time and frequency domains, such that the frequencies of a signal can be seen. However the Fourier transform cannot provide information on which frequencies occur at specific times in the signal as time and frequency are viewed independently. To solve this problem the Short Term Fourier Transform (STFT) introduced the idea of windows through which different parts of a signal are viewed. For a given window in time the frequencies can be viewed. However Heisenburg's Uncertainty Principle states that as the resolution of the signal improves in the time domain, by zooming on different sections, the frequency resolution gets worse. Ideally, a method of multiresolution is needed, which allows certain parts of the signal to be resolved well in time, and other parts to be resolved well in frequency. The power and magic of wavelet analysis is exactly this multiresolution.

Images contain large amounts of information that requires much storage space, large transmission bandwidths and long transmission times. Therefore it is advantageous to compress the image by storing only the essential information needed to reconstruct the image. An image can be thought of as a matrix of pixel (or intensity) values. In order to compress the image, redundancies must be exploited, for example, areas where there is little or no change between pixel values. Therefore images having large areas of uniform colour will have large redundancies, and conversely images that have frequent and large changes in colour will be less redundant and harder to compress.

Wavelet analysis can be used to divide the information of an image into approximation and detail subsignals. The approximation subsignal shows the general trend of pixel values, and three detail subsignals show the vertical, horizontal and diagonal details or changes in the image. If these details are very small then they can be set to zero without significantly changing the image. The value below which details are considered small enough to be set to zero is known as the threshold. The greater the number of zeros the greater the compression that can be achieved. The amount of information retained by an image after compression and decompression is known as the 'energy retained' and this is proportional to the sum of the squares of the pixel values. If the energy retained is 100% then the compression is known as 'lossless', as the image can be reconstructed exactly. This occurs when the threshold value is set to zero, meaning that the detail has not been changed. If any values are changed then energy will be lost and this is known as 'lossy' compression. Ideally, during compression the number of zeros and the energy retention will be as high as possible. However, as more zeros are obtained more energy is lost, so a balance between the two needs to be found.

The first part of the report introduces the background of wavelets and compression in more detail. This is followed by a review of a practical investigation into how compression can be achieved with wavelets and the results obtained. The purpose of the investigation was to find the effect of the decomposition level, wavelet and image on the number of zeros and energy retention that could be achieved. For reasons of time, the set of images, wavelets and levels investigated was kept small. Therefore only one family of wavelets, the Daubechies wavelets, was used. The images used in the investigation can be seen in Appendix B. The final part of the report discusses image properties and thresholding, two issues which have been found to be of great importance in compression.

2. Background

2.1. The Need for Wavelets

Often signals we wish to process are in the time-domain, but in order to process them more easily other information, such as frequency, is required. A good analogy for this idea is given by Hubbard[4], p14. The analogy cites the problem of multiplying two roman numerals. In order to do this calculation we would find it easier to first translate the numerals in to our number system, and then translate the answer back into a roman numeral. The result is the same, but taking the detour into an alternative number system made the process easier and quicker. Similarly we can take a detour into frequency space to analysis or process a signal.

2.1.1 Fourier Transforms (FT)

Fourier transforms can be used to translate time domain signals into the frequency domain. Taking another analogy from Hubbard[4] it acts as a mathematical prism, breaking up the time signal into frequencies, as a prism breaks light into different colours.

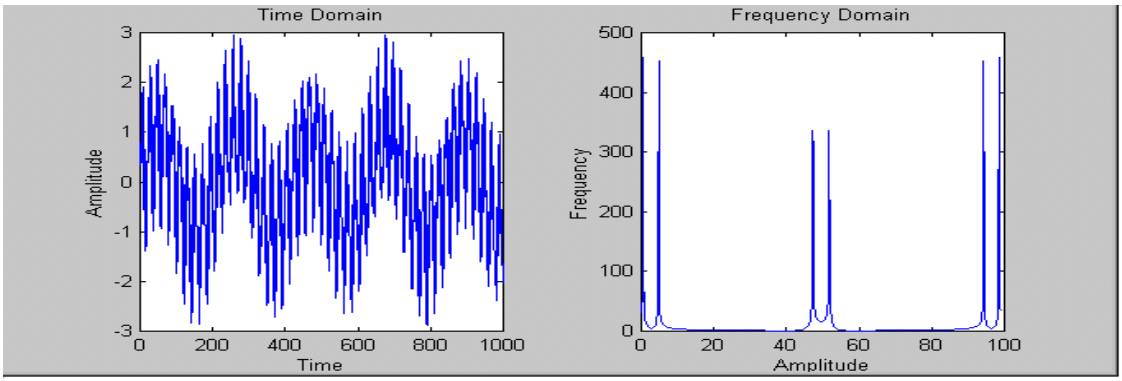


Figure 2.1 The left graph shows a signal plotted in the time domain, the right graph shows the Fourier transform of the signal.

The following equations can be used to calculate the Fourier transform of a time-domain signal and the inverse Fourier Transform [2]:

$$X(f) = \int_{-\infty}^{\infty} x(t).e^{-2j\Pi ft}.\delta t$$

$$x(t) = \int_{-\infty}^{\infty} X(f).e^{2j\Pi ft}.\delta f$$

x is the original signal
t is time
f is frequency
X is the Fourier transform.

Fourier transforms are very useful at providing frequency information that cannot be seen easily in the time domain. However they do not suit brief signals, signals that change suddenly, or in fact any non-stationary signals. The reason is that they show only what frequencies occur, not when these frequencies occur, so they are not much help when both time and frequency information is required simultaneously. In stationary signals, all frequency components occur at all times, so Fourier Transforms are very useful. Hubbard[4] helps to make this idea clearer by using the analogy of a musician; if a musician were told what notes were played during a song, but not any information about when to play them, he would find it difficult to make sense of the information. Luckily he has the tool

of a music score to help him, and in a parallel with this the mathematicians first tried to use the Short Term Fourier Transform (STFT), which was introduced by Gabor.

The STFT looks at a signal through a small window, using the idea that a sufficiently small section of the wave will be approximately a stationary wave and so Fourier analysis can be used. The window is moved over the entire wave, providing some information about what frequencies appear at what time.

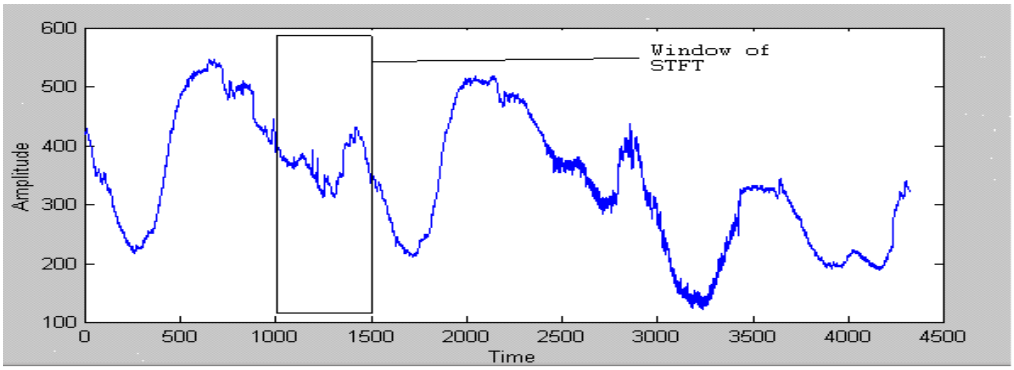


Figure 2.2 Example of a window used for STFT

The following equation can be used to compute a STFT. It is different to the FT as it is computed for particular windows in time individually, rather than computing overall time (which can be alternatively thought of as an infinitely large window). x is the signal, and w is the window.

$$STFT_x^w(t,f) = \int [x(t).w^*(t-t')].e^{-j2\Pi ft}\delta t \qquad [2]$$

This is an improvement as a time domain signal can be mapped onto a function of time and frequency, providing some information about what frequencies occur when. However using windows introduces a new problem; according to Heisenberg's Uncertainty principle it is impossible to know exactly what frequencies occur at what time, only a range of frequencies can be found. This means that trying to gain more detailed frequency information causes the time information to become less specific and visa versa. Therefore when using the STFT, there has to be a sacrifice of either time or frequency information. Having a big window gives good frequency resolution but poor time resolution, small windows provide better time information, but poorer frequency information.

2.1.2 Multiresolution and Wavelets

The power of Wavelets comes from the use of multiresolution. Rather than examining entire signals through the same window, different parts of the wave are viewed through different size windows (or resolutions). High frequency parts of the signal use a small window to give good time resolution, low frequency parts use a big window to get good frequency information.

An important thing to note is that the 'windows' have equal area even though the height and width may vary in wavelet analysis. The area of the window is controlled by Heisenberg's Uncertainty principle, as frequency resolution gets bigger the time resolution must get smaller.

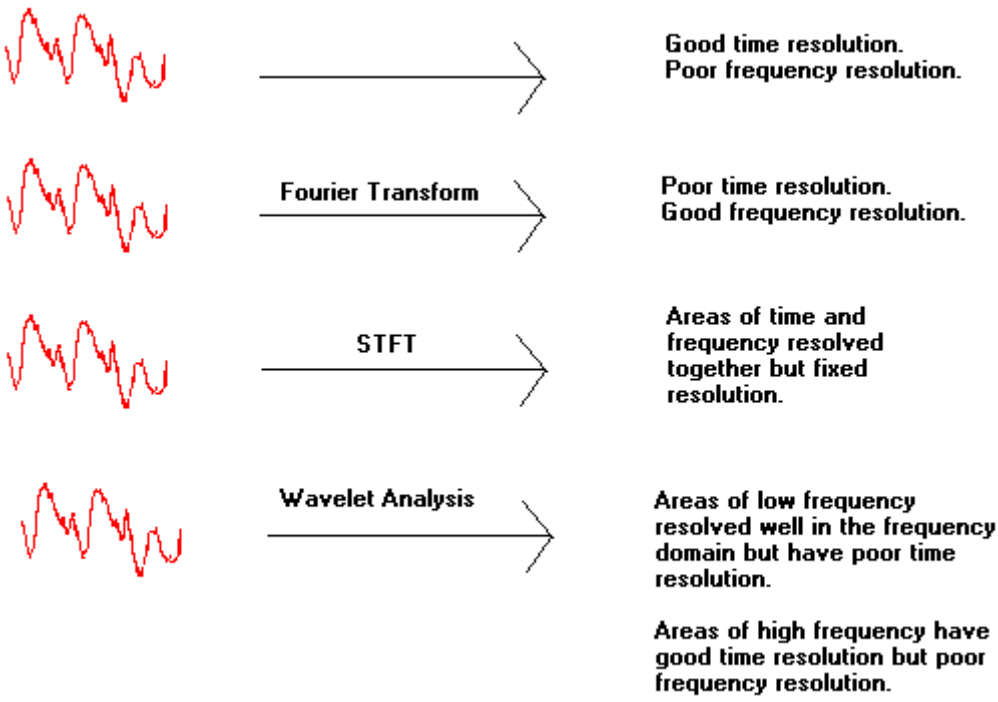


Figure 2.3 The different transforms provided different resolutions of time and frequency.

In Fourier analysis a signal is broken up into sine and cosine waves of different frequencies, and it effectively re-writes a signal in terms of different sine and cosine waves. Wavelet analysis does a similar thing, it takes a 'mother wavelet', then the signal is translated into shifted and scale versions of this mother wavelet.

2.1.3 The Continuous Wavelet Transform (CWT)

The continuous wavelet transform is the sum over all time of scaled and shifted versions of the mother wavelet ψ. Calculating the CWT results in many coefficients C, which are functions of scale and translation.

$$C(s,\tau) = \int_{-\infty}^{\infty} f(t)\psi(s,\tau,t).dt$$

The translation, τ, is proportional to time information and the scale, s, is proportional to the inverse of the frequency information. To find the constituent wavelets of the signal, the coefficients should be multiplied by the relevant version of the mother wavelet.

The scale of a wavelet simply means how stretched it is along the x-axis, larger scales are more stretched:

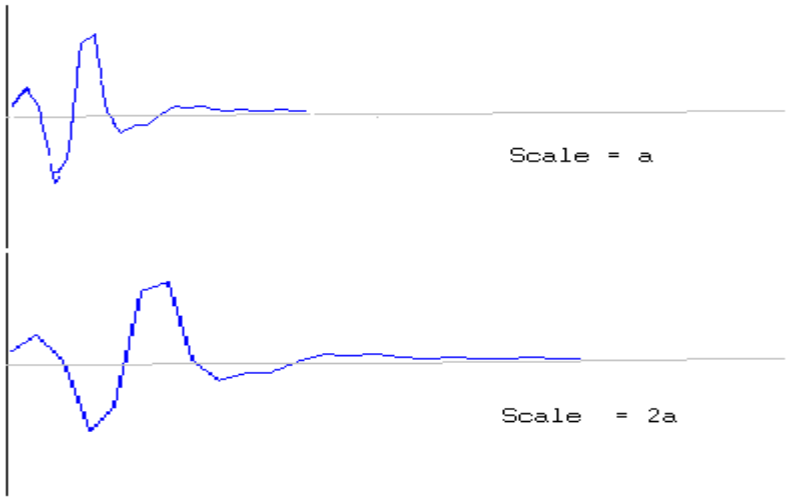


Figure 2.4 The db8 wavelet shown at two different scales

The translation is how far it has been shifted along the x-axis. Figure 2.5 shows a wavelet, figure 2.6 shows the same mother wavelet translated by k:

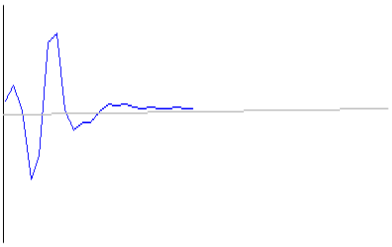


Figure 2.5

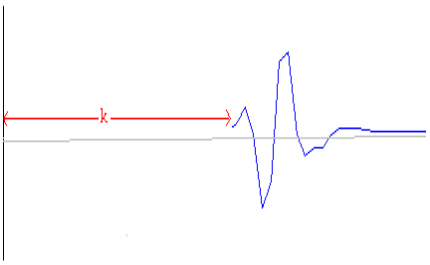


Figure 2.6 The same wavelet as in figure 2.5, but translated by k

The *Wavelet Toolbox User's Guide* [7] suggests five easy steps to compute the CWT coefficients for a signal.

1. Choose a wavelet, and compare this to the section at the start of the signal.
2. Calculate C, which should measure how similar the wavelet and the section of the signal are.
3. Shift the wavelet to the right by translation τ, and repeat steps 1 and 2, calculating values of C for all translations.
4. Scale the wavelet, and repeat steps 1-3.
5. Repeat 4 for all scales.

The coefficients produced can form a matrix at the different scale and translation values; the higher coefficients suggest a high correlation between the part of the signal and that version of the wavelet. Figure 2.7 shows a signal and a plot of the corresponding CWT coefficient matrix. The colours in the coefficients matrix plot show the relative sizes of the coefficients. The signal is very similar to the wavelet in light areas, dark area shows that the corresponding time and scale versions of the wavelet were dissimilar to the signal.

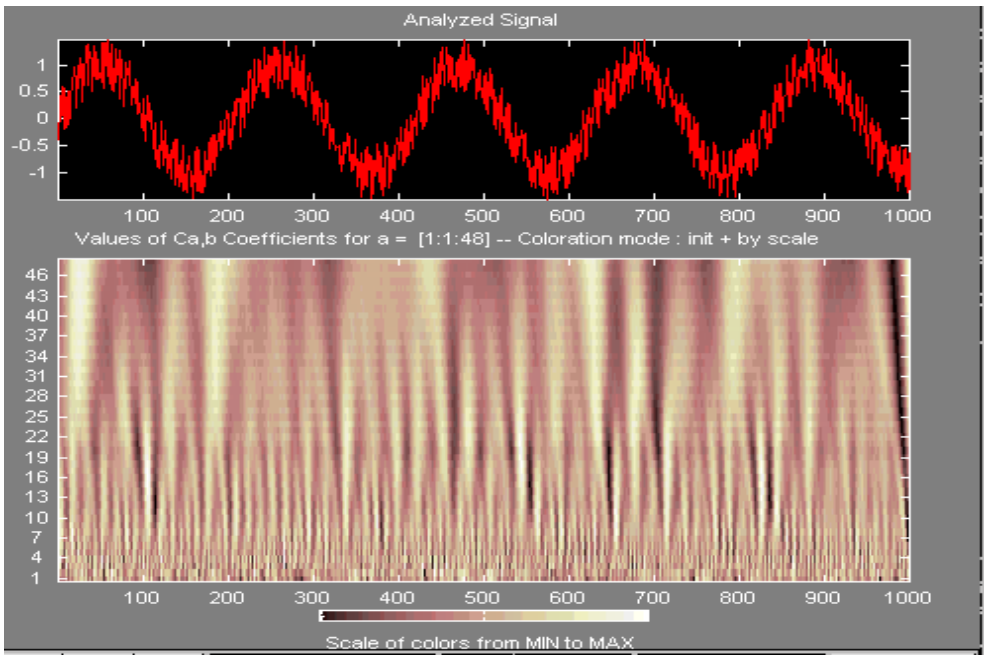


Figure 2.7 Screen print from Matlab Wavelet Toolbox GUI. The top graph shows the signal to be analysed with the CWT. The bottom plot shows the coefficients at corresponding scale and times. The horizontal axis is time, the vertical axis is scale.

2.1.4 Sampling and the Discrete Wavelet Series

In order for the Wavelet transforms to be calculated using computers the data must be discretised. A continuous signal can be sampled so that a value is recorded after a discrete time interval, if the Nyquist sampling rate is used then no information should be lost. With Fourier Transforms and STFT's the sampling rate is uniform but with wavelets the sampling rate can be changed when the scale changes. Higher scales will have a smaller sampling rate. According to Nyquist Sampling theory, the new sampling rate N_2 can be calculated from the original rate N_1 using the following:

$$N_2 = \frac{s_1}{s_2} N_1$$

where s_1 and s_2 are the scales. So every scale has a different sampling rate.

After sampling the Discrete Wavelet Series can be used, however this can still be very slow to compute. The reason is that the information calculated by the wavelet series is still highly redundant, which requires a large amount of computation time. To reduce computation a different strategy was discovered and Discrete Wavelet Transform (DWT) method was born.

2.1.5 DWT and subsignal encoding

The DWT provides sufficient information for the analysis and synthesis of a signal, but is advantageously, much more efficient.

Discrete Wavelet analysis is computed using the concept of filter banks. Filters of different cut-off frequencies analyse the signal at different scales. Resolution is changed by the filtering, the scale is changed by upsampling and downsampling. If a signal is put through two filters:

(i) a high-pass filter, high frequency information is kept, low frequency information is lost.
(ii) a low pass filter, low frequency information is kept, high frequency information is lost.

then the signal is effectively decomposed into two parts, a detailed part (high frequency), and an approximation part (low frequency). The subsignal produced from the low filter will have a highest frequency equal to half that of the original. According to Nyquist sampling this change in frequency range means that only half of the original samples need to be kept in order to perfectly reconstruct the signal. More specifically this means that upsampling can be used to remove every second sample. The scale has now been doubled. The resolution has also been changed, the filtering made the frequency resolution better, but reduced the time resolution.

The approximation subsignal can then be put through a filter bank, and this is repeated until the required level of decomposition has been reached. The ideas are shown in figure 2.8.

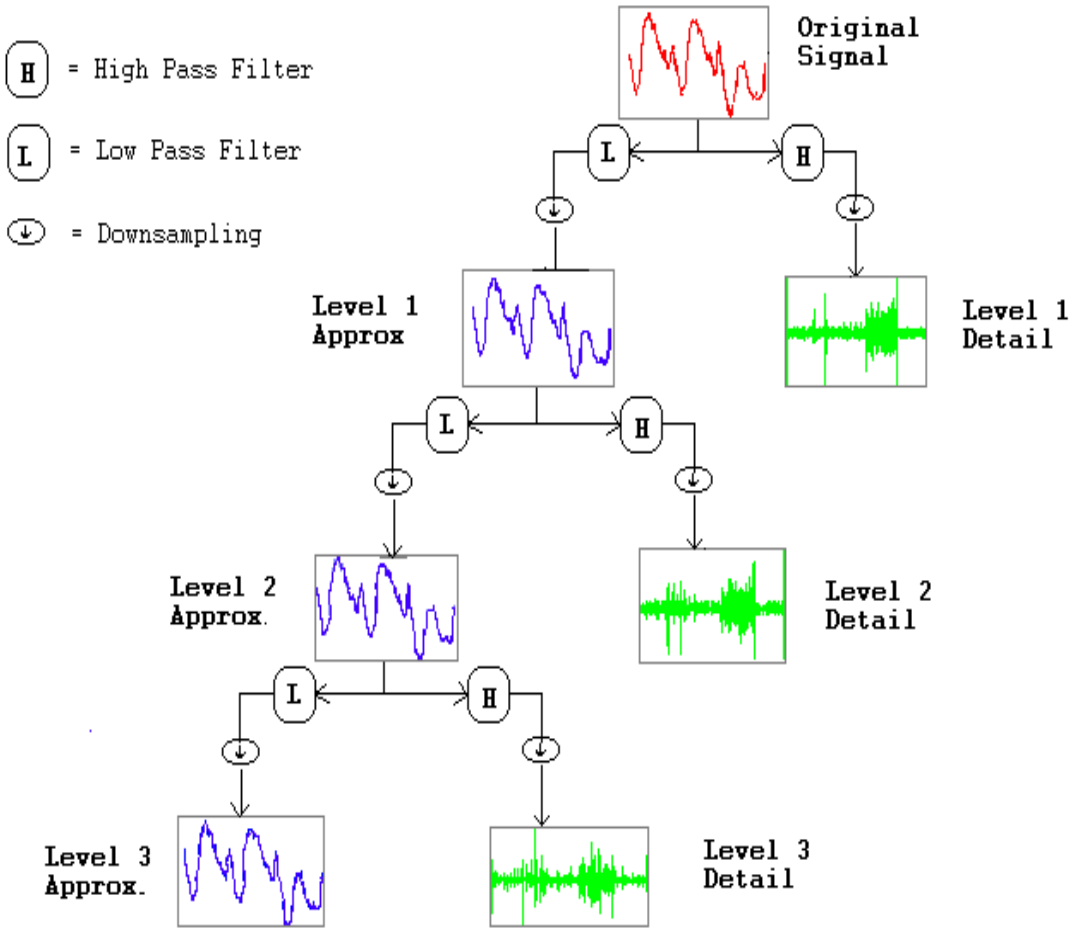


Figure 2.8

The DWT is obtained by collecting together the coefficients of the final approximation subsignal and all the detail subsignals.

Overall the filters have the effect of separating out finer and finer detail, if all the details are 'added' together then the original signal should be reproduced. Using a further analogy from Hubbard[4] this decomposition is like decomposing the ratio 87/7 into parts of increasing detail, such that:

$$87 / 7 = 10 + 2 + 0.4 + 0.02 + 0.008 + 0.0005$$

The detailed parts can then be re-constructed to form 12.4285 which is an approximation of the original number 87/7.

2.1.6 Conservation and Compaction of Energy

An important property of wavelet analysis is the conservation of energy. Energy is defined as the sum of the squares of the values. So the energy of an image is the sum of the squares of the pixel values, the energy in the wavelet transform of an image is the sum of the squares of the transform coefficients. During wavelet analysis the energy of a signal is divided between approximation and details signals but the total energy does not change. During compression however, energy is lost because thresholding changes the coefficient values and hence the compressed version contains less energy.

The compaction of energy describes how much energy has been compacted into the approximation signal during wavelet analysis. Compaction will occur wherever the magnitudes of the detail coefficients are significantly smaller than those of the approximation coefficients. Compaction is important when compressing signals because the more energy that has been compacted into the approximation signal the less energy can be lost during compression.

2.2 Image Compression

Images require much storage space, large transmission bandwidth and long transmission time. The only way currently to improve on these resource requirements is to compress images, such that they can be transmitted quicker and then decompressed by the receiver.

In image processing there are 256 intensity levels (scales) of grey. 0 is black and 255 is white. Each level is represented by an 8-bit binary number so black is 00000000 and white is 11111111. An image can therefore be thought of as grid of pixels, where each pixel can be represented by the 8-bit binary value for grey-scale.

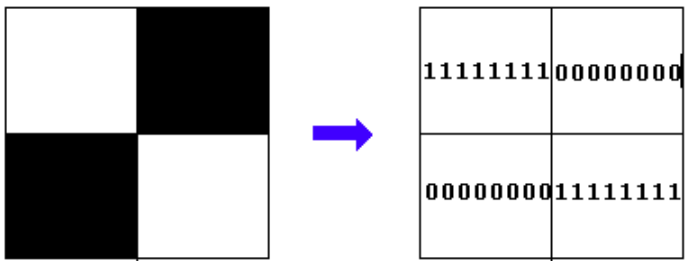


Figure 2.9

The resolution of an image is the pixels per square inch. (So 500dpi means that a pixel is 1/500th of an inch). To digitise a one-inch square image at 500 dpi requires 8 x 500 x500 = 2 million storage bits. Using this representation it is clear that image data compression is a great advantage if many images are to be stored, transmitted or processed.

According to [6] "Image compression algorithms aim to remove redundancy in data in a way which makes image reconstruction possible." This basically means that image compression algorithms try to exploit redundancies in the data; they calculate which data needs to be kept in order to reconstruct the original image and therefore which data can be 'thrown away'. By removing the redundant data, the image can be represented in a smaller number of bits, and hence can be compressed.

But what is redundant information? Redundancy reduction is aimed at removing duplication in the image. According to Saha there are two different types of redundancy relevant to images:

(i) Spatial Redundancy – correlation between neighbouring pixels.
(ii) Spectral Redundancy - correlation between different colour planes and spectral bands.

Where there is high correlation, there is also high redundancy, so it may not be necessary to record the data for every pixel.

There are two parts to the compression:

1. Find image data properties; grey-level histogram, image entropy, correlation functions etc..
2. Find an appropriate compression technique for an image of those properties.

2.2.1 Image Data Properties

In order to make meaningful comparisons of different image compression techniques it is necessary to know the properties of the image. One property is the image entropy; a highly correlated picture will have a low entropy. For example a very low frequency, highly correlated image will be compressed well by many different techniques; it is more the image property and not the compression algorithm that gives the good compression rates. Also a compression algorithm that is good for some images will not necessarily be good for all images, it would be better if we could say what the best compression technique would be given the type of image we have. One way of calculating entropy is suggested by [6] :

If an image has G grey-levels and the probability of grey-level k is P(k) the entropy H_e is:

$$H_e = -\sum_{k=0}^{G-1} P(k)\log_2[P(k)] \qquad [6]$$

Information redundancy, r, is

$$r = b - H_e \qquad [6]$$

where b is the smallest number of bits for which the image quantisation levels can be represented.

Information redundancy can only be evaluated if a good estimate of image entropy is available, but this is not usually the case because some statistical information is not known. An estimate of H_e can be obtained from a grey-level histogram.

If h(k) is the frequency of grey-level k in an image f, and image size is MxN then an estimate of P(k) can be made:

$$\tilde{P}(k) = \frac{h(k)}{MN} \qquad [6]$$

Therefore,

$$\tilde{H}_e = -\sum_{k=0}^{G-1} \tilde{P}(k)\log_2[\tilde{P}(k)] \text{ and } \tilde{r} = b - \tilde{H}_e \qquad [6]$$

The Compression ratio $K = b / H_e$

2.2.2 Compression techniques

There are many different forms of data compression. This investigation will concentrate on transform coding and then more specifically on Wavelet Transforms.

Image data can be represented by coefficients of discrete image transforms. Coefficients that make only small contributions to the information contents can be omitted. Usually the image is split into blocks (subimages) of 8x8 or 16x16 pixels, then each block is transformed separately. However this does not take into account any correlation between blocks, and creates "blocking artifacts" , which are not good if a smooth image is required.

However wavelets transform is applied to entire images, rather than subimages, so it produces no blocking artefacts. This is a major advantage of wavelet compression over other transform compression methods.

Thresholding in Wavelet Compression

For some signals, many of the wavelet coefficients are close to or equal to zero. Thresholding can modify the coefficients to produce more zeros. In Hard thresholding any coefficient below a threshold λ, is set to zero. This should then produce many consecutive zero's which can be stored in much less space, and transmitted more quickly by using entropy coding compression.

An important point to note about Wavelet compression is explained by Aboufadel[3]:

"The use of wavelets and thresholding serves to process the original signal, but, to this point, no actual compression of data has occurred".

This explains that the wavelet analysis does not actually compress a signal, it simply provides information about the signal which allows the data to be compressed by standard entropy coding techniques, such as Huffman coding. Huffman coding is good to use with a signal processed by wavelet analysis, because it relies on the fact that the data values are small and in particular zero, to compress data. It works by giving large numbers more bits and small numbers fewer bits. Long strings of zeros can be encoded very efficiently using this scheme. Therefore an actual percentage compression value can only be stated in conjunction with an entropy coding technique. To compare different wavelets, the number of zeros is used. More zeros will allow a higher compression rate, if there are many consecutive zeros, this will give an excellent compression rate.

2.2.3 Images in MATLAB

The project has involved understanding data in MATLAB, so below is a brief review of how images are handled. Indexed images are represented by two matrices, a colormap matrix and image matrix.

(i) The colormap is a matrix of values representing all the colours in the image.
(ii) The image matrix contains indexes corresponding to the colour map colormap.

A colormap matrix is of size Nx3, where N is the number of different colours in the image. Each row represents the red, green, blue components for a colour.

e.g. the matrix $\begin{bmatrix} r1 & g1 & b1 \\ r2 & g2 & b2 \end{bmatrix}$

represents two colours, the first have components r1, g1, b1, and the second having the components r2, g2 and b2.

The wavelet Toolbox only supports indexed images that have linear, monotonic colormaps. Often colour images need to be pre-processed into a grey scale image before using wavelet decomposition.

The Wavelet Toolbox User's Guide [7] provides some sample code to convert colour images into grey scale (section 2-87), this will be useful if I need to put any images into MATLAB.

2.3. Wavelets and Compression

Wavelets are useful for compressing signals but they also have far more extensive uses. They can be used to process and improve signals, in fields such as medical imaging where image degradation is not tolerated they are of particular use. They can be used to remove noise in an image, for example if it is of very fine scales, wavelets can be used to cut out this fine scale, effectively removing the noise.

2.3.1 The Fingerprint example

The FBI have been using wavelet techniques in order to store and process fingerprint images more efficiently. The problem that the FBI were faced with was that they had over 200 Million sets of fingerprints, with up to 30,0000 new ones arriving each day, so searching through them was taking too long. The FBI thought that computerising the fingerprint images would be a better solution, however it was estimated that checking each fingerprint would use 600Kbytes of memory and even worse 2000 terabytes of storage space would be required to hold all the image data.

The FBI then turned to wavelets for help, adapting a technique to compress each image into just 7% of the original space. Even more amazingly, according to Kiernan[8], when the images are decompressed they show "little distortion". Using wavelets the police hope to check fingerprints within 24 hours.

Earlier attempts to compress images used the JPEG format; this breaks an image into blocks eight pixels square. It then uses Fourier transforms to transform the data, then compresses this. However this was unsatisfactory, trying to compress images this way into less than 10% caused "tiling artefacts" to occur, leaving marked boundaries in the image. As the fingerprint matching algorithm relies on accurate data to match images, using JPEG would weaken the success of the process.

However wavelets don't create these "tiles" or "blocks", they work on the image as a whole, collecting detail at certain levels across the entire image. Therefore wavelets offered brilliant compression ratios and little image degradation; overall they outperformed the techniques based on Fourier transforms.

The basic steps used in the fingerprint compression were:

(1) Digitise the source image into a signal s
(2) Decompose the signal s into wavelet coefficients
(3) Modify the coefficients from w, using thresholding to a sequence w'.
(4) Use quantisation to convert w' to q.
(5) Apply entropy encoding to compress q to e.

2.3.2 2D Wavelet Analysis

Images are treated as two dimensional signals, they change horizontally and vertically, thus 2D wavelet analysis must be used for images. 2D wavelet analysis uses the same 'mother wavelets' but requires an extra step at every level of decomposition.

The 1D analysis filtered out the high frequency information from the low frequency information at every level of decomposition; so only two subsignals were produced at each level.

In 2D, the images are considered to be matrices with N rows and M columns. At every level of decomposition the horizontal data is filtered, then the approximation and details produced from this are filtered on columns.

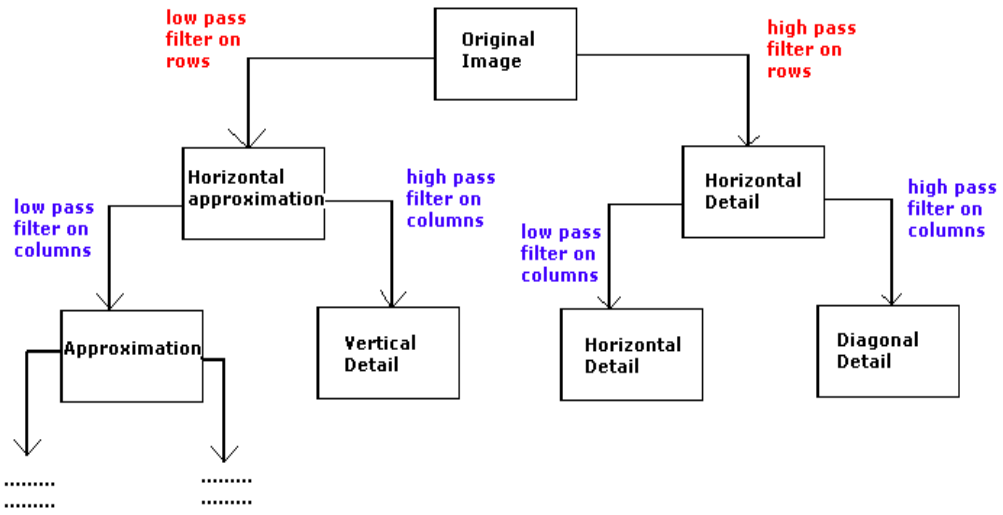


Figure 2.11

At every level, four sub-images are obtained; the approximation, the vertical detail, the horizontal detail and the diagonal detail. Below the Saturn image has been decomposed to one level. The wavelet analysis has found how the image changes vertically, horizontally and diagonally.

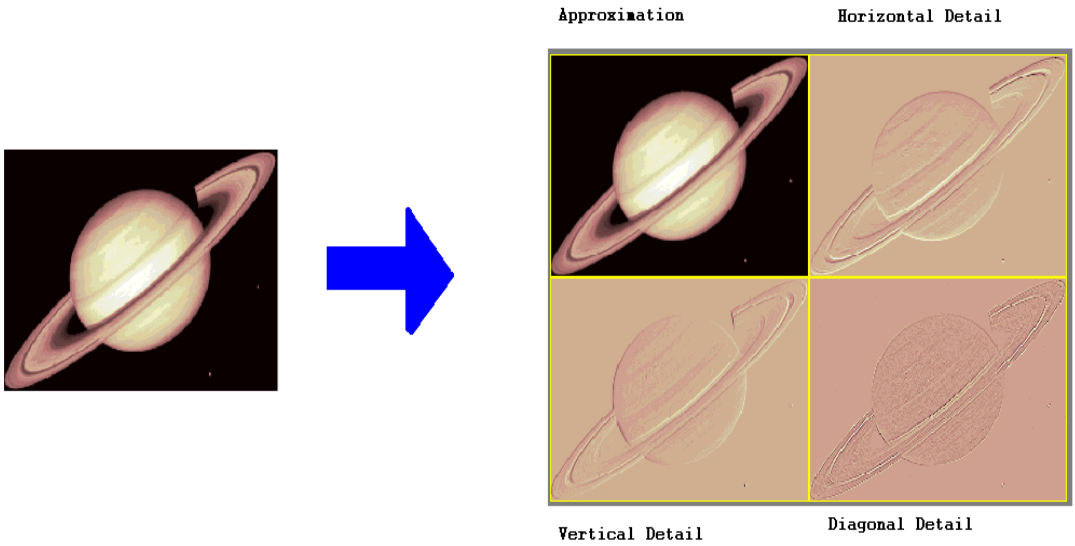


Figure 2.12 2-D Decomposition of Saturn Image to level 1

To get the next level of decomposition the approximation sub-image is decomposed, this idea can be seen in figure 2.13:

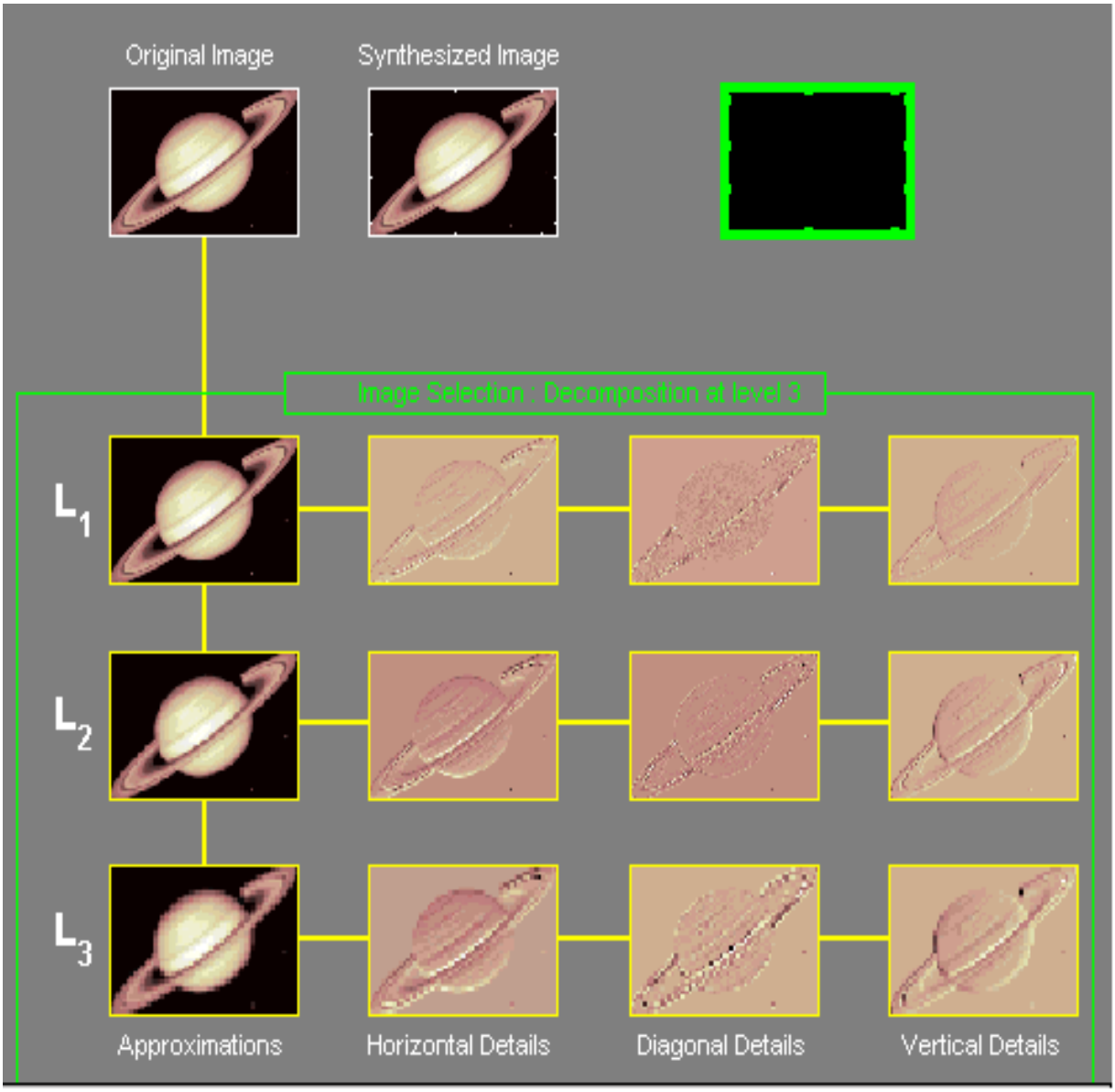


Figure 2.13 A screen print from the Matlab Wavelet Toolbox GUI showing the saturn image decomposed to level 3. Only the 9 detail sub-images and the final sub-image is required to reconstruct the image perfectly.

2.3.3 Wavelet Compression in MATLAB

MATLAB has two interfaces which can be used for compressing images, the command line and the GUI. Both interfaces take an image decomposed to a specified level with a specified wavelet and calculate the amount of energy loss and number of zero's.

When compressing with orthogonal wavelets the energy retained is [7]:

$$\frac{100 * (\text{vector - norm(coeffs of the current decomposition, 2)})^2}{(\text{vector - norm(original signal, 2)})^2}$$

The number of zeros in percentage is defined by [7]:

$$\frac{100 * (\text{number of zeros of the current decomposition})}{(\text{number of coefficients})}$$

To change the energy retained and number of zeros values, a threshold value is changed. The threshold is the number below which detail coefficients are set to zero. The higher the threshold value, the more zeros can be set, but the more energy is lost. Thresholding can be done globally or locally. Global thresholding involves thresholding every subband (sub-image) with the same threshold value. Local thresholding involves uses a different threshold value for each subband.

3. Implementation

3.1. Choosing the Images

The images used were selected from a range of over 40 images provided in Matlab. A script was programmed to calculate the entropy H_e of an image using the equations:

$$\widetilde{P}(k) = \frac{h(k)}{MN}$$

$$\widetilde{H}_e = -\sum_{k=0}^{G-1} \widetilde{P}(k) \log_2[\widetilde{P}(k)] \qquad [6]$$

The script read through the image matrix noting the frequency, h ,of each intensity level, k. Images that were not already in a matrix format had to be converted using the imread function in Matlab. For example to convert an image file tire.tif to a Matlab matrix, X, the following code could be use:

```
X =double( imread('tire.tif'));
```

The entropy could then be calculated using:

```
He = imageEntropy(X);
```

The entropy results showed a range of entropies from 0.323 to 7.7804 and a sample of 9 images was selected to represent the full range with intervals as even as possible. Two further images were added to act as controls and help confirm or disprove the effect of image entropy on compression. Since the image entropy of control 1 ('Saturn') was similar to the image 'Spine' similar results would tend to confirm the significance of image entropy. This also applied to control 2 ('Cameraman') and 'Tire'.

Image	Image Entropy
Text	0.323
Circbw	0.9996
Wifs	1.9754
Julia	3.1812
Spine	4.0935
Saturn (control 1)	4.114
Woman	5.003
Facets	6.0264
Tire	6.9265
Cameraman (control 2)	7.0097
Bonemarr	7.7804

3.2. Collecting Results

3.2.1 The Results Collected

The results that were collected were values for percentage energy retained and percentage number of zeros. These values were calculated for a range of threshold values on all the images, decomposition levels and wavelets used in the investigation.

The energy retained describes the amount of image detail that has been kept, it is a measure of the quality of the image after compression. The number of zeros is a measure of compression. A greater percentage of zeros implies that higher compression rates can be obtained.

3.2.2 How the Results were collected

Results were collected using the wdencmp function from the Matlab Wavelet toolbox, this returns the L^2 recovery (energy retained) and percentage of zeros when given the following inputs:

1. 'gbl' or 'lvd' for global or level dependent thresholding
2. an image matix
3. a wavelet name
4. level of decomposition
5. threshold value(s)
6. 's' or 'h' for sort or hard thresholding
7. whether the approximation values should be thresholded

An automated script (calcres.m) was written which took as inputs an image, a wavelet and a level. It calculated 10 appropriate threshold levels to use (see section 3.3) and then collected the energy retained and percentage of zeros using wdencmp with the given image, wavelet, decomposition level for each of the 10 threshold values.

For example, *RES =calcres (X, 'db1', 2)* would give a results matrix RES for the image matrix X, wavelets 'db1' and level 2 of decomposition. The RES matrix has three columns, the first is threshold value, the second is percentage of zeros and the third is energy retained.

3.3 Choosing the threshold values

There are a number of different options for thresholding. These include

1. The approximation signal is thresholded or not thresholded
2. Level dependent or global threshold values
3. Threshold different areas of an image with different threshold values

3.3.1 Thresholding for Results set 1

The first set of results used the simplest combination which was to use global thresholding and not to threshold the approximation signals. To get an even spread of energy values, the function 'calcres' first calculated the decomposition coefficients up to the level given as input. The script then calculated 10 values to be evenly spread along the range 0…max_c, where max_c is the maximum value in the coefficients matrix. (see figure 3.1)

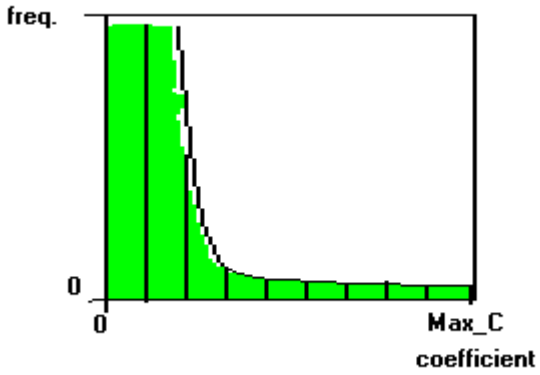


Figure 3.1 The spread of coefficients, 10 points are sampled**.**

However it was observed that the results for energy retained and percentage zeros only changed significantly between the first 4 or 5 thresholds. The most dramatic compression and energy loss occurred before reaching a threshold of 0.25max_c. The reason for this must be because a large percentage of coefficients and energy occurs at the lower values, therefore setting these values to zero would change a lot of the coefficients, and increase the compression rate. Setting the higher values to zero had little effect on compression because there weren't many coefficients to affect.

Therefore the threshold calculations were changed to zoom in on the first quarter of possible thresholds (figure 3.2). The idea of this was to get a better view of how the energy was lost with thresholding, the optimal compromise between energy loss and compression, if any were possible, was more likely to be contained within this section.

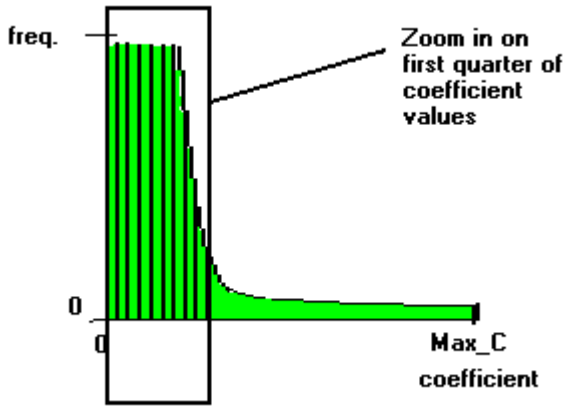


Figure 3.2

3.3.2 Thresholding for Results set 2

This used local thresholding (see figure 3.3). The function 'calcres2' was written to calculate the coefficient matrices for each level and direction (horizontal, vertical and diagonal), the maximum value

for each level was noted. Again 10 sample of threshold values were taken from the range 0..max_c for each local value. The first thresholds were all zero, the 10th set contained the maximum values for each.

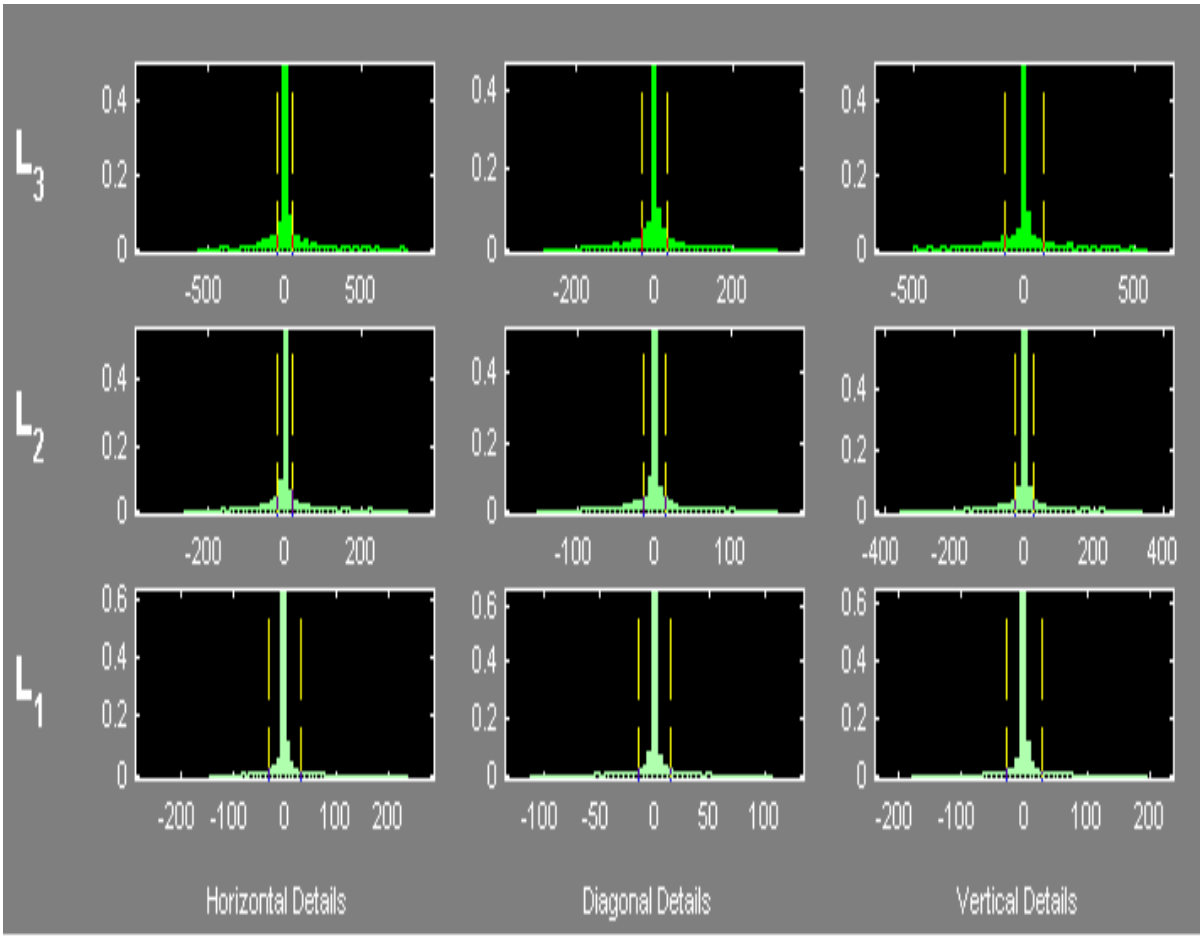


Figure 3.3 Each level and direction subsignal can be thresholded differently. The Level 3 horizontal would be tested with thresholds in the range 0 to 880 but the level 1 diagonal only requires a range of thresholds 0 to 120.

3.4 Collecting results for all images

The function 'calcres' (or 'calcres2') was used to get one set of results, for a particular combination of image, wavelet and decomposition level. However results were required for many images with many wavelets at many decomposition levels. Therefore an automated script was written (getwaveletresults.m), which follows the following algorithm:

```
For each image,
        For each wavelet,
                For level=1 to 5,
                        RES = calcres(image, wavelet, level)
                        Save RES
                end
        end
end
```

The list of images and wavelets to use are loaded from a database, using the Database Toolbox. (see section 3.5). The results are then saved in the database.

3.5 Using the Database Toolbox

The Database Toolbox was used to provide a quick and relatively simple solution to problems involved with saving the results. Originally the results were saved to a large matrix but in Matlab the matrix couldn't include strings, so the names of images and wavelets could not be included in the matrix. The initial solution was to give each image and wavelet a numeric identifier that could be put into the matrix. The image names and wavelet names could then be written in separate files, and the ID values linked to their position in this file. However this lead to concurrency problems, if the image file was changed then the results pointed to the wrong images and thus were wrong. The second idea was to save a matrix of strings in the same file that the results matrix was saved, however if the strings are of different lengths they can not go into the same matrix. Thus database toolbox allows Matlab to communicate with a database e.g. MS Access database in order to import and export data. An advantage of saving results into a database of this sort is that SQL statement can be used to retrieve certain sets of data and thus makes the analysis of results a lot easier.

The database contained 3 tables: Images, Results and Wavelets

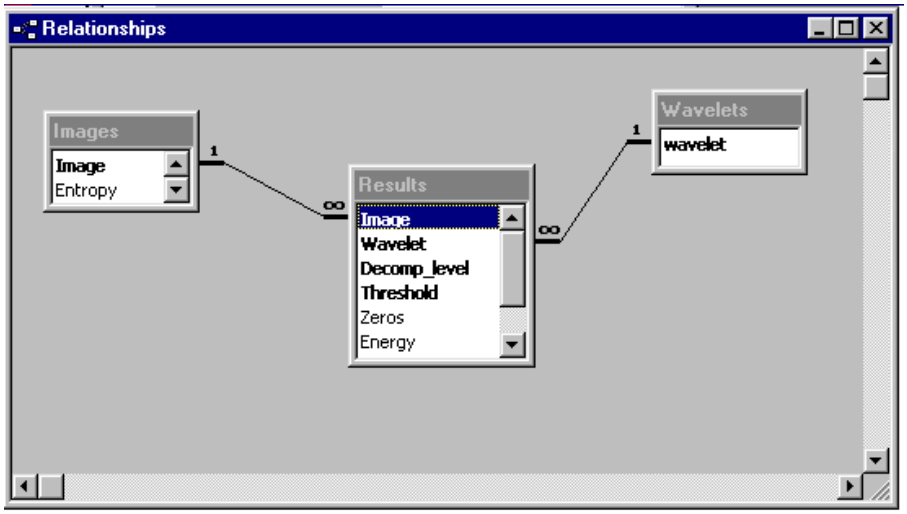


Figure 3.5

3.6 Method of Analysis

In order to analyse the results, graphs were plotted of energy retained against percentage of zeros. In an ideal situation an image would be compressed by 100% whilst retaining 100% of the energy. The graph would be a straight line similar to:

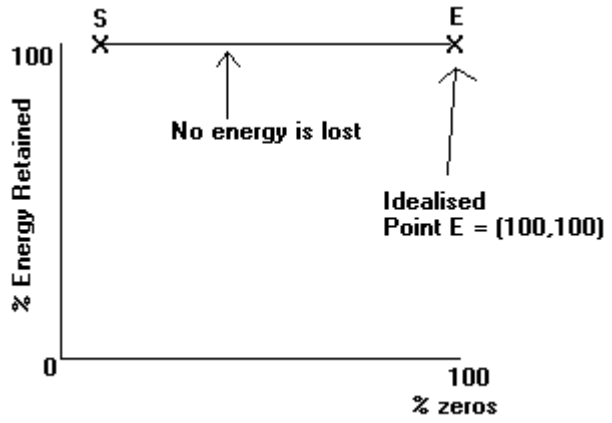


Figure 3.6

The best threshold to use would be one that produced 100% zeros whilst retaining 100% of the energy , corresponding to point E above. However, since energy retained decreases as the number of zeros increases, it is impossible for the image to have the same energy if the values have been changed by thresholding. Thus all the graphs produced had the general form shown as in Figure 3.7.

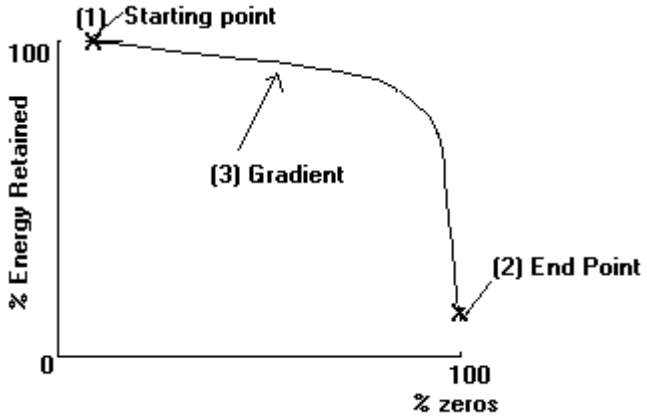


Figure 3.7

Important things to analyse about the curves were:

1. The starting point: the x co-ordinate gives the percentage of zeros with no thresholding and therefore shows how much the image could be compressed without thresholding.
2. The end point: this shows how many zeros could be achieved with the investigated thresholds and how much energy is lost by doing so.
3. The changing gradient of the curve: this shows how rapidly energy is lost when trying to compress the image. It is the ability of wavelets to retain energy during compression that is important, so energy being lost quickly suggests that a wavelet is not good to use.

3.6.1 Defining the Best Result

In figure 3.7 the starting point indicates the best energy retention and the end point indicates the best compression. In practice compressing an image requires a 'trade-off' between compression and energy loss. It may not be better to compress by an extra small amount if this would cause a dramatic loss in energy.

The Distance, ***D***

For the purposes of this study it was decided to define the best 'trade-off' to be the point T on the curve which was closest to the idealised point E. This was calculated using simple Pythagoras Theorem with the ten known points on each graph. For the rest of the study the distance between each point T and point E will be know simply as ***D***.

$$D = \sqrt{(100 - t_x)^2 + (100 - t_y)^2}$$

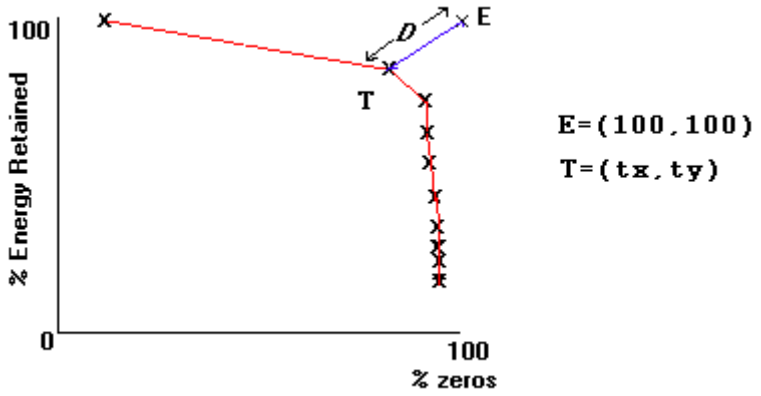


Figure 3.8

By calculating ***D*** for each result we can take the minimum to be the best, in other words the most efficient at retaining energy while compressing the signal. This will not necessarily be the best to use for all situations but be the most efficient found in the results.

Energy Loss per percentage Zero Ratio

A second possible measure of the best result is the ratio of Energy Loss per percentage Zero. This can be calculated for each result by the following equation:

$$\text{Energy Loss per percentage Zero} = \frac{100\% - \%\text{Energy Retained}}{\%\text{Zeros}}$$

This gives a measure of how much energy is lost by compressing the images, so a smaller value for the ratio would mean the compression is less lossy, and therefore better.

4. Analysis of Results: Set 1 (Global Thresholding)

Results Set 1	
Images	text, circbw, wifs, julia, spine, saturn, woman, facets, tire, cameraman, bonemarr
Wavelets	db1. db3, db5, db7, db9
Decomposition Levels	1,2,3,4,5
Thresholding	Global. Based on coefficients of final level. Sampled between 0 and 0.25max_Coefficient.

4.1 The Effect of Changing the Decomposition Levels

At first glance the results showed a clear pattern, in that decomposing the images to greater levels increased the compression but reduced the energy retention. An example of this is shown below (figure 4.1).

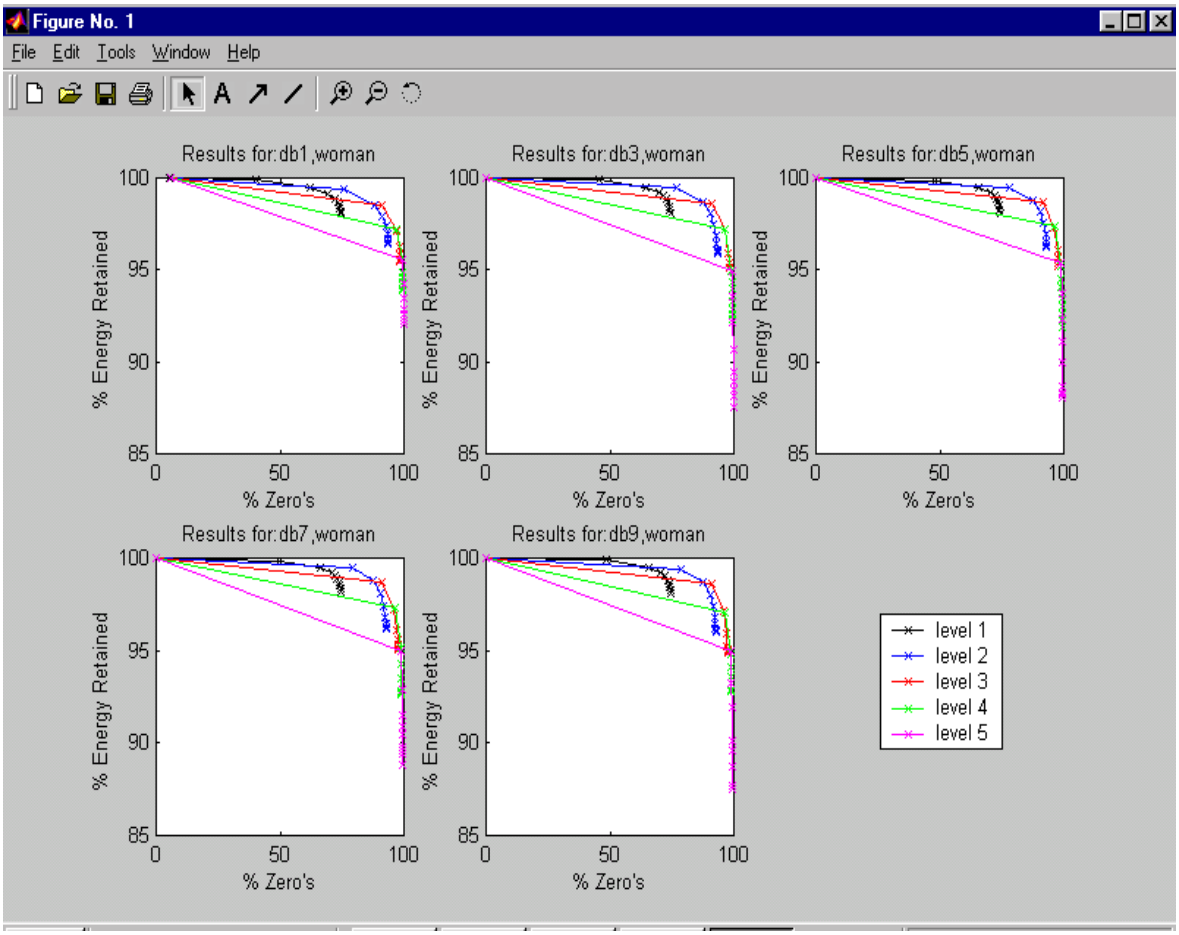


Figure 4.1

After analysing the graphs it could be seen that at higher decomposition levels:

1. The percentage of zeros at 100% energy retention was higher. This suggested a better compression rate had been gained by simply analysing at a deeper level without the need for thresholding (as this point corresponds to a global threshold of 0).
2. The end point was at a higher percentage of zeros but lower energy retained. This suggested that more compression was obtained by decomposing an image to greater levels but by doing so much energy was lost.
3. The gradient is steeper at higher levels, suggesting that more energy is lost for every % compression gained.

However on deeper analysis the conclusions 1-3 about the graphs could be false. This is due to how the global threshold values were calculated. The threshold values were chosen from the range 0..max_C, where max_C is the maximum coefficient in the decomposition. However this value comes from the final approximation subsignal, and increases with level of decomposition. Even though the coefficients

for level one details were the same however many levels were reached, the thresholds depended on the number of levels used. This resulted in an increased percentage of zeros and a reduction in the energy. The patterns shown in Figure 4.1 came not from the decomposition level changes but because of how the thresholds had been changed. It was decided that patterns could be seen more easily if local thresholding was used, this would mean each coefficient matrix was thresholded with values relevant to the values it contained, which wouldn't be affected by decomposing to greater levels. The beauty of how wavelets divide the information up into approximation and detail subsignals means that local thresholding is very simple and yet can provide great power in choosing the number of zeros and energy retention.

It was difficult to come to any firm conclusions about the gradient as only 10 points had been sampled and therefore simply joining the points did not show the true shape of the curve.

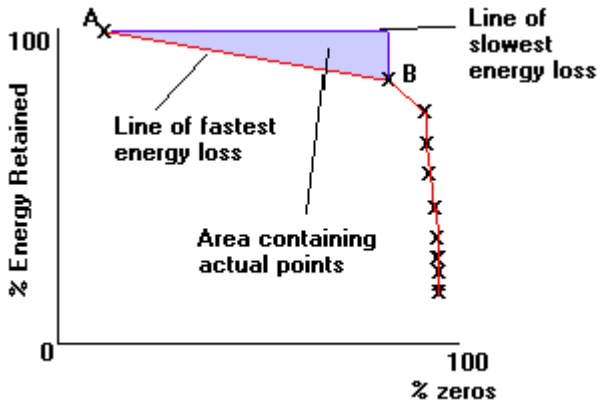


Figure 4.2

For example in figure 4.2, a straight line could be used to join points A and B, this would be pessimistic view of how the energy is lost. The true curve may not be this steep, in fact it might be that little energy is lost up to a certain point after which there is a sudden drop in energy retained. The real curve is likely to be somewhere in between the two, showing a gradual change in energy as the compression is increased through thresholding. The actual gradient will depend on the image and wavelet used, as these decide the co-efficient values, and therefore how thresholds will affect them. A sudden and relatively large change in energy would suggest that there was a substantial number of coefficients of a certain value and the threshold value used had just removed them. A very smooth and gradual change in energy and compression suggests that the energy is spread evenly throughout values, such that there are a large number of small valued coefficients and a small number of large valued coefficients.

In conclusion, in calculating results set 1 the effect of thresholding and in particular global thresholding had been severely underestimated. What has been learned from the results is that the choice of threshold values is extremely important in compression. Wavelets and decomposition levels divide the energy of an image between different subsignals, which in itself can provide compression if the detail coefficients are zero. However, to get the best compression rates thresholding is required. Thresholds hold the power to decide which energy is kept and which is lost. There is a huge range of potential thresholds and each detail matrix deserves to have its own local threshold if it is to provide the best mix between energy retained and percentage of zeros.

Local thresholding implies that the detail matrices for a given level, direction, image and wavelet will be thresholded with the same values whatever the maximum level of decomposition is used. Thus any differences seen in the graphs of results with local thresholding will be a result of the decomposition level rather than the threshold values used.

4.2 The Effect of Wavelet Types

No conclusive pattern could be found in from looking at the %Energy Retained - %Zeros graphs with different wavelets. The wavelets definitely changed the energy and number of zeros but how well the wavelet worked seemed very dependent on the images. This is not surprising because the ability of a wavelet to compact energy depends on the energy spread within the image. As explained in section 4.1 more reliable conclusions can be drawn from compressing the images with local thresholds.

4.3 The Effect of Changing the Images

The images affected how energy was lost with thresholding, this is because the image affects the values and spread of coefficients in the approximation and detail matrices. There didn't appear to be any pattern between the image entropy and how the compression differed. This suggested that the image entropy calculated did not characterise the image's compressibility with wavelets. The image entropy was a measure of disorder in the image, it looked at the frequency of different intensity values. However wavelets depend on how the energy is changing, not just what energy is contained within the image. A further discussion on image properties can be found in section 6 of this report.

5. Analysis of Results: Set 2 (Local Thresholding)

Results Set 1	
Images	text, circbw, wifs, julia, spine, saturn, woman, facets, tire, cameraman, bonemarr
Wavelets	db1. db3, db5, db7, db9
Decomposition Levels	1,2,3,4,5
Thresholding	Local. Based on coefficient matrix for each level and direction. Sampled between 0 and max_C of each matrix.

5.1 The Effect of the Decomposition Level

The higher the decomposition level the higher percentage of zeros obtained with no thresholding. This is because decomposing to greater levels means that a higher percentage of coefficients come from detail subsignals. Detail subsignals generally have a smaller range of values than the approximation subsignals, ideally zero values. Therefore this pattern shows that as decomposition level increases, more detail is filtered out with value zero.

The energy loss at the maximum threshold was higher at greater levels of decomposition. This again is because at higher levels of decomposition there is a higher proportion of the coefficients in the detail subsignals. So a higher percentage of the energy is lost by removing all the detail.

Figure 5.1 is an example of how the decomposition levels affected the number of zeros and energy retained for the 'woman' image. This can be compared with figure 4.1 to show the difference between local and global thresholding. The graphs for local thresholding show a much smoother and gradual loss in energy and increase in compression than the graphs for global thresholding.

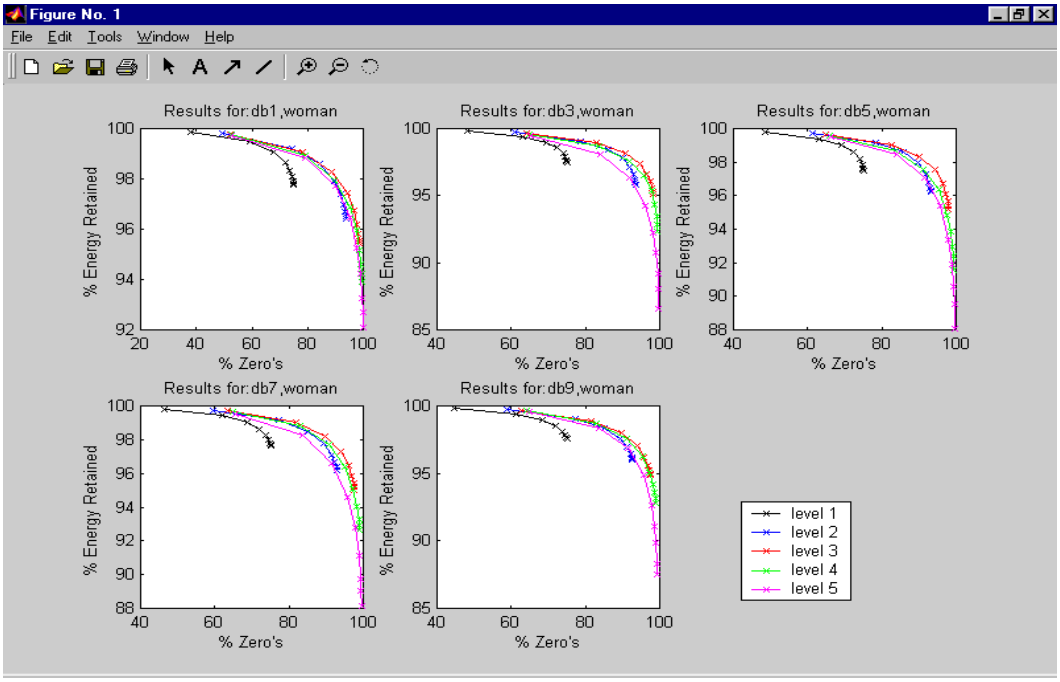


Figure 5.1 The effect of decomposition levels on the 'woman' image

In section 3, the measurement ***D*** was defined in order to find the best 'trade off' between number of zeros and the energy retained. The lowest values for ***D*** were found at decomposition levels 3 and 4. Looking at figure 5.2 it can be seen that level 1 showed the smallest range of values for ***D***, but also often the highest. So if the smallest change in energy is required then level 1 is good to use, otherwise better solutions can be found by decomposing to more levels. However decomposing too much is bad because more energy is lost. The best 'trade-offs' were therefore found at decomposition levels 3 and 4.

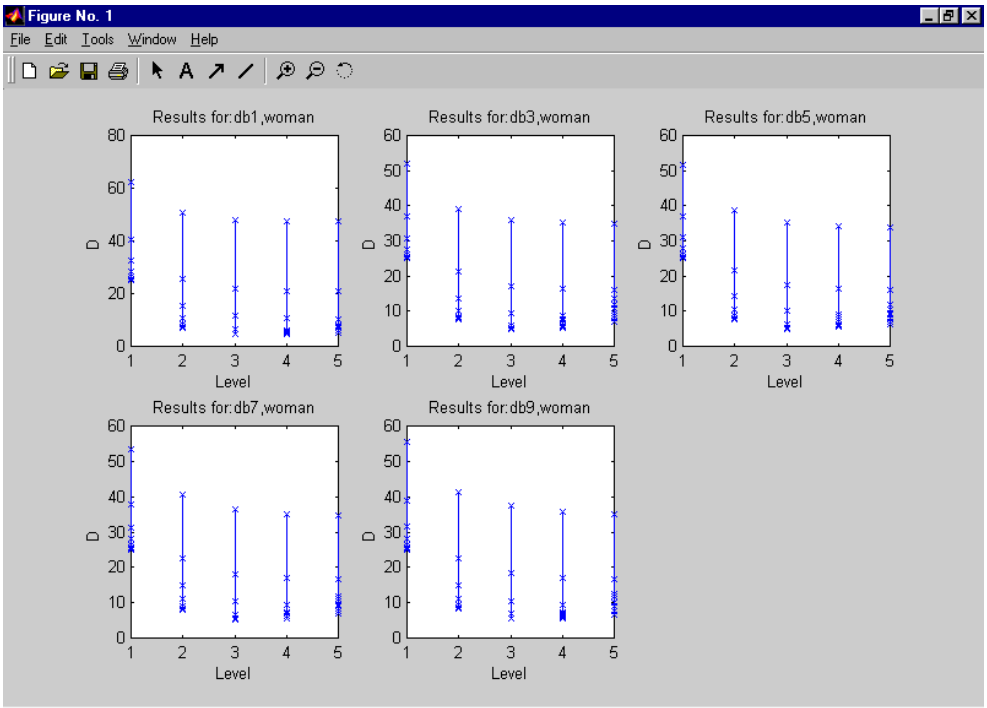


Figure 5.2

A point to note about these ***D*** values is that although it is interesting to look at the general trends, the specific range of values possible may not be shown by the results. This is because only 10 points were sampled from a huge number of potential threshold combinations, meaning that there may be an optimal value that has not been found in the results. Thus although ***D*** could be useful in showing which is the best trade-off, all the possible threshold combinations would need to be looked at to find the optimal value.

The %Energy loss to % Zeros ratio, defined in section 3, showed a pattern for all images at all wavelets, an example can be seen in figure 5.3. The minimum and maximum ratio increased with level and so did the range of values. This suggests that best ratio is always obtained from lower level of decomposition. The ratio is related to gradient of the %Energy retained - %Zeros curves (the negative inverse of the gradient), so the ratio gives a measure of how steeply the gradient is changing and hence how easily energy is lost. So while a higher proportion of the detail coefficients are from the lower decomposition levels the energy contained in the lower level detail matrices is smaller then the energy contained in the higher level detail matrices.

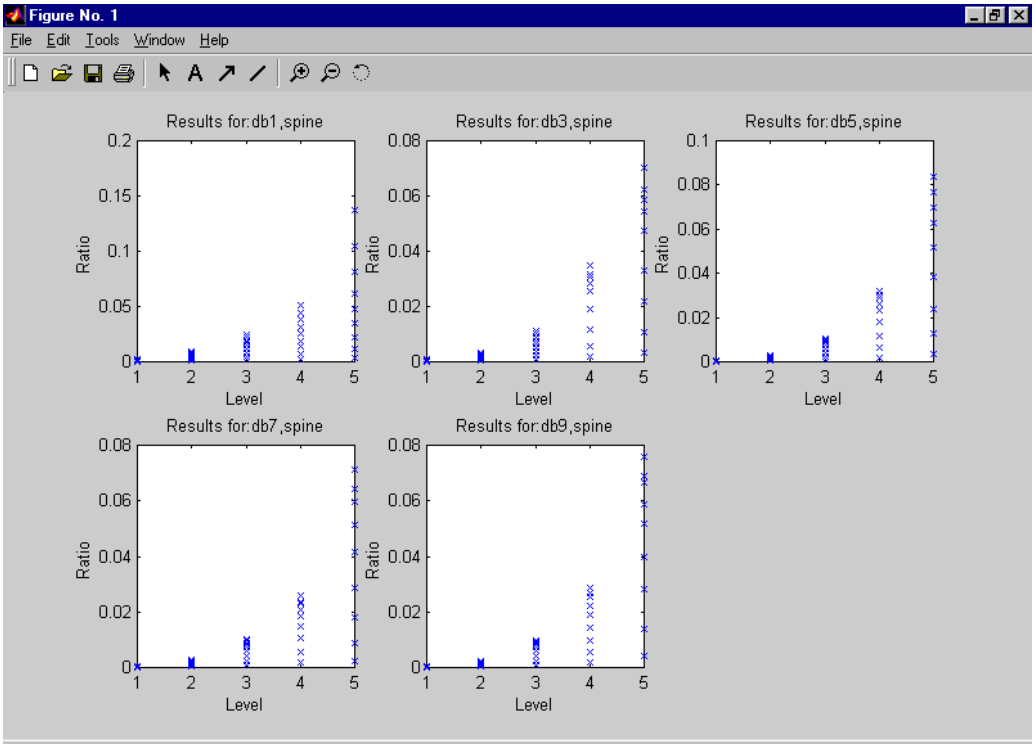


Figure 5.3

5.2 The Effect of the Wavelet

The wavelet definitely changed the shape of the %Energy Retained - %Zeros curve but no particular pattern could be seen overall. The results that can be obtained from different wavelets depends on image and level used. This is because different wavelets look at the energy changes in the image differently. Wavelets with longer supports (e.g db9 has longer supports than db1) take into account the changes between a greater number of pixels and are therefore less localised, but they also change faster so they may be better in approximating fast energy changes. The results have been plotted in figures 5.4 and 5.5 for two different images, to show how the image and level can have a great effect on the compression.

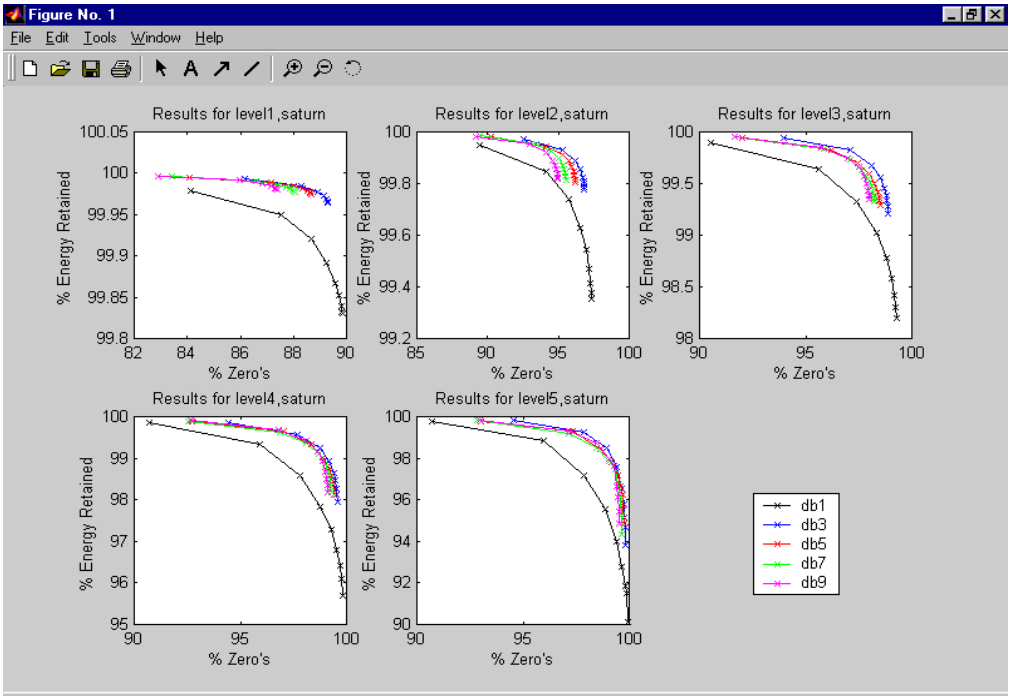


Figure 5.4

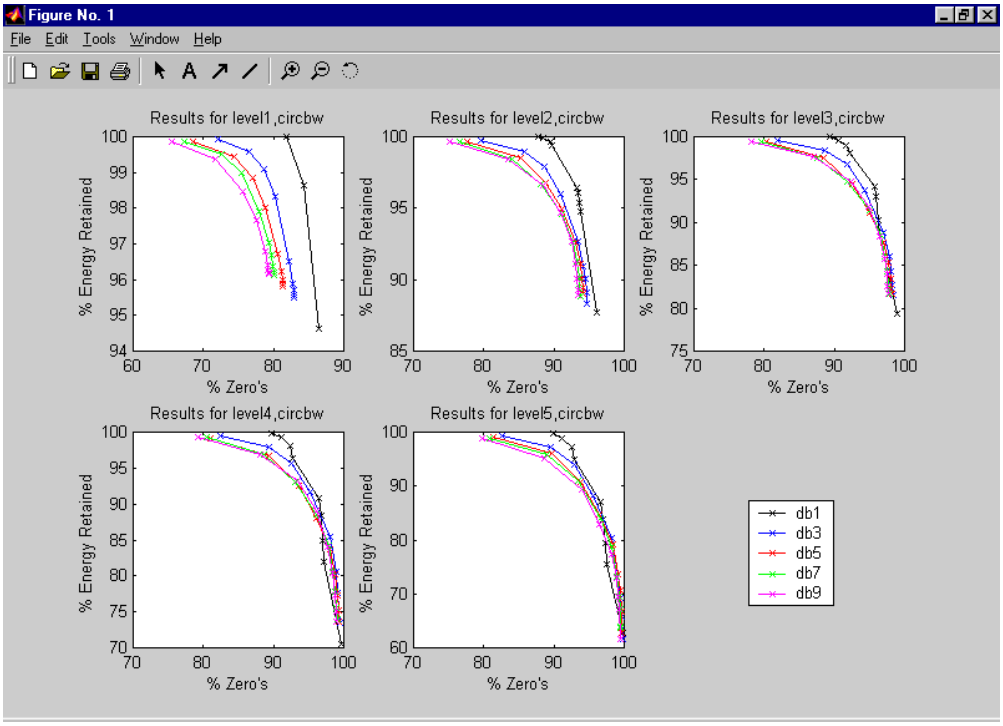


Figure 5.5

The above two figures of results for 'saturn' and 'circbw' show that the wavelets work differently on different images and the best wavelet to use will not be the same for all images.

5.3 The Effect of Changing the Image

No general trend could be found in the results, other than the image had a huge effect on the compression and energy rates. While analysing the lowest ***D*** value for each image might provide information on how the images compare, these ***D*** values may not be the optimal value and hence the information may not be useful. However there are two features of the %Energy retained - %Zeros curves that are not dependent on the threshold value, but simply the combination of image, wavelet and decomposition level, these are the start point and end point of the curves. The start point occurs where the threshold was zero, so this shows how compressible the image is without thresholding (were 100% of the energy is retained). These zeros will either be from a detail subsignal where there are areas of uniform intensity in the image, or from the approximation subsignal, where the image contains zero values. The end point is where all the detail has been thresholded, so this shows how much energy has been compacted into the final approximation signal.

Therefore the range of possible %zero values and the range of possible %energy retained values can be said with certainty. The results are shown in the table below:

Image	Min. %Zeros.	Max. %Zeros.	Min. %Energy Retained	Max. %Energy Retained
Bonemarr	0	99.89	86.03	100
Cameraman	0	99.90	85.15	100
Circbw	13.94	99.91	61.70	100
Facets	0	99.90	79.95	100
Julia	0	99.89	79.55	100
Mixed *[2]	0	99.90	46.86	100
Plain *[1]	0	99.90	93.71	100
Random *[3]	0	99.90	73.34	100
Saturn	40.67	99.93	90.14	100
Spine	29.31	99.92	86.31	100
Text	53.9	99.94	14.43	100
Tire	0	99.88	59.89	100
Wifs	49.43	98.64	55.03	100
Woman	0	99.90	86.56	100

*[1] Plain is an image where all values are 255.
*[2] Mixed is an image with consecutive values changing between 0 and 255, similar to a chessboard
*[3] Random is an image created from random numbers.

Five of the images had a minimum number of zeros greater than 0. This means that without thresholding a number of zeros could be obtained, simply with the combination of image and wavelet. These five particular images have large areas of black, which provides zeros in the approximation subsignal. For an image such as 'Text' which loses up to 85.57% of its energy during compression, comfort can be taken in the fact that it has come with an in-built compression of 53.9% simply from the intensity values it contains.

Looking at the maximum zeros it can be seen that with the exception of 'Wifs' (with 98.64) compressions of over 99% are possible for all images. Therefore wavelets are extremely effective at compression the image data. This is because wavelets are good at gradually filtering more data out until there is only a small approximation signal left.

The cost of high compression (energy loss) varies greatly between the images. 'Plain' and 'Saturn' have the highest minimum energy suggesting that they are best suited to compression with Daubechies wavelets, over 93.71% energy retention is guanteed for 'Plain' whatever the thresholding strategy.

If the minimum and maximum values are looked at more closely for the 'plain' image it can be seen that the Haar wavelet (db1) is ideally suited to this: (see the following table)

Plain			
Wavelet	Min Zeros.	Max Zeros.	Min Energy Ret
db1 (Haar)	75	99.90	100
db3	0	99.79	93.71
db5	0	99.66	94.74
db7	0	99.50	95.46
db9	0	99.32	93.91

Because of the way that the Haar wavelet performs averaging and differencing between pixel values to form the approximation and detail coefficients, all the detail coefficients are zero, which means 100% of the energy can be compacted into the approximation subsignal giving a compression of 99.90%. The other wavelets are however not as good, they try to find more sudden changes in the image and look at how the intensity values are changing over a greater number of pixels. As the 'plain' image has uniform intensity values, the pixels values are not changing so the wavelets which try to approximate changes are not as useful.

5.4 Conclusions

Although global thresholding can be used successfully to compress images it is difficult to find a global threshold that will give near optimal results because of how the different detail subsignals differ. Global thresholding leads to unnecessary energy losses in order to obtain a certain compression rate. Therefore it is more logical to use local thresholds.

Although the thresholding strategy was improved by using a local rather than global technique the results only used a very small proportion of the possible threshold combinations. There are many clever solutions to thresholding available. For example, the threshold was calculated as a fraction of the maximum coefficient. However this did not take into account the energy contained in each detail signal. It may be better, for example, to threshold the first 50% of some signals, while leaving others that contain a lot of energy not thresholded. Using this idea any value of energy retention or alternatively any value of compression can be obtained by thresholding the details with less energy the most, until the required value is obtained.

The decomposition level changes the proportion of detail coefficients in the decomposition. Decomposing a signal to a greater level provides extra detail that can be thresholded in order to obtain higher compression rates. However this also leads to energy loses. The best 'trade-off' between energy loss and compression is provided by decomposing to levels 3 and 4. Decomposing to fewer levels means provides better energy retention but not as great compression, decomposing to higher levels provides better compression but more energy loss.

The type of wavelet affects the actual values of the coefficients and hence how many detail coefficients are zero or close to zero and therefore how much energy and zeros can be obtained. Wavelets that work well with an image redistribute as much energy as possible into the approximation subsignal, while giving a large proportion of the coefficient value to describe details.

An image is a collection of intensity values and hence a collection of energy and energy changes. The image has a huge effect on the compression and how well energy can be compacted into the approximation subsignal. For example the minimum energy retained ranged from 14.43% for 'text' to 93.71% for 'plain', showing a variation of 79.28%. It is probably the most important factor for how much energy can be retained during compression. The minimum percentage of zeros is similarly dependent on the image. 'Text' had a minimum percentage zeros of 53.9 which is due to the fact that the image contains 94% zeros to begin with. So in fact it may be better not to compress 'Text'.

The results were interesting to analyse but it was felt further investigation was required into the following areas:

1. The image properties that affect compression (see section 6)
2. The best thresholding techniques (see section 7)
3. The best wavelet basis to use for a given images.

6. Image Properties

So far only the image entropy of an image has been considered, this measurement took into account the different sizes of the images, but it did not look at the range of intensity values (or bitrate) of the images. If an image has intensity values in the range 0-255 then it's bit rate is 8, as the range of 256 values can be represented by binary numbers of length 8. This concept is important in considering compression, for example the text image had only values 0 and 1 so this only requires a bitrate of 1. Having only small values is a great advantage when trying to encode information using Huffman encoding, so this is already, in a sense, compressed because of it range of intensity values without the need for wavelets and thresholding to make the values any smaller. A concept which does take into account this bit rate is the redundancy of the image. If an image has a high redundancy this means that there is much information given in an image that is not required in order to reconstruct the image. The calculation for this is:

$$r = b - H_e \qquad [6]$$

where b is the bit rate and H_e is the image entropy.

Another value that can be calculated is the compression ratio, K, which should give an idea of how compressible the image is given the entropy and bit rate:

$$K = b / H_e$$

The following values for redundancy and compression ratio were calculated.

Image	**Image Entropy, He**	**Bit rate, b**	**Redundancy, r**	**Compression ratio, K**
Text	0.323	1	0.677	3.096
Circbw	0.9996	1	0.0004	1.600
Wifs	1.9754	4	2.025	2.025
Julia	3.1812	6	2.819	1.886
Spine	4.0935	6	1.907	1.466
Saturn	4.114	8	3.886	1.944
Woman	5.003	8	2.997	1.599
Facets	6.0264	8	1.973	1.327
Tire	6.9265	8	1.074	1.154
Cameraman	7.0097	8	0.990	1.141
Bonemarr	7.7804	8	0.220	0.973

Looking at the results for redundancy, it is suggested that 'Circbw' is not very redundant, which means that it can not be approximated very much if all the energy of the image is to be retained, thus it will lose energy quickly during compression. Looking at 'Saturn' we can see that this is highly redundant compared to the other images with r=3.886, energy will not be lost quickly during compression. However the compression ratios for the images do not show the same pattern, this is because although there may be spatial redundancy between neighbouring pixels, if the bit rate is higher then the intensity values required to approximate the image will be higher. High values are not encoded as efficiently as small values so the compression ratio is not necessarily higher for redundant images. Redundancy indicates how easy it is to approximate something without losing energy, it does not mean that lots of values to zero without losing energy.

There still doesn't seem to be a pattern between the compression results discussed in sections 4 and 5 and the redundancy or compression ratios. This is probably because the image entropy which is used for the redundancy measurements and compression ratio is based simply on the frequency of each intensity value in the image. It does not provide any information about where the intensity values occur. However the power of wavelets comes from their mulitresolution, their ability to look at which intensities occur at which times (or positions) in the image. So to predict the behaviour of a technique that uses multliresoluion, a measurement that can also use multiresolution is needed. That is, a way of

looking at where the intensity values occur, and more importantly the differences between neighbouring pixels is required.

Wavelets work by looking at the values of neighbouring pixels, and splitting those values into an approximation value and a detail value. If the pixel values are similar then the value of the detail is small. Thus an image with intensity values that only have small changes between pixel values is easier to compress with wavelets than those that have dramatic and irregular changes. This is because with these images the approximation signal will contain most of the energy, the detail signals will have values close to zero and therefore not much energy. Thresholding the detail signals will therefore have little effect on the energy, but provide more zeros. So compression can be obtained with little cost in energy loss. Thus if an image contains a high frequency of a certain intensity value, then this could help to provide a good compression rate, but it depends on where they are in the image. If they are all together then there will be an area of the same intensity value, and this means that the detail values will be zero. If they are randomly spread throughout the image, next to pixels of dissimilar intensities, then the fact that there was a high frequency of a certain intensity will not be enough to provide good compression.

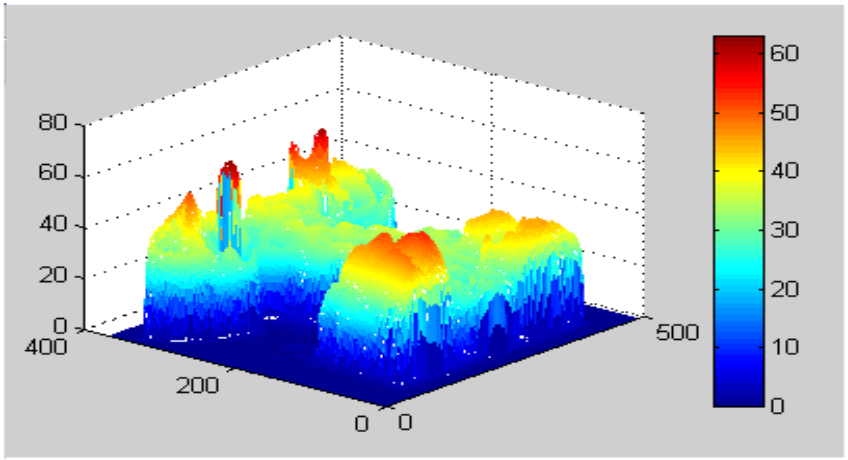
Looking at the spread of intensities in the 'Spine' image (see figure 6.1(a)) it can be seen that there are some edges of relatively large intensity change but within the edges there are regions of uniform intensity or small changes. So the differences between neighbouring pixels are not dramatic and the detail values are small, allowing lots of zeros will little energy loss. Although peaks of high intensity are reached there are no sharp or dramatic changes in the energy values required to get there.

In the 'wifs' image which was harder to compress, the intensity values are change a lot (figure 6.1 (b)), which can be seen in the 'spikes' of intensity levels, rather than a smooth progression from small to large intensities.

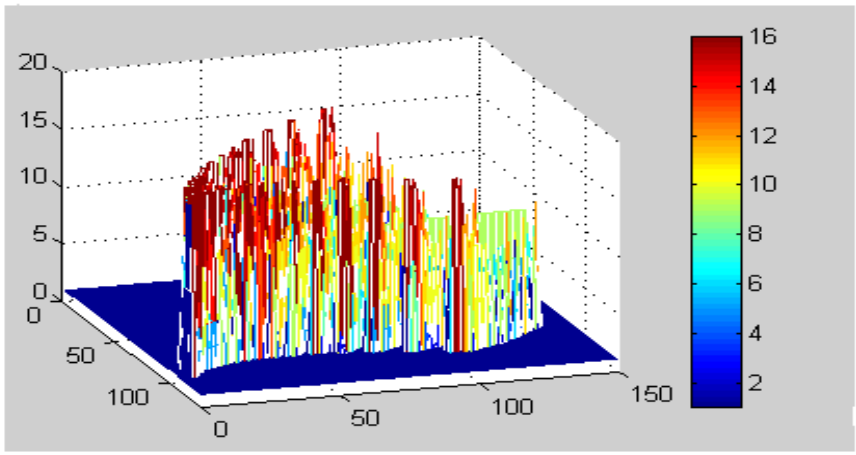
To show how the intensity values are changing an automated script was written that traversed an image matrix and created a new matrix where each element is the amount of intensity change corresponding to each pixel in the image matrix. The intensity change was calculated by summing the differences between a given pixel p, and the eight pixels surrounding p. The maximum possible intensity change is therefore eight times the range of intensity values in the image. For example given an image containing intensity values 0..255 then the range is 256. The worst case of intensity change would be where a pixel of intensity 255 is surrounded by eight pixels of intensity 0 as the change can be no bigger than between 0 and 255.

In 'spine' the maximum possible intensity change would be 504, but the actual maximum intensity change is 169, proving further evidence that the spine image changes gradually. Contrasting this with 'wifs' where the maximum possible change is 120, the actual maximum change is 103.

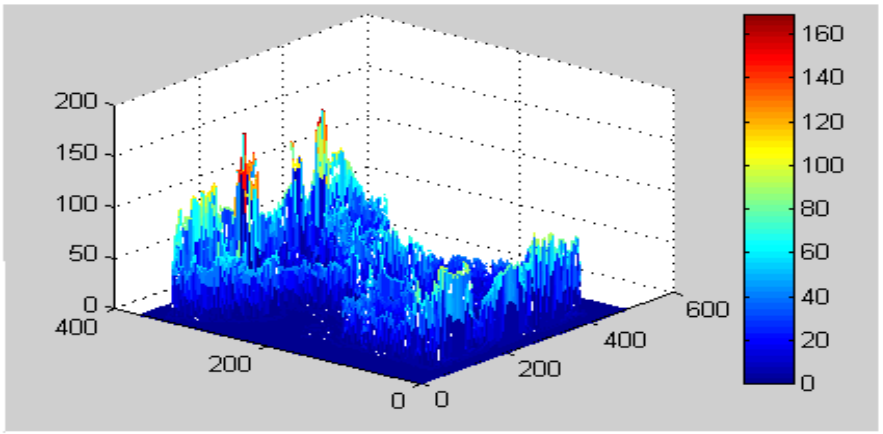
Figure 6.1



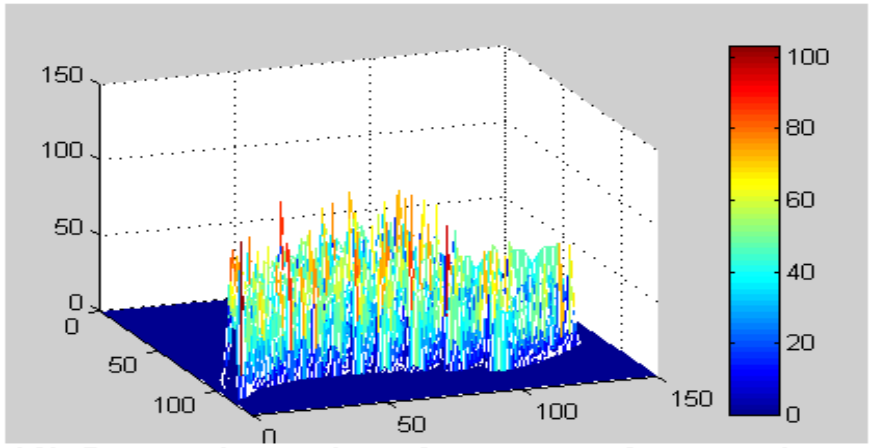
(a) Intensities in the spine image. Some edges and smooth changes, areas of uniform intensity.



(b) Intensities in the wifs image. 'Spikes' of intensity showing dramactic intensity changes between neighbouring pixels.



(c) Sum of intensity changes for each pixel. Most are blue, corresponding to small changes. Spine can therefore be easily approximated as the details are small.



(d) Larger intensity changes can be seen in the wifs image. These are shown by the read, yellow and green 'spikes' in the graph.

7. Thresholding Strategies

7.1 Finding a Near Optimal Threshold

Chang et al [12,13,14] proposed a method of finding a near optimal threshold for image denoising and compressing that was adaptive to the image and each detail subband. In their papers Chang et al explained that techniques introduced by Donoho and Johnstone for thresholding had performed well on 1D signals. The techniques used thresholding to filter additive iid Gaussian noise. It was explained that a threshold 'acts as an oracle which distinguishes between the insignificant coefficients likely due to noise, and the significant coefficients consisting of important signal structures'[12]. These insignificant coefficients can be thought of as noise, or coefficients that can be removed during compression without significantly altering the energy of the image. It is interesting to note that thresholding can be used to simultaneously compress and denoise an image.

Unlike Donoho and Johnstone, the thresholding strategy proposed by Chang et al was suited to images rather than 1D signals but the idea of filtering Gaussian noise was still used. In fact, the strategy used an observation that wavelet coefficients of any subband can be described by a generalised Gaussian distribution. It also follows from this that the average mean square error (MSE) can be approximated by a Bayesian squared error risk. The strategy is known as BayesShrink, and it's goal is to find the threshold that minimises the Bayesian risk. It was found that for each subband the near optimal threshold using soft thresholding was $T_B = \sigma^2/\sigma_X$ where σ^2 is the noise variance and σ_X^2 is the signal variance.

σ^2 is the noise variance but this may be difficult to measure if there is no uncorrupted image or optimally compressed image to measure against. Therefore σ^2 can be estimated from the diagonal detail subband of the first decomposition level, HH_1.

$$\hat{\sigma} = \frac{Median(|Y_{ij}|)}{0.6745} \quad , \quad Y_{ij} \in \text{subband } HH_1 \quad [12]$$

The BayesShrink strategy provides a near optimal threshold for preserving energy while removing unnecessary detail. This is fine when the energy retained or compression rate does not need to be of a specific value, the best trade off is acceptable.

7.2 Finding a Threshold for a Required Energy Retained

Walker proposed in [1] a method of choosing threshold values by using cumulative energy profiles. His example was a 1D signal with the Haar transform and a global threshold. This idea was to take the coefficient of the transform and arrange them into decreasing order:

$$L_1 \geq L_2 \geq L_3 \geq L_n \quad [1]$$

L_1 is the largest absolute coefficient value. If two values are equal then they left in the original order. The cumulative energy profile is then:

$$\left(\frac{L_1}{\varepsilon_f}, \frac{L_1 + L_2}{\varepsilon_f}, \frac{L_1 + L_2 + L_3}{\varepsilon_f}, \ldots\ldots\ldots, 1 \right) \qquad \text{adapted from [1]}$$

If the amount of energy retained by compression is required, as a decimal to be E_{rq}, then the coefficients required to be kept in order to retain this energy can be found using,

$$\frac{L_1 + L_2 + \ldots\ldots + L_k}{\varepsilon_f} = E_{rq}$$

The coefficients are accumulated until E_{rq} is reached, the final coefficient that is required to reach this energy is L_k. Any coefficients smaller than L_k are not required and can be thresholded to zero. Thus the threshold should be chosen to be less than L_k if the minimum retained energy is E_{rq}.

The example form [1] was a global thresholding technique but the same strategy could be adapted to local thresholding on 2D signals. An energy profile could be calculated for each detail subband and thresholding done to make sure that the total energy retained of the signal did not fall below E_{rq}.

If ε_A is the percentage energy of the approximation signal, $\varepsilon_1 \ldots\ldots \varepsilon_{3n}$ the percentage energy of the details subbands from decomposition level 1 to n, then :

$$100 = \varepsilon_A + \varepsilon_1 + \varepsilon_2 \ldots\ldots + \varepsilon_{3n}$$

If a fraction y of the total energy is to be retained then a certain proportion x of each detail subband needs to be retained:

$$y = \varepsilon_A + x(\varepsilon_1 + \varepsilon_2 \ldots\ldots + \varepsilon_{3n})$$

So to calculate the proportion x, for each detail subband:

$$x = \frac{y - \varepsilon_A}{\varepsilon_1 + \varepsilon_2 + \ldots + \varepsilon_{3n}}$$

However the sum, $\varepsilon_1 + \varepsilon_2 \ldots\ldots + \varepsilon_{3n}$ is equal to $100-\varepsilon_A$. So x can be calculated from the required energy y and the energy of the approximation subband ε_A,

$$x = \frac{y - \varepsilon_A}{100 - \varepsilon_A}$$

Thus to find the threshold for each subband, the cumulative energy profile should be calculated, then the threshold should be chosen such that the proportion of energy retained is equal to x:

$$\frac{L_1 + L_2 + \ldots\ldots + L_k}{\varepsilon_s} = x$$

where $L_1,..,L_k$ are the coefficients in the subband, ε_s is the total energy in the subband and the threshold should be chosen to be less than L_k. The more energy that is contained within the approximation subband, the smaller the value of x, which means the smaller amount of energy has to be retained by the detail coefficients. This should give better results than the local thresholding used to collect results set 2, as the thresholding is done dependent on the spread of energy in the detail signal, not simply as a fraction of the maximum coefficient.

There were also some other thresholding strategies suggested by the wavelet toolbox such as the 'Birge-Massart method', 'equal balance sparsity norm' and 'remove near 0'. Unfortunately although it would have been interesting to investigate and compare these thresholding strategies there was not sufficient time left to research them.

8. Evaluation and Possible Extensions

The general aim of the project was to investigate how wavelets could be used for image compression. I believe that I have achieved this. I have gained an understanding of what wavelets are, why they are required and how they can be used to compress images. I understand only too well the problems involved with choosing threshold values, as Change et al. [12] state "while the idea of thresholding is simple and effective, finding a good threshold is not an easy task".

I have also gained a general understanding of how decomposition levels, wavelets and images change the division of energy between the approximation and detail subsignals. So I did achieve an investigation into compression with wavelets, but the way in which I went about it could, with hindsight, have been better.

The importance of the threshold value on the energy level was something of which I did not have an appreciation before collecting the results. To be more specific I understood that thresholding had an effect but didn't realise the extent to which thresholding could change the energy retained and compression rates. Therefore when the investigation was carried out more attention was paid to choosing the wavelets, images and decomposition levels than the thresholding strategy. Using global thresholding is not incorrect, it is a perfectly valid solution to threshold, the problem is that using global thresholding masked the true effect of the decomposition levels in particular on the results. This meant that the true potential of a wavelet to compress an image whilst retaining energy was not shown.

The investigation then moved to local thresholding which was better than global thresholding because each detail subsignal had its own threshold based on the coefficients that it contained. This meant it was easier to retain energy during compression. However even better local thresholding techniques could be used. These techniques would be based on the energy contained within each subsignal rather than the range of coefficient values and use cumulative energy profiles to find the required threshold values. If the actual energy retained value is not important, rather a near optimal trade off is required then a method called BayesShrink [12,13,14] could be used. This method performs a denoising of the image which thresholds the insignificant details and hence produces Zeros while retaining the significant energy.

I was perhaps too keen to start collecting results in order to analyse them when I should have spent more time considering the best way to go about the investigation. Having analysed the results it is clear that the number of thresholds analysed (only 10 for each combination of wavelet, image and level) was not adequate to conclude which is the best wavelet and decomposition level to use for an image. There is likely to be an optimal value that the investigation did not find. So it was difficult to make quantative predictions for the behaviour of wavelets with images, only the general trends could be investigated.

I feel however that the reason behind my problems with thresholding was that thresholding is a complex problem in general. Perhaps it would have been better to do more research into thresholding strategies and images prior to collecting results. This would not have removed the problem of thresholding but allowed me to make more informed choices and obtain more conclusive results.

At the start of the project my aims were to discover the effect of decomposition levels, wavelets and images on the compression. I believe that I have discovered the effect each have, but have not been able to make quantative statements. For example, I wanted to be able to analysis an image and say "with this sort of image it is best to use the wavelet w, decomposition level y and threshold t". However all I can really say is that "image x is harder to compress than image y with wavelet w". The reasons for this are mainly due to time constraints, if I had more time I would conduct more research into finding the best basis function for an image.

There are many extensions to the project, each of which would be a project by itself. The first area would be finding the best thresholding strategy. How should the best thresholds be decided? There are many different strategies that can be compared such as the 'Birge-Massart method', 'equal balance sparsity norm' and 'remove near 0'. Perhaps certain areas of the image could be thresholded differently based on edge detection rather than each detail subsignal.

Wavelet packets could be investigated. These work in a similar way to the wavelet analysis used in this investigation but the detail subsignals are decomposed as well as the approximation subsignals. This would be advantageous if there tends to be a lot of energy in the detail subsignals for an image.

How well a wavelet can compact the energy of a subsignal into the approximation subsignal depends on the spread of energy in the image. An attempt was made to study image properties but it is still unclear as to how to link image properties to a best wavelet basis function. Therefore an investigation into the best basis to use for a given image could be another extension.

Only one family of wavelets was used in the investigation, the Daubechies wavelets. However there are many other wavelets that could be used such as Meyer, Morlet and Coiflet.

Wavelets can also be used for more than just images, they can be used for other signals such as audio signals. They can also be used for processing signals not just compressing them. Although compression and denoising is done in similar ways, so compressing signals also performs a denoising of the signal.

Overall I feel that I have achieved quite a lot given the time constraints considering that before I could start investigating wavelet compression I had to first learn about wavelets and how to use Matlab. I feel that I have learned a great deal about wavelets, compression and how to analyse images.

9. Conclusions

Wavelet analysis is very powerful and extremely useful for compressing data such as images. Its' power comes from its multiresolution. Although other transforms have been used, for example the DCT was used for the JPEG format to compress images, wavelet analysis can be seen to be far superior, in that it doesn't create 'blocking artefacts'. This is because the wavelet analysis is done on the entire image rather than sections at a time. A well known application of wavelet analysis is the compression of fingerprint images by the FBI.

The project involved writing automated scripts in Matlab which could calculate a great number of results for a range of images, Daubechies wavelets and decomposition levels. The first set of results calculated used global thresholding, however this was found to be an inferior way of calculating threshold values. To improve upon this, a second result set was calculated, using local thresholding. This second results set proved to be more useful in understanding the effects of decomposition levels, wavelets and images. However, this was still not the optimal thresholding in that it is possible to get a higher energy retention for a given percentage of zeroes, by thresholding each detail subsignal in a different way.

Changing the decomposition level changes the amount of detail in the decomposition. Thus, at higher decomposition levels, higher compression rates can be gained. However, more energy of the signal is vulnerable to loss. The wavelet divides the energy of an image into an approximation subsignal, and detail subsignals. Wavelets that can compact the majority of energy into the approximation subsignal provide the best compression. This is because a large number of coefficients contained within detailed subsignals can be safely set to zero, thus compressing the image. However, little energy should be lost. Wavelets attempt to approximate how an image is changing, thus the best wavelet to use for an image would be one that approximates the image well. However, although this report discusses some relevant image properties, there was not time to research or investigate how to find the best wavelet to use for a particular image.

The image itself has a dramatic effect on compression. This is because it is the image's pixel values that determine the size of the coefficients, and hence how much energy is contained within each subsignal. Furthermore, it is the changes between pixel values that determine the percentage of energy contained within the detail subsignals, and hence the percentage of energy vulnerable to thresholding. Therefore, different images will have different compressibilities.

There are many possible extensions to this project. These include finding the best thresholding strategy, finding the best wavelet for a given image, investigating other wavelet families, the use of wavelet packets and image denoising.

References

[1] Walker, J.S. *A Primer on Wavelets and Their Scientific Applications* . Boca Raton, Fla. : Chapman & Hall/CRC, 1999

[2] Robi Polikar, *"The Wavelet Tutorial"* . http://engineering.rowan.edu/~polikar/WAVELETS/Wttutorial.html

[3] Aboufadel, Edward. *Discovering Wavelets, by Edward Aboufadel and Steven Schliker*. New York; Chichester : Wiley, 1999
.

[4] Hubbard, Barbara Burke. *The World According to Wavelets*. A.K Peters Ltd, 1995.

[5] Saha, Subhasis. "Image Compression – from DCT to Wavelets : A Review" **http://www.acm.org/crossroads/xrds6-3/sahaimgcoding.html**

[6] Sonka, M. Hiaual, V. Boyle, R. Image Processing, Analysis and Machine Vision, 2nd edition. Brooks/Cole Publishing Company.

[7] Misiti, M. Misiti, Y. Oppenheim, G. Poggi, J-M. Wavelet Toolbox User's Guide, Version 2.1. The Mathworks, Inc. 2000.

[8] Vincent Kiernan. "*FBI puts the squeeze on suspect prints"*. Article from New Scientist magazine, vol 145 issue 1968, 11/03/1995, page 21

[9] Ian Stewart. *"Little waves, big squeeze*". Article from New Scientist magazine, vol 149 issue 2019, 02/03/1996, page 24

[10] Mulcahy, Colm. "*Image compression using the Haar wavelet transform*". Spelman Science and Math Journal. Found at: http://www.spelman.edu/~colm/wav.html

[11] A.Bruce and H, Gao. "Applied Wavelet Analysis with S-Plus". Springer –Verlag New York, Inc. 1996
.

[12] S. G. Chang, B Yu and M Vetterli. "*Adaptive Wavelet Thresholding for image Denoising and Compression*". IEEE Transactions on Image Processing, Vol. 9, No. 9, September 2000

[13] S. G. Chang, B. Yu and M Vetterli. "*Spatially Adaptive Wavelet Thresholding with Context Modelling for Image Denoising*"

[14] S. G. Chang, B. Yu and M Vetterli. "Image Denoising via Lossy Compression and Wavelet Thresholding". "

[15] H. L. Resnikoff and R.O. Wells, Jr. "Wavelet Analysis The Scalable Structure of Information". Springer –Verlag New York, Inc. 1998

[16] I.Daubechies. "Ten Lectures on Wavelets", vol. 61 of CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, 1992.

[17] Powerpoint Presentation, "Introduction to Wavelets" found at www.mathworks.com/products/index.html

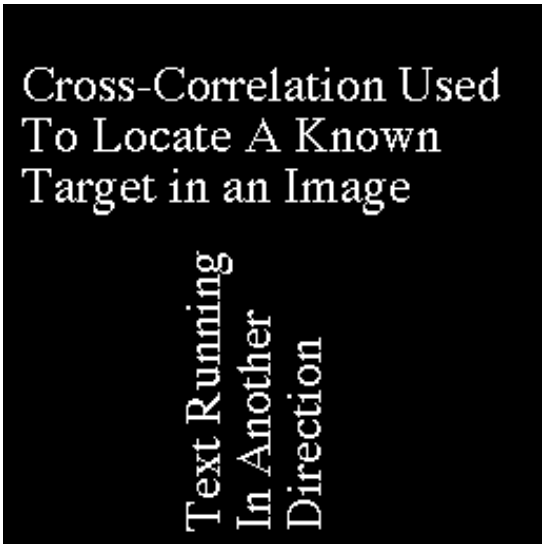
Appendix A – Matlab Functions

Name	Description
calcres	Calculates the energy retained and percentage of zeros for 10 global threshold values. e.g RES = calcres(image, 'wname', decomposition_level) RES contains 3 columns for threshold value, percentage of zeros and percentage energy retained.
calcres2	Calculates the energy retained and percentage of zeros for 10 local thresholds. e.g RES = calcres2(image, 'wname', decomposition_level) RES contains 3 columns for threshold ID (1..10), percentage of zeros and percentage energy retained
createMixedImage	Creates a matrix of size 256x256. The values flip between 0 and 255, creating a chessboard effect.
createPlainImage	Creates a matrix of size 256x256 where all the values are set to 255.
getIntensityChanges	Returns a matrix of intensity change values. The intensity change for each pixel is calculated by summing the magnitude of the difference between a pixel and the eight surrounding pixel values. e.g CH = getIntensityChanges(IMAGE) where IMAGE is the matrix of pixel values.
imageChanges	Returns the sum of intensity change for a given image matrix. e.g C = imageChanges(IMAGE)
ImageEntropy	Returns the image entropy value for the image E.g He = imageEntropy(IMAGE)

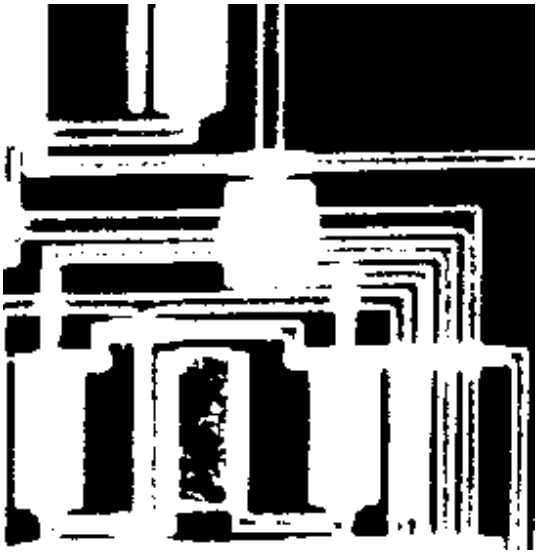
Functions Requiring Database Toolbox

Name	Description
getwaveletresults	Calls calcres for every combination of image and wavelet found in the database and level of decomposition 1 to.5. e.g getwaveletresults('results1')
Getwaveletresults2	Calls calcres2 for every combination of image and wavelet found in the database and level of decomposition 1 to.5. e.g getwaveletresults2('results2')
getTotalChanges	Gets the total intensity change for every image in the databse e.g getTotalChanges('results1')

Appendix B – Images Used

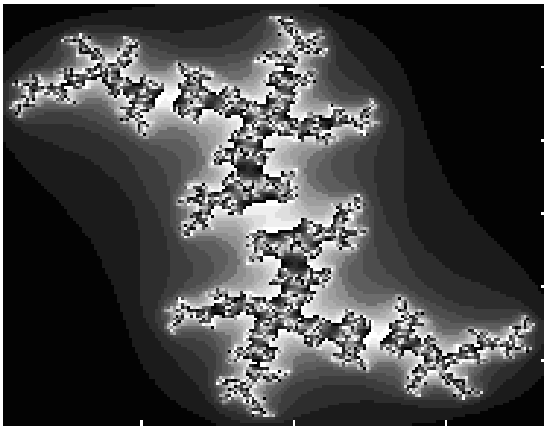


text

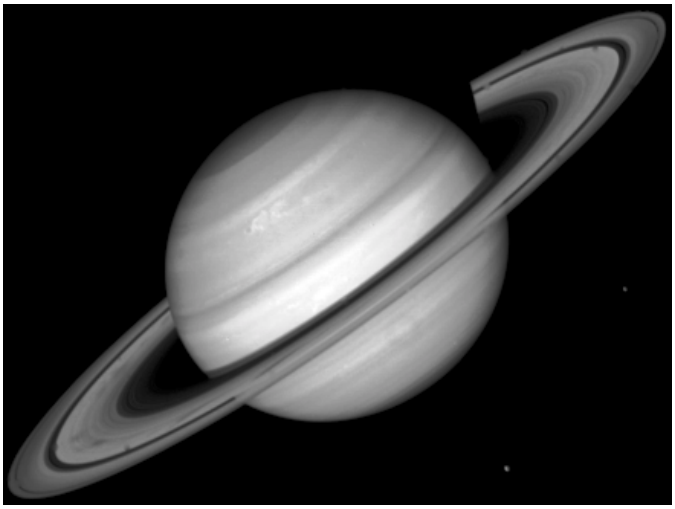


circbw

spine



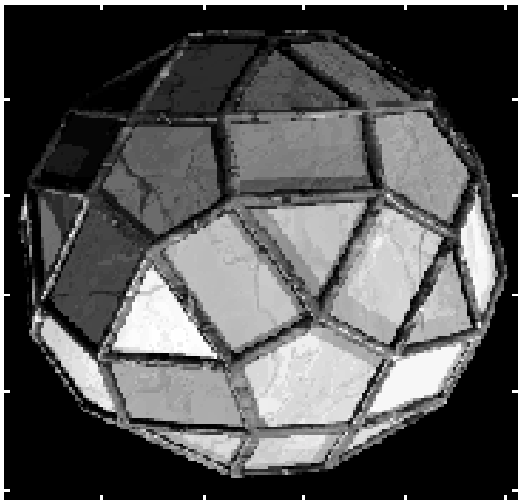
julia



saturn

wifs

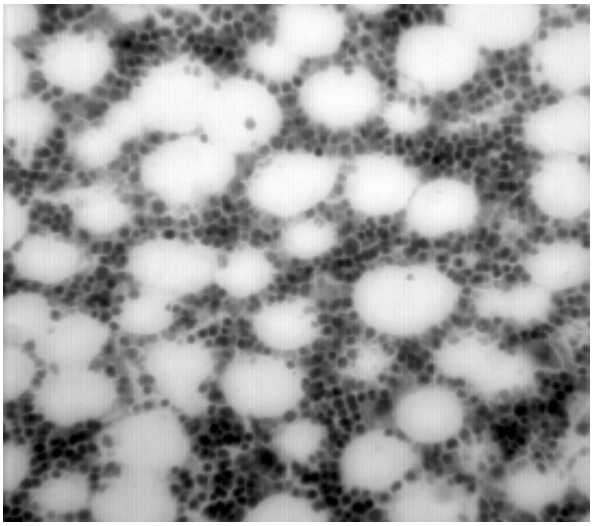
woman



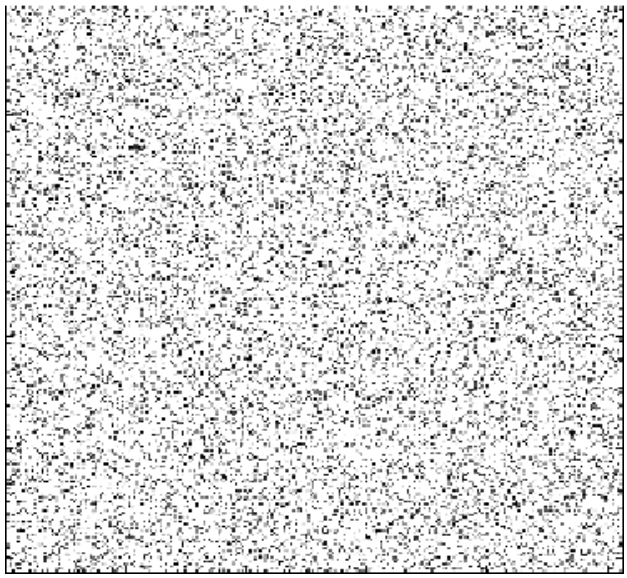
facets

tire

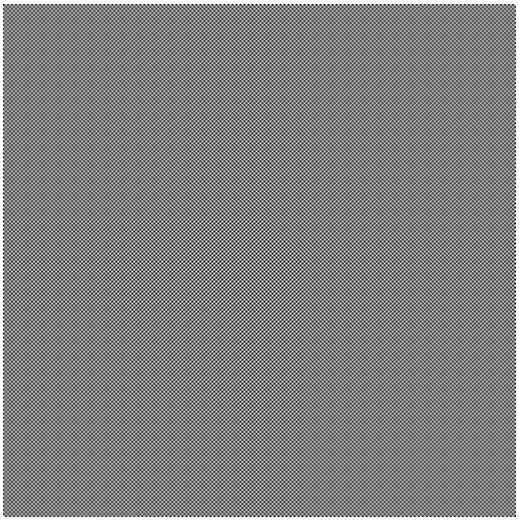
cameraman



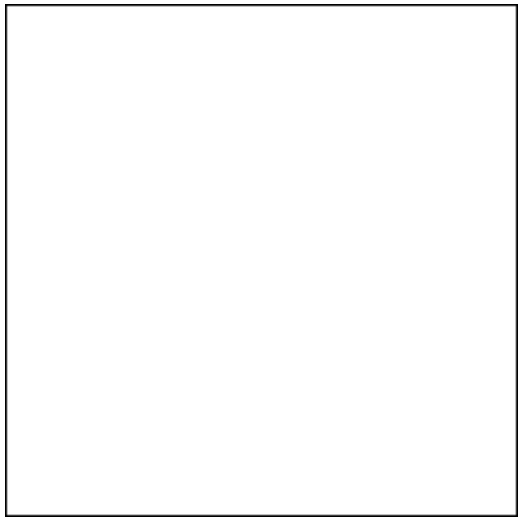
bonemarr



random



mixed



plain