Technical Report No. 526

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Extending Lossless Image Compression

A. J. Penrose

December 2001

This Technical Report is a copy of a PhD dissertation.

William Gates Building
J J Thomson Avenue
Cambridge CB3 0FD
United Kingdom

UNIVERSITY OF CAMBRIDGE

Extending Lossless Image Compression

Andrew John Penrose

Gonville and Caius College
University of Cambridge

January 2001

A report based on a dissertation submitted for the degree of Doctor of Philosophy.

Abstract

"It is my thesis that worthwhile improvements can be made to lossless image compression schemes, by considering the correlations between the spectral, temporal and interview aspects of image data, in extension to the spatial correlations that are traditionally exploited."

Images are an important part of today's digital world. However, due to the large quantity of data needed to represent modern imagery the storage of such data can be expensive. Thus, work on efficient image storage (image compression) has the potential to reduce storage costs and enable new applications.

Many image compression schemes are lossy; that is they sacrifice image information to achieve very compact storage. Although this is acceptable for many applications, some environments require that compression not alter the image data. This *lossless* image compression has uses in medical, scientific and professional video processing applications.

Most of the work on lossless image compression has focused on monochrome images and has made use of the spatial smoothness of image data. Only recently have researchers begun to look specifically at the lossless compression of colour images and video. By extending compression schemes for colour images and video, the storage requirements for these important classes of image data can be further reduced.

Much of the previous research into lossless colour image and video compression has been exploratory. This dissertation studies the problem in a structured way. Spatial, spectral and temporal correlations are all considered to facilitate improved compression. This has lead to a greater data reduction than many existing schemes for lossless colour image and colour video compression.

Furthermore, this work has considered the application of extended lossless image coding to more recent image types, such as multiview imagery. Thus, systems that use multiple views of the same scene to provide 3D viewing, have been provided with a completely novel solution for the compression of multiview colour video.

Acknowledgements

There are many people I would like to thank, for making the time spent on this work more enjoyable than it might otherwise have been. I thank my supervisor Dr Neil Dodgson for allowing me the freedom to pursue my research interests unhindered and for his valuable assistance in proofreading. I would like to thank the whole of the Rainbow Graphics research group for an interesting work environment. I would also like to thank my family and friends for many a word of encouragement.

Declaration

Except where stated, this dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration.

To Laura

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Pictures have been with us since the dawn of time. However, the way that pictures have been represented and displayed has changed greatly. Originally every picture was unique, being both represented and displayed in a physical way, such as paint on a cave wall or etchings in stone. However, in recent times pictures have been dealt with electronically. One consequence of this is that the representation used for transmission or storage of the image can be separated from the means of display. One example of this is traditional broadcast colour television, where the representation that is transmitted does not relate directly to the intensities of the red, green and blue electron guns in a television set.

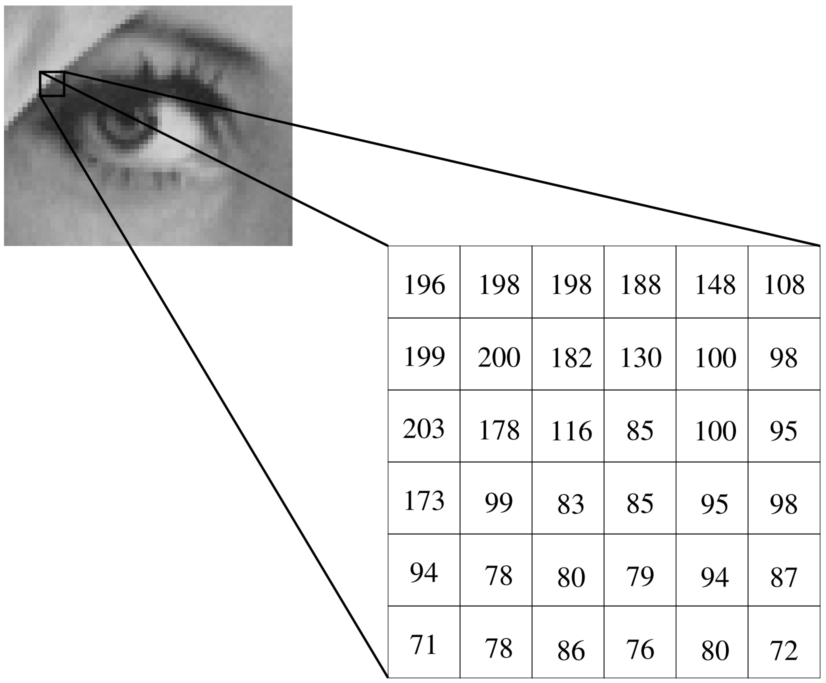
By storing images in digital form, the possibilities for image representation increase dramatically. An image can be stored in any representation, provided there is an algorithm that can convert it to a form usable by a display. This process of changing the representation of an image is called *image coding* and if the result uses less storage space than the original it is called *image compression*.

1.1 The Nature of Digital Images

A real image can be characterised as a continuous two-dimensional (2D) function $f(x, y)$. To become a digital image, this function must be digitised. This is achieved by measuring the value of the function at a fixed number of locations (spatial sampling) and limiting the result to a fixed set of values (amplitude quantisation). The relationship between pixels values and an image is illustrated in Figure 1.1. A full coverage of these topics can be found in any good image processing book[GW92, SHB93].

The number of samples taken determines the resolution of the image. For example, using a rectangular grid of equally spaced sampling points, with 1024 sampling points per row and 768 per column, yields an image with a resolution that is written 1024×768. Each sample is generally thought of as representing the intensity of a picture element or *pixel*.

The size of the set of values that can be taken by a pixel is almost always a power of 2. If a pixel can have 2^n values, then it requires n bits of storage. Thus, a two-tone image



196	198	198	188	148	108
199	200	182	130	100	98
203	178	116	85	100	95
173	99	83	85	95	98
94	78	80	79	94	87
71	78	86	76	80	72

Figure 1.1: Alternative views of image data.

(e.g. a fax) would have a binary value for each pixel, and is often referred to as a 1 bit image. Continuous-tone greyscale images generally use 8 bits per pixel (bpp) and colour images use 24 bits per pixel (8 each for red, green and blue). Medical and scientific images typically use more bits per pixel, sometimes up to 16 bpp for greyscale.

Taken together, the values of all the pixels in an image constitute the *raw data* representation of the image. The amount of storage required by this raw data can be calculated as the product of the number of pixels and the bits used per pixel. As an example, consider a continuous-tone colour image of dimensions 1024x768 using 24 bpp. The storage requirements for the raw data of such an image would be:

$$1024 \text{ x } 768 \text{ x } 24 = 18874368 \text{ bits} = 2359296 \text{ Bytes} = 2.25 \text{ MB}$$

This may not seem like a great deal of storage space, but as the number of images that need to be stored increases, the total storage requirements soon become overwhelming. For example, it is estimated that NASA receives over a terabyte of digital imagery, every day, from Earth orbiting satellites alone[Tat94]. Therefore, an efficient representation for image storage (and transmission) is important.

1.1.1 The Eye of the Beholder

Data can only be compressed if it contains some form of redundancy. Images contain two forms of redundancy: source and psychovisual. Source redundancy is found in the correlations between pixel values; that is, the source data. For example, a given pixel value is often close to the value of neighbouring pixels. Therefore, by representing a pixel value relative to its neighbours, source redundancy can be exploited and more efficient storage can be accomplished.

Psychovisual redundancy exists when an image is to be viewed by a human observer, because some small changes in an image have no effect on the viewer's perception of the image. This means that two slightly different images may give the same subjective impression as each other. However, if one image is more compressible than the original it resembles, a greater storage saving can be made by storing the near matching image rather than the original. This is known as *lossy* compression, whereas *lossless* compression, making careful use of only source redundancy, guarantees that the decoded image will be exactly the same as the original.

The use of lossy compression for applications where the imagery is only intended to be viewed by a human observer is both sensible and wide-spread. Prominent examples of this include pictures on the world wide web, image storage in digital cameras, and the emerging digital broadcast television. However, some applications still benefit from, or require, lossless compression. Such applications include scientific and medical image storage and high quality video production techniques.

1.2 Background to Lossless Image Compression

The foundation of image compression is information theory, as laid down by the likes of Shannon in the late 1940s[Sha48, Ver98]. Information theory tells us that the information of an event is:

$$\log_2 \frac{1}{p(e)} \text{ bits} \tag{1.1}$$

where $p(e)$ is the probability of the event occurring. Thus, the information content of an event is directly proportional to our surprise at the event happening. A very unlikely event carries a lot of information, while an event that is very probable carries little information.

Encoding an image can be thought of as recording a sequence of events, where each event is the occurrence of a pixel value. If we have no model for an image, we might assume that all pixel values are equally likely. Thus, for a greyscale image with 256 grey levels, we would assume $p(e) = 1/256$ for all possible pixel values. The apparent information required to record each pixel value is then $\log_2 256 = 8$ bits. Clearly, this is no better than the raw data representation mentioned above.

However, due to the spatial smoothness common in images, we expect a given pixel to be much like the one before it. If the given pixel value conforms to our expectation of being close to the previous value, then little information is gained by the event of learning the current pixel's value. Consequentially, only a little information need be recorded, so that the decoding process can reconstruct the pixel value.

This idea of using previous pixel values to lower the information content of the current pixel's encoding has gone under several names: Differential Pulse Code Modulation (DPCM), difference mapping and more generally predictive coding. From early work in the 50s and 60s on television signal coding[O'N66, Oli52, Har52], to modern lossless image compression schemes, predictive coding has been widely used. The common theme has always been to use previous data to predict the current pixel and then only the prediction error (or prediction residual) need be encoded.

Predictive coding requires the notion of a current pixel and past pixels and this implies a one-dimensional (1D) sequence of pixels. However, image data is two-dimensional. To correct for this mis-match a 1D path is needed that visits every pixel in the 2D image. By far the most common path is *raster-scan ordering*, which starts at the top left of an image and works left to right, top to bottom, over the whole image.

1.2.1 Entropy and Symbol Coding

One way to get a quantitative measure of the benefits of prediction is to use *entropy*. This commonly used measure of information content, again provided by Shannon, says that for a collection of independent, identically distributed (i.i.d.) symbols $x_0, x_1, \ldots, x_i$, each with a value in the range $0 \leq j \leq (N-1)$, the average information content per symbol is:

$$\sum_{j=0}^{(N-1)} p(x_i = j) \log_2 \frac{1}{p(x_i = j)} \text{ bits} \tag{1.2}$$

where $p(x_i = j)$ is the probability of a given symbol having value j. The i.i.d. assumption implies a *memoryless source* model for the data. That is, it does not use information based on preceding symbols (memory) to model the value of the current symbol. Note that this assumption is almost never true at any stage in image coding, however it is a useful simplifying model at this stage.

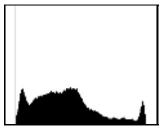
Equation 1.2 shows that it is the distribution of pixel values, the $p(x_i = j)$, that is important. It can be inferred that distributions that are nearly flat (all symbols nearly equiprobable) have average information measures approaching $log_2 N$, whereas sharply peaked distributions have much lower entropies. Figure 1.2 gives an example of this and shows that simple predictive coding produces a residual image with a sharply peaked distribution.

Having changed the distribution of symbols so that their entropy is reduced, it remains to store them efficiently. An effective way to do this involves Variable Length Codes (VLCs), where short codes are given to frequent symbols and longer codes are given to infrequent symbols. In 1952 Huffman[Huf52] described his algorithm for producing optimal codes of integer length. However, in the late 1970s, researchers at IBM succeeded in implementing Arithmetic coding, which by removing the integer length constraint allows more efficient symbol coding.

Image compression schemes tend to be innovative in the stages leading up to symbol coding. The final symbol coding stage is generally implemented using traditional algo-

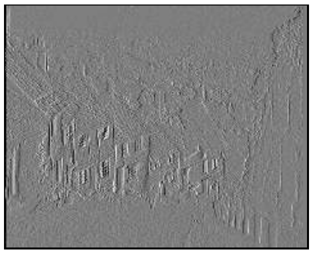
a) Without Prediction:

Pixel Distribution:

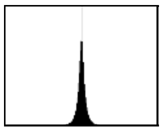


7.53 bits
per pixel

b) With Prediction:



Residual Distribution:



5.10 bits
per pixel

Figure 1.2: The benefit of prediction. In a) the entropy of the pixel values is close to the raw data size of 8 bpp. In b) Using the previous pixel value as a predictor, the prediction errors (adjusted for display) have a much lower entropy.

rithms. Such schemes are documented in any good book on data compression[BCW90, Nel91] and for completeness some of the most common approaches are briefly documented in Appendix A.

1.3 Covering New Ground

Traditional lossless image compression uses only spatial correlations to model image data. While this is all that can be done for greyscale images, modern imagery is often more complex than a still greyscale image. Colour images and video are now common and have correlations beyond just the spatial domain. Colour images have spectral correlations between the colour bands and video has temporal correlations between successive frames.

Another type of imagery that is becoming more popular is multiview stereo. Two or more views of a scene are stored and then displayed with suitable hardware. The viewer can then see the scene in 3D or a machine vision algorithm can attempt to calculate the distance to objects in the scene. There is correlation between the multiple viewpoints that could be exploited for compression.

By considering all the forms of correlation present in modern imagery, lossless image compression can progress beyond traditional methods. Methods that are improved in this way, which will be referred to as *extended* lossless image compression schemes, should be capable of improved compression when applied to advanced types of imagery, such as colour images, video and multiview sequences.

Extended image compression schemes have been investigated before, but most of the work is either for lossy compression, application specific coding or considers only one or two of the correlation types. The question of how best to combine information from some or all of these correlations, to improve lossless image compression, remains unanswered.

The purpose of this work is to examine ways in which the full range of correlations can be exploited to produce extended lossless image compression methods, that are applicable to a wide class of images. It is expected that this will lead to superior compression performance for advanced image types.

1.3.1 A Considered Approach

It quickly becomes apparent that by considering four main types of image correlation, each of which having been the subject of numerous works of research, the scope of this work is quite broad. To give focus to the goal of integrating techniques to provide extended lossless image compression, a number of guiding principles were used during the course of this work.

- Generic Simplicity

  The way in which multiple correlation types are integrated should be independent of the correlation types available. That is, the integration method should be generic and sufficiently simple that it can easily handle multiple types of correlation.

- Adaptability

  The resultant compression schemes should adapt to offer good performance for a range of image types. Encodings that are tuned to a specific image class will not be investigated.

- Solid Foundations

  Many established algorithms exist for many of the problems in extended image compression. Where appropriate, existing approaches will be used. Innovation will be saved for those areas where it is needed.

- Usability

  The intended increase in compression should not come at the cost of unacceptable running times for the compression software. Thus, schemes that require extensive analysis prior to compression will not be used.

1.4 Dissertation Outline

Image compression is a well established field and the literature available is too numerous to cover exhaustively. However, as per the third guiding principle above, algorithms from the literature will be an important part of the following work. In order to clearly differentiate those approaches that come from the literature and those that are novel contributions, the relevant literature will be surveyed separately. To this end, the literature regarding lossless greyscale image compression and the compression of more advanced image types (e.g. colour images and colour video) are surveyed in Chapters 2 and 3 respectively.

Chapter 4 shows how adaptive selection of predictor models can be used as a framework for integrating multiple types of correlation in imagery. In Chapter 5, the use of higher order correlations and their impact on entropy coding are examined. Chapter 6 introduces certain practical considerations and quantifies the performance of the compression scheme resulting from this work. In Chapter 7, ideas for further work are aired, including the concept of error resilient lossless image coding. The main conclusions that can be drawn from this work appear in Chapter 8.

Chapter 2

Lossless Greyscale Image Compression

This chapter presents a survey of the literature relating to lossless greyscale image compression. The literature in this field is quite extensive and it is impossible to cover exhaustively due to space constraints. However, to ensure a reasonably thorough coverage, what are considered to be good examples of all the major techniques are discussed. In particular both the old and new JPEG standards for lossless image compression are covered in some detail.

Before the literature itself is covered, a simple model of lossless compression is presented in Section 2.1. This model will help simplify the discussion of the following image compression schemes.

A good review of lossless greyscale image compression was published in [MS95c]. From this, and other parts of the literature, it is apparent that most modern lossless image compression schemes are based on some form of predictive coding. As such, the origins of predictive coding are presented in Section 2.2 and the refinements used in many recent predictive coding schemes are detailed in Sections 2.3 and 2.4.

Having discussed the main concepts behind successful lossless image compression methods, three complete schemes are described in Section 2.5.

Although the material presented up to Section 2.5 covers many mainstream ideas, other noteworthy contributions exist and some of these are detailed in Section 2.6.

The literature relating to colour images, video and multiview stereo imagery is covered in Chapter 3.

2.1 A Model for Lossless Image Compression

Lossless image compression can be decomposed into three main stages: Mapping, Modelling and Coding, as shown in Figure 2.1. Fitting compression schemes to this model is often useful for comparing the similarities and differences between two schemes and thus can help

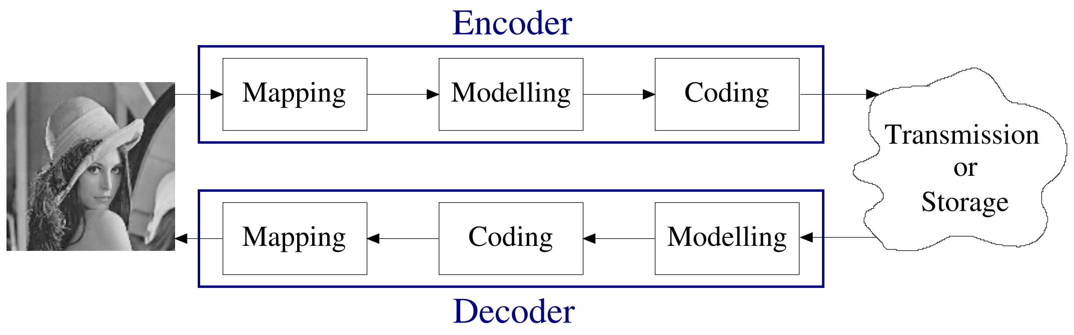


Figure 2.1: Basic model for lossless image compression.

simplify the discussion of complex compression schemes. Note the asymmetry between the encoder and the decoder, caused by the coding stage's dependence on the modelling stage.

The mapping stage provides a reversible mapping of the image data, such that the result is less correlated than the original data. The mapping can be as simple as replacing each pixel with the difference between the current and previous pixel (difference mapping), although more complex mappings often yield better results.

As previously discussed in Section 1.2 and illustrated in Figure 1.2, by decorrelating the image data, the mapping stage changes the statistical properties of the pixel data. This makes the data to be encoded closer to being i.i.d. and therefore closer to the kind of data that can be efficiently coded by traditional symbol coding techniques.

The modelling stage attempts to characterise the statistical properties of the mapped image data. It attempts to provide accurate probability estimates to the coding stage, and may even slightly alter the mapped data. By being mindful of higher order correlations, the modelling stage can go beyond the memoryless source model and can provide better compression than would be apparent from measuring the entropy of the mapped image data using Equation 1.2.

The symbol coding stage aims to store the mapped pixel data efficiently, making use of probability estimates from the modelling stage. Symbol coders are also sometimes called statistical coders (because they use source statistics for efficient representation) and entropy coders (because they aim to represent data using no more storage than allowed by the entropy of the data). Coding schemes take a sequence of symbols from an input alphabet and produce codewords using an output alphabet. They aim to produce a code with the minimum average codeword length. The reader who is unfamiliar with the standard symbol coding schemes (Huffman coding, Arithmetic coding and Lempel-Ziv based methods) should consult Appendix A.

2.1.1 Requirements for Lossless Decoding

In order for an image compression scheme to be lossless, a decoder must be able to produce the original image from the data transmitted by the encoder. To ensure this, the

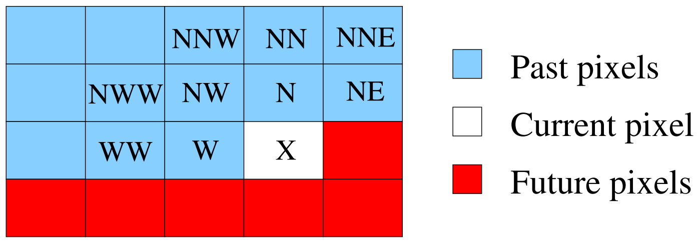


Figure 2.2: Locations relative to the current pixel, X.

encoder must only make predictions on the basis of pixels whose value the decoder will already know. Therefore, if all past pixels have been losslessly decoded, the decoder's next prediction will the same as that made by the encoder.

Also of importance, is that the encoder and decoder agree to the nature of the variable length coding scheme to be used. This is easy when a fixed coding scheme is used, but if an adaptive scheme is used, where the meaning of codes change over time, then the decoder must make the same adaptations. This can be achieved either by the encoder making a decision based on future pixels and transmitted that change to the decoder (forward adaptation) or by the encoder and decoder both making changes, in a predefined way, based on the values of previous pixels (backward adaptation).

2.2 The Origins of Predictive Coding

Cutler is commonly given as the first to do work on image compression by predictive coding[Cut52]. Also in 1952, researchers at Bell Telephone labs published work on the use of predictive systems to reduce the bandwidth requirements for television signals[Oli52, Har52]. In 1966, frustrated by the fact that TV signals were still being transmitted without efficient coding, O'Neal[O'N66] re-examined the problem.

O'Neal considered not just the previous pixel in the TV signal, but a neighbourhood around the current picture element. Such neighbourhoods are common in the literature, although the labelling of the pixel positions varies. For consistency, all neighbourhoods described from here on will be assumed to use the labelling given in Figure 2.2.

By considering pixels in the previous scan line, as well as those to the left in the current scan line, the prediction becomes 2 dimensional. This is a significant advance on the 1 dimensional difference mapping model described in Section 1.2.

O'Neal used a linear predictor formulated as:

$$\hat{X} = w_1 W + w_2 N + w_3 NW + \dots \tag{2.1}$$

where $\hat{X}$ is the predicted value of X and the w_i are weights. The optimal weights for a given scene were found in advance, and not changed during the encoding of a scene. It should be noted that for O'Neal's work, the prediction error $e = X - \hat{X}$ was not symbol coded but quantised and hence the resultant compression scheme was lossy.

In a limited study, using three low resolution stills digitised from sample TV signals, O'Neal found that there was little utility in using samples beyond W and N for prediction, given the model in Equation 2.1. He also noted that the distribution of prediction errors was Laplacian (i.e. a two-sided exponential distribution) as evidenced in Figure 1.2.

O'Neal commented on the penalty of DPCM coding over PCM coding. That is, by removing the redundancy inherent in the signal one also removes some protection against the effects of transmission noise. However, the opinion was given that quantisation noise was the main cause for loss of quality and that therefore DPCM coding was in general desirable.

2.2.1 The Lossless JPEG Standard

The first major predictive image coding standard came from the Joint Photographic Experts Group (JPEG)[Wal91]. This standardisation effort is best known for its lossy coder (based on the quantisation of DCT coefficients), however the standard did contain a lossless component (LJPEG)[1].

LJPEG can use any of 8 predictors, as shown in Table 2.1. For computational convenience, the pixel weights used in the prediction are either 0, 1/2 or 1. From the results in [MS95b] we see that JPEG7 appears to be the best predictor on average.

The encoder must choose one predictor for an image, and store that choice with the compressed output data. This is an example of global forward adaptation of the predictor; global because the choice is fixed for the whole image and forward because the encoder can scan the image in advance to determine the best predictor. Although this allows some flexibility, it is inefficient as different regions in an image may benefit from different predictors. For example, in a region of strong vertical correlation (e.g. tree trunks in a forest) predictor 2 may be best, while in a noisy region (e.g. foreground grass) the averaging effect of predictor 7 may yield better results.

The prediction residuals can be stored with variations of either Huffman coding or Arithmetic coding. These two versions of LJPEG not only use different symbol coders, they also employ radically different error modelling.

The Huffman flavour of LJPEG requires the encoder to determine the best coding for the prediction residuals in advance and transmit the encoding as part of the message. This is another example of global forward adaptation.

By contrast when LJPEG uses Arithmetic coding, it employs a more advanced error modelling scheme that is capable of altering the way prediction errors are coded on a pixel-by-pixel basis. It does so by making use of information from the encoding of previous pixels and is therefore an example of local backward adaptation. The exact method is described in [LGS92] and more details are given in Section 2.4.

[1]It should be noted that the lossless part of the standard was not the result of a competitive evaluation programme like that used for the lossy JPEG methods.

Predictor	$\hat{X} =$	Performance
JPEG0	0	19.37
JPEG1	W	14.41
JPEG2	N	14.61
JPEG3	NW	15.64
JPEG4	W+N-NW	14.32
JPEG5	W+((N-NW)/2)	13.95
JPEG6	N+((W-NW)/2)	14.01
JPEG7	(W+N)/2	13.83

Table 2.1: The predictors specified by LJPEG. The performance figures are from [MS95b] and show the entropy of the prediction errors, averaged over six 24 bpp colour images. The three colour planes in these images were each predicted separately.

Although offering reasonable lossless compression, LJPEG was never widely used outside the research community. A version of the Huffman flavour of LJPEG is available[2] and has been used as a means of comparison in many papers. However, a freely available implementation of the Arithmetic coding flavour does not seem to exist.

The main lesson to be learnt from LJPEG is that global adaptations are insufficient for good compression performance. This fact has spurred most researchers to look at more adaptive methods.

To advance upon simple schemes like LJPEG, alternative methods for prediction and error modelling are needed. The next two sections will look at how adaptation has been used to approach these issues.

2.3 Adaptive Prediction

All of the predictors mentioned so far are linear functions. However, images typically contain non-linear structures (e.g. edges). This has lead to efforts to find good non-linear predictors. Methods involving vector quantisation[SN94] and neural networks[RD93] have been tried. Although the results in [RD93] look promising, the multi-layer perceptron network used required 6000 passes over the image data to train it, and the overhead for sending this training data does not seem to be factored into the results. Also, the paper only studied the performance of the scheme for one image.

Another way to model the non-linearity of image structure is to switch between linear predictors based on image characteristics. Although the actual predictors are still linear functions, the switching mechanism can attempt to deal with non-linear features such as edges. Switching schemes can be backward or forward adaptive, depending on whether they make their decision on the basis of past or future data respectively.

[2]ftp://ftp.cs.cornell.edu/pub/multimed/

2.3.1 Backward Adaptive Techniques

Backward adaptive schemes use previously transmitted pixels to chose a predictor. This means that the encoder and decoder can make the same decision without any explicit communication.

Zhang

An early scheme is given by Zhang[Zha82] and summarised in [MPG85]. Zhang defines four predictors, one for each of four different image characteristics. The different predictors are designed to cope with flat regions, horizontal edges, non-horizontal edges and regions of texture. Heuristics, based on a small number of pixel pair differences, are given for determining the characteristic of the current image region. Although detailed and complex, Zhang's scheme does not perform as well as more simple predictors[MS95c].

Median Adaptive Predictor

One of the most widely reported [Mar90, MS95c, WSS96, MWSM97] switching schemes is the Median Adaptive Predictor (MAP). In 1990 Martucci[Mar90] suggested using the median value of a set of predictive functions as the actual prediction. Three suitable predictive functions are W, N, and $W + N - NW$.

However, in [WSS96] the predictive scheme is given as:

$$\hat{X} = \begin{cases} \min(W, N) & \text{if } NW \geq \max(W, N) \\ \max(W, N) & \text{if } NW \leq \min(W, N) \\ N + W - NW & \text{otherwise} \end{cases} \tag{2.2}$$

Simple arithmetic reasoning will show that the above formulation is the same as choosing the median of W, N, and $W + N - NW$. However, as noted in [WSS96] this way of looking at MAP gives an alternative reasoning for the way the scheme functions. Suppose, NW is the brightest pixel (highest value) then we assume an edge feature. The maximum of W and N then partners NW on the bright side of the edge, while the minimum of W and N joins X on the dark side. Thus, $\min(W, N)$ forms a good prediction for X. Similar reasoning holds for the case when NW is the darkest of the three pixels. In the case where NW is not an extreme value, $W + N - NW$ is used, which models the local pixel values as a plane and predicts X accordingly. This description of MAP as a scheme for adapting to edge features, lead to it being called the Median Edge Detection (MED) predictor in [WSS96].

In order to avoid confusion, MED will be used to describe the specific predictor described above, whereas MAP will be used for the concept of using the median value from a set of predictors.

Gradient Adjusted Prediction

This idea of explicitly looking for edges in the image data was also used by Wu in [Wu96]. He uses the local horizontal and vertical image gradients, given by:

$$\begin{aligned} d_h &= |W - WW| + |N - NW| + |NE - N| \\ d_v &= |W - NW| + |N - NN| + |NE - NNE| \end{aligned} \tag{2.3}$$

to help predict X:

```
if($d_v - d_h > 80$)                          //sharp horizontal edge
    $\hat{X} = W$
else if($d_v - d_h < -80$)                    //sharp vertical edge
    $\hat{X} = N$
else
{
    $\hat{X} = (W + N)/2 + (NE - NW)/4$       //assume smoothness first
    if($d_v - d_h > 32$)                      //horizontal edge
        $\hat{X} = (\hat{X} + W)/2$
    else if($d_v - d_h > 8$)                  //weak horizontal edge
        $\hat{X} = (3\hat{X} + W)/4$
    else if($d_v - d_h < -32$)                //vertical edge
        $\hat{X} = (\hat{X} + N)/2$
    else if($d_v - d_h < -8$)                 //weak vertical edge
        $\hat{X} = (3\hat{X} + N)/4$
}                                                                    (2.4)
```

By classifying edges as either strong, normal or weak, GAP does more modelling than MED. This extra modelling gives GAP better performance than MED, although typically not by a large margin. The extra work also makes GAP more computationally expensive. The use of MED in JPEG-LS indicates that in terms of a joint complexity-performance judgment, MED has the upper hand.

History Based Blending

An idea presented in [STM97] is to blend the predictions of several predictors to form an overall prediction. This approach can be seen as asking the advice of several experts and then combining their advice. The final prediction ($\hat{X}$) is a linear combination of the results of the individual predictive functions ($\hat{X}_i$), i.e.:

$$\hat{X} = \sum a_i \cdot \hat{X}_i \tag{2.5}$$

The vector of weights $\underline{a} = (a_1, a_2, \ldots)$ can be calculated by solving a linear system of the form $\underline{P} \cdot \underline{a} = \underline{Q}$ where $\underline{P}$ is the matrix of past predictions and $\underline{Q}$ is the matrix of past pixel values. In [STM97] the authors describe how to avoid computing $\underline{P}$ and $\underline{Q}$ in their entirety, for every pixel. Instead these matrices are approximated by a number of counts. These counts are updated in such a way that past information is slowly depreciated. That the past information is used at all, gives the *history* element of the scheme.

The cost of computing the weights $\underline{a}$ is given as $O(n^3)$ when there are n predictors. To keep computational costs down and yet permit a larger number of predictors, the HBB concept is cascaded. Three units, each with predictors designed for a specific image characteristic, produce a prediction via history based blending. The output of these three units is again blended in the same way to produce the final prediction value. The three units are:

Unit	Predictors		
Noise unit	$(W+N)/2$	$(2W+N+NE)/4$	$(W+N+NW+NE)/4$
Smooth unit	$W+N-NW$	$2W-WW$	$2N-NN$
Edge unit	W	N	NE

The completed scheme given in [STM97] is shown to be superior in terms of compression performance to other presented results in which MED and GAP were used. This indicates that HBB may be more effective than either MED or GAP, but obviously at the cost of a great deal of computation.

2.3.2 Forward Adaptive Switching

In LJPEG the choice of predictor for an image is chosen by the encoder (forward adaptation) and transmitted as overhead to the decoder. If we allow the encoder to adapt its choice of predictor to the local characteristics of the image data, better prediction may result. However, the overhead caused by transmitting these predictor choices to the decoder will increase.

Block Based Adaptation

One simple scheme, mentioned in [MS95c], is to split the image into 8 x 8 pixel blocks and chose the best of the LJPEG predictors to use for the block. In [MS95c] the trivial predictor LJPEG0 is replaced by $W+(NE-NW)/2$. This predictor choice is then encoded as 3 bits of overhead ($3/(8 \times 8) = 0.047$ bpp overhead). In [MS95c] the *best* predictor is determined to be the one that gives the least sum of absolute prediction errors for the block[3]. This method has been shown to give results that are better than MAP [MS95c].

In [Lee95] a similar scheme is given, except 16 predictors are available. Furthermore, the overhead of sending the predictor index for each block is itself coded by an adaptive Arithmetic code, hence lowering the overhead.

[3]Note, the fact that a given predictor has the least sum of absolute prediction errors, does not necessarily imply that the same predictor gives the least contribution to the total entropy of the prediction residuals.

Prediction Patterns

Still better results were reported in [MS95c] for a scheme based on *prediction patterns* [MS95a]. This scheme again divides the image into blocks, but instead of a single predictor each block is assigned a prediction pattern index. A prediction pattern specifies how multiple predictors should be used to best predict pixel values in a block of pixels.

The complex parts of the scheme, choosing a codebook of prediction patterns and choosing the best prediction pattern for a given block, are familiar problems in the domain of vector quantisation[Cla95]. In [MS95a], the LBG algorithm is used for codebook design and exhaustive search is used to select the best prediction pattern for a given block.

2.3.3 Adaptive Prediction Summary

Backward adaptation naturally seems to have many advantages over forward adaptation. Backward adaptation carries no overhead, allows an individual predictor selection to be made for each pixel and permits a single-pass image coder. However backward adaptation has its problems as well.

This is especially evidenced in Zhang's scheme, which although well reasoned shows poor performance. The predictors and predictor selection heuristics also appear well-designed, but the predictors are made very specific to the expected image properties of a given type of region, as reported by the heuristics. Thus, when the heuristics fail, very poor prediction is likely.

In contrast MED uses more general purpose predictors that fare well in most conditions. Thus, a poor predictor selection decision will result in very poor prediction less frequently that Zhang's scheme. This is taken further by HBB, which blends predictors in an attempt to mitigate the effects of unsuitable predictions.

Forward adaptation can prevent the worst prediction errors by looking ahead at the data to come. However, forward adaptation must always balance its accuracy against the overhead incurred.

2.4 Advanced Error Modelling

To work efficiently, symbol coders such as Huffman and Arithmetic coding require accurate estimates of the distributions of prediction errors. However, this distribution is rarely constant over all regions in an image. For example, in a smooth area of an image (where most predictors work well) the prediction error $\varepsilon = X - \hat{X}$ is highly likely to be 0. Whereas in a textured or noisy region (where most predictors have difficulty) ε is non-zero with a high probability. Given these changing distributions, adaptive error modelling is required for good compression performance.

By using adaptive error modelling to take account of changing prediction error distributions, we no longer assume prediction errors to be identically distributed. Thus, we are now going beyond the i.i.d. assumption of zero-order entropy (see Section 1.2.1) and can expect compression performance that improves on the entropy of the prediction errors as given by Equation 1.2.

One simple solution would be to assume that the prediction errors come from a single source whose statistics change over time. Frequency counts of errors could be kept and a model of the distribution built accordingly. However, this assumes that the distribution of prediction errors changes gradually along the scan path used. This is very unlikely to occur in most images.

A better solution is to assume that the prediction errors come from multiple sources. Each source has its own distribution and relates to different regions in the image. Constructing such a model requires two main problems to be solved; determining which source to use for a given pixel and adapting the individual sources to the image characteristics.

Most schemes solve the first problem by using some characteristic of the local pixel values (or the local prediction errors) in a way not unlike the switching prediction schemes of the previous section. Thus the pixel's relation to its neighbours - its *context* - is important. Because the probabilities $p(\varepsilon = i)$ are conditioned in this context, these are often called *conditioning contexts*. Algorithms for solving this issue can be termed *context determination* algorithms and some examples are given in Section 2.4.1.

The solution to the second problem, conditioning the error probabilities based on the context, generally uses some form of frequency counting on a per context basis. The exact details tend to vary with the actual compression scheme used.

One problem that must be overcome by all context coding schemes, is that of context dilution. This occurs when the combination of contexts and frequency counts becomes too large. In this case, contexts are entered infrequently and accurate statistics cannot be acquired. This leads to poor error modelling.

This problem can be avoided by limiting either the number of contexts or the number of frequency counts. However, if too few contexts are available, the error modelling will also be inefficient.

It was mentioned earlier that prediction errors have traditionally been assumed to be distributed as a two sided Laplacian centred at zero. However, many authors have found that something different happens when errors are context modelled. Contexts that represent active regions in the image often have error distributions which have a centre that is non-zero. Such distributions show evidence of bias and a measure of this bias is given by the mean prediction error in a given context. The process of *bias cancellation* adjusts prediction residuals to counteract the contextual bias, hence providing a more accurate error model.

The issue of context modelling is given a good theoretical coverage in [WRA96]. The use of *tree models*[4] for lossless image compression is suggested. Their approach builds a tree, based on the image data received, thus keeping the number of contexts down to the level required. The results presented in [WRA96] were at the time the best available. However, the use of more heuristic context modelling schemes, for example [Wu96] that have shown better compression performance, indicates that the full rigor of the methods

[4]Tree models contain Markov Models as a special case.

in [WRA96] are unnecessary for good performance.

2.4.1 Examples of Context Determination Algorithms

The purpose of a context determination algorithm is to assign the current pixel to one of a fixed set of contexts, based on the current neighbourhood. This should be done in such a way that pixels in active regions get put into active contexts and likewise pixels in smooth regions get put into smooth contexts. In this way, we expect all prediction errors in a given context to resemble the output of a single source. Hence, modelling all errors in a given context as coming from a single distribution permits efficient coding.

Error Buckets

Examining the prediction errors experienced at neighbouring pixels is one way to ascertain the local image activity. Such a scheme was first presented in [TLR85], and makes use of *error buckets*. The full range of errors $-255 \leq \varepsilon \leq 255$ (for 8 bpp images) is divided into 5 error buckets (a crude form of quantisation). The context of the current pixel is calculated as the cross-product of the error buckets of the closest three pixels (W, N and NW). Thus, there are $5 \times 5 \times 5 = 625$ contexts. Better results are reported when 11 error buckets are used.

The use of error buckets for modelling also appeared in the SUNSET coder[Lan91, Lan88] and the Arithmetic coding version of LJPEG[LGS92]. Both these approaches use the bit position of the most significant bit in the absolute prediction error as the error bucket index.

Local Gradients

The purpose of context determination is to identify smooth regions in the image from active regions. This can be done by considering the local image gradients. LOCO-I[WSS96] uses 4 gradients, $g_1 = NE - N$, $g_2 = N - NW$, $g_3 = NW - W$ and $g_4 = W - WW$. Calculating a context based on the cross-product of these gradients would result in having just under 7×10^{10} contexts! This is far more than can be practically handled and would certainly cause problems with context dilution.

To overcome this, the 4 gradients were quantised into a number of roughly equiprobable buckets. g_1, g_2 and g_3 were each quantised into 9 regions, while g_4, being further from X, is quantised into 3 regions. This results in $9^3 \times 3 = 2187$ contexts.

This was still deemed to be too many, so context merging was used to cut the number practically in half. By assuming,

$$p(\varepsilon = d | C = [q_1, q_2, q_3, q_4]) = p(\varepsilon = -d | C = [-q_1, -q_2, -q_3, -q_4]) \tag{2.6}$$

where C is the current context and the q_i are the quantised g_i, contexts that are symmetric around zero can be merged. This merging process reduces the number of contexts to 1094.

Local Texture and Error Energy

In [Wu96] Wu suggests generating contexts based on a combination of local texturing and error energy. Moreover, the scheme presented used such hybrid contexts only for bias cancellation, error distribution modelling is based on the local error energy metric alone.

To characterise local texture, Wu considers a set of eight local events:

$$\mathcal{S} = \{N, W, NW, NE, NN, WW, 2N - NN, 2W - WW\} \tag{2.7}$$

An 8 bit binary number $Q_t = b_7 b_6 b_5 \dots b_0$, is then formed by:

$$b_k = \begin{cases} 0 & \text{if} \quad \mathcal{S}_k \geq \hat{X} \\ 1 & \text{if} \quad \mathcal{S}_k < \hat{X} \end{cases} \tag{2.8}$$

In [Wu96] GAP is used to determine $\hat{X}$. By using $\hat{X}$ as a threshold, Q_t is firmly linked to the prediction and therefore the prediction error. The local error energy (Δ) is determined by:

$$\Delta = a \min(d_h, d_v) + b \max(|\varepsilon_W|, |\varepsilon_N|) \tag{2.9}$$

where d_h and d_v are as given in Equation 2.3 and ε_W and ε_N are the prediction errors from W and N respectively. The coefficients a and b are determined by an offline optimisation process. Δ is then quantised into 8 levels to form Q_d; the quantisation levels again being determined by offline optimisation with a training set. Wu based his choice of d_h, d_v, $|\varepsilon_W|$ and $|\varepsilon_N|$ as the basis for Q_d, because of a high average correlation coefficient (> 0.3) between these values and $|\varepsilon|$.

This scheme uses $256 \times 8 = 2048$ contexts for bias cancellation, but only 8 contexts for error distribution modelling.

2.5 Some Examples of Complete Schemes

The building blocks of many lossless image compression schemes have now been detailed, but no complete schemes have been described. For reasons of space, and to avoid excessive replication of material, a sample of three complete schemes is given below.

JPEG-LS[ITU96] is the new lossless image coding standard from the JPEG group and replaces LJPEG. JPEG-LS (heavily based on LOCO-I[WSS96]) is an example of an image coder built from a simple switching predictor with context based error modelling. Many other examples of coding schemes built along similar lines exist[WM97, STM97, LGS92, Lan91, Wal91].

Although JPEG-LS was designed to offer good compression performance at a reasonable level of computational complexity, it still has more computational requirements than some schemes. FELICS[HV93] is an example of a lossless image compression scheme that makes use of some interesting insights to provide acceptable compression at great speed. It is detailed in Section 2.5.2 .

Input Symbol	for $k = 0$	for $k = 1$	for $k = 2$
0	*0*	0*0*	00*0*
1	*10*	1*0*	01*0*
2	*110*	0*10*	10*0*
3	*1110*	1*10*	11*0*
4	*11110*	0*110*	00*10*
5	*111110*	1*110*	01*10*
6	*1111110*	0*1110*	10*10*

Table 2.2: Example Golomb-Rice codes for three values of k. Note, bits constituting the unary section of the code are shown in italics.

On the other end of the complexity scale is a scheme named TMW[MT97] after its creators. This considers image coding as a two part process and is briefly discussed in Section 2.5.3.

2.5.1 The New JPEG Standard - JPEG-LS

Due to the perceived inadequacy of the LJPEG standard, the JPEG committee put out a call for proposals for a new lossless image compression standard. LOCO-I[5][WSS96] was put forward as one of the contenders and the new standard JPEG-LS[ITU96] is heavily based on it.

JPEG-LS uses the MED predictor and a gradient based context determination scheme. The main difference between the context determination of JPEG-LS compared to LOCO-I is that JPEG-LS does not use the g_4 gradient.

In order to help keep complexity low and reduce model cost, JPEG-LS uses a symbol coder that requires only a single parameter to describe an adaptive code. The scheme used is Golomb-Rice (GR) coding (often known only as Rice coding) and the parameter required is k. GR codes are a subset of the Huffman codes and have been shown to be the optimal Huffman codes for symbols from a geometric distribution.

Golomb-Rice codes are generated as two parts; the first is made from the k least significant bits of the input symbol and the latter is the remainder of the input symbol in unary format. Some example codes are given in Table 2.2. Selecting the correct k to use when coding a prediction error is very important. The ideal value for k is strongly related to the logarithm of the expected prediction error magnitude[WSS96].

In JPEG-LS four counts are kept for every context. The variable N counts the number of times a context is visited and A accumulates the absolute value of the prediction errors in the context. The expectation for the prediction error magnitude is given by A/N and k is thus:

[5]LOw COmplexity LOssless COmpression for Images.

$$\min(k|2^k N \geq A) \tag{2.10}$$

In [WSS96] it is noted that this can be conveniently written in C as:

```
for(k=0; (N<<k)<A; k++);
```

As for the other two per-context counts; B accumulates the prediction errors in the context (actual values, not magnitudes) and C tracks the prediction error bias. C is updated when B reaches a threshold. Whenever C is updated, A and B are also updated to take account of the now cancelled bias.

Although, JPEG-LS is already highly adaptive, it has the potential for inefficiency due to contextual error statistics changing as image coding progresses. To counter this, the contextual counts are *reset* after the context has been visited a predetermined number of times. This resetting procedure halves all the counts for the context, thus reducing the importance of past data and allowing new statistical information to play a greater role.

One limitation of GR codes is that, like general Huffman codes, they are limited to a minimum code length of one bit per symbol. In very highly redundant regions of an image this can represent a serious penalty. This effect is limited by the introduction of a *run mode*. The run mode is entered when a special context is entered. This *run-context* is the context for which the local (quantised) image gradients are all zero. Once in run-mode the next symbol is a GR coded run-length that determines how many identical symbols follow. As the run mode can be entered even when no run is present, runs can be of zero length.

Weinberger et al. compared LOCO-I to many other lossless schemes, including the LJPEG standard. LOCO-I was found to compress better than all the other methods except CALICS [Wu96]. Weinberger et al. claim that the slightly better performance of CALICS is at the cost of an extra order of magnitude in computational complexity. Its good performance coupled with a relatively simple implementation appears to have been persuasive, as the new standard for lossless JPEG [197] is strongly based on LOCO-I.

2.5.2 The Fast and Efficient Image Compression System

In [HV93] Howard presents his Fast, Efficient, Lossless Image Compression System. FELICS is built around the simple observation that the probability distribution of the current pixel value X, is roughly flat within the range formed by the two nearest pixel's values (W and N) and decays exponentially outside that range.

For typical images, Howard finds that the value X is *in range* (i.e. between the values of W and N) about 50% of the time. A single bit is thus used for each pixel to determine whether X is *in range* or not. As the *in range* probability distribution is roughly flat, a binary code is both theoretically suitable and computationally simple. Howard uses an adjusted binary code that gives a slight bias to values in the centre of the range (where a slight hump in the probability distribution is found).

If X is not *in range*, a further bit is sent to indicate whether X falls below the bottom of the range or above the top of the range. The distribution of X was found to be symmetrical outside the range, therefore the use of a single bit is again justified. As the distribution for values of X falls exponentially beyond the given range, Rice codes give an efficient way to code the value of X, with respect to either the lower or upper boundary of the range. The parameter for the Rice code is determined by contextual adaptation; the context for a given pixel is determined by δ , where $\delta = \max(\mathrm{W,N}) - \min(\mathrm{W,N})$. Note, δ is not quantised.

Results given in [HV93] indicate that FELICS has roughly the same compression performance as the original lossless JPEG scheme but has about five times the data throughput on a given machine. It is notable that for images that are highly compressible by LJPEG, FELICS's relative compression performance deteriorates. This is in part due to the assumption that for *typical* images the *in range* probability is about 1/2. An atypical image (i.e. one that is highly compressible) is likely to have an *in range* probability that is distinctly different from 1/2 and therefore the use of one bit to code it is inefficient. In [HV93], Howard ignores these images on the argument that they are rare. Also, to model the *in range* probability would add considerably to the computational complexity of the scheme.

In [HV93] Howard notes that by directly modelling the probability distribution for X, FELICS neatly combines the prediction and error modelling steps of lossless compression. However, an alternative view of FELICS presents itself, if we consider it to be a predictor switching scheme. In essence FELICS uses three predictors W, N and $(W + N)/2$ (the latter is evidenced by the bias given towards values in the centre of the range). Forward adaptation is used to determine whether or not to use $(W + N)/2$ for prediction. If X is *in range* $(W + N)/2$ is used, although it is not necessarily the best predictor. As well as indicating a predictor selection that one bit also limits the prediction error and indicates a flat model for the prediction error distribution.

If X is out of range, forward adaptation is again used to make a predictor selection decision, this time choosing between W and N. This extra bit, along with the out of range knowledge, effectively determine the sign of the prediction error, leaving only its magnitude to be coded by Rice coding. Thus, the efficiency of FELICS comes largely from the multiple inferences that can made for each bit of overhead information.

2.5.3 Two Part Coding

The ultimate product of forward adaptation is the two part coding scheme. The encoder does image analysis on the input and attempts to capture the essence of the image (its global characteristics). This model of the image is then transmitted to the decoder. Following this, the image is compressed according to the model built for it. Such a scheme is presented in [MT97].

A set of linear predictors are used, the weights for which are computed by the encoder. Instead of predicting a single value for each pixel, the predictors calculate a probability distribution. Each predictor produces a prediction distribution centred on a value calculated by linear prediction. The distribution formed is a variant of the t-distribution. The width of the distribution is determined by a weighted combination of local prediction errors; the weights involved being part of the image model.

These distributions are blended together, yielding a composite distribution. Note, this takes predictor blending one step further than HBB[STM97]. However, different predictors will best apply to different areas, so a notion of predictor efficiency is built into the system. The prediction distributions are then blended by a linear function, which depends partly on a weight calculated during the analysis phase (for global efficiency) and partly on the recent image past (for local efficiency).

It is not necessary to compute the composite prediction distribution for all potential values of the current pixel. Instead it is calculated for various ranges, thus allowing the determination of each actual pixel to be decomposed into a sequence of binary events. These binary events are encoded with an arithmetic coder.

All the weights used by TMW are continuous and this aids the image analysis phase. A variety of optimisation techniques are used to fine tune these weights. Note, that some parameters that appear in the image model, such as the number of predictors to use, are not optimised and must be provided by the user.

The compression results of TMW surpassed those of the previously best known image compression scheme CALIC, for all the images tested in [MT97]. Although not clearly stated in [MT97], personal comments from the author suggest that the image analysis phase of the scheme is computationally very intensive. This makes the current incarnation of TMW unsuitable for many widespread applications.

2.5.4 Considering Extensions

When considering the three schemes just described, with a view to extending them to colour images, video, etc., some issues become apparent. It is hard to see how FELICS could maintain its simplicity and efficiency if extra dimensions were added to the input data. Although highly adaptive, TMW is already highly complex. An extended version would exhibit very high computational complexity. Both schemes, though successful for greyscale images, seem inappropriate as a foundation for extended lossless image coding.

However, JPEG-LS, and similar schemes such as CALICS and SUNSET, seem a better starting point. By adding spectral, temporal and interview considerations to prediction, predictor selection and error modelling, extended lossless image compression should be attainable.

2.6 Miscellaneous Techniques

Although many mainstream ideas have already been covered in this chapter, there are still other techniques which should be included to give a comprehensive review. Such techniques and schemes are detailed in this section.

Probably the most widely used standard lossless image compression scheme is the Graphics Interchange Format. Although much used, it is not a good example of a state-of-the-art lossless compression scheme, as detailed in Section 2.6.1. Hierarchical decomposition is a powerful tool in any form of image processing. Although not frequently used in lossless compression, some examples of hierarchical methods are given in Section 2.6.2. Another form of decomposition is bit plane decomposition. This and associated compression schemes are mentioned in Section 2.6.3.

Finally, in Section 2.6.4 we examine the interesting but rarely used concept of scan path optimisation.

2.6.1 The Graphics Interchange Format (GIF)

The first widely used standard for lossless image compression was the Graphics Interchange Format (GIF) standard invented by Compuserve[Rim92]. It is based on Welch's popular extension of the LZ78 coding scheme. GIF uses a colour palette, that contains a maximum of 256 entries. Each entry specifies one colour using a maximum of 8 bits for each of red, green and blue. The colour palette must be built prior to coding and is sent along with the compressed data. Note that images with more than 256 colours cannot be losslessly coded with GIF. The LZW coding is applied directly to the pixel data, therefore there is no mapping stage. Due to the inherently adaptive nature of LZW, it can be seen as combining the modelling and coding stages into one.

LZW (and hence GIF) works well for computer generated images (especially icons) which have a lot of pixel sequences repeated exactly. However, for real pictures it performs less well. This is due to the noise, inherent in any form of image capture, breaking the repeating sequences of pixels that LZW depends on. Also, the limitation of 256 different pixel values became a problem as cheaper memory made 24 bit (16777216 colours) images more popular.

2.6.2 Hierarchical Methods

Hierarchical forms of image compression have not been very popular for lossless applications. This is because they generally have inferior performance to sequential schemes of similar complexity. The one main advantage of hierarchical schemes is that of progressive decoding[Qiu99]. That is, a coarse representation of the image can quickly be built up, allowing a user to determine if the rest of the image should be decoded in full. This can be very useful if data is coming over a slow link or if the image in question is very large.

One simple hierarchical method is quadtree coding. An image region (typically square) is either represented as a single value, or if that is not satisfactory, it is split into four regions. If present, the four children of the original region are then processed recursively. This technique has application in some areas, but is generally unsuitable for lossless image compression because real images will require almost all nodes to be split. This causes the overhead of representing the quadtree to exceed any advantage in representing flat regions of the image with a single value.

$$\begin{array}{ccccc}
L & H_1 & H_3 & H_1 & L \\
H_1 & H_2 & H_1 & H_2 & H_1 \\
H_3 & H_1 & H_4 & H_1 & H_3 \\
H_1 & H_2 & H_1 & H_2 & H_1 \\
L & H_1 & H_3 & H_1 & L
\end{array}$$

Figure 2.3: A small example of hierarchical interpolative decomposition. The L pixels remain after four passes. The H pixels are replaced by their prediction error; the subscript determines in which passes the pixel is removed.

$$\begin{array}{ccc}
A & & B \\
& H & \\
D & & C
\end{array}
\qquad\qquad
\begin{array}{ccc}
& A & \\
D & H & B \\
& C &
\end{array}$$

Figure 2.4: Neighbours in a hierarchical scheme.

Interpolative Coding

Another way to form a coarse representation of an image is to simply remove pixels from the image and use the remaining pixels to form a lower resolution image. The removed pixels can then be predicted by some interpolating method, based on the pixels remaining. We call the pixels remaining after one pass, the L band and the prediction errors that replace the removed pixels the H band. By using multiple passes a truly hierarchical scheme emerges.

One scheme that has been mentioned in several papers[Rob97, HV91] removes every even pixel on every odd row and every odd pixel on every even row. An example of this decomposition is shown in Figure 2.3. It should be noted that the sampling lattice must be rotated by 45^o after each pass.

The decoder is sent the remaining L pixels first (usually DPCM coded) and then H pixels; from H_4 to H_1 in the current example. Note, every H pixel (excluded edge effects) is surrounded by 4 pixels at a higher level, either as a square or as a diamond. These four (see Figure 2.4) can be used to interpolate (predict) the H pixel and hence only the prediction error need be sent. In [Rob97] Robinson introduces some other predictors, in particular it is noted that:

$$\hat{H} = \begin{cases} (A+C)/2 & \text{if } |A-C| < |B-D| \\ (B+D)/2 & \text{if } |B-D| < |A-C| \end{cases} \tag{2.11}$$

is much better than the linear combination $\hat{H} = (A+B+C+D)/4$.

One drawback of interpolative schemes is that their intermediate resolution images suffer from aliasing. This is due to the lack of filtering in the sub-sampling process.

The S-Transform

The S-Transform, which has similarities to the multiresolution Haar representation, can be used for hierarchical methods. Consider a sequence of $2N$ symbols $s[n], s = 1, \ldots, 2N$. $s[n]$ can be represented as two sequences of symbols, one that represents the low frequency component (averaging) of s and one that holds the high frequency component (differencing). These sequences are defined as:

$$l[n] = \left\lfloor \frac{s[2n] + s[2n+1]}{2} \right\rfloor \tag{2.12}$$

$$h[n] = s[2n] - s[2n+1] \tag{2.13}$$

and the reverse transform is:

$$\begin{aligned} s[2n] &= l[n] + \left\lfloor \frac{h[n]+1}{2} \right\rfloor \\ s[2n+1] &= s[2n] - h[n] \end{aligned} \tag{2.14}$$

Although the S-Transform is inherently 1-dimensional, by transforming first the rows and then the columns a 2-dimensional transform is achieved. After the rows and columns have both been transformed, one quarter of the image is entirely low frequency components. This band can be transformed recursively to form a multiresolution image pyramid. An example is given in Figure 2.5.

Although potentially useful for lossless image compression, the use of the S-Transform as a mapping stage is generally inferior to simple DPCM methods. However, the performance was improved by Said and Pearlman [SP96] who introduced the S+P-Transform. The P is added to indicate the presence of prediction that is used to lower the entropy of the representation. High frequency components ($h[n]$) are predicted from previously seen coefficients of both high and low frequency.

2.6.3 Bit Plane Decomposition

One of the major problems in modelling image data is the relatively large alphabet of possible pixels values (typically 2^8). One way to avoid this problem is to decompose a k bpp image into k different bit planes. The k 1 bpp images that now represent the original can then be compressed using run-length or context based coding. If the pixel values are coded with a Gray code, rather than the standard binary code, the resultant bit planes are spatially more uniform and better compression is seen[GW92].

One problem with bit-plane decomposition is that the entropy of the resultant k planes is generally larger than that of the original image. In [Yu95] it is given that decomposing a

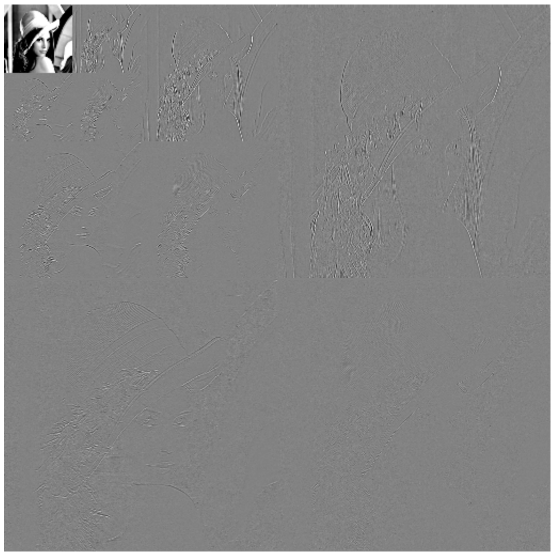


Figure 2.5: Result of a multiresolution S-Transform.

k bpp image into $2^k - 1$ planes can overcome any penalty normally associated with binary decomposition techniques.

An interesting hybrid of bit plane decomposition and predictive coding is presented in [SA92]. Every pixel is coded as a prefix and a suffix. The prefix represents the number of contiguous bits, going from the most significant to the least, that the current pixel has in common with a reference pixel. The suffix is then all but one of the lower order bits; if the first n bits of the current pixel match the reference, then the $n+1^{\text{th}}$ bit of the current pixel is implicitly not the $n+1^{\text{th}}$ bit of the reference. For example, if we had:

reference pixel	=	10110010
current pixel	=	10111010

the prefix would be 4 and the suffix would be 010. In [SA92] the prefix is Huffman coding and the reference pixel can be chosen as either W or N. One scheme presented in [SA92] alternates between potential reference pixels when the value of the prefix falls below a preset threshold.

2.6.4 Scan Path Optimisation

Almost all the compression schemes mentioned previously have used the raster scan ordering; one exception being HBB[STM97]. Theoretical work suggests that an optimal pixel ordering would be related to a Hilbert space filling curve. However, such a method is considered impractical for the compression of greyscale image data[MS95c].

Instead of trying to find a universally optimal scan ordering, Memon et. al. have tried optimising the scan path to take account of a given image and its features. The problem of finding an optimal path can be attacked with graph theory. Each pixel is considered a vertex and a weighted edge joins every pixel to its four closest neighbours (above, below, left and right). The weight on each edge is the difference between the two relevant pixel values. This weight can also be seen as the prediction error from simple DPCM when using a scan model that traverses the edge.

It is useful to find a scan model that minimises the absolute weight of prediction residuals. A minimum absolute weight model can be found by computing a minimum weight spanning tree of the graph described above. Fortunately, efficient algorithms are known for this [Sed92].

The drawback of this scan based method, when encoding single images, is that the cost of recording the scan path outweighs most of the performance gained by using it. However, Memon et. al. have applied their scan method to colour images with successful results (see Section 3.1.3).

2.7 Summary

In this chapter we have seen some of the very many techniques used to tackle the problem of lossless greyscale image compression. Many successful schemes were seen to use adaptive prediction, followed by advanced error modelling through context based conditioning. By extending the predictive functions and context models to include spectral, temporal and interview elements, we might reasonably hope to achieve the goal of extended lossless image coding.

Chapter 3

Beyond Greyscale Image Compression

This chapter covers the literature underlying the compression of colour images, video and multiview imagery. Compared to the work on greyscale image compression, the literature on lossless colour image compression, discussed in Section 3.1, is very sparse. It is mostly recent and much of it is related to the specific case of compressing multiband satellite data.

While the literature on lossless video compression is practically non-existent, a huge amount exists on lossy video compression, including Clarke's excellent book "Digital Compression of Still Images and Video" [Cla95]. This is covered briefly in Section 3.2, with particular emphasis being placed on techniques that may help extend lossless methods to video. The nature of multiview stereo imagery, which can be seen as a special case of video data, is introduced in Section 3.3.

Finally, a summary of the literature covered is given in Section 3.4.

3.1 Extensions for Colour

Any form of colour (or multiband) image can be thought of as function $f(b, x, y)$ where b ranges over the colour bands $(0 \leq b \leq B-1)$ and x and y range over the width and height respectively. Most lossless image compression schemes, as discussed in previous chapters, are designed to deal with functions of the form $f(x, y)$ and so consider a colour image as B functions $f_b(x, y)$. However, by utilising the interband correlation additional compression can be gained. This process can also be thought of as using interband relationships to better decorrelate the data prior to entropy coding.

3.1.1 Colour Space Transforms

To have physical meaning, the values of the B pixels at a fixed spatial location should represent a point in some colour space[FvDFH93]. We shall consider the default colour space to be the Red, Green and Blue (RGB) space. This is often represented as a cube, with three orthogonal axis giving the amount of red, green and blue. The cube is usually scaled to have sides of unit length, $(0, 0, 0)$ being black and $(1, 1, 1)$ being white. For image storage it is often the case that 8 bits are used for each colour component (24 bpp in total) and therefore we can consider the standard RGB cube to be quantised and scaled by a factor of 255.

The RGB colour space is related to our physical perception of colour and is used by display devices such as CRTs and LCDs. However, other colour representations are needed for other tasks. For example, when printing, the RGB scheme (an additive colour scheme) is unusable. Instead, a subtractive scheme (so called because colour components are removed from the white of the paper) based on cyan, magenta and yellow is used. To improve printing results, a fourth component is added that represents black. This gives the CMYK (Cyan-Magenta-Yellow-blacK) colour scheme.

Pixel values are often transformed into colour spaces that aim to separate luminance and chrominance information. Conversion to YIQ and YUV have been popular for television broadcast applications, where Y presents luminance and I,Q,U and V are chrominance components. Y can be calculated from RGB by:

$$Y = 0.299R + 0.587G + 0.114B \tag{3.1}$$

For the YUV system, the components U and V are described by colour differences:

$$\begin{aligned} U &= B - Y \\ V &= R - Y \end{aligned} \tag{3.2}$$

However, this formulation allows U and V to be negative. The YCrCb formulation gets around this by scaling and zero-shifting the components:

$$\begin{aligned} Cr &= (V/1.6) + 0.5 \\ Cb &= (U/2) + 0.5 \end{aligned} \tag{3.3}$$

Note, the 0.5 shift assumes a colour space of unit dimension; using 8 bit imagery requires a shift of 128.

By converting to a colour space such as YCrCb, the luminance and chrominance components are decorrelated. This allows the chrominance components to be handled at a different resolution to the luminance components. This is advantageous for lossy coding schemes as the perceptual quality of an image is related more strongly to the luminance than the chrominance part of the signal. Hence, the Cr and Cb bands are often subsampled to half the horizontal and vertical resolution of the luminance band. This gives great advances in compression but is not applicable for lossless coding.

Although, colour space conversion is often seen as a decorrelating technique, the entropy of the resultant image can be greater than that of the original[Tur94]. Furthermore as the

Y, Cr and Cb bands have little correlation, subsequent interband methods are unlikely to prove effective. As a consequence, traditional colour space conversions seem ineffective for lossless compression purposes.

Optimal Transforms

Theoretically, the transform best able to decorrelate the various colour bands in an image, would be the Karhunen-Loève Transform (KLT). In [APL98] a reversible approximation to the KLT is given and its usefulness in lossless image compression is assessed.

In [APL98] the encoder computes the matrix for the lossless KLT for every region in the image (regions are either generated by simple block division or colour segmentation) and transmits an approximation of the matrix to the decoder. This scheme takes into account the possibility that colour statistics will vary over the extent of an image.

The results of several experiments are reported in [APL98], mostly using a CMYK (32 bpp) image *musicians*. It was found that the lossless KLT, followed by Arithmetic coding, did compress the data but was inferior to the use of the JPEG7 predictor applied to each band separately. The lossless KLT was then used in concert with spatial decorrelation (JPEG7). It was found that the best results were obtained by using the KLT after the spatial prediction step. In this way, the KLT is used to assist error modelling.

The conclusions drawn in the paper were that spatial (intraband) correlations are more important than spectral (interband) correlations and that their lossless KLT aided compression if it followed a spatial decorrelation step. However, in the results provided it is clear that a purely intraband version of CALIC was often able to surpass the KLT enhanced coder presented.

The results of [APL98] are somewhat counter-intuitive. It seems likely that spectral decorrelation should have a greater utility than to be used to decorrelate spatial prediction residuals. It also suggests, that global application of spectral modelling may not be the best alternative for advanced coding schemes.

3.1.2 Interband Prediction

In order to use spectral coherence to directly decorrelate pixel values, a predictive function is required. To describe such a function, we shall use the same location naming conventions as in the last chapter (see Figure 2.2). X, W, N, etc. are assumed to be in the current band and X_r, W_r, N_r, etc. are pixel values at the same locations but in a reference band. All pixels up to and including the current, in the reference band, must be known by the decoder.

An interband extension of JPEG-LS, presented in [MWSM97], defines an interband predictor:

$$\hat{X} = \frac{W + (X_r - W_r) + N + (X_r - N_r)}{2} \tag{3.4}$$

This can be seen as using the average of the horizontal and vertical intensity gradients in the reference band, to model the same gradients in the current band. In [MWSM97] it is noted that despite its simplicity, the predictor defined by Equation 3.4 performs better than some more complex alternatives they considered.

In a more recent paper [WMC98] an interband version of CALIC is presented. This extension of CALIC utilises a more complicated interband predictor, which starts by defining two potential predictive estimates:

$$\begin{aligned} \hat{X}_h &= W + \alpha(X_r - W_r) \\ \hat{X}_v &= N + \alpha(X_r - N_r) \end{aligned} \tag{3.5}$$

where α is related to the correlation between the colour bands. In the presence of a strong edge, one of these two values is chosen, otherwise the values are combined. This is computed by:

```
if($(|X_r - N_r| - |X_r - W_r| > T)$          //sharp horizontal edge
    $\hat{X} = \hat{X}_h$
else if($(|X_r - N_r| - |X_r - W_r| < -T)$    //sharp vertical edge
    $\hat{X} = \hat{X}_v$                                              (3.6)
else
    $\hat{X} = (\hat{X}_v + \hat{X}_h)/2$
```

where T is a threshold whose value is not given. This use of local gradients in prediction is strongly reminiscent of GAP.

It is worth noting that under the simplifying assumptions that $\alpha = 1$ and given a degree of image smoothness, the predictor of interband CALIC is equivalent to the one used in interband JPEG-LS.

3.1.3 Interband Switching Methods

In both [MWSM97] and [WMC98] it was determined that selective use of interband prediction is superior to its unconditional use. Indeed, it was found that unconditional use of interband prediction can lead to a reduction in compression performance, as compared to the use of intraband prediction. Therefore, we shall now consider some methods to switch between inter- and intraband predictors.

In [MWSM97] two methods are compared for switching purposes. Firstly, the correlation coefficient between the current and reference bands, considering 10 pixel locations close to the current pixel, is calculated. If this correlation coefficient is greater than some threshold, the interband predictor is used, otherwise intraband prediction is performed.

The second method given in [MWSM97] requires the storage of the absolute prediction errors for both intra- and interband predictors, for the current and previous rows. The errors from W, N, NW and NE are summed and the predictor with the lowest total is used.

This was found to give roughly equivalent results to the correlation based switching scheme. The use of local error requires less computation but more storage than the correlation method.

In [WMC98] local correlation is again used to switch between prediction schemes. However, the local neighbourhood is reduced from 10 to 8 pixels for the correlation coefficient computation and the threshold value is given as 1/2.

Interband Methods for Improved Intraband Prediction

Interband correlations can be used purely for predictor switching. An example is given in [MS95b], in which the Previous Best Predictor (PBP) technique is introduced. PBP first uses JPEG7 to predict a reference band. For subsequent bands, a set of predictors is used on the current pixel location in the reference band. The predictor which gives the minimum predictor error in the reference band is used for the current band. In [MS95b], the LJPEG predictors (see Table 2.1) were used[Wal91]. It is noted that using other sets of predictors lead to no appreciable improvement in performance

Also in [MS95b] a method based on scan models is proposed, in a fashion similar to that mentioned for greyscale images (see Section 2.6.4). Firstly, JPEG7 was used to predict a reference band. Using this reference band an optimal scan path for simple DPCM is generated. This scan path is then used, with DPCM, to predict the current band. In [MS95b] the scan model approach was found to outperform PBP.

Both the schemes mentioned in [MS95b], PBP and the interband scan model, rely on the idea that what worked well for the reference band will work well for the current band. Although good compression savings are shown, both the methods make unconditional use of the interband information. This could lead to poor results in cases where the image bands have poor correlation.

3.1.4 Satellite Imagery

The compression of satellite images has often been cited as an application that requires lossless coding, although research employing lossy techniques for such compression has been published[AMH97, Abo95]. One example of a lossless study is in given in [RC96], which considers the compression of AVIRIS (Airborne Visible/Infrared Imaging Spectrometer) data. This data contains 224 bands and is hence called *hyperspectral*.

Although the aim of the work in [RC96] is very specific (that is, to compression AVIRIS data) the approach taken is quite general. Prediction is followed by adaptive error coding, in much the same way as previous schemes.

Five predictors were considered independently. These included predictors that were purely spatial (JPEG7), purely spectral and predictors that used both forms of correlation. Of the five linear predictors, two had constant weights applied to the previous pixels used in the prediction. The other three had the potential to optimise their weights. The optimal value of these weights was found via least-squares minimisation, on a row-by-row basis.

The best weights were then quantised and transmitted as overhead. Hence, this can be seen as forward adaptation.

The best performing predictor had the form:

$$\hat{X} = a + bW + cX_r + dW_r \tag{3.7}$$

As the weights can be optimised for each line, varying amounts of spatial and spectral correlation could be used as required. Hence, this use of adaptive weights in a single linear predictor, represents an alternative means of selection between spatial, spectral and hybrid prediction.

3.1.5 Band Ordering

The problem of band ordering can be seen as determining which band to use as a reference for which other band, when spectral prediction is to be used. Of the work referenced so far, static (pre-determined) band orderings have been used. For example, in [MS95b] the red band is used as a reference for the green band and the green band is then used as a reference for blue. Whereas, in [MWSM97], the green band is used as a reference for both the red and blue bands.

If we have B bands there are $B!$ possible band orders. The most obvious solution to the band order problem is to use all possible band orderings and chose the one that works best. The decoder will also need to be told which choice was made. This approach carries the cost of running the encoding algorithm $B!$ times. While this may be acceptable for RGB image ($3! = 6$) or CMYK images ($4! = 12$) some satellite images have over 200 bands (hyperspectral images) and so alternative schemes are required.

In [Tat94] the compressed size of every band is considered, allowing for every other band to be the reference band. The problem is then structured as a weighted graph problem for which efficient solutions exist. As there are roughly $O(B^2)$ band pairs, the method of [Tat94] reduces the band ordering problem from one that is $O(B!)$ to $O(B^2)$.

When decompressing an image stored in RGB or CMYK format it is likely that all colour bands will require decoding. However for hyperspectral imagery, if a single band is required, problems could arise from band ordering. For example it may be that the required band depends on the decompression of over a hundred previous bands. This is clearly undesirable. One solution, given in [Tat94], is to divide the image into partitions of a fixed number of bands. This problem turns out to be NP-hard if more than 2 bands are allowed in each partition. Hence, a scheme based on band pairing is suggested for when extraction of single bands is likely to be a major issue.

3.2 Moving Pictures

The idea of using information from previous frames in a video sequence was mentioned at least as far back as O'Neal's work in 1966[O'N66]. However, lacking statistics on interframe correlations, he did not study the idea in detail. Another barrier to the use of interframe methods at that time, was the cost of storing the previous video frame.

For interframe prediction, the obvious choice for a predictor is the pixel value at the same location as the current pixel, but from the previous frame. We can write this as $\hat{X} = X_{t-1}$, where the $t-1$ indicates a value from the previous frame in the time sequence. A problem with this predictor is that it fails in areas of motion. Any image region containing a moving object will result in poor prediction, because the predictor has assumed the current frame looks like the previous frame. Worse still, if there is global motion caused by camera movement (panning), $\hat{X} = X_{t-1}$ will be likely to fail in all regions of the image.

To solve this problem and make efficient use of interframe correlations, it is necessary to have a model for the motion in a scene. This idea underlies the topics of *motion estimation* and *compensation*.

3.2.1 Motion Estimation and Compensation

In order to make use of interframe correlations despite the presence of motion, said motion must be modelled. The process of modelling the motion is called *motion estimation*. Given the previous frame and the motion model, the decoder can compensate for the motion and produce a *motion compensated* interframe prediction.

Motion estimation was considered to be "one of the main recent developments in image coding" in a review published in 1985[MPG85]. A good tutorial coverage is also given in [Cla95]. Research in motion estimation took off in the mid-70s and is still the subject of many publications today, indicating that it is still an important problem without an ideal solution.

To model the motion for a given pixel or image region, motion estimation generally assumes knowledge of the relevant pixel values in the current frame. Therefore, motion estimation must normally be carried out by the encoder (hence this is a forward adaptive method). Motion estimation can be computationally very complex and therefore a simple model of motion is generally used. This model limits motion to 2D translational motion of solid objects. Other forms of object motion, for example rotation, translation along the camera axis and non-rigid movements, all result in reduced accuracy of the motion compensated prediction. Although no details will be given here, some recent work has considered affine motion models that overcome most of these limitations[Csi97, WSG99].

If motion is modelled only as 2D translation, then a full description of the motion between two frames is provided by having a *motion vector* for each pixel in the current frame. Each motion vector (a.k.a. displacement vector) gives a horizontal and vertical distance from the current location of an image feature, to its location in the previous frame. Note, it is important that the motion vectors point backwards in time. Otherwise, if they showed how each element in the previous frame had moved in the time up to the current frame, gaps would appear in the motion compensated frame.

Early work on motion estimation aimed to minimize the motion compensated residual for a given pixel by a recursive technique based on gradient descent. Although popular for a while, practical implementations of this method seem to have been limited by the computing power of the late-70's. So when a simpler method named *block based motion estimation*, was introduced by Jain and Jain in 1981[JJ81], it soon became the mainstream approach for tackling motion estimation.

Block Based Motion Estimation

In [JJ81] the current frame is divided into rectangular blocks of $M \times N$ pixels. Each block is then matched against blocks from the previous frame and a distortion measure is calculated. The blocks from the previous frame used for matching, correspond to the location of the current block in the current frame, displaced by i pixels horizontally and j pixels vertically. The shift that gives the minimum distortion, $\min_{i,j} D(i,j)$, is selected as the motion vector for the block.

Jain and Jain used mean square error (MSE) as their distortion metric. Hence, using a $M \times N$ block size and a motion vector range of $(\pm k, \pm l)$, the distortion $D(i,j)$ is given by:

$$D(i,j) = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} (I_t(m,n) - I_{t-1}(m+i, n+j))^2 \qquad -k < i < k, -l < j < l \quad (3.8)$$

where $I_t(x,y)$ and $I_{t-1}(x,y)$ range over the current and previous frames respectively, relative to the position of the current block. The squaring operation is often replaced by an absolute operation, to give mean absolute difference (MAD) as a distortion metric. By keeping the shift (i,j) bounded, the search is limited to a rectangular region of the previous frame. Bounding the search limits computational requirements, but it forces an assumption to be made about the maximum likely motion of an object in a scene.

Block based matching is clearly an optimisation technique. As it has been described thus far, the method constitutes a full search approach. Many alternative searching strategies have been suggested to lessen the searching effort[JJ81, LL97]. We will consider two such schemes here.

In [JJ81], the logarithmic search is introduced. It is based on the assumption that the distortion $D(i,j)$ increases monotonically as the shift (i,j) moves away from the true motion vector. The search is iterative and starts with the $(0,0)$ shift as the best estimate of the true motion vector. At each iteration in the search, five possible shifts are considered; the current best estimate of the true motion vector and four locations around it as shown in Figure 3.1. If the minimum $D(i,j)$ comes from the current best estimate, or from a location at the edge of the search range, the step size is reduced. Otherwise, the shift giving the minimum $D(i,j)$ is chosen as the best estimate for the next iteration. The algorithm terminates when the best estimate for the motion vector is found to give the least $D(i,j)$ and the step size has reached a minimum threshold.

There are many other fast motion estimation algorithms given in the literature[LF96, OA97]. On example is the three-step search. It is very similar in principle to the logarithmic

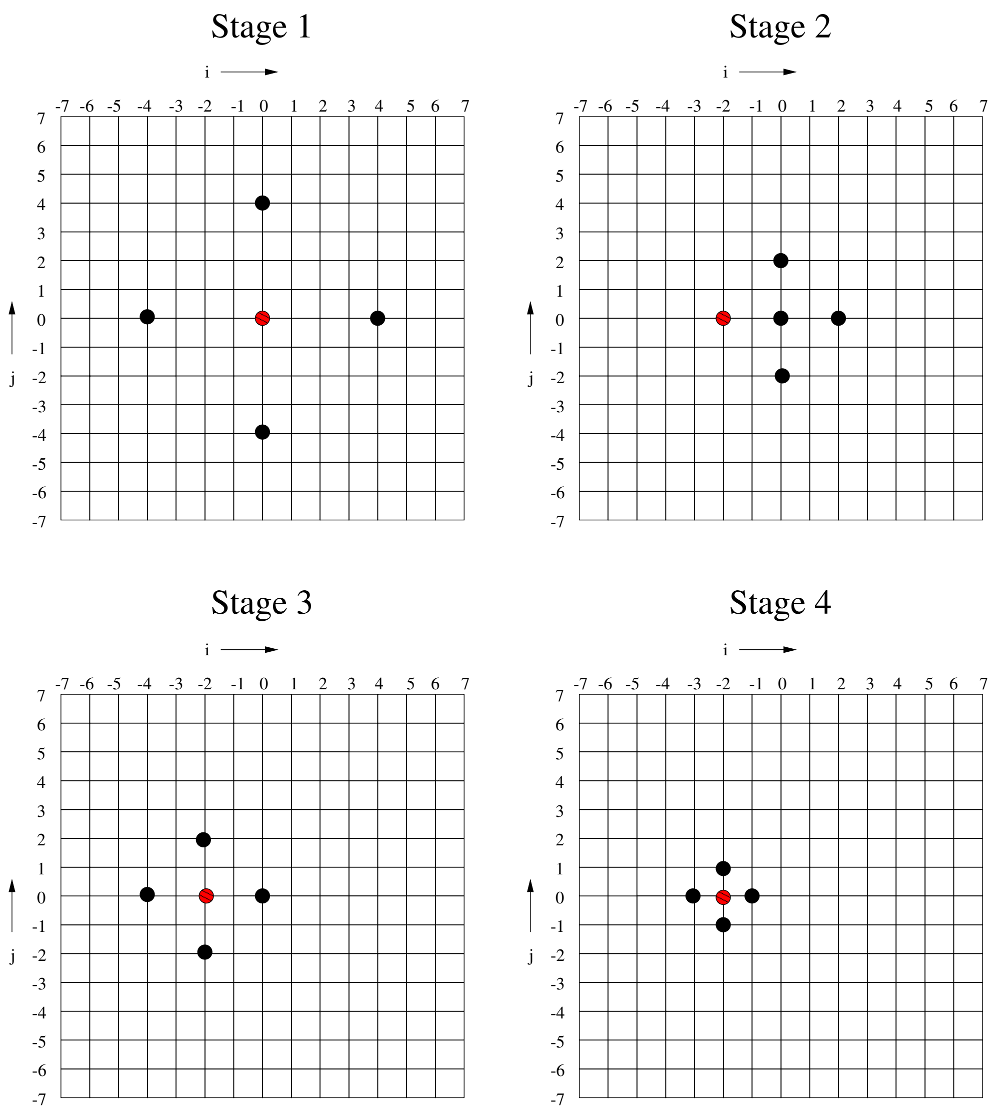


Figure 3.1: An example of the logarithmic searching strategy for block based motion compensation. An initial step size of 4 is used, along with a minimum step size of 1. The displacement that yields the lowest distortion at each stage is shown in red. The algorithm terminates on the fourth stage because the step size has reached the minimum and the current best estimate has the lowest associated distortion. The final motion vector is (-2,0).

search. The differences are that eight neighbours, rather than four, are used around the best estimate and that the step size is reduced at each iteration regardless of where the minimum $D(i,j)$ was found.

Although these heuristic searching methods reduce the computational burden of motion estimation, they do not guarantee to find the optimum motion vector [1]. Given modern computation power and the potential for parallelising the search with suitable hardware, full search is often the preferred optimisation strategy.

Another speed up that is frequently used is to perform a simple *zero motion* test. The distortion $D(0,0)$ is calculated and if it is found to be below some threshold, it is assumed that there is no motion in the block. However, this may cause problems if there is very gradual motion in a scene.

In theory there is no reason why the shifts (i,j) have to be integer shifts. Greater accuracy is obtained by allowing non-integer shifts, although interpolation of the reference frame is needed. A compromise that is sometimes used is to allow half pixel resolution, thus limiting the set of cases that an interpolation algorithm must deal with.

Block based motion estimation is often criticised as being “brute force and ignorance”. However, an alternative view is that it is founded on sound principles, but sacrifices some intellectual niceties for performance. It is often noted that objects in motion tend to have uniform motion over their extent. E.g. All pixels in a car traversing the scene will have the same motion vector. Therefore, it makes sense to estimate motion of regions of the image. Dividing the image into rectangular blocks is a crude segmentation, but allows simple implementation. Recent work on region based motion estimation, employing sophisticated segmentation techniques, will not be covered here but can be found in [BM97, Csi97, DD99].

A more problematic limitation of block based matching is that the motion vector found is bound by the limits of the search and only represents the optimum vector according to the distortion metric; that is, it may not represent the true motion of the block. A classic example of this last point is a large moving object with a uniform interior. Because the interior of the object is uniform, block based matching will not detect motion there. Only the motion at the edges will be found. These factors can be alleviated by hierarchical methods.

Hierarchical Motion Estimation

By starting with a large block size and a coarse motion vector resolution, hierarchical block matching for motion estimation can model large scale motion[Bie88]. This is achieved without the massive burden of using a large search space at a fixed resolution. The block size and motion vector resolution are progressively reduced until 1 pixel (or half pixel) resolution motion estimates are available for blocks of a small size. At each stage, the motion estimates from the previous level are used as a starting point for further estimation.

A more sophisticated technique that uses a multilevel image pyramid, is presented

[1]Note, even the motion vector found by full search is only optimum with respect to the distortion metric used.

in [WC90]. A three level pyramid is constructed, where the bottom level is the original image. The bottom level is split into 4 by 4 pixel blocks. Each pixel in the next level up represents the average of the corresponding block in the bottom level. The top level is constructed from the middle level in the same fashion as the middle level is constructed from the bottom level.

The current and reference frames are each converted to this pyramid representation. Initial estimates for the motion vectors at the top level are found by direct matching of elements. The matching criteria is the match with the least absolute difference. A range of ±1 is used for both horizontal and vertical components. This range corresponds to a range of ±16 in the original image.

For the middle level, the motion estimates from the top level are used as starting estimates. These estimates are again refined by direct matching, but this time the range is ±2. This process is repeated for the lowest level, giving motion vectors for every pixel.

The results presented in [WC90] show that the entropy of the motion compensated residuals generated with the pyramid based motion estimates were far lower than those using a pel-recursive motion estimation scheme.

However, as the authors note, it is impractical to transmit such a dense motion field to the decoder. They present a practical video coder based on their pyramid representation. The differences between the top levels of the current and previous frame pyramids are sent. This allows the decoder to perform top level motion estimation. Using the motion vectors calculated, the decoder can perform motion compensation to predict the middle pyramid level. This allows only the motion compensated residuals for the middle level to be sent. Motion estimation is carried out based on the middle level and those motion estimates are used to motion compensate the lowest level. One more set of motion compensated residuals is then sent. Note, motion vectors need never be sent to the decoder.

Although the pyramid based scheme seems useful for motion estimation, the test results for the compression scheme mentioned above showed almost no improvement over a hierarchical block matching based scheme.

Transmission of Motion Vectors

As motion estimation is generally a forward adaptive technique, the encoder must transmit the motion vectors found by motion estimation to the decoder. This leads to various compromises. It might seem ideal to have an accurate motion vector available for every pixel. However, the data requirement for the motion vectors would easily outweigh the benefits of the accurate motion vectors. Thus choosing a motion field density (block size) is important. Also, the accuracy of motion vectors must be taken into account. For example, half-pixel accuracy motion vectors allow better motion compensation, but enlarge storage requirements for the motion vectors.

Once block size and motion vector accuracy have been decided, the raw data rate for the motion vectors is easily calculated. However, we expect accurate motion estimates to show some correlation over the frame. Consequentially, differential (or predictive) coding of the horizontal and vertical components of the motion vectors seems appropriate. However, the literature seems divided on the issue. A study in [CP89], reported in [Cla95], suggests that two-dimensional coding of single motion vectors is better than differential coding. However, more recent work suggests an alternative conclusion. For example, in [HP99] a multi-scale motion field is efficiently encoded via prediction and context based modelling.

3.2.2 Practical Video Coding Schemes

Although all major video compression standards are lossy, it is useful to observe how they operate. We shall briefly consider the Moving Picture Experts Group (MPEG) standard[MPFD96].

MPEG has three primary types of frame, Independent, Predicted and Bi-directionally predicted frames. The basic features of these frame types are:

- I frames: Intraframe coding is used for all blocks. Lossy coding, via quanitsed DCT, is used.
- P frames: Frame is predicted by motion compensation, using the previous I or P frame as a reference.
- B frames: Frame is predicted by motion compensation, using both the previous and following I or P frames as reference. B frames are never used as a reference for motion compensation.

Motion estimation and compensation are performed on *macroblocks* of 16 by 16 pixels (of luminance information). As motion compensation can sometimes be ineffective, for example if an object moves into the scene, or some background is exposed by a moving object, intra coding may be preferable to interframe coding. Thus, each macroblock of an P or B frame can be intra or interframe coded. The decision of which to use, is made by the encoder and passed on as part of a macroblock header.

The allowed range for the motion vectors and their accuracy (whole or half pixel) is encoded into a per frame header for P and B frames. Motion vectors are coded differentially using the vector from the previous macroblock if available.

An MPEG sequence is composed of one or more *group of pictures* (GOPs). A typical GOP (in display order) would be: I-B-B-B-P-B-B-B-P-B-B-B-P. However, the B frames require a P frame that is forward in time, hence the order in which the frames are coded must be: I-P-B-B-B-P-B-B-B-P-B-B-B. Clearly, the difference in coding and display order requires an MPEG decoder (and encoder) to be able to buffer several frames of video.

The use of I frames at regular intervals in a coded video might seem inefficient, as these frames do not use any interframe correlations. However, using regular I frames provides multiple access points into the video stream. This removes the requirement that all previous frames be decoded to access a given frame[2]. Fast forward and rewind capabilities are thus enabled by the use of regular I frames.

3.2.3 Lossless Video Coding

There are surprisingly few papers in the literature regarding lossless video coding. One in particular that does approach the subject is [MS96], which considers the compression of colour video. This makes the paper particularly interesting as it covers spatial, spectral and temporal correlations and thus is the most *extended* of the lossless compression schemes encountered so far.

[MS96] first compares intraband, interband and interframe prediction schemes independently. They introduce 3D linear predictors, which are generated from the original JPEG set of predictors.

They also apply the Previous Best Predictor (PBP) technique, introduced by the same authors in [MS95b], to video data. This means that the JPEG set of predictors are used on the current pixel location in the previous video frame. The predictor with the least prediction error is used for the current pixel in the current frame. They call this PBPTEMP as they also use the PBP technique for spectral decorrelation (PBPSPEC).

Memon and Sayood note that all the temporal predictors given above fail to work well in regions of high motion, hence they also incorporate motion compensated prediction (using simple block matching estimation) into their study.

The first major result given is that when spectral prediction is used independently, it is by far better than either spatial or temporal prediction, even for scenes of low motion. In fact from one set of results[3] it could be concluded that spatial prediction was better than simple temporal prediction (i.e. excluding motion compensation). If other researchers encountered similar results at an early stage, it may account for the low number of papers on the subject of lossless video compression.

To get better performance, a hybrid scheme is used. Motion compensation is used in tandem with PBPSPEC, a single bit being set to indicate which was used. JPEG7 is used in the case that neither of the aforementioned predictors is applicable. The hybrid scheme is found to have the prediction errors with the lowest entropy. The difference between the hybrid scheme and the best of the non-switching schemes was typically 1bpp.

After determining the prediction scheme, error modelling was considered. The methods used can be seen as a simple context model with eight contexts. A variety of context determination algorithms were considered. The one that performed best, averaged prediction errors from a $k \times k$ block in the reference colour band and then thresholded the result. Memon and Sayood note that this method has again used spectral correlations to great effect and that of all the methods in the paper, they feel the greatest room for improvement lies with the error modelling step.

[2]The situation for MPEG is actually a little different, as a GOP is not necessarily independent of all previous GOPs.

[3]Note, results are only given for two video sequences.

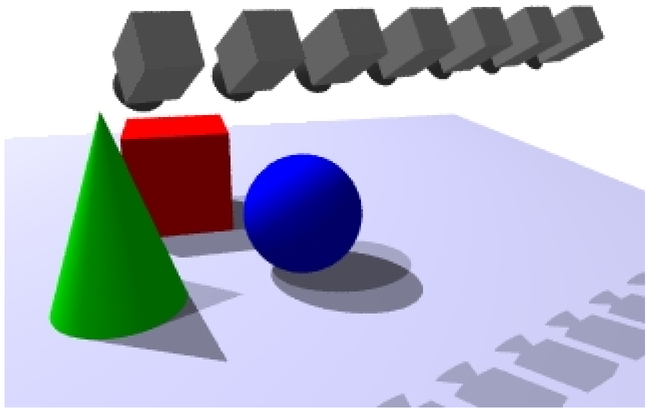


Figure 3.2: Camera setup for a multiview system.

3.3 Multiview Imagery

So far we have considered imagery that contains spatial, spectral and temporal variation. This covers nearly the full range of human visual experience. The only element missing is stereo; the use of multiple images to add a feeling of depth to a displayed scene. Although only two images are necessary to produce a stereo effect, more images are sometimes used to allow greater flexibility, e.g. to allow multiple viewers to use a single display.

We will use the term *multiview sequence* for a set of images that are all views of the same still scene, but taken from slightly displaced viewpoints. Such a sequence would be collected by the cameras in Figure 3.2. When considering a time sequence, the term *multiview video* will be used. If the number of views, which will generally be arbitrary, is just two, the terms *stereo image* and *stereo video* will be used instead of multiview sequence and multiview video respectively.

A multiview sequence can be thought of as a time sequence, in which the camera has moved, with constant velocity, in front of a still scene. However, a multiview video clearly has an extra dimension as compared to standard video. Although multiview imagery has similarity to video, it has an important difference. Video aims to present small movements to fool the eye into seeing smooth motion over several frames. Whereas stereoscopic imagery presents a disparity large enough to enable the viewer to perceive depth.

The movement of an object due to a change in viewing position, is called *disparity* and is related to the distance from the viewpoints to the object. Disparity estimation and compensation are the analogues of motion estimation and compensation. Approaches to disparity estimation are given in Section 3.3.2. In order to get the most out of multiview correlations, we must consider the structure of multiview imagery. This is best achieved by considering a combination of capture devices and display devices.

3.3.1 Stereoscopic Display Methods and Camera Geometry

A 3D display system presents each of the viewer's two eyes with a slightly different image. If these images represent two suitably displaced views of a scene, the observer should be able to *fuse* the image pair and perceive depth within the scene. This use of separate views for each eye provides what is called the *binocular* depth cue. Even though a single image can contain many depth cues (occlusion, perspective, etc.), the binocular depth cue is known to be one of the stronger cues and can help disambiguate confusing depth information.

There are many approaches to displaying stereo imagery. A classic scheme uses spectral multiplexing of a single display surface. That is, one image is shown in red and the other in green. A viewer with red-green filter glasses then receives a different monochrome image in each eye. Other solutions use temporal multiplexing[MDTL96] of a single display or use multiple displays to present multiple images.

When considering multiview imagery, it is worth thinking about the image source; be it capture by camera or generation by computer graphics.

Given a linear set of laterally displaced cameras, there are two principal geometries available; converging or parallel camera axes. Many authors [$M^{+}93$, DML97] have noted the potential problems of the converging axes case:

- A display method utilising a single display can show distortions that must either be corrected or suffered by the viewer.

- The disparity of a scene feature, between adjacent views, may have a slight vertical component. In contrast, disparity is purely horizontal in the parallel axes case.

This suggests that the parallel axes case is generally more appropriate. However, this is not always realised; for example all of the widely used DISTIMA test imagery was captured with a converging camera pair. The confusion surrounding this issue is evident in two references from the DISTIMA group [Fra96, TGS97]. In [Fra96] the vertical component of disparity is said to be less than half a pixel and therefore negligible; however, in [TGS97] the vertical component is assumed to be in the range ± 2.

Even when the parallel axes setup is used, there is the question of whether to use on-axis or off-axis projection. All the test imagery used by Harashima's group [FH94a, FH94b, NKH96b, NH94, NKH96a] appears to use on-axis projection, however off-axis projection is required for display systems that share a single display screen.

It seems likely that until a single display technology dominates the field, much of this confusion will remain. In particular, the confusion will manifest itself as assumptions and assertions that only hold if a particular camera geometry is used. However, with everyone using a different geometry, references must be very carefully read.

3.3.2 Disparity Estimation and Compensation

Disparity estimation has received a good deal of study [Fra96, RH97, MH96]. In particular Franich's thesis [Fra96] presents much interesting material. He gives a good review of block based disparity estimation techniques. This is concluded by saying that disparity maps generated by such techniques are not sufficiently accurate, due to incorrect (spurious) matches, for use with advanced techniques such as view interpolation[4].

A genetic algorithm is then proposed for disparity estimation. A clever hybridisation of the usual reproduction, cross-over and mutation operations is employed to aid performance. The resultant disparity maps are shown to be superior to those generated by block matching.

The *disparity space image* is also introduced in [Fra96], as a representation of the correlations between horizontal lines from multiple viewpoints, with identical vertical positioning. Disparity and occlusion information is relatively easily visualised by and extracted from disparity space images. A number of algorithms are presented for the extraction of said information. Of these, the best disparity maps come from an algorithm that again uses genetic algorithm techniques.

3.3.3 Two View Stereo

Two view stereo compression has been more widely researched than the arbitrary view case. Most of the methods are lossy and incorporate some form of dispartiy compensation [SSMJ97, WO97, ÖS93].

Perkins' Thesis

Of the relatively small body of work published on the compression of stereo pairs, one of the most fundamental references is due to Perkins [Per92]. He introduces the CONCOD (CONditional CODer) structure to describe a scheme that first codes one image of the stereo pair and then codes the other based on the first. An information theoretic argument is then given, which shows that although the CONCOD structure allows optimal coding in the lossless case, it is inherently sub-optimal when a degree of distortion is allowed. Perkins follows this statement with the concession that although mostly sub-optimal, compression schemes based on the CONCOD structure are of practical importance; largely due to their relatively simple implementations.

Perkins presents two techniques for the compression of stereo pairs. The first uses a simple block matching technique to estimate disparities between the two images. One of the images is then compressed, by quantisation of transform coefficients following a DCT, and stored together with the disparity estimates. The other image is predicted, in the transform domain, from the first using the disparity map (i.e. it is disparity compensated) and only the errors are coded and stored.

[4] A technique that uses available images to construct views from intermediate viewpoints, that were not available in the original data.

The second technique involves mixed-resolution coding and makes use of a curious feature of the human visual system. It is stated that if one image from a stereo pair is displayed at low resolution, the brain can still fuse the pair. Furthermore, the observer will perceive the result as being closer to the high resolution image than the low resolution partner. This motivates a scheme that derives substantial compression by simply sub-sampling one image from the stereo pair, although this is obviously not applicable for lossless coding.

Using the Worldline

A paper by Siegel et al. [SGSJ94] gives a fairly general introduction to the compression of stereo pairs and thus covers much the same ground as Perkins' work. However, they do introduce the *Worldline*[5] correlation. This considers a moving object, in time displaced left and right images. The concept being, if the horizontal motion of the object suitably compensates perspective, then the view of the object in one image will be very similar to the view of the object in the other image at a future time. Unfortunately, they give no practical method for utilising this interesting correlation.

3.3.4 Multiview Stereo Compression

A major study of multiview sequence compression has been undertaken at the University of Tokyo, under the direction of Harashima [FH94a, FH94b, NKH96b, NH94, NKH96a]. In [FH94a, FH94b] a method is presented that reduces a multiview sequence to a structure and a texture map. The mesh structure is generated by placing vertices at points of minimum variance in a so called *normalised object space*. Coordinates are normalised using geometric equations, although disparity estimation is required for the depth component.

With this scheme, decompression is a simple rendering operation. Furthermore, intermediate views can easily be generated by rendering the data from a point between two original viewpoints. However, as their own results show, the nature of the triangular mesh used to represent scene structure is unable to capture complex occlusions.

Later papers [NKH96b, NH94] present a similar, but more sophisticated, method that overcomes some earlier problems. Firstly, a multiview sequence is subjected to a 3D segmentation. This is performed by giving an attribute vector to all image points and then partitioning the attribute space using the K-means algorithm[6]. Once again, the image data is represented by a mesh structure and texture data. However, to deal with occlusions the surfaces determined by the segmentation are grouped into layers, which are mesh segments at particular depths.

Although the work presented in [NKH96b, NH94] is very interesting, some unfortunate details remain. Not least the fact that the grouping of segmented regions is done by hand. Also, as in [FH94a, FH94b] a reduction in data rate is achieved purely by means of an

[5]A term that they admit to having stolen from General Relativity.

[6]A common optimisation technique from the pattern matching and machine learning literature.

alternative representation; no complete coding system appears to be developed. The work carried out by Harashima's group seems to have been done in anticipation of its usefulness, but without access to a multiview autostereoscopic display. This is evidenced by a complete lack of human observer test results to evaluate their lossy methods.

3.4 Summary

In this chapter we have seen that predictive coding is still the preferred method for lossless compression of colour images, videos and multiview sequences. Although interband predictors resemble the intraband predictors of the previous chapter, the most common means of utilising interframe and interview correlation are very different. Motion and disparity compensation are forward predictive schemes, requiring more work for the encoder and the transmission of extra information (motion vectors and disparity estimates).

Observing the publication dates of many of the papers referenced, indicates a recent surge in interest for the lossless compression of colour images. However, work on video and multiview sequences is less common. Indeed, while papers exist that consider lossless compression for colour video[MS96] and lossy compression for stereo video[SGSJ94], none are apparent for lossless multiview colour video compression.

Chapter 4

Approaches to Predictor Selection

As predictor selection mechanisms have been used to good effect, in previous compression schemes (see Chapters 2 and 3) for both greyscale and more advanced image types, it makes good sense to further consider their use here. In Section 4.1 the use of predictor selection, for extended lossless image coding, is validated and a commentary on the design of predictor selection schemes is given Section 4.2.

In Section 4.3 some novel selection schemes using backward adaptation are detailed and compared to existing methods. Section 4.4 presents the same discussion for forward adaptive methods. Hybridised methods are presented in Section 4.5 and are followed by the chapter summary in Section 4.6.

This chapter contains results produced by the author using purpose built software. To have relevance, the tests were carried out using a set of imagery which is used throughout this dissertation and includes images used elsewhere in the field. A commentary on the test imagery used is given in Appendix B.

4.1 Predictor Selection as a Solution for Extended Coding

The two previous chapters have shown a numbers of ways for predicting the value of the current pixel based on spatial, spectral, temporal and interview relationships. For most pixels two or more of these predictive options will be available and the problem becomes deciding which predictor to use.

Although the literature makes it quite clear that predictor selection is beneficial for both greyscale and colour image compression, the case is not well made for video and multiview imagery. Thus it is prudent to test this basic assumption early on and to determine whether predictor selection has some benefit over other more obvious ways of proceeding.

A fixed order of preference might naïvely be assigned to the predictors, using the most advanced type of correlation available. For example, we might chose to use interview correlations (disparity estimated prediction), dropping back to temporal (motion compensated prediction) when neighbouring views are not present. When temporal prediction is unavailable (i.e. an I frame is being coded) spectral prediction would be used. Only for the reference band in an I frame would spatial prediction be needed. The result of implementing such a scheme, let us call it *Naïve1*, will be considered shortly.

Given the results of [MS95b] as discussed in Section 3.2.3, a less naïve scheme would use spectral prediction in preference to temporal prediction. Furthermore, as we expect there to be more disparity between two views than there is motion between two successive frames, the use of temporal prediction in preference to interview methods seems preferable. This new wisdom leads to a scheme that shall be referred to as *Naïve2*.

To allow the required comparison, the two naïve schemes above are compared to a very simple selective scheme. Block Based Selection (BBS) considers the sum of the magnitude of errors from each applicable predictor, over an 8 by 8 block. The predictor with the minimum absolute error is chosen for the block. The uncoded overhead of $\lceil log_2 n \rceil$ bits, where n is the number of possible predictors, is included in the results shown in Table 4.1. Note, further details regarding BBS are given in Section 4.4.

The two naïve schemes and BBS all have access to a small set of predictors. The actual predictors used and the reasons for choosing them are described in Section 4.2. The results for the three schemes are shown in Table 4.1. The results given show the entropy of the prediction residuals, averaged over all bands, frames and views present in the imagery. This means that these results should be considered against a raw data size of 8 bpp. This is so even for the colour imagery that, having red, green and blue bands, might otherwise be considered as 24 bpp in raw data terms.

The results in Table 4.1 clearly show that even the naïve use of extended prediction gives superior average compression, compared to the use of only spatial prediction. The fact that Naïve2 generally outperformed Naïve1, supports the comments made in [MS96] that spectral correlation is stronger than temporal. However, there are exceptions to this generalisation. For example the colour *granny* video, which is computer generated and has particularly high temporal correlation, performs better with Naïve1 than Naïve2.

The result most relevant to the current discussion however, is that BBS, despite its simplicity, yields a lower average entropy of prediction residuals than either of the naïve schemes. This is a clear indication that adaptive predictor selection is preferable to a fixed assignment of predictors.

However, there are examples where BBS is slightly bettered by a naïve scheme (the *cats* image and the *ctflesh* sequence). This implies that the slight overhead involved with BBS may lead to inefficiency in some cases and still better predictor selection mechanisms should be sought.

4.2 A Design Philosophy for Predictor Selection

Having shown that even simple predictor selection is worthwhile, it is now appropriate to consider one of the most important elements of a prediction selection scheme; that is, the

Image	Raw	MED	Naïve1	Naïve2	BBS
Colour Images					
air2	6.05	4.44	4.49	4.49	4.30
baboon	7.64	6.43	6.14	6.14	6.09
cats	4.59	3.54	2.59	2.59	2.60
cmpnd2	2.90	2.07	1.63	1.63	1.58
house	6.41	4.45	4.45	4.45	4.28
lena	7.27	4.80	4.58	4.58	4.56
Average	**5.81**	**4.29**	**3.98**	**3.98**	**3.90**
Greyscale Video					
claire	6.22	2.73	2.28	2.28	2.28
granny	7.20	3.53	2.64	2.64	2.40
mall	7.15	3.67	5.50	5.50	3.68
mobile	7.12	4.95	4.42	4.42	4.33
salesman	6.81	4.68	3.94	3.94	3.88
Average	**6.90**	**3.91**	**3.76**	**3.76**	**3.31**
Colour Video					
claire	6.10	2.86	2.73	2.71	2.49
football	7.12	5.05	5.58	4.88	4.52
granny	7.27	3.52	2.57	2.98	2.18
mobile	7.02	4.98	4.47	4.20	4.06
susie	6.82	4.02	4.14	3.46	3.32
Average	**6.87**	**4.09**	**3.90**	**3.65**	**3.31**
Multiview					
ctflesh	2.67	1.55	0.56	1.55	0.58
granny	7.29	3.55	3.35	3.31	2.74
skull	6.08	3.63	3.90	2.83	2.82
Average	**5.35**	**2.91**	**2.60**	**2.56**	**2.05**
Colour Multiview Video					
granny	7.26	3.53	3.27	2.98	2.08

Table 4.1: A comparison of the entropies of prediction residuals, following fixed and adaptive predictor usage.

Image	MED	PP	IF1	MC	MED + PP	MED + IF1	MED + MC
Greyscale Video							
claire	2.73	2.43	2.48	2.46	2.34	2.40	2.28
granny	3.53	2.88	2.83	2.79	2.46	2.65	2.39
mall	3.67	5.79	4.21	5.55	3.68	3.68	3.67
mobile	4.95	5.38	5.30	4.53	4.91	4.95	4.33
salesman	4.68	4.15	4.59	4.08	4.05	4.36	3.89
Average	**3.91**	**4.13**	**3.88**	**3.88**	**3.49**	**3.61**	**3.31**

Table 4.2: A comparison of the entropies of prediction residuals, following various predictor usage.

set of predictors that can be selected from. It is important not to have too many predictors. In a backward adaptive setting, too many predictors would complicate heuristics and/or greatly increase the number of counts that need to be kept. If forward adaptation is used, too many predictors would lead to increased overhead for signalling the predictor selection made by the encoder.

Given that we wish to keep the number of predictors under consideration low, it makes sense to determine the best predictors available and use them with selection schemes. However, this procedure has its pitfalls. As an example of this, we shall consider the case of greyscale video. Three interframe predictors are compared: the Previous Pixel (PP) $\hat{X} = X_{t-1}$, a predictor we shall call InterFrame 1 (IF1):

$$\hat{X} = \frac{W + (X_{t-1} - W_{t-1}) + N + (X_{t-1} - N_{t-1})}{2} \tag{4.1}$$

and Motion Compensated prediction (MC).

IF1 can be seen as taking the closest horizontal and vertical gradients in the previous frame and applying them to the current frame[1]. This models static regions and allows for small scale changes in lighting. Also, due to the averaging effects of the predictor, we expect it to perform better than PP in regions of very slight motion.

Table 4.2 shows the results of applying each of these three predictors to greyscale video. MED is also used as a reference. Finally, each of the three interframe predictors are used in conjunction with MED, using BBS for switching. Note, the first frame of each sequence is decorrelated with MED only.

The results show that IF1 performs better than PP. This would imply that IF1 would be preferable to PP in a prediction selection scheme. Contrary to expectation however, when used in conjunction with spatial prediction, the combination of MED+PP is superior to MED+IF1. This can be explained by considering the strong points of each predictor. PP does well in areas unchanged from frame to frame. Whereas in areas of motion, PP

[1]IF1 is actually the same as the interband predictor used in [MWSM97], but with the reference imagery being the previous frame, rather than a reference colour band.

will perform poorly and MED will be chosen instead. IF1 (due to its inherent averaging) will do less well than PP in unchanged areas and because it does not model motion, it will perform less well than MED in areas of motion. Thus, despite the better individual performance of IF1, the combination of MED+PP is superior to MED+IF1.

This result implies that it is better to have predictors that model one type of image region well, rather than a predictor with good average performance over all regions. This forces us to re-evaluate many of the predictors in the literature, as these have often been constructed to provide good general performance rather than to model one type of region well.

For example, when considering spatial prediction we might think of JPEG7, which has been reckoned to be the best of the JPEG set[MS95b]. The averaging property leads to good overall performance, but makes the predictor less suitable for use in a predictor selection scheme.

A second interesting result from Table 4.2 is given by considering the results of using MED+PP and MC. As can be seen, the use of two simple predictors and a simple selection scheme gives prediction performance that is superior to the computationally more expensive motion compensation (although MED+MC provides still better prediction). This implies that extended lossless image compression holds the potential to yield either much better compression, or comparable levels of compression with lower computation requirements.

4.2.1 The Final Predictor Set

To help compare predictor selection schemes, it was decided to chose a fixed set of predictors and use this set for all the following tests. This helps indicate which predictor selection scheme is best and removes the need to jointly optimise predictor selection alongside the predictor set used.

For spatial prediction, MED was chosen. Although MED has good general performance, the predictors it uses are all very simple. W, N and $W + N - NW$ are good models for horizontal, vertical and plane-like smoothness respectively.

When using spectral prediction, the predictor introduced in [MWSM97], as given in Equation 3.4 was used. From here on, this predictor shall be referred to as InterBand 1 (IB1).

For temporal and interview prediction, motion and disparity compensated prediction were used respectively. As can be inferred from Chapter 3, these schemes are still the subject of active research. It is not the intention here to further the art in motion or disparity estimation and thus fairly standard estimation/compensation procedures were used. For motion estimation, full search block matching with an 8×8 block was used. Half-pixel resolution was employed and the resultant motion vectors were in the range $(\pm 32, \pm 32)$. For disparity estimation, the same basic approach was used, but considering only horizontal shifts. As disparity compensation carries less overhead per block than motion compensation, a smaller block size (6×6) was found to be acceptable.

It is also worth considering use of the previous pixel (PP) as a predictor for temporal sequences. Although simple, it does model static scenes very well and carries none of the overhead of motion compensation.

In the subsequent tests, all available predictors were used. All the predictor selection schemes that follow employ an order of precedence, that will become important when predictors give equally good predictions. This order is: MED, IB1, PP, MC, DC. One consequence of this is that PP will be selected over MC in areas of zero motion in a video sequence.

4.2.2 A Standard Scan Order

Just as in standard predictive coding, the order in which pixels are processed is an issue for predictor selection schemes. However, in the extended scope now being considered, a one dimensional path must be found through the spatial, spectral, temporal and interview dimensions of the input imagery. This clearly leads to a multitude of possible scan orderings.

A simple solution to this problem is a logical extension of the raster-scan ordering mentioned in Chapter 1. Each band in a colour image is processed from the top left corner, moving left to right, top to bottom, until the bottom right corner is reached. For a colour image, the bands are processed in the order red, green and blue; each band being entirely processed before the next is started. For temporal sequences every frame is processed from the first to the last and for multiview sequences every view is processed from the left to the right. For multiview video sequences, all views at a given time step must be processed before moving to the next time step, otherwise interview correlation would be lost.

Such a simple solution is unlikely to provide an optimal ordering, however it has many advantages. It provides previous values for MED based predictions and guarantees that the entire previous band, frame or view is available as a reference for the relevant prediction operation. Also the scheme is general to all imagery types, without requiring any special cases.

Given these advantages, the simple scan ordering mentioned above was used for all the tests in this work. This has the effect of removing another variable from consideration and thus helping the following tests focus on the nature of various predictor selective mechanisms.

4.3 Past Evidence

Predictive coding is based on the ideal of using information, received up to a given point, to model future pixels and hence reduce the information required to reconstruct future portions of the image. The application of this ideal to predictor selection yields backward adaptive methods and is discussed below.

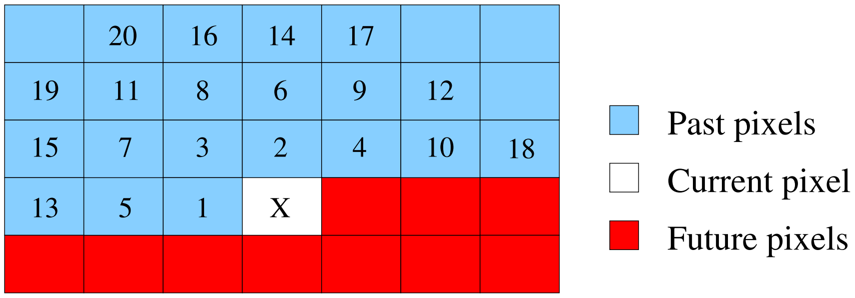


Figure 4.1: The pixel locations used for various regions with local error based predictor switching. A region of size n comprises all pixels that are numbered $\leq n$. For example, the region of size three, uses W, N and NW.

4.3.1 Local Predictor Error

A simple backward adaptive method that was shown to work well in [MWSM97], makes use of the absolute prediction error in the locale of the current pixel. In [MWSM97] the prediction error in the region $\{W, N, NW, NE\}$ was considered for both MED and IB1. The predictor with the minimum absolute prediction error in the region was chosen for the current pixel. It was shown that this technique slightly outperformed a more computationally expensive technique based on correlation. This latter method used the correlation between the current and reference colour bands as the switching criteria.

Given its simplicity and good performance, it seems useful to further explore this technique for predictor switching. We will thus consider the extension of local error based switching to video and multiview imagery.

One issue that remains open is the best size for the region over which to sum the prediction errors. It is noted in [MWSM97] that a region larger than four showed no appreciable benefit. Whether this will remain true when the technique is extended beyond colour images is not clear.

To determine the best region size, a Local Error Prediction Selection (LEPS) scheme was implemented with varying sizes of error accumulating region. Each region is built incrementally, by adding one pixel onto the last. Figure 4.1 shows the pixels used; the numbers show the size of the region in which the pixel is first used.

The average reduction in the entropy of prediction residuals, for each category of image data, is shown in Figure 4.2. The results for colour images, which benefit the least from increasing the region size, show no significant benefit in a region size greater than four. This is in agreement with the findings of [MWSM97].

The other types of imagery do show the benefits of larger region sizes. The fact that more complex imagery gains a greater benefit as the region size increases can be explained by considering the number of predictors that are under consideration. For example, colour images can only make use of two predictors and an information gathering region of four

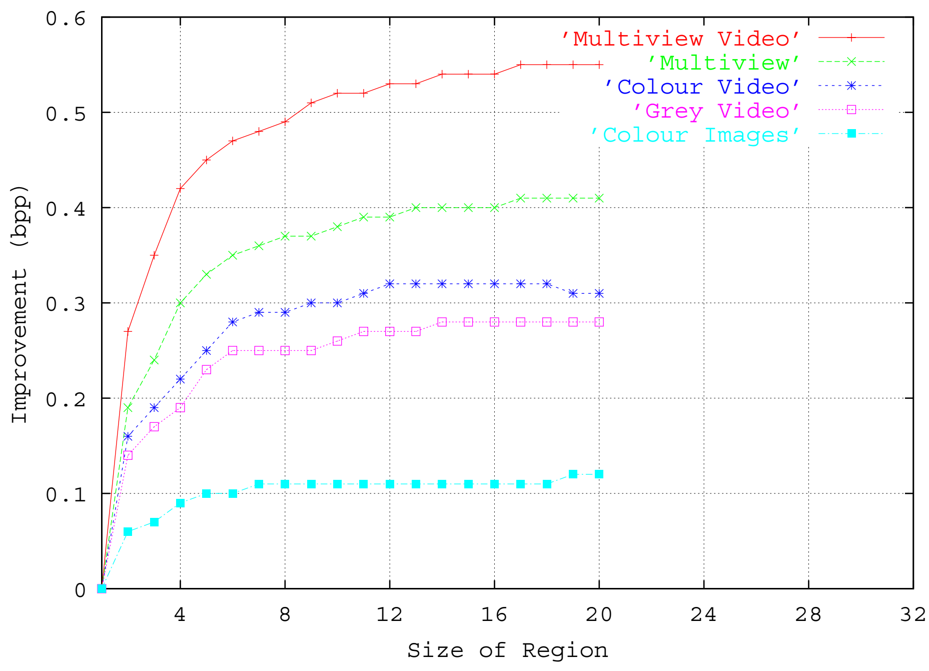


Figure 4.2: The improvement of using ever larger regions for local error based predictor switching.

pixels seems adequate for this. However, multiview video[2] can make use of more predictors. This implies that more information is needed for the predictor selection decision and hence a larger region is required.

However for very large region sizes, virtually no benefit is seen by adding to the region. Indeed, for colour video sequences a drop in performance is seen. This lack of continuing improvement is due to the reduced locality enforced on the predictor selector by using a larger region to gather error averages. An obvious solution to this is to weight the prediction errors by distance from the current pixel. However, a scheme that weighted each error, based on the Euclidean distance between the error location and the current pixel, did not perform as well as the unweighted local error predictor selection scheme.

Determining the *best* value for the region size involves a trade off between performance and computation time. As the region size grows, so to does the computation required to calculate the best predictor according to the LEPS approach. The compromise used for the following tests was a region size of twelve. At this size, the results for colour images and colour video have flattened off and little extra gains are evident from using a larger region with other types of imagery.

[2]The particularly good results for multiview video are based on just one sequence (*granny*). Hence, they may not be truly representative of the performance of a LEPS system.

Further Extending the Locale

As well as extending the area used for error accumulation, we can also extend the current pixel's locale to the previous band or frame. By adding in a given predictor's error from X_r or X_{t-1} (or both) we gather more information about the local performance of the given predictor. This should help make a better predictor selection decision.

Although this approach would seem to offer great promise for colour images and video, two main concerns arise. Firstly, in the case of colour images, the interband predictor can only draw on meaningful prediction residuals from the previous band, from the third band onwards. As colour images often consist of only three bands this is something of a limitation. However intraband, interframe and interview predictors can make use of residuals from the previous band in all but the first band of a colour image.

The second major concern relates to video. The use of residuals from the current location in the previous frame, only offers benefit under the assumption of very low motion. The information gained from such residuals in the presence of motion maybe misleading and lead to poor selection decisions. Furthermore, in a similar situation to that above, interframe predictors can not make use of interframe prediction errors until the third frame of video (the second being the first for which interframe prediction is possible).

As the zero motion assumption is always false for multiview images, use of the prediction errors from the current location in the previous view will not be considered.

The results of using local error predictor selection (LEPS) and extended LEPS (ELEPS) are shown in Table 4.3. Both schemes use a spatial region size of 12, and ELEPS also uses the error at the current location in the previous band and/or frame if available. Before discussing these results, we shall consider a second class of backward adaptive selection schemes.

4.3.2 Context Based Predictor Selection

The scheme discussed in the previous section, determines the context of the current pixel in a simple way (average prediction error for each predictor) and then on that basis, makes a fixed decision (use predictor with least average error). However, as seen in Chapters 2 and 3, there are more sophisticated ways to determine a context and more flexible ways of using the contextual information.

One way to implement a context based predictor selection (CBPS) scheme is to keep track of prediction errors in a given context, for a set of different predictors. For example, consider using an error bucketing context determination scheme, as discussed in Section 2.4.1. The context is determined for each predictor as before and the predictor chosen is that with the least mean absolute error in its current context.

By collecting prediction error magnitudes in a given context, a CBPS scheme makes more selective use of past information than a LEPS approach. However, this is done at the expense of locality; the error accumulated and used for predictor selection may be spatially distant from the current pixel.

Image	LEPS	ELEPS	CBPS
Colour Images			
air2	4.30	4.30	4.31
baboon	6.10	6.10	6.13
cats	2.59	2.59	2.59
cmpnd2	1.59	1.57	1.58
house	4.29	4.29	4.32
lena	4.57	4.57	4.59
Average	**3.91**	**3.90**	**3.92**
Greyscale Video			
claire	2.32	2.31	2.34
granny	2.42	2.41	2.40
mall	3.82	3.81	3.82
mobile	4.39	4.39	4.44
salesman	3.90	3.88	3.90
Average	**3.37**	**3.36**	**3.38**
Colour Video			
claire	2.48	2.42	2.44
football	4.54	4.51	4.58
granny	2.19	2.15	2.17
mobile	4.07	4.03	4.06
susie	3.36	3.34	3.37
Average	**3.33**	**3.29**	**3.32**
Multiview			
ctflesh	0.61	0.61	0.59
granny	2.75	2.72	2.75
skull	2.86	2.84	2.87
Average	**2.07**	**2.06**	**2.07**
Colour Multiview Video			
granny	2.11	2.08	2.11

Table 4.3: A comparison of the entropies of prediction residuals, following different backward adaptive predictor selection mechanisms.

In order to determine the potential benefits of CBPS over LEPS, a CBPS scheme was implemented and the results are shown in Table 4.3. Context determination was via error bucketing and the prediction errors at W, N and NW were quantised into 11 buckets as suggested in [TLR85]. As [TLR85] does not give actual quantisation thresholds for the error bucketing, the approach discussed in [LGS92] was used. This approach is to use exponents of 2 as the thresholds. The justification for this is that prediction errors, generally having a Laplacian distribution (negative exponential) would be expected to equally populate error buckets of exponentially increasing size. The buckets used were:$\{-255,\ldots,-16\}$, $\{-15,\ldots,-8\}$, $\{-7,\ldots,-4\}$, $\{-3,-2\}$, $\{-1\}$, $\{0\}$, $\{1\}$, $\{2,3\}$, $\{4,\ldots,7\}$, $\{8,\ldots,15\}$, $\{16,\ldots,255\}$.

This idea of using contexts for predictor selection is independent of the context determination used. Indeed, the use of the JPEG-LS context determination scheme[ITU96] along with Wu's method[Wu96] were also tried. However, these methods failed to offer any apparent advantage over error bucketing in a CBPS system [3].

4.3.3 Comparing Backward Adaptive Methods

From the results shown in Table 4.3, we can see that ELEPS produces prediction residuals with slightly less entropy than LEPS. This indicates that the extra information from the prediction error at the same location in the previous band and/or frame is useful. The fact that the improvement is slight might have been expected, as the extra error or two that ELEPS uses, are additional to the 12 prediction errors LEPS makes use of. Hence, the influence of the extra prediction errors will be small. This may suggest that including extra prediction errors from the previous band/frame into the accumulator might be useful. However, this is unlikely to be the case, as experience suggests that the prediction error at W is better correlated to $X - \hat{X}$ than the prediction error at W_r or W_{t-1}.

When comparing ELEPS to CBPS, we see that CBPS is unable to match the average performance of ELEPS. Clearly, the more local data collected by ELEPS is more appropriate than the contextual information acquired by CBPS.

In summary, extended local error predictor selection is the best performing backward adaptive method, out of the three methods discussed. ELEPS will be further built on in Section 4.5, when hybrid methods will be investigated.

4.4 Predictor Map Compression

If we determine the best predictor for each pixel in an image, we can construct a *predictor map.* If we store this predictor map during encoding, the best possible prediction (given the available predictors) is guaranteed for every pixel. This represents the ideal forward adaptive predictor selection scheme.

[3]It should be noted that all of these context determination schemes were designed for error modelling and not predictor selection.

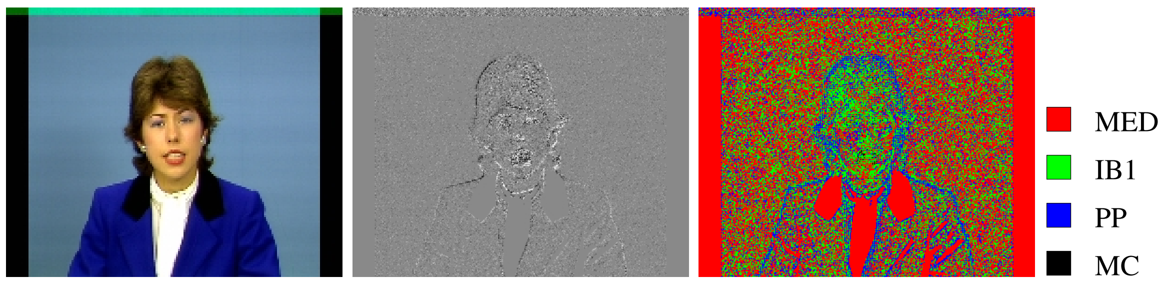


Figure 4.3: Left: A frame from the Claire sequence. Middle: The predictor errors for optimal forward adaptive predictor selection (entropy is 1.38 bpp). Right: A map of optimal predictors (entropy is 1.41 bpp). Although the predictor residuals have very low entropy (lower than that required to store the map!) the sum of the residuals and the predictor map (2.79 bpp) is greater than using interband prediction (2.52 bpp) in this example.

Unfortunately, problems arise with this simplistic notion of a predictor map. For example, as shown in Figure 4.3, the entropy of the predictor map is sufficient that when added to the prediction residuals, the total is not competitive with schemes already seen.

However, it is not necessary to store the prediction map losslessly. In fact, it is not wise to do so. Unlike the pixel values in an image, the values of elements in the prediction map are not necessarily unambiguous. Two predictors may be equally good in some instances. As a further example, consider the case where predictor A has been the best predictor for several pixels in a row. Suppose that for the current pixel, predictor B is slightly superior to predictor A. The cost of signalling the use of predictor B (rather than continuing the run of predictor A usage) may well outweigh the benefit of coding the slightly smaller prediction error generated by predictor B.

Clearly, schemes are needed that efficiently store a useful representation of the predictor map. Unfortunately, as is seen in Figure 4.3, the predictor map does not have the spatial smoothness of an image and therefore predictive decorrelation is not an option.

4.4.1 Block Based Selection

By finding the best predictor for a block of pixels, we can form a very crude, sub-sampled representation of the predictor map. This version of the predictor map has a storage requirement that can be made arbitrarily low, simply by increasing the block size. Though simple, Block Based Selection (BBS) has already be shown to outperform more naïve approaches to extended lossless image compression (see Section 4.1).

Blocks of 8 by 8 pixels were used, as they were found to work well. The best predictor was determined to be the one that gave the least absolute error over the block. For each block the overhead of $\lceil log_2 n \rceil$ bits, where n is the number of possible predictors, is added to the entropy of the prediction errors to get the final outcome. Even with four predictors, the overhead is only 2 bits per block ($2/64 = 0.03$ bpp) although this could be reduced slightly by coding the overhead bits depending on the statistics of predictor usage. Also, overhead is further reduced when using any form of forward adaptive predictor selection, as motion vectors and disparity estimates are only required for regions that use a motion or disparity compensating predictor. For regions where this overhead is not required, it is not sent.

The main problem with BBS is that there is never a correct block size. In plain regions a large block size would lower the overhead of forward adaptation. Whereas, in an active region, the extra overhead of using a smaller block size could be justified by the improved accuracy of the predictor map. These joint goals can be achieved using quadtree methods.

4.4.2 Quadtree Predictor Map Coding

Quadtree based coding starts in the same way as BBS, only with a larger block size. In successive iterations of the algorithm the block size is halved. At a given stage, one block in the previous iteration covers the same area as a four blocks in the current iteration. The prediction error from the one block (using one predictor) can then be compared with those of the four smaller blocks (using up to four predictors).

Once again the least absolute error is used as the criterion for choosing the best predictor for a given block. However, the decision to split a larger block into its four child blocks is more complicated, as extra overhead must be considered. If a parent block is split into four children, four predictor selection decisions need to be encoded, rather than just one. Also, splitting a parent block requires that an extra four split/no split events are encoded. This is unless the parent block is twice the minimum block size (chosen to be 2×2). If this is so, the child blocks cannot be split and therefore those events need not be encoded.

To minimise the overhead associated with the quadtree, the probability of the split decision can easily be estimated by counting the frequency of splits. Hence, the overhead for a given split decision can be calculated as $-\log_2 p(split)$ bits and likewise for the alternative case. As we expect splits to be more common at the top of the tree and less common lower down, the frequency counts are reset at each level.

To increase the likelihood of a net gain, a larger block is only split if the mean absolute error of prediction, of the four small blocks, is less than their parent by more than a certain threshold. This threshold is chosen so as to account for the increase in overhead, per pixel. Neglecting the overhead for the quadtree itself, the per block overhead required to indicate predictor selection information is independent of block size. Therefore, the per pixel overhead is inversely related to the area of the block. As we wish to compare mean absolute errors, rather than the number of bits of information each would require, some constant will be needed to relate the predictor error to increasing average overhead. That is, to split a larger block, the child blocks must have a mean absolute error which is less than their parent by at least $C/$(area of a child block). Empirically, $C = 2$ was found to work well.

The results of using this heuristic and a starting block size of 32 x 32 pixels to construct

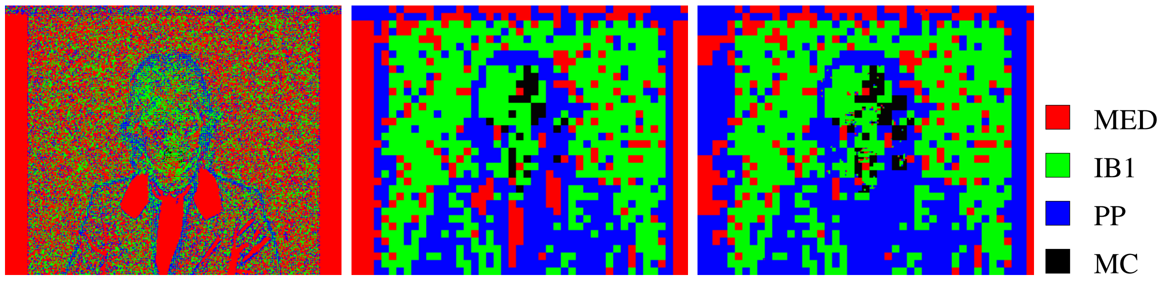


Figure 4.4: Maps of optimal predictors for each pixel (left), block size of 8×8 as used by BBS (middle) and variable block size as used by QTBS (right).

a quadtree based predictor selector (QTBS) are detailed next.

4.4.3 Forward Comparison

The results for BBS and QTBS, along with hybrid schemes to be discussed in the next section, are given in Table 4.4. The ability of QTBS to select appropriate block sizes for predictor selection, relevant to the image properties, gives it only a slight advantage over BBS. This is evident from the reduced entropy of the prediction residuals.

By looking at the predictor selections made, we can gain a better understanding of how QTBS beats BBS. Maps of the predictors selected by both schemes are shown in Figure 4.4. What is clear, is that QTBS used much smaller block sizes around the face area in the *claire* sequence; that is the area in which the most motion occurs. QTBS was thus better able to determine which areas would benefit from motion compensation.

However, Figure 4.4 also shows that QTBS often uses the same block size as BBS. This implies that 8×8 is a good size for BBS to use by default, at least for the *claire* sequence.

4.5 Hybrid Schemes

Hybrid schemes, utilising both forward and backward adaptation, give us the tantalizing possibility of providing the best of both approaches. That is, fine grain decisions and a certain element of foresight.

In fact hybrid predictor selection schemes have already been considered in this chapter. Although BBS is a simple forward adaptive predictor selection scheme, by using MED for spatial prediction it becomes something else. BBS with MED is a multi-layered hybrid predictor. The first layer uses forward adaptation to determine which type of correlation to exploit and if spatial prediction is used, the backward adaptive nature of MED is utilised.

Another good example of a hybrid scheme is given in [MS96] (see Section 3.2.3) where forward adaptation is used to switch between motion compensation and PBPSPEC. An obvious extension of this, is to exchange motion compensation for disparity compensation, allowing for multiview sequences to be coded. For multiview video, a two step forward

	Forward		Hybrid	
Image	BBS	QTBS	PBP+	Oracle
Colour Images				
air2	4.30	4.29	4.39	4.29
baboon	6.09	6.08	6.25	6.08
cats	2.60	2.59	3.11	2.59
cmpnd2	1.58	1.55	1.84	1.57
house	4.28	4.27	4.53	4.29
lena	4.56	4.55	4.76	4.56
Average	**3.90**	**3.89**	**4.15**	**3.90**
Greyscale Video				
claire	2.28	2.27	2.27	2.31
granny	2.40	2.37	2.45	2.42
mall	3.68	3.67	3.77	3.82
mobile	4.33	4.36	4.32	4.38
salesman	3.88	3.88	3.87	3.87
Average	**3.31**	**3.31**	**3.34**	**3.36**
Colour Video				
claire	2.49	2.47	2.44	2.45
football	4.52	4.51	4.72	4.52
granny	2.18	2.15	2.24	2.18
mobile	4.06	4.06	4.26	4.05
susie	3.32	3.31	3.55	3.35
Average	**3.31**	**3.30**	**3.44**	**3.31**
Multiview				
ctflesh	0.58	0.57	1.87	0.61
granny	2.74	2.72	3.30	2.75
skull	2.82	2.80	3.15	2.85
Average	**2.05**	**2.03**	**2.77**	**2.07**
Colour Multiview Video				
granny	2.08	2.04	2.24	2.10

Table 4.4: A comparison of the entropies of prediction residuals, following different forward adaptive and hybrid predictor selection mechanisms.

adaptation process is required. The first stage switches between PBPSPEC and a compensated predictor. In the latter case, a further stage must determine whether motion or disparity compensated prediction should be used. For the purposes of comparison, this scheme was implemented and the results are shown in Table 4.4, under the heading of PBP+.

4.5.1 Consulting the Oracle

We can consider backward adaptation to be like a wise man, using past knowledge and heuristics to guess the future. However, if the wise man is unsure of the future, he might consult an oracle. The Oracle has perfect knowledge of the future (i.e. it is a forward adaptive process), but only gives simple answers to specific questions, e.g. "Which of these two predictors should I use?" Consulting the Oracle is expensive (in terms of overhead) and our wise man would only do so if he was particularly unsure about the best way forward.

What makes this scenario different from what has already been investigated in this chapter, is that heuristics for the confidence the system has in a predictor selection decision, are now required. Also, we must determine how the system can make good use of an oracle.

The backward adaptive schemes seen so far all consider the mean absolute error of the available predictors, although the methods for calculating this average vary. Although the predictor with the minimum MAE is clearly the best candidate, that with the next lowest MAE may also be a strong contender. The confidence heuristics can be based on the difference between the predictions of the top two predictors. If the two predictions are close, the decision is less important, however if they are greatly different then a wrong decision will more notably affect compression.

To implement the scheme, ELEPS (see Section 4.3.1) is used for backward adaptive predictor selection. If the difference between the top two predictors is greater than a threshold T, the Oracle is consulted. The Oracle uses forward adaptation to determine which of the two top predictors, as determined by ELEPS, should be used. The cost of this binary decision is weighed up against the benefit of a lower predictor error and T is changed accordingly. This aims to ensure that the oracle is consulted only as often as is prudent.

Specifically, whenever the Oracle is consulted, the cost of doing so is calculated as the information contained in the Oracle's answer, that is $-\log_2 p(answer)$ bits. This is based on the previously observed probabilities that the Oracle will recommend staying with the first predictor or switching to the second choice predictor. If the first choice predictor is recommended, then no benefit is gained. However, there is a benefit if the second favourite predictor is chosen. The improvement gained in this case is that the lower prediction error of the second predictor will now be coded. As prediction errors are generally distributed as a Laplacian (negative exponential) distribution, a good estimate for the magnitude of the advantage gained is the difference between the logarithms of the prediction errors for each predictor. If the cost associated with visiting the Oracle is greater than the benefit gained, the uncertainty threshold T is raised, to make visits to the Oracle less frequent.

Similarly, after the current pixel is processed, if the Oracle was not visited, the decoder can calculate the cost and benefit that would have been associated with going to the Oracle. If the benefit is found to outweigh the cost, then T is lowered. This aims to make visits to the Oracle more frequent in future.

It was empirically observed that a good starting value for T is 48. To avoid overly rapid changes in the uncertainty threshold, the increments and decrements to the T are both made in 0.1 steps.

4.5.2 Results of Hybridisation

The performance of PBP+ was very disappointing, as it was clearly outperformed by other techniques. As the only extension applied in this work to PBP+ was the addition of disparity estimation, the results for colour images and video parallel the results in [MS95b] and [MS96] respectively. As PBP+ is shown to be inferior to some of the other techniques developed here for predictor switching, we can expect the compression schemes so derived to outperform those presented in [MS95b, MS96].

When comparing the Oracle method against ELEPS (see Table 4.3), we find that the Oracle performs favourably for colour images, saving 0.02 bpp on the *baboon* image for example. However the Oracle's performance on both greyscale and colour video is less impressive. It actually increases the entropy by 0.03 bpp for the colour versions of both the *claire* and *granny* video sequences. Such losses are also seen on most of the multiview sequences.

It is interesting to note that the Oracle shows its benefit for colour images, where only two predictors are employed. Therefore, if the first place predictor is chosen against by the Oracle, the second place predictor is necessarily the best choice. However, the other forms of imagery utilise more predictors and hence the best predictor may be neither the first nor second place predictors, as determined by ELEPS. Further work might productively look for improved methods for determined the second, alternative predictor to be considered by the Oracle.

Hybrid predictor selection schemes, as explored here, clearly fail to bridge the performance gap with the forward adaptive methods. Consequentially, as we continue to build towards extended lossless compression schemes, QTBS should be seen as the best performing predictor selector.

4.6 Summary

In this chapter we have demonstrated that predictor selection is an appropriate method for decorrelating image data, whether it be greyscale, colour, video or multiview imagery. A number of selection approaches have been examined. Schemes using both backward and forward adaptation have been developed, as have hybrid methods using both forms of adaptation.

The end result, although close, suggests that forward adaptation via a quadtree structure is the best method. This quadtree based selection (QTBS) is particularly able to exploit the predictor selection paradigm when employed against more advanced imagery types such as video and multiview data.

However, the fact that the performance of QTBS is nearly matched by many of the other schemes considered, suggests that these methods are approaching the limit of what is possible with relatively simple schemes.

Chapter 5

Extending Error Modelling

The necessary next step in extended lossless image compression is error modelling. Section 5.1 shows how a simple error model can be built for prediction residuals. This is applied both globally and with context conditioning. To improve compression performance, extended context models are discussed in Section 5.2.

In Section 5.3 other techniques often used in error modelling are explored. The nature of bias in the residuals from a predictor selective mechanism is compared to that from a standard predictive scheme. The effect of bias cancellation for extended image compression is considered. Mechanisms for capturing extreme redundancy and improving the adaptability of context modelling are also discussed.

Finally, in Section 5.4 we consider the use of forward adaptive methods for error modelling, using the schemes developed in the last chapter for predictor selection as a reference point. In particular the idea of joint predictor-error model selection schemes is investigated.

5.1 Modelling the Distribution

All the results presented so far have been in terms of entropy. The entropy has been calculated at the end of each band and averaged over all bands. These results assume perfect knowledge of the distribution of errors over the band. Therefore they represent the best possible performance, given the assumption of a fixed distribution for the prediction errors.

However, real image compression schemes need to model the distribution of the prediction errors at all times. These models are generated from statistical counts of past errors. The models are generally kept relatively simple and utilise as few counts as possible. In particular for context conditioned schemes keeping down the number of counts kept per context is important, if context dilution is to be avoided (see Section 2.4).

This need to keep counts low, makes maintaining a perfect model of the error distribution infeasible. As a consequence we might well expect real coding schemes to give results slightly inferior to those entropy based results seen previously. However, highly adaptive error models have the ability to adapt to changing statistics and overcome the fixed distribution assumption made by entropy measures. As such, real compression schemes could improve on the entropy based results in some cases.

5.1.1 Laplace Distributions

When considered globally, prediction errors are generally found to be distributed as a Laplace distribution (two-sided negative exponential) with zero mean[O'N66, HV91]. Such a distribution has a probability density function of[HV91]:

$$f_{\sigma^2}(x) = \frac{1}{\sqrt{2\sigma^2}} \exp\left(-\sqrt{\frac{2}{\sigma^2}}|x|\right) \tag{5.1}$$

where x is a random variable (i.e. the prediction error) and σ^2 is the variance of the distribution. σ^2 depends on the activity in the image and for effective compression must be well estimated.

The above distribution is continuous and infinite. However, prediction errors are discrete and finite. To overcome this discrepancy, we shall estimate the probability of discrete prediction errors, $p(\varepsilon)$ to be:

$$p_{\sigma^2}(\varepsilon) = P_{\sigma^2} \exp\left(-\sqrt{\frac{2}{\sigma^2}}|\varepsilon|\right) \tag{5.2}$$

where P_{σ^2} is a normalising constant given by:

$$1/P_{\sigma^2} = \sum_{\varepsilon=-\alpha}^{\alpha} \exp\left(-\sqrt{\frac{2}{\sigma^2}}|\varepsilon|\right) \tag{5.3}$$

where the range of prediction errors is $-\alpha \leq \varepsilon \leq \alpha$. Figure 5.1 shows a comparison of the continuous and discrete distributions given by Equations 5.1 and 5.2 respectively.

To get a distribution for the current pixel's prediction error, we must estimate σ^2. An obvious estimate is $\hat{\sigma}^2 = E[\varepsilon^2] - E[\varepsilon]^2 = \bar{\varepsilon^2} - \bar{\varepsilon}^2$. However, the Laplacian model implicitly assumes that $\bar{\varepsilon} = 0$, so the estimate $\hat{\sigma}^2 = \bar{\varepsilon^2}$ will be used. The $\bar{\varepsilon}^2$ term in the equation is brought back in to provide bias cancellation in Section 5.3.2.

In order to calculate $\hat{\sigma}^2$ it is sufficient to keep two accumulators per context. Namely, $N_{context}$ the number of times the context has been visited and $E^2_{context}$ the sum of all errors squared.

Armed with an estimate for σ^2 and Equation 5.2, the error model can be used to generate a probability distribution prior to the encoding of each prediction error. Using the information theory law, that the information needed to communicate an event with probability p is $-\log_2(p)$ bits, the size of an optimum coding of the prediction errors (assuming the two-sided Laplacian model) can be calculated.

However, due to the necessity of calculating P_{σ^2} it is not practical to let σ^2 vary arbitrarily. A recurring idea, seen in many other works[MS95b, WSS96], is to use a set

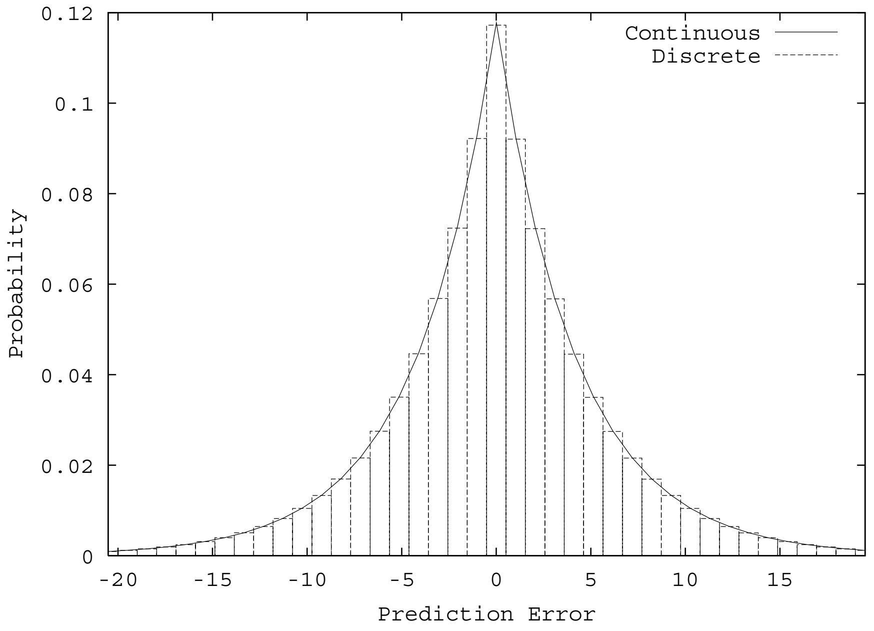


Figure 5.1: Continuous and discrete plots of the probability distribution for prediction errors, given the Laplacian model. $\sigma = 6$ in both cases and the horizontal scale has been truncated for display.

of fixed distributions. To this end, distributions with $\sigma = 0.5, 1, 2 \ldots, 255$ were used and the associated values for P_{σ^2} were pre-calculated. A distribution with $\sigma = 0$ was not used as this case gives $p(\varepsilon = 0) = 1$ and $p(\varepsilon \neq 0) = 0$. Clearly this assumes more than can necessarily be learned from the data!

The results of applying the error model under discussion are shown in Table 5.1. The application of the model is split into two cases. Global application is considered on the left of the table. For the purposes of comparison, the global entropy of the prediction residuals is given under the heading of *Entropy*. These results generally show a slight increase in the information content, indicating that the model is not a perfect fit to the data. Significant deviations will be discussed shortly.

Context conditioned application of the model is on the right of the table. This is the case generally used for lossless compression schemes. The context determination scheme used is the same error bucketing scheme used for predictor selection in Section 4.3.2. These results show a significant gain over the global error modelling and reinforce the utility of separating prediction residuals by context.

It is interesting to note that in most cases, the less than perfect fit of the current model would not have been noticed if the second column of results had not been given. That is, the benefits of context conditioned error modelling generally outweigh the loss of performance due to a slightly inaccurate model.

	Global		Context Conditioned	
Image	Entropy	Modelled	Modelled	Model-B
Colour Images				
air2	4.29	4.68	3.90	3.90
baboon	6.08	6.08	6.00	6.00
cats	2.59	3.05	2.03	2.03
cmpnd2	1.55	3.07	2.56	1.12
house	4.27	4.28	4.15	4.15
lena	4.55	4.58	4.48	4.48
Average	**3.89**	**4.29**	**3.85**	**3.61**
Greyscale Video				
claire	2.27	2.42	2.17	2.17
granny	2.37	2.62	2.20	2.20
mall	3.67	3.72	3.70	3.70
mobile	4.36	4.43	4.36	4.36
salesman	3.88	3.89	3.77	3.77
Average	**3.31**	**3.42**	**3.24**	**3.24**
Colour Video				
claire	2.47	2.59	2.34	2.34
football	4.51	4.59	4.33	4.33
granny	2.15	2.36	2.03	2.03
mobile	4.06	4.10	3.97	3.97
susie	3.31	3.35	3.27	3.27
Average	**3.30**	**3.40**	**3.19**	**3.19**
Multiview				
ctflesh	0.57	2.08	1.51	1.51
granny	2.72	2.84	2.53	2.53
skull	2.80	2.88	2.58	2.58
Average	**2.03**	**2.60**	**2.21**	**2.21**
Colour Multiview Video				
granny	2.04	2.18	1.96	1.96

Table 5.1: The information content of the images in the test set, following QTBS prediction. Several models are considered, including both global and context conditioned approaches.

5.1.2 Bostelmann's Technique

One particular example of where the current model breaks down is the *cmpnd2* image. This compound image contains a significant region of black text (pixel value 0) on a white background (pixel value 255). This leads to great many prediction errors of -255 and 255. Clearly this is not well modelled by the zero centred Laplacian!

One approach to address this problem is a simple technique, sometimes attributed to Bostelmann[LGS92]. This technique makes use of the fact that pixel values have a limited range, say 0 to α. Prediction errors therefore have a nearly doubled range $-\alpha \leq \varepsilon \leq \alpha$. However, this is redundant as a small example will show.

In the case of *cmpnd2*, where α is 255, it is acceptable to remap the prediction errors with:

$$\varepsilon' = \begin{cases} \varepsilon + 256 & \text{if } \varepsilon < -128 \\ \varepsilon - 256 & \text{if } \varepsilon > 127 \\ \varepsilon & \text{otherwise} \end{cases} \tag{5.4}$$

Consequentially, if a black pixel is predicted to be white, the residual is $\varepsilon = -255$ but $\varepsilon' = 1$. Having coded ε', when the decoder adds ε' to the predicted value (255) the result (256) will be out of range. The decoder now takes this value modulo the number of allowable values (256) and gets 0 - the value for black. This amounts to folding the tails of the residual distribution back into the central region.

The results of using Bostelmann's technique prior to error modelling is shown under the *Model-B* heading in Table 5.1. This shows a definite improvement in the case of the *cmpnd2* image, although all other images are unaffected. This is because most images have very few cases of prediction errors that would be remapped by the Bostelmann technique.

One result that still shows a problem with the error modelling scheme used is that of *ctflesh*. This computer generated image is somewhat awkward. It contains regions of near constant intensity that are demarcated by sudden boundaries. As a consequence, although the predictor errors for this image are mainly centred around zero, there are secondary peaks around the levels related to these steps in the image. Such multi-modal distributions are not well captured by the Laplacian model. However, as we shall see in subsequent sections, some additions to the basic error modelling described so far do work to improve on the compression performance for the *ctflesh* sequence.

5.2 Context Determination for Advanced Image Types

The context determination schemes presented in Chapter 2, use a combination of prediction errors, local image gradients and spatial texture to determine an error modelling context. In these prior examples, the metrics are calculated using only intraband information. The same context determination schemes could be used for colour images, video etc. However, at first glance this would seem to make no use of any interband, interframe or interview information.

Before further considering context determination for advanced image types, it is worth reminding ourselves about the purpose of context determination. The assumption is made that prediction errors in a similar context, will have the same distribution. Hence, by determining a context that partitions prediction residuals into groups, such that each group is well modelled by a single distribution, we can better model the errors. Better modelling yields better probability estimates for error coding and hence allows better compression.

So how can we gain additional information about a pixel's context from interband, interframe and interview relationships? As we are using predictors that utilise correlations beyond just spatial relationships, we can gain information implicitly from local prediction errors. This is because a predictor only works well when the correlation it is designed to exploit is high. Thus, a predictor's error is inversely proportional to the local correlation it is suited to exploit. For example, if there are high prediction errors for a spectral predictor, in locations local to a given pixel in the current band, this implies a low spectral correlation. These correlation measures can be very useful for context determination.

The current argument suggests that a standard context determination scheme like error bucketing (see Section 2.4.1) would give extended context determination effectively for free. It is worth noting that the context of a given pixel would then depend on which predictor was being used for that pixel.

Alternatively, we may prefer a more explicit means of extending context determination. The gradient based metric might seem promising, but it entails complications. A large local intraband gradient (e.g. $W - NW$) in a greyscale image would imply a relatively high likelihood of a large prediction error for the current pixel. However, a large interband gradient (e.g. $W - W_r$) does not necessarily imply the same thing. The waveform of the reference band might have the same shape as the current band (high correlation) but a different amplitude. Hence, the important metric would be variation in interband gradients, such as $(W - W_r) - (N - N_r)$.

For video the notion of gradient becomes less useful still. In areas of low motion we expect a low temporal gradient, but in areas of motion the metric would take on arbitrary values.

Instead of concerning ourselves with the complexities of adapting local gradient methods, we shall instead focus on methods utilising prediction errors. Thus, we consider extended error bucketing.

5.2.1 Extended Error Bucketing

When considering extending error bucketing, the main worry is context dilution. Standard error bucketing (SEB), as used in Section 4.3.2, uses 11 buckets for each of the three locations (W, N and NW) used in the context determination. That generates $11^3 = 1331$ contexts. If as part of an extension, a bucket is added to the standard scheme, $11^4 = 14641$ contexts would be generated. This large increase in the number of contexts would be likely to result in context dilution and therefore poor performance.

To overcome this problem, the number of error buckets per location can be reduced.

If 7 buckets are used instead of 11, the number of contexts generated when 4 locations are considered is $7^4 = 2401$. Although this is still more contexts than the 11^3 case, it is a significant improvement over the 11^4 alternative.

The quantisation thresholds used for the 7 bucket case, also used the exponential argument discussed in Section 4.3.2. However, due to the more crude quantisation, exponents of 4 were used. Therefore, the actual buckets are:$\{-255,\ldots,-16\}$, $\{-15,\ldots,-4\}$, $\{-3,-2,-1\}$,$\{0\}$, $\{1,2,3\}$, $\{4,\ldots,15\}$, $\{16,\ldots,255\}$.

It was decided to keep the three spatial locations (W, N and NW) used by SEB in Section 4.3.2. In a manner similar to ELEPS (see Section 4.3.1) the fourth location was chosen to be the current pixel location in the previous band and if that was not available the same location in the previous frame. In the event that neither were available (i.e. the first band in the first frame of a sequence) an extra spatial location was used (NE).

This results in the same complications as experienced by ELEPS. Namely, interband and interview predictors cannot access their own prediction errors from the previous band/frame until the third band or frame respectively.

A new complication that now arises, is related to the limited presence of motion vectors and disparity estimates. As mentioned in Chapter 4, QTBS allows such overhead to be transmitted only when it will be used. As a consequence, if motion compensation is being used for the current pixel, there is no guarantee that motion data will be available for any of its spatial neighbours. To allow encoding without any extra overhead, this issue is solved by using the motion information at the current pixel to get motion compensated prediction errors for context determination. The same process is used for disparity compensation.

If prediction errors are required from X_r, there is no problem. This is because both motion vectors and disparity estimates are shared over all bands of the current image.

The final instance of this problem relates to using the motion compensated residual from X_{t-1}. Again, there is no guarantee that motion information is present for the current pixel location in the previous frame. However, in the case that motion compensation is being used, the motion compensated error from X_{t-1} is unlikely to be useful information, as the feature in question must have moved (otherwise a different predictor would be in use). Hence, the prediction error at X_{t-1} is considered as unavailable when motion compensation is being used.

An extended error bucketing (EEB) scheme was constructed as discussed above and the results of this scheme are compared to SEB in the next section.

5.2.2 Comparing Context Determination Schemes

The utility of a given context determination scheme can be measured by the compression performance obtained by an image coder, where the context determination scheme is used to condition the errors[1]. Using the two-sided Laplacian model, with Bostelmann's

[1]It is worth noting here that the use of context based entropy measures is not a good idea. This can be done by calculating the entropy of prediction errors in individual contexts and then averaging these contextual entropies, weighted by context occurrence. However, this pre-supposes perfect knowledge of as

Image	SEB		EEB	GRAD
	11^3	7^4		
Colour Images				
air2	3.90	3.86	3.84	3.88
baboon	6.00	6.00	6.00	5.98
cats	2.03	2.02	2.02	2.03
cmpnd2	1.12	1.11	1.11	1.22
house	4.15	4.14	4.15	4.12
lena	4.48	4.47	4.47	4.46
Average	**3.61**	**3.60**	**3.60**	**3.62**
Greyscale Video				
claire	2.17	2.13	2.11	2.15
granny	2.20	2.21	2.21	2.41
mall	3.70	3.70	3.70	3.70
mobile	4.36	4.36	4.36	4.37
salesman	3.77	3.78	3.78	3.87
Average	**3.24**	**3.24**	**3.23**	**3.30**
Colour Video				
claire	2.34	2.34	2.24	2.28
football	4.33	4.34	4.31	4.39
granny	2.03	2.02	2.02	2.19
mobile	3.97	3.98	3.96	4.03
susie	3.27	3.27	3.26	3.30
Average	**3.19**	**3.19**	**3.16**	**3.24**
Multiview				
ctflesh	1.51	1.26	1.26	1.66
granny	2.53	2.56	2.53	2.61
skull	2.58	2.61	2.59	2.64
Average	**2.21**	**2.14**	**2.13**	**2.30**
Colour Multiview Video				
granny	1.96	1.96	1.95	2.07

Table 5.2: A comparison of several context determination schemes is achieved by considering the information content of prediction residuals, following context conditioned error modelling. All context coding is based on prediction residuals from QTBS.

technique, as discussed in the previous section, SEB and EEB were compared.

The results in Table 5.2 include SEB-11^3 as used in the previous section. SEB-7^4 (where the fourth bucket always uses the error from NE) is also included to determine whether any context dilution effects are evident when the number of contexts increases from 1331 to 2401. Results for EEB are presented and to further the comparison, the gradient based context determination scheme from JPEG-LS is also included under the heading GRAD.

The results from SEB-7^4 are very similar to those from SEB-11^3. Many images actually show slight gains from the extra specificity gained by adding the fourth error bucket. Only a few images show signs of performance loss. Where performance is lost, it is not clear whether this results from context dilution or the more coarse quantisation of the errors into buckets. Whatever the reason, the use of a 7^4 bucketing scheme seems quite acceptable.

Comparing the results from SEB-7^4 and EEB, we see that EEB is generally as good as or better than SEB-7^4. The most noticeable improvement is for colour video, where prediction residuals from the previous band or frame will almost always be available.

The context determination scheme from JPEG-LS, which was also used in an interband extension of the new standard[MWSM97], was unable to match the average performance of EEB. One example of an exception to this generalisation is the *house* image, for which prediction residuals are better modelled by the JPEG-LS scheme than by EEB. In fact SEB-7^4 performs better on *house* than EEB. This shows that purely spatial context determination can still be the best of the available options, in some cases.

5.3 Further Contextual Considerations

In this section we will consider three other aspects of context based error coding. Namely bias, runs and the resetting of contextual data for addition adaptation.

5.3.1 A Different Bias

Contextual bias in prediction errors is the result of a systematic failure of the predictor to correctly model some aspect of the image. In most work, a single predictor is used and bias can be seen as a slight deficiency in that predictor. In the current work, the prediction for a given pixel is the output of many predictors and a predictor selection scheme. To further the discussion of the current work, we should understand how the use of predictor selection affects contextual bias in prediction errors.

As predictor selection schemes try to find the best predictor in a given circumstance, we might expect the prediction used to be superior to a non-selective mechanism. As a consequence, if the predictor is better matched to the current image characteristics, we would expect the average absolute bias to be less for switching predictor schemes.

many distributions as there are contexts; a considerable amount of assumed information. Hence, context based entropy measures can be wildly optimistic about the potential compression performance of a given method.

Image	JPEG7	MED	QTBS
Colour Images			
air2	7.13	6.25	3.16
baboon	5.97	6.05	3.56
cats	8.22	7.46	1.81
cmpnd2	9.92	6.80	3.29
house	4.44	2.80	1.32
lena	5.70	5.21	2.62
Average	**6.90**	**5.76**	**2.63**
Greyscale Video			
claire	3.27	2.86	0.59
granny	5.72	4.99	2.48
mall	1.72	0.85	0.84
mobile	4.95	2.58	2.67
salesman	4.40	3.94	0.84
Average	**4.01**	**3.04**	**1.48**
Colour Video			
claire	3.70	3.03	0.62
football	6.82	6.49	3.74
granny	6.12	5.83	2.13
mobile	6.28	5.69	2.43
susie	3.42	3.18	0.72
Average	**5.27**	**4.84**	**1.93**
Multiview			
ctflesh	3.21	1.19	0.76
granny	6.22	5.25	2.30
skull	2.19	2.21	0.56
Average	**3.87**	**2.88**	**1.20**
Colour Multiview Video			
granny	6.21	5.08	1.51

Table 5.3: The average absolute contextual bias following various prediction schemes. EEB is used for context determination in all cases.

To test this assumption, the average absolute[2] contextual bias is determined for a number of prediction schemes. JPEG7 will be used as an example of a simple, non-switching scheme. MED and QTBS will be used as examples of simple and complex switching schemes respectively. Context determination is carried out by the EEB scheme developed in the previous section.

The results of this test are presented in Table 5.3. Each result represents the average absolute contextual bias, averaged over all bands/frames in the image/sequence. These results clearly show that the contextual bias of prediction errors is greatly reduced by using more adaptive predictors, as argued above. MED shows a slight improvement over JPEG7 and QTBS reduces the contextual bias still further. The superior performance of QTBS in this regard can be put down to its active selection of predictors via forward adaptation and its greater choice of predictors as compared to MED.

5.3.2 The Effect of Bias Cancellation

Although the use of QTBS has been shown to reduce the contextual bias in prediction residuals, it has not eliminated it. Hence, it may still be worthwhile to use bias cancellation to further improve compression.

A fairly standard approach was taken to this problem. A count of the sum of all prediction errors was kept for each context. This accumulator, averaged over the number of instances of the context encountered up to a given point, gives the contextual bias. This bias is then subtracted from the prediction error before coding. It is also used, to offset the calculation of $\hat{\sigma}^2$. Bias cancellation provides the $\bar{\varepsilon}^2$ that enables the estimation of $\hat{\sigma}^2 = \bar{\varepsilon^2} - \bar{\varepsilon}^2$ as discussed in Section 5.1.

The results from applying this actively re-centred distribution model are given in Table 5.4. From these results, bias cancellation can be seen to be worthwhile. With just one exception the test images showed improvement. It is interesting to compare these improvements with the average absolute contextual bias figures in Table 5.3. The sequence with the largest bias after QTBS is *football* and this same sequence shows the greatest improvement after bias cancellation (0.14 bpp). However, there is no simple trend to the data when taken as a whole. *cmpnd2* also shows a large bias following QTBS, but shows only slight improvement (0.02 bpp) after bias cancellation. Alternatively, *salesman* has very low contextual bias but shows a large improvement (0.12 bpp) in Table 5.4.

From this analysis we can conclude that simple bias cancellation should be included in the current compression approach. However, it is clear that the method used is unable to deliver savings in proportion to the amount of bias present. Hence, further work may yet yield improved bias cancellation.

[2]To have meaning the absolute bias must be used. If the raw bias was averaged, positive and negative biases would cancel to leave the global, not contextual, deviation from a zero mean distribution.

Image	EEB	After Bias Cancellation	With Run Mode	Context Resets
Colour Images				
air2	3.84	3.79	3.80	3.80
baboon	6.00	5.95	5.95	5.92
cats	2.02	1.98	1.90	1.89
cmpnd2	1.11	1.09	0.99	0.99
house	4.15	4.13	4.13	4.12
lena	4.47	4.42	4.42	4.42
Average	**3.60**	**3.56**	**3.53**	**3.52**
Greyscale Video				
claire	2.11	2.10	2.08	2.06
granny	2.21	2.20	2.20	2.18
mall	3.70	3.69	3.69	3.70
mobile	4.36	4.34	4.34	4.34
salesman	3.78	3.66	3.66	3.65
Average	**3.23**	**3.20**	**3.19**	**3.19**
Colour Video				
claire	2.24	2.22	2.22	2.20
football	4.31	4.17	4.17	4.15
granny	2.02	2.01	2.01	2.01
mobile	3.96	3.84	3.84	3.86
susie	3.26	3.16	3.16	3.16
Average	**3.16**	**3.08**	**3.08**	**3.08**
Multiview				
ctflesh	1.26	1.27	1.24	0.88
granny	2.53	2.52	2.52	2.49
skull	2.59	2.53	2.54	2.51
Average	**2.13**	**2.11**	**2.10**	**1.96**
Colour Multiview Video				
granny	1.95	1.94	1.95	1.94

Table 5.4: Results of bias cancellation, run mode and contextual resets.

5.3.3 A Case of the Runs

The error model that emerges from the previous discussion is highly adaptive and caters for a wide range of image activity. However in some cases, for example the *cmpnd2* image, more could be done. This image in particular contains many areas where there is no image activity. That is, it contains quite a lot of blank space. To be coded effectively, a model that is parameterised to give a $p(\varepsilon = 0)$ of nearly 1 is required. However, this is very dangerous, as when the region ends and the predictor error is non-zero the cost to code this transition could be very high indeed.

In such cases of extreme redundancy, other approaches are needed. One such approach is to use a *run mode*. Such a method is used in JPEG-LS[ITU96] and briefly described in Section 2.5.1. When entering a special context JPEG-LS assumes a run of pixels with the same value and codes the run length. This use of forward adaptation allows the efficient coding of extreme redundancy without the potential pitfall of encoding a large prediction error using a distribution in which it is very unlikely.

As an alternative, we might consider that a run is likely in any context for which $\hat{\sigma}^2$ is close to zero. For the purpose of practicality, $\hat{\sigma}^2 < 0.5$ is deemed to be close to zero.

We must also consider what sort of run to expect. JPEG-LS uses a spatial run of identical pixel values. However in the current work, we might consider an interband run, for which spectral prediction yields zero error. Such a run requires different handling to those of JPEG-LS. To make this difference, a run is here defined as a sequence of zero prediction errors, rather than identical pixel values.

If the current pixel is in a context where $\hat{\sigma}^2 < 0.5$, a run can only start if $\varepsilon = 0$. So, instead of entering the run mode on entry to the context, the prediction error is first encoded and the run mode is only entered if ε is indeed equal to zero.

If a run is present it is just as likely to extend down as it is to extend across from the current location. Hence, it seems more appropriate to consider a run not as a sequence starting at the current pixel but as a region with the current pixel at the top left corner. For the sake of convenience the *run region* was chosen to be a square.

To summarise the implementation, the run mode is entered when the current pixel is in a context for which $\hat{\sigma}^2 < 0.5$ and following bias cancellation, $\varepsilon = 0$. Once in run mode, the encoder searches for the largest allowable $n \times n$ square, with the current pixel as the top left corner, such that all prediction errors are zero in the region so defined. To enable efficient coding of the run length (which must be at least 1), an upper limit is put on n (32). Furthermore, counts of the frequency of occurrence of given run lengths are kept to allow entropy coding of this overhead.

The consequence of adding this run mode capability to the current modelling schemes, is show in Table 5.4. As might be expected, images with considerable flat regions (*cats* and *cmpnd2*) show considerable improvement, while most of the other test data is unaffected. The relatively small reduction in bit rate for *ctflesh* was quite surprising as the sequence appears to be very amenable to run coding. However, close examination of the source images shows that apparently flat regions are affected by noise. An apparently random collection of pixels are one grey level brighter than the majority of the surrounding region.

These pixels will break the runs and hence reduce the effectiveness of the run coding scheme.

Further Extension of the Run Mode

The runs that have just been considered are partially extended, in that they can be runs of zero error from an interband, interframe or interview predictor. However, we might consider further extensions. The most obvious is to combine runs in different bands. With this scheme a run mode would be entered if the prediction errors at one location, in all bands, were zero

Such a scheme would be expected to offer superior performance for images such as *cats* and *cmpnd2*, where the flat regions of black and white have very high spectral correlation. However, the fact that this scheme assumes total spectral correlation in flat regions of the image may well cause a drop in performance for other images.

The results in [MWSM97] support just such a conclusion. When considering an interband version of JPEG-LS, the standard JPEG-LS run mode is synchronised across all the bands in the image. *cats* and *cmpnd2* do indeed show modest improvements in bit-rate. However, overall the effect across the entire test set used in [MWSM97] suggests that such a run mode is inferior to the original scheme used in JPEG-LS, as discussed in Section 2.5.1.

5.3.4 Resetting of Contextual Data

One danger of the count based statistics accumulated by the contextual scheme currently in use, is that old data is always present. This long memory could be harmful if the contextual statistics were to change over time. One popular way to alleviate this is to halve the counts kept in a given context, when the occurrences of that context reach a certain threshold. In this way the effect of old statistics is lessened and subsequent counts will have a greater effect.

This additional piece of adaptivity, also used in [ITU96], was added to the current compression scheme. A context occurrence threshold of 64 was used.

The results of using these contextual resets are given in Table 5.4. In almost every case a slight benefit is seen from the extra ability to adapt. Only for a couple of sequences is the loss of the extra long term memory found to be negative.

5.3.5 Contextual Summary

We have seen how contextual modelling of prediction residuals can be accomplished in the new environment of extended lossless image compression. For context determination a slight benefit is seen by considering information from the previous band or frame. By modelling runs of zero prediction error, further advantage is made of extended correlation. This is because runs can occur in areas of extreme spectral or temporal correlation as well as areas that are spatially flat.

Although no obvious use of extended information has been made in bias cancellation or contextual resetting, these techniques were still found to be of benefit in the currently proposed model.

5.4 Forward Methods for Error Modelling

Although backward adaptive methods for error modelling are dominant in the literature, having seen how well forward adaptive methods fared at predictor selection, it makes sense to test the application of forward methods to the current problem. The basic method is the same as that seen in the previous chapter. That is, for each region of the image the best parameters (for an error model in this case) are calculated and transmited to the decoder.

Obviously, with such schemes the nature of the error model is important. As the model parameters must be sent to the decoder, a model that requires only one parameter is favourable. Fortunately, the 2SL-B model used previously is just such a scheme and is now used with forward adaptation.

Previously, the backward adaptive approaches had access to distributions with $\sigma = 0.5, 1, 2 \ldots, 255$. This is not appropriate for forward adaptation, as such a fine quantisation of σ would generate a large overhead for indicating the choice of error modelling parameters to the decoder. Instead, only eight values of σ used, corresponding to the powers of two $1, 2, \ldots, 128$.

Initial tests showed that the largest value of σ was almost never used. Given this, the meaning of the event $\sigma = 7$ was changed to indicate a run mode. If a run mode is indicated, it implies that the prediction errors are zero for all pixels in the region. Therefore no further decoding is necessary for the relevant pixels.

Previously, when a quadtree structure was employed for predictor selection, the decision of whether to split a parent block into its four children was governed by a simple heuristic. That is, if the average difference in prediction errors from the parent and the children was less than a threshold, a split occurred. With error modelling the actual bits required to encode the prediction residuals are known. Hence the splitting decision can be made in a way related to the actual cost of the additional overhead required if a split takes place.

If a split takes place, four extra split decisions will need to be encoded (unless the child block size is the minimum allowed). The error model parameter will also be needed for each of the four children, but not for the parent. Estimating the cost of the model parameter at 3 bits (8 possible values) and the cost of the split decisions at 1 bit each, the splitting threshold becomes 13 bits.

In order to test forward error modelling the results of QTBS followed by QuadTree Base Modelling (QTBS-QTBM) are given in Table 5.5.

Image	QTBS-QTBM	Joint QTM
Colour Images		
air2	3.91	3.91
baboon	5.95	5.96
cats	1.94	1.94
cmpnd2	1.06	1.06
house	4.13	4.13
lena	4.50	4.50
Average	**3.58**	**3.58**
Greyscale Video		
claire	2.16	2.15
granny	2.20	2.18
mall	3.71	3.71
mobile	4.35	4.34
salesman	3.87	3.87
Average	**3.26**	**3.25**
Colour Video		
claire	2.34	2.32
football	4.35	4.34
granny	2.04	2.02
mobile	4.02	4.02
susie	3.30	3.30
Average	**3.21**	**3.20**
Multiview		
ctflesh	1.11	1.11
granny	2.62	2.60
skull	2.63	2.63
Average	**2.12**	**2.11**
Colour Multiview Video		
granny	1.98	1.96

Table 5.5: The results of forward error modelling.

5.4.1 Joint Adaptation

Using separate forward adaptation mechanisms for predictor selection and error modelling may be somewhat wasteful. It is quite possible that regions that are well serviced by a particular predictor would often produce residuals that fit a given error model. The potential for gain is even greater with a quadtree based scheme, as the overhead for one of the quadtrees can be saved if joint predictor selection and error modelling is performed.

In essence the joint scheme works in a very similar way to before. However, for each region, *every* combination of available predictor and error model is considered with the best choice being included in the output. The other difference is that the cost of encoding three additional predictor selection decisions must be taken into account when determining the split threshold. The exact cost of these decisions is determined by the number of predictors available at a given time. The results from joint QuadTree Modelling (Joint QTM) are also shown in Table 5.5.

5.4.2 Results for Forward Error Modelling

The results in Table 5.5 show that Joint QTM is slightly better on average than QTBS-QTBM. However, the difference is slight and for the *baboon* image, the reduced flexibility of having to specify predictor and error modelling parameter together for each block, has had a negative effect.

Considering the results of forward error modelling against those for backward modelling, as given in Table 5.4, we see that the latter is superior. Although the results for Joint QTM are comparable to those for modelling with EEB context coding, the backward schemes gain a significant advantage when bias cancellation, run mode and contextual resets are added. Joint QTM already contains a run mode and it is difficult to see how forward bias cancellation could be added without incurring intolerable overhead. A hybrid scheme utilising contextual bias cancellation and forward adaptation of the error modelling parameter would be interesting, but it is not clear that it would bridge the gap to the backward adaptive schemes seen earlier.

5.5 Summary

In this chapter we have seen that the standard two-sided Laplacian model, with the addition of Bostelmann's technique, is an acceptable way to model the prediction residuals from QTBS.

The problems of extending context coding were discussed. A solution, Extended Error Bucketing was found to produce slight benefits over the standard approach. The nature of contextual bias in the prediction residuals for switched predictors (particularly QTBS) were analysed and found to be lower than for standard prediction schemes. However, it was also shown that bias cancellation is still a necessary technique for top performance. Run modes and contextual resets were also added to the discussion.

Forward error modelling approaches were briefly considered and found to be inferior to the backward adaptive method based on context coding. Hence, it is concluded that the best available method for extended error modelling uses the 2SL-B model, with EEB context determination and also includes bias cancellation, run mode coding and contextual resets as described in Section 5.3.

Chapter 6

Practical Results

Now that all the relevant techniques have been advanced, it is necessary to generate full compression schemes and therefore real compression results.

Section 6.1 discusses the practicalities involved in producing a complete compression scheme from the tools previously discussed. Having introduced certain practicalities, the real compression results are compared against the theoretical results from the previous chapter.

In Section 6.2 several aspects of the developed compression scheme are analysed. Primarily this analysis considers the predictor usage and the overhead present in the output. Finally, in order to have external relevance, the results presented must be compared to other compression schemes from the literature. This is accomplished in Section 6.3.

6.1 Towards a Practical Coding Scheme

The results of the previous chapter were based on a theoretical information measure, utilising the probability of events given by a particular model. A practical compression scheme needs an encoder as a final step. Of the many available coding schemes (see Appendix A for an overview) the one that will allow the best match to the theoretical result of the previous chapter is Arithmetic coding. This coding scheme is able to code symbols using a fractional number of bits and is well known to approach the optimum coding limit.

To avoid unnecessary effort, an existing package was used for Arithmetic coding. The package used was produced by John Danskin and is based on a Bell, Cleary, and Witten style Arithmetic Coder[BCW90]. It makes use of frequency based histograms and provides coding of symbols from alphabets of arbitrary size. The probability distributions generated by the models of the last chapter were converted into the required frequency histograms simply by multiplying all the probabilities by a large constant (10000).

The impact of moving to a practical encoding of the data is shown in Table 6.1. For comparison the theoretical results from the last chapter are shown. To recap, these results are based on QTBS followed by EEB context modelling, using 2SL-B. Bias cancellation,

Image	Theory	With Arithmetic Coding	Including Regular I-frames
Colour Images			
air2	3.80	3.82	3.82
baboon	5.92	5.93	5.93
cats	1.89	1.91	1.91
cmpnd2	0.99	0.99	0.99
house	4.12	4.15	4.15
lena	4.42	4.45	4.45
Average	**3.52**	**3.54**	**3.54**
Greyscale Video			
claire	2.06	2.09	2.12
granny	2.18	2.21	2.29
mall	3.70	3.73	3.73
mobile	4.34	4.32	4.35
salesman	3.65	3.68	3.75
Average	**3.19**	**3.21**	**3.25**
Colour Video			
claire	2.20	2.22	2.24
football	4.15	4.17	4.17
granny	2.01	2.03	2.09
mobile	3.86	3.87	3.88
susie	3.16	3.19	3.19
Average	**3.08**	**3.10**	**3.11**
Multiview			
ctflesh	0.88	0.85	0.85
granny	2.49	2.55	2.55
skull	2.51	2.54	2.54
Average	**1.96**	**1.98**	**1.98**
Colour Multiview Video			
granny	1.94	1.98	2.02

Table 6.1: A comparison of theoretical results from the previous chapter against real results using Arithmetic coding. The column on the right also includes I-frame coding at regular intervals, which is a requirement for practical video coding.

run mode and contextual resets are also used.

Generally the impact of using a practical encoding method is very small; generally around 0.03 bpp. This is as expected given the near ideal nature of Arithmetic coding.

Analysis of Table 6.1 does present some anomalous results. Namely, *mobile* (greyscale) and *ctflesh* which actually improve under practical coding. The reason for this is the way overhead is processed. The theoretical results from the previous chapter used entropy based measures to account for the size of the motion vectors and disparity estimates. In the practical coding case, the coding of these types of overhead is based on frequency counts that are updated as progress is made through the image. As *mobile* (greyscale) and *ctflesh* make heavy use of motion compensation and disparity compensation respectively (see Section 6.2.1) this effect is most pronounced for these two cases.

6.1.1 The Need for Regular I-frames in Video

In this work so far, there has been a conscious attempt to avoid making special allowances for one type of imagery over another. However, video must be seen as being a slightly special case. When a single image is decoded, the whole image is generally required. Likewise multiview sequences; two or more images[MDTL96] must be displayed at once to gain a stereoscopic effect. Video however, can be meaningfully viewed in segments, rather than requiring the whole.

As a concession to this difference, multiple access points will be allowed to a coded video sequence. To enable this, the Group of Pictures (GOP) structure of conventional lossy video coders will be borrowed (see Section 3.2.2). The GOP size will be the same as MPEG's default GOP size (12). However, as lossless bi-directional coding of video frames has not been considered, the GOPs will consist of one I frame followed by 11 P frames.

As can be seen from the results in Table 6.1, the cost of regular I-frames is not generally excessive. Indeed, given the benefit that is gained from multiple entry points into video data, it seems a price worth paying.

6.2 Analysis

In this section we shall see how QTBS actually makes use of the five predictors available to it. This is followed by a study of the overhead carried by the current scheme. That is, how many bits are there that do not directly encode prediction errors.

6.2.1 Predictor Usage Analysis

We saw in Chapter 4 that QTBS is capable of producing prediction errors with relatively low entropy. Now, as an integral part of the presented compression scheme, it is appropriate to investigate how it uses the predictors that is has access to.

Table 6.2 shows the usage of the available predictors for the whole image test set. The figures shown give the percentage predictor usage, average over all bands, frames and views. Note, the figures are rounded and may not sum to exactly 100%.

For colour images we see that intraband prediction is on average used more than interband prediction. However, it is important to note that these figures are biased towards intraband prediction, because that is the only prediction available for the first band in the image. The first band represents one third of the whole image data for these RGB images, so the maximum possible usage of interband prediction is 66.7%. With an average usage of 40.7%, interband prediction is used by QTBS for slightly over half of the pixels in the blue and green bands of the colour images used for the test.

For greyscale video, temporal prediction is used twice as much as intraframe prediction on average. The notable exception to this is the *mall* sequence. This sequence contains whole scene movement and therefore PP would be inappropriate as a predictor. That MC is not used for *mall* indicates that there is either a lot of interframe noise or that the motion estimation used was unable to capture the movement in the sequence. It is possible that a more sophisticated motion estimation scheme would change this result.

Also of note for the greyscale sequences, is that PP and MC are on average used in equal proportion. As mentioned in Chapter 4, QTBS will choose PP in preference to MC if their prediction residuals are equally small. However, PP carries no overhead and is well used. This shows the importance of including it alongside MC.

Moving onto the colour video sequences, it can be seen that the preference for interframe predictors is replaced by a tendency to use interband prediction. The exception to this generalisation is the *granny* sequence. *granny* is computer generated and therefore has no camera noise. It also has no global motion (i.e. camera movement). These two factors taken together, give a high probability that a given pixel will be identical to the same location in the previous frame. This ensures that PP is the preferred predictor, even over IB1. This is not the case though for all the other sequences (even those with no camera movement) that are captured by conventional means. However, this is not to say that the other sequences gain no benefit from interframe prediction. Indeed, *mobile* uses MC more than MED.

The computer generated versus camera captured dichotomy is again present when multiview sequences are considered. The computer generated multiview sequences, having perfect alignment and no camera noise, are able to make good use of disparity compensation. However, the camera captured sequence *skull* barely uses disparity estimation at all. Camera alignment is the most likely cause of this result.

It is interesting to note that *ctflesh* makes such heavy use of disparity compensation. As has been mentioned previously, this image set contains many strong edges in each view. These edges are the sort of feature that intraband predictors (even MED) find hard to deal with. However, disparity compensation is able to track these features across the sequence and hence is generally the best predictor for *ctflesh*.

The results for the colour multiview video sequence *granny* is quite similar to the colour video version of the same. However, disparity estimation is used and therefore shows that

Image	MED	IB1	PP	MC	DC
Colour Images					
air2	62.0	38.0	-	-	-
baboon	52.6	47.4	-	-	-
cats	66.4	33.6	-	-	-
cmpnd2	64.8	35.2	-	-	-
house	64.3	35.7	-	-	-
lena	46.0	54.0	-	-	-
Average	**59.4**	**40.7**	**-**	**-**	**-**
Greyscale Video					
claire	18.9	-	68.3	12.9	-
granny	18.8	-	70.1	11.1	-
mall	99.2	-	0.1	0.7	-
mobile	14.8	-	3.6	81.7	-
salesman	13.4	-	24.6	62.0	-
Average	**33.0**	**-**	**33.3**	**33.7**	**-**
Colour Video					
claire	38.5	33.2	22.5	5.7	-
football	23.0	51.1	5.7	20.2	-
granny	10.9	17.5	64.9	6.8	-
mobile	7.0	54.7	1.8	36.5	-
susie	20.4	64.3	7.1	8.2	-
Average	**20.0**	**44.2**	**20.4**	**14.3**	**-**
Multiview					
ctflesh	19.3	0.0	-	-	80.7
granny	19.7	25.6	-	-	54.7
skull	29.8	66.1	-	-	4.0
Average	**22.9**	**30.6**	**-**	**-**	**46.5**
Colour Multiview Video					
granny	4.0	13.4	62.7	3.7	16.2

Table 6.2: Average percentage of predictor usage for QTBS.

is has some advantage over the other predictors in some cases.

In summary, all of the five predictors included are used in varying proportions by QTBS, on the test data. That the predictors are used in different proportions for different images shows the power of predictor selection in allowing an adaptive, extended lossless images compression scheme with good general performance.

6.2.2 Analysis of Incurred Overhead

The presented compression scheme has many elements that use forward adaptation. All such elements generate overhead that must be included in the output of the encoder. So far, relatively little attention has been paid to this component of the output bit-stream. In order to gauge the penalty of carrying this overhead, its contribution to the output has been determined.

Of the forward adaptive elements, QTBS is probably the most obvious and must transmit both its quadtree structure and predictor selections as overhead. Also two of the predictors used require additional information to function. Motion Compensation and Disparity Compensation require motion vectors (MVs) and disparity estimates (DEs) respectively. Thanks to the nature of QTBS, this overhead need only be transmitted when the relevant predictor is actually used. Finally the run mode introduced in Section 5.3.3 also makes use of forward adaptation to gauge the size of the run region. The contribution, in terms of bpp, to the output bit-stream of all these forms of overhead is recorded in Table 6.3.

For QTBS, the overhead of the quadtree structure, which is modelled on the basis of the observed frequency of splits at a given level, is quite small. The overhead for signalling the predictor selection decision, which is done without any probabilistic modelling, is also generally small. However, for colour video sequences it does become the largest single form of overhead. A reduction in the cost of this overhead could be had by modelling based on frequency counts. This would lead to an average improvement of around one hundredth of a bit per pixel for colour video sequences.

The overhead requirements for motion vectors and disparity estimates are related to the usage of MC and DC seen in Table 6.2. These forms of overhead are already modelled by frequency counts, but more sophisticated modelling schemes could further reduce the overhead. One such scheme, to use predictive coding for the MVs and DEs, is complicated by the fact that the information is only present when it is used. Therefore, the full set of neighbouring values may not be present, making prediction difficult.

Another possibility is to not send the motion vectors/disparity estimates at all. The decoder could use neighbouring pixel values to compute the relevant estimation and use the result as an estimate for the current pixel. However, it is not clear that the savings this would create would balance against the potential loss that could be made due to less accurate motion and disparity estimations. This scheme would also have the undesirable consequence that the decoder would have to expend a great deal of computation on estimation routines, every time the image was to be decompressed.

Image	Quadtree	Predictor	MVs	DEs	Run Mode	Total
Colour Images						
air2	0.0047	0.0072	-	-	0.0278	0.0397
baboon	0.0059	0.0108	-	-	0.0000	0.0167
cats	0.0002	0.0008	-	-	0.0038	0.0048
cmpnd2	0.0006	0.0011	-	-	0.0248	0.0265
house	0.0065	0.0085	-	-	0.0000	0.0150
lena	0.0058	0.0076	-	-	0.0013	0.0147
Average	**0.0040**	**0.0060**	**-**	**-**	**0.0096**	**0.0196**
Greyscale Video						
claire	0.0021	0.0054	0.0087	-	0.0194	0.0356
granny	0.0042	0.0110	0.0114	-	0.0039	0.0305
mall	0.0005	0.0027	0.0011	-	0.0000	0.0043
mobile	0.0066	0.0171	0.0631	-	0.0000	0.0868
salesman	0.0031	0.0078	0.0368	-	0.0006	0.0483
Average	**0.0033**	**0.0088**	**0.0242**	**-**	**0.0048**	**0.0411**
Colour Video						
claire	0.0090	0.0211	0.0026	-	0.0332	0.0659
football	0.0124	0.0381	0.0141	-	0.0027	0.0673
granny	0.0057	0.0136	0.0043	-	0.0412	0.0648
mobile	0.0124	0.0343	0.0209	-	0.0000	0.0676
susie	0.0042	0.0113	0.0074	-	0.0012	0.0241
Average	**0.0087**	**0.0237**	**0.0099**	**-**	**0.0157**	**0.0580**
Multiview						
ctflesh	0.0003	0.0011	-	0.0439	0.1140	0.1593
granny	0.0080	0.0167	-	0.0292	0.0225	0.0764
skull	0.0038	0.0062	-	0.0070	0.0420	0.0590
Average	**0.0040**	**0.0080**	**-**	**0.0267**	**0.0595**	**0.0982**
Colour Multiview Video						
granny	0.0064	0.0200	0.0027	0.0108	0.0401	0.0800

Table 6.3: An analysis of the overhead incurred by the presented compression scheme.

As expected, the overhead associated with the use of the run mode is directly related to the presence of flat regions in the image. Frequency counts are already employed to model this overhead and it is not clear that substantial benefit could be gained by an alternative modelling scheme. Probably the best way to reduce overhead in this case is to put more stringent limits on when the run mode is entered. This would cut down the number of overly small runs that were encoded.

In summary, although the overhead from forward adaptive elements of the proposed compression method can never be wholly eliminated, reductions could be made by further modelling. Given the level of modelling already employed, it is not obvious that substantial gains could be made; any gains would most likely become apparent for the more complex imagery types, which currently carry more overhead than the simpler types.

6.3 External Comparisons

The culmination of the work described so far, is a Generalised, Extended Lossless Image Compression scheme, which shall now be referred to as GELIC. In order to put this into a useful context, it is necessary to compare the results presented with other schemes from the literature.

An obvious benchmark is JPEG-LS, the new JPEG standard for lossless image compression. The compression results for JPEG-LS, given in Table 6.4, were generated with the LOCO-I/JPEG-LS software (v1.00) provided by Hewlett-Packard labs[1]. The default parameters were used in all cases.

Comparing the results from GELIC and JPEG-LS, we see that the latter is beaten on every test, except *mall*. The poor performance of GELIC on *mall* can be related to the negligible use of extended prediction used for this sequence (99.2% MED usage, 0.8% PP and MC).

In general GELIC shows superior results to JPEG-LS. This is especially true for the colour video and multiview sequences. For *ctflesh* the file size generated by GELIC is less than half that required by JPEG-LS.

Taking colour video as an example, the 0.66 bpp improvement that GELIC provides may not seem a revolutionary amount. However, with many millions of pixels in even a short colour video sequence, this benefit soon becomes large in real terms.

6.3.1 Other Extended Schemes

As the current work makes use of extra correlations, that are not considered in JPEG-LS, the previous comparison may seem unfair. To ensure a fair comparison, GELIC should also be weighed up against schemes which have purposely been extended to consider more than just intraband relationships. As the majority of such schemes have only been developed for colour images, only this part of the test set will be used in the comparison.

[1]Available at http://www.hpl.hp.com/loco/locodown.htm.

Image	GELIC	JPEG-LS
Colour Images		
air2	3.82	4.11
baboon	5.93	6.17
cats	1.91	2.61
cmpnd2	0.99	1.32
house	4.15	4.19
lena	4.45	4.54
Average	**3.54**	**3.82**
Greyscale Video		
claire	2.12	2.32
granny	2.29	3.18
mall	3.73	3.61
mobile	4.35	4.66
salesman	3.75	4.39
Average	**3.25**	**3.63**
Colour Video		
claire	2.24	2.46
football	4.17	4.80
granny	2.09	3.11
mobile	3.88	4.69
susie	3.19	3.77
Average	**3.11**	**3.77**
Multiview		
ctflesh	0.85	2.05
granny	2.55	3.14
skull	2.54	3.17
Average	**1.98**	**2.79**
Colour Multiview Video		
granny	2.02	3.12

Table 6.4: A comparison of the presented scheme versus the JPEG-LS standard for lossless image compression.

Image	GELIC	I-JPEG-LS	I-CALIC	Scan
Colour Images				
air2	3.82	3.90	-	-
baboon	5.93	-	-	-
cats	1.91	1.92	1.81	-
cmpnd2	0.99	1.05	0.92	-
house	4.15	-	-	4.35
lena	4.45	-	-	-

Table 6.5: Comparison of the presented method with several compression schemes in the literature.

The extended version of JPEG-LS and CALIC are good contenders. These schemes are referred to as I-JPEG-LS and I-CALIC respectively in Table 6.5. As executables for these methods are not publicly available, the results presented have been taken from the relevant papers (I-JPEG-LS[MWSM97], I-CALIC[WMC98]). These papers, being published during the course of the current investigation, are certainly contemporary with the presented results in this work. Also included in Table 6.5 is a result from the scan-based method[MS95b] described in Section 3.1.3.

All of the methods used for comparison have been developed with the intention of compressing 3 or 4 band colour images; primarily RGB and CMYK. However, the driving force behind the current work has been to accommodate all natures of imagery with an unified compression scheme. As such, the other methods might be expected to have a slight advantage for colour images.

However, the results in Table 6.5 show that GELIC is able to produce superior compression to some of the schemes aimed specifically at colour images; although I-CALIC is not beaten. In particular, the superior performance of I-CALIC on the *cmpnd2* image could be due to a special binary mode in I-CALIC, aimed specifically at such images.

Results for colour video compression are presented in [MS96], using the *susie* and *football* sequences. However, a comparison is made difficult as these results are presented as a graph of bit-rate against frame number. However, it is worth noting that the results for GELIC presented in Table 6.4, which are averaged over all frames included in the sequence, are less than the minimum bit rate reported in [MS96].

6.4 Summary

Based on the techniques discussed in the previous chapters, a concrete compression scheme named GELIC has been generated for the lossless compression of all forms of imagery in the test set. Analysis of the scheme shows that the five predictors included, are all well used by different items of test imagery. Furthermore, the amount of overhead present in the scheme was determined. Although not generally excessive, this overhead could be further attacked, by more sophisticated modelling, to yield better compression performance.

A comparison with the new lossless standard JPEG-LS showed that GELIC offers clear improvements over standard lossless schemes. A comparison with approaches designed specifically to handle colour images showed GELIC to be comparable, but not the best, of these recent developments. However, when compared to the only known scheme for colour video compression, the results from GELIC were found to be superior.

Chapter 7

Further Work

In the previous chapters of this dissertation, we have seen a thorough study of how lossless image compression can be extended. Extra compression has been obtained by making use of the additional correlations present in colour images, video and multiview imagery. The novel methods presented have been shown to have advantages over existing approaches. However, it would be unreasonable to suggest that no further improvements can be made in the field of extended lossless image compression.

Section 7.1 details some extensions to the work presented so far. Section 7.2 carefully considers an orthogonal step in extending lossless image compression; the use of error resilient coding methods.

7.1 General Improvements

Many possible improvements to the current work are apparent. Indeed, some have already been mentioned, such as ways to reduce overhead in Section 6.2.2. The following three ideas are interesting avenues for subsequent work.

7.1.1 In the Same Context

QTBS was chosen as the preferred predictor selection mechanism, as it was the best of those considered in Chapter 4. However, in Chapter 5 context based methods were found to be superior to the use of quadtree based error modelling. Therefore the scheme presented in Chapter 6 requires the handling of both a quadtree structure and a context model.

Given the results in Chapters 4 and 5, it is clear that the advantage of context coding over quadtree based error modelling, is larger than the benefit of using quadtree based predictor selection instead of contextual methods. Hence, a unified scheme based on contextual predictor selection and error modelling would yield an effective encoding scheme with a more simple implementation than the scheme presented in Chapter 6.

When considering such a scheme, it seems likely that it would have performance at least comparable to that given in Chapter 6. It might perform slightly worse, as it uses a predictor selection scheme that is generally inferior to QTBS. However, some benefit may be gained by using the same context for prediction selection and error modelling.

Another modification that could be made is to keep separate contextual counts for each predictor. This technique was shown to give a small improvement in the interband extension of JPEG-LS[MWSM97]. However, this effectively multiplies the number of contexts by the number of predictors. While a slight benefit was seen for colour images with two predictors in [MWSM97], it is not clear that similar gains would be given by using the technique for colour video, or multiview sequences, which make use of many more predictors.

7.1.2 Expanding the Test Set

Although this work has considered a larger set of test imagery than is usual, more could be done to better characterise the general performance of the methods presented. Hyperspectral satellite images and medical time sequences are particular examples of types of imagery missing from the current test set. Further study with these and other additions to the test set could produce interesting results.

7.1.3 Keeping it Simple

One application of the current work is to archive film libraries. Greyscale and colour video could be efficiently and losslessly stored using the techniques described in previous chapters. However, these techniques have achieved extra compression, over standard greyscale compression techniques, at the expense of some extra computational complexity, whereas a more simple approach could have advantages.

A video archive is likely to be encoded once and decoded many times. Furthermore, such a system would clearly benefit from the ability to decompress the archived video at full frame rate and this in turn would be made most convenient by a scheme with a computationally simple decoder. The parts of the decoding process that add the most computational load are motion compensation, context coding and Arithmetic coding. By eliminating or replacing these elements, the goal of fast lossless playback of video could be realised.

It was shown in Section 4.2 that the combination of MED and PP are superior to MC by itself and only a little behind MED combined with MC. Therefore by using PP instead of MC, some compression performance is traded for extra decoding speed.

To simplify the error modelling and coding stages, we could consider using a forward adaptive modelling scheme with Golomb-Rice coding (see Section 2.5.1). In was shown in Chapter 5 that forward adaptive methods are inferior to backward, context adaptive methods. However, without the overhead of context determination and count keeping, a forward adaptive method can offer less computational complexity for the decoding process.

The preceding ideas were combined to good effect in a scheme presented in [Pen99b]. A block based adaptive method was used to jointly select predictors and error model

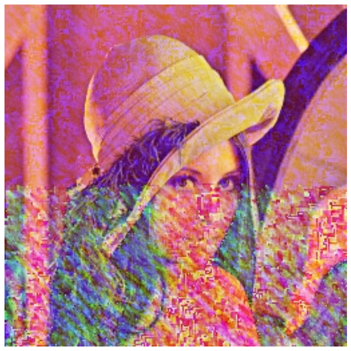


Figure 7.1: A single bit was artificially flipped half-way through the file produced by a GELIC encoder. The disastrous effect of the error on the decoded image can be clearly seen.

parameters. The available predictors were MED, IB1 and PP. The resulting compression was shown to be superior to JPEG-LS in almost all of the sequences compared (*football* being the notable exception), although not as good as the results presented in Chapter 6.

Not only does the scheme presented in [Pen99b] generally offer superior compression compared to JPEG-LS, it has the potential to offer faster decompression, as it does not have the overhead of managing contextual information. This suggests that useful further work could look at simple extended lossless compression schemes that offer superior compression and decoding speed compared to modern lossless greyscale compression.

7.2 A New Direction - Error Resilience

All of the compression schemes discussed so far are designed to operate in a noiseless environment. That is, they require the decoder to receive exactly the data that was sent from the encoder. If a single bit is mis-sent, a lossless decompression will not occur. In fact, even a single bit error can have a dramatic effect, as shown in Figure 7.1.

Hardware is necessarily imperfect and therefore the effect of errors must be handled. This is normally done transparently by the network or storage system being used. In the simplest case, an error detecting check sum accompanies the data. The receiver calculates a check sum from the data received and if that does not match the checksum sent, the data is requested again. More advanced schemes use Forward Error Correcting (FEC) codes[PW72]. These introduce some redundancy into the data such that the original data can still be retrieved if a small number of transmission errors occur.

However, re-reading data is not always productive; for example if storage media is physically damaged. Also the application of FEC codes, while generally an excellent solution, may not always be effective. To be effective it would be necessary to know the error rate of the channel in advance. If the error rate is not known, or is variable, then the error protection will either be too strong (poor overall compression) or too weak (chronic image corruption).

In theoretical terms this is not a problem with a clear solution. Compression aims to remove redundancy, but redundancy is needed to mitigate the effects of transmission errors. The correct balance between compression and robustness to errors will generally depend on the application involved. For example medical imagery and images beamed from far beyond Earth may require special consideration!

Previous research on error resilient compression has often considered lossy video coding [YNL97]. In this case re-transmission would cause objectionable pauses in playback and in any case would not be applicable for broadcast applications. Furthermore heavy FEC coding is generally unacceptable due to the additional bit-rate it would require.

Error resilient lossless image compression could be useful for many applications, such as archiving and remote scientific investigation. In the absence of error, lossless decoding would be achieved. However, in the presence of error the system would offer a *best-effort* decompression, yielding an image close to the original. Although the decompression is not lossless when errors have been introduced, it is hoped that the output of an error resilience lossless decoder would be significantly better than that of normal schemes, as exemplified in Figure 7.1.

By considering the effects of transmission errors in the context of lossless image compression, a better understanding can be gained of the issues that must be addressed to enable error resilient lossless image compression. Furthermore, the range of options to tackle the problems of transmission errors can be appreciated.

7.2.1 Lossless Compression Techniques and Transmission Errors

In Chapter 2, the notion of a lossless compression scheme as a combination of mapping, modelling and coding was introduced. We shall now consider the effects of transmission errors on each part of the compression process. As this process is inherently sequential, an incorrectly decoded pixel can lead to problems for all following pixels. Hence, there will be a particular focus on ways in which a single error can propagate to affect the decoding of many pixels.

Image Mapping

If a transmission error has caused a pixel value to be incorrectly decoded, any prediction based on that pixel will lead to error propagation. This is particularly disastrous if a predictor such as MED is used. As MED uses W, N and NW for its prediction, any pixel decoded in error can propagate this error to all pixels below and to the right of the initially affected pixel.

A solution to this is to re-synchronise the predictor by sending raw pixel values at predefined intervals. For example, if the first pixel in every row is sent in raw format and if W is used as the prediction for X, transmission errors can only propagate to the end of the current row.

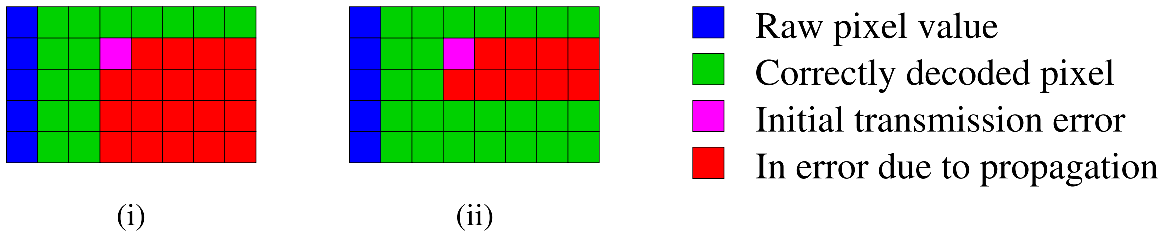


Figure 7.2: (i) MED is used for prediction throughout and a single transmission error propagates to all pixels below and to the right of the originally affected pixel. (ii) MED and W are used for prediction on alternate lines (top line MED). An error occurs when W is being used and this propagates to the end of the line. The following line uses MED, which consults the line above and hence the error continues to propagate. The fourth line shown uses W and hence no further error propagation occurs.

However, the performance of using W is much lower than MED. As a compromise we could consider using W and MED as predictors on alternate lines of the image. A line using W as the predictor would prevent the downwards spread of mis-predictions due to errors that MED would allow, as illustrated in Figure 7.2. By varying the ratio of lines using W to lines using MED, the degree of error propagation can be traded against prediction performance.

Another solution to this problem, as presented in [MKC97], is to split the image into small rectangular regions. Predictors are then employed that only use previous pixel values from inside the current region. This ensures that an incorrectly decoded pixel can not cause error propagation outside of its region.

Another issue with the raster-scan based prediction that has been pursued earlier in this work, is that of data truncation. If only half the data is received, only the first half of the image can be decoded. An alternative is to use a hierarchical image representation that allows for progressive transfer. In that way a low resolution representation of the whole image could be had if some fraction of the original data were received. One such representation is the S+P-Transform mentioned in Section 2.6.2.

Another effect of using a hierarchical image representation is that errors no longer propagate spatially, instead they propagate to lower levels of the representation. This can be used to good advantage if the decoder has some error detection capabilities. If an error is detected, no further decoding of any affected pixels should take place. This means that an error results in a region that is only decoded to a coarse resolution. This would seem to have advantages over a region that is left blank or contains artefacts caused by the initial error.

Modelling

As was shown in Chapter 5, context based modelling of prediction errors is highly effective for lossless coding. However, context determination depends on pixels local to the current location and if one of those pixels had been erroneously decoded, an incorrect context determination could take place. Furthermore, contextual statistics depend on a potentially large, disconnected set of previous pixels. A pixel decoded in error would compromise these statistics for all subsequent pixels in the same context.

These issues are very serious and indicate that context based modelling is not a technique that is applicable to an error resilient application of lossless image coding. An alternative is to use forward adaptive error modelling. This approach removes any dependence on the correct decoding of all previous pixel values. However, the error modelling decision of the encoder must somehow be protected during transmission if this scheme is to be robust.

As shown in Chapter 5, forward error modelling performs less well than context coding. However the difference in performance is not large enough to outweigh the possible advantages, as mentioned, in an error resilient application.

Symbol Coding

A transmission error will obviously cause the prediction residual involved to be decoded incorrectly. Worse still, it may cause the symbol coder to lose synchronisation. That is, it may read too many or too few bits from the incoming bit stream. As a consequence, when it starts to decode the next symbol, it will be starting from the wrong place in the input bit-stream and another incorrect decoding is almost certain.

Clearly there is a large potential for error propagation in the symbol coding stage. Unfortunately, it is a consequence of using self-terminating, variable length codes. Although using a system based on fixed length codes would alleviate the issue, compression performance would fall as a result.

To solve this problem, the image compression scheme needs to be able to re-synchronise the decoder in the event of an error. This can be achieved by dividing the prediction residuals into regions. For each region an index into the bit-stream is then provided, in some robust manner. The decoder can then be sure of starting each region of residuals from the correct place in the input bit-stream.

As described, this solution requires extra overhead for the re-synchronisation indices. As these must be sent robustly, they will require some form of FEC coding to protect them. However, other solutions to this problem have been proposed, such as the Error Resilient Entropy Coder (EREC)[RK96] that manages this re-synchronisation with practically no overhead at all.

7.2.2 Possible Solutions

By considering the discussion above, an initial design for an error resilient lossless image compression scheme can be formulated. Hierarchical image representations have been shown to have advantages over standard predictive mapping methods and so the S+P Transform (Section 2.6.2) makes a good choice for the mapping stage. The error modelling can be accomplished with a forward adaptive, block based scheme that is very similar to

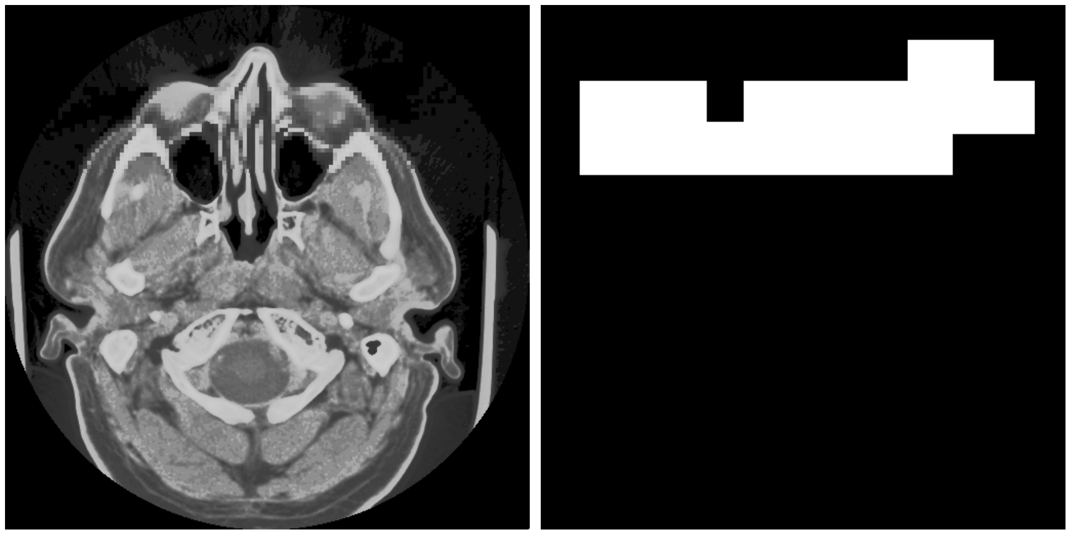


Figure 7.3: Left: The medical image *ct* decoded over a noisy channel. Right: The proposed scheme marks regions known to be decoded only to a coarse resolution due to errors in transmission.

the quadtree modelling described in Section 5.4. By using such a scheme, the residuals from the S+P Transform are partitioned into blocks, which can also be used for the re-synchronisation of the coding stage.

Each block requires a header that contains the length of the block (so the start of the next block is known to the decoder) and the error modelling parameter chosen by the block based error modelling method. To ensure robustness, the header for each block must be protected by FEC coding.

An implementation of the above scheme was presented in [Pen99a]. Golomb-Rice codes were used for the symbol coding stage and Golay codes[PW72] were used for the FEC coding. Error detection was based partly on whether the decoder used the correct number of bits when decoding a block and partly on parity bits included in the block header.

The compression obtained was found to be comparable to LJPEG, but inferior to JPEG-LS. The error detection was used to prevent decoding of both information found to contain transmission errors and any subsequent image data that depended on error contaminated data. As a result, practically no image artefacts related to noise were observed; transmission errors instead resulted in regions that were decoded to only a coarse resolution. Figure 7.3 shows an example of this.

While the results of this work were promising, it was found that at the error rates for which the above scheme returned a reasonable image, a conventional lossless image coder followed by FEC coding was generally able to offer lossless decoding at roughly the same bit-rate. Clearly further work will be required if this gap is to be bridged.

7.2.3 Further Extensions

Making progress towards an extended lossless image compression scheme that is error resilient, is likely to be difficult. The methods discussed earlier in this work for exploiting spectral, temporal and interview correlations, alongside spatial redundancy, necessarily increase the potential for error propagation. For example, an incorrectly decoded value can now lead to pixels in the same band and future bands, frames and views all being incorrectly predicted.

One solution to this would require an extended hierarchical representation. Armed with a representation that enabled multiple resolutions in the spatial, spectral, temporal and interview dimensions, the approach discussed above could be employed.

However in the event of data truncation, some applications may prefer to have some regions at high resolution, than to have the whole image at a potentially lower resolution. For example remote sensing applications may require the full resolution of the image data in order to identify ground based features from space. Another example is digital document storage; where in the event of an error, it would be better to have most of the text legible rather than all of the text at an unreadably low resolution. Such high resolution data sets are often very large and previous research has considered ways to provide multiple access points to such imagery, hence allowing image fragments to be independently decoded[AF97]. If the meta-data required to provide these multiple access points is stored with FEC coding, then the same access points can also provide means for re-synchronisation in the event of error. If this is coupled with error detection, to signal image regions affected by error, the overall scheme also has the advantage of providing data guarantees.

Such error detection can even be integrated into the coding process. An ingenious idea presented in [KCR98] is to include an error symbol in the alphabet for an Arithmetic coder. This symbol is never encoded, but if it is decoded the decoder knows a transmission error has occurred. By varying the probability assigned to this error symbol the degree of error detection can be varied arbitrarily. Results given in [KCR98] show this to be a more effective means of error detection than traditional cyclic redundancy checks.

7.3 Summary

The work presented in this dissertation has offered many new ideas for lossless compression of modern image types. However, it seems likely that by following some of the recommendations presented in Section 7.1, still better compression performance, or decoding speed, could be obtained.

The goal of error resilient lossless image coding has been discussed in Section 7.2. This feature would enable greater use of compression for reliable archiving and transmission of image data.

Chapter 8

Contributions and Conclusions

In this chapter the primary contributions made to the state-of-the-art in lossless image compression are outlined. This is followed by a summary of the conclusions drawn from this work as a whole.

8.1 Contributions

By considering state-of-the-art lossless image compression techniques, alongside methods that normally accompany lossy schemes (e.g. motion and disparity compensation), the literature review presents a unique window on previous knowledge, with a greater scope than has been seen before. Hence, Chapters 2 and 3 constitute an excellent starting point for any future research into a broad range of image compression techniques.

Predictor selection for greyscale and colour images has been previously studied in [MS95c] and [MWSM97, WMC98] respectively. However, never before has predictor selection been thoroughly studied in a way that considers spatial, spectral, temporal and interview correlations. This study produced a forward adaptive method using a quadtree structure for predictor selection, that was found to perform better than any alternatives considered.

Error modelling was also studied in an extended context. It was shown that extended error bucketing offers a simple and yet effective way to make extra use of prediction errors to gain improved context determination. Other error modelling techniques were also considered in this study. In particular the notion that predictor selection with access to intraband, interband, interframe and interview predictors can lower the contextual bias of prediction residuals, is a useful result.

By combining the best of these techniques, a novel extended lossless image compression scheme named GELIC has been presented. GELIC is able to efficiently deal with a wider range of image types than previously considered by a single image compression scheme. The results from GELIC are on average significantly better than the current JPEG standard for lossless image compression, JPEG-LS, which considers only spatial correlations. This shows that by additionally considering spectral, temporal and multiview correlations, compression benefits for lossless image coding can be gained. GELIC's results are also comparable with the most recent methods aimed specifically at colour image compression. For lossless compression of video and multiview sequences, GELIC currently has no real competition.

Finally, the ideas presented in the previous chapter offer some preliminary results and a variety of options for further study, that could further enhance the field of extended lossless image compression.

8.2 Conclusions

The results of this work show that unconditional use of non-spatial predictors does not lead to effective use of extended correlations. This gives the conclusion that some form of predictor selection is necessary to give extended lossless compression that is both efficient and flexible.

By using prediction errors outside the current band/frame, context determination can yield more appropriate contexts and hence deliver improved compression performance through better error modelling. However, this benefit is rather small. This suggests that the predictor selection scheme developed is able to remove most of the extra correlation present in the advanced imagery types considered. The residual higher-order correlations, of the sort exploited by error modelling, appear to be little more than those left after traditional spatial prediction. This leads to the conclusion that significant gains based on extended error modelling are unlikely, unless radically new insights into the nature of higher-order residual correlations can be gathered.

The results given in Chapter 6 support the conclusion that worthwhile improvements can be made to lossless image compression schemes, by considering the correlations between the spectral, temporal and interview aspects of image data, in extension to the spatial correlations that are traditionally exploited. This is in accordance with the thesis proposed at the beginning of this dissertation.

Based on the ideas given in the previous chapter, it can also be concluded that challenges remain in the field of extended lossless image compression. Further investigation is likely to yield still improved compression and hence the field is still worthy of research.

Appendix A

Common Symbol Coders

Probably the most widely known example of a symbol coder is Morse code. Designed by Samuel Finley Breese Morse in the 1840's, Morse code uses the frequencies of letters in English to efficiently encode text. Common letters such as e $(\cdot)$ and t $(-)$ have short codes, while less common letters have longer codes, for example Q $(- - \cdot -)$.

For use with digital systems, the output alphabet of a code should be only 0s and 1s. Although it appears binary, the output alphabet of Morse codes uses three symbols; $\cdot$, $-$ and an inter-symbol space. The inter-symbol space is needed to indicate the end of a letter, for example after receiving a $\cdot$ it is not clear if this represents a single e or if it is the start of an s $(\cdots)$. One way to eliminate the need for an inter-symbol space is to construct *prefix* codes. A prefix code has the property that no code word is the prefix of another code. Hence, once a codeword has been read, the original input symbol can be unambiguously identified without reading anymore information. This is called instantaneous decoding and is a highly desirable property for a symbol coding scheme.

Image compression schemes tend to be innovative in the mapping and modelling stages. Whereas the coding stage of most image coders is generally based on a traditional coding technique. Probably the three most influential of these coding schemes are Huffman Coding, Arithmetic Coding and the Lempel-Ziv based methods. All of these schemes are documented in any good book on data compression[Nel91], however due to their importance, these three methods will be briefly detailed here.

A.1 Huffman Coding

In 1952 Huffman gave an algorithm that produces an encoding that is optimal under certain conditions. The algorithm requires the probabilities of all the symbols in the input alphabet to be known beforehand. The input symbols are then sorted according to their probabilities. The two least frequent symbols become the leaves of a binary tree. The tree replaces the two symbols in the list and is given the sum of the probabilities of its leaves as its probability. This procedure is repeated until there is only one element left; a binary tree with the all input symbols as leaves. By labelling the paths from the root of the tree

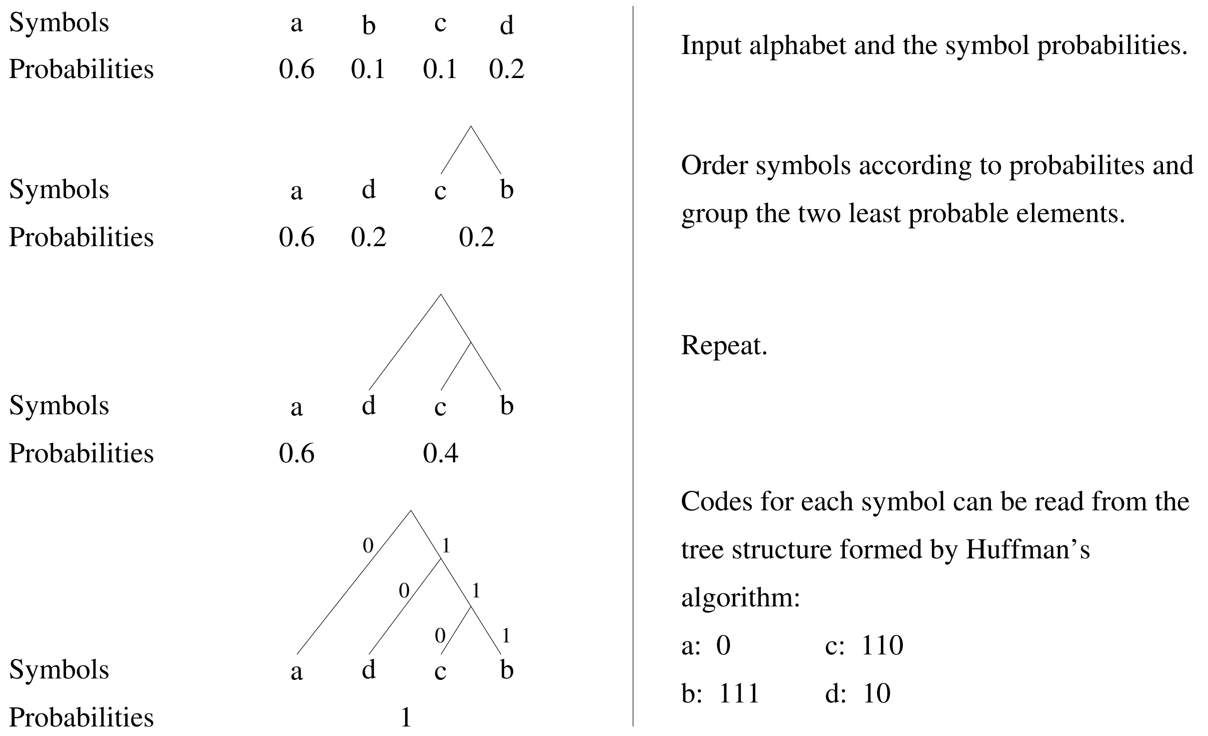

Symbols a b c d
Probabilities 0.6 0.1 0.1 0.2
Input alphabet and the symbol probabilities.
Symbols a d c b
Probabilities 0.6 0.2 0.2
Order symbols according to probabilites and group the two least probable elements.
Symbols a d c b
Probabilities 0.6 0.4
Repeat.
Symbols a d c b
Probabilities 1
0
1
0
1
0
1
Codes for each symbol can be read from the tree structure formed by Huffman's algorithm:
a: 0 c: 110
b: 111 d: 10

Figure A.1: An example of constructing a Huffman code.

to the leaves with 0s and 1s a binary codeword can be given to each input symbol. An example of this is given in Figure A.1. Decoding a codeword can be seen as traversing the tree from the root to a leaf, using the bits of the codeword to determine which branch to follow at each node.

It should be noted that Huffman's algorithm builds codes on a bottom-up basis. Shannon-Fano coding, which slightly pre-dates Huffman's work and proceeds in a top-down fashion, can produce codes that are inferior to those produced by Huffman's algorithm.

Huffman codes are only optimal if the probabilities of all the input symbols are an integer power of 1/2; which is an unusual occurrence. Furthermore, Huffman codes are bound to have a minimum length of one bit. As a consequence, for coding distributions in which the most probable symbol has a probability much greater than 1/2, considerable inefficiency may occur from using Huffman coding.

A second drawback of Huffman coding is the relative difficulty in adapting it to changing input statistics. Although it is conceptually simple to rebuild the code tree whenever the source statistics change, it is computationally expensive to do so.

In spite of its drawbacks, Huffman's algorithm generally produces good codes, is simple to implement and is not covered by any patents. As a consequence of these facts (especially the later) it is widely used.

A.2 Arithmetic Coding

Arithmetic coding removes the limitations of Huffman coding by considering not individual symbols but the whole input message. It aims to represent the sequence of input symbols as a pointer to a region in the interval $0 \rightarrow 1$. The length of the encoded message is determined by the precision required to record the necessary region. The size of the region and hence the required precision, is directly related to the probability of the message.

In order to specify the region that the message lies in, two variables *low* and *high* are used. These record the low and high values of the range in which the message lies. Initially, *low* and *high* have values of 0 and 1 respectively. As input symbols are processed by the encoder, *low* and *high* are adjusted to refine the message's range, based on the probabilities of the symbols already seen.

To see how this works, an example is useful. Consider an input alphabet with four symbols, $\{a, b, c, d\}$, each with the same probabilities as the example in Figure A.1. The interval $0 \rightarrow 1$ is now divided up so that each symbol has a share proportional to its probability, as illustrated in Figure A.2.

Now suppose the message to be coded is 'aabd'. After processing the first symbol, *low* and *high* will have values of 0 and 0.6 respectively. The symbol probabilities are now scaled by the size of the range demarcated by *low* and *high*, before the next symbol is processed. Hence, the second 'a' in the message causes *high* to be updated to 0.36, while *low* remains at 0. This process is continued until the whole input message is encoded. As shown in Figure A.2 the final values of *low* and *high* are 0.2448 and 0.252 respectively. Hence the message 'aabd' is coded by any number in between these two values. Unfortunately, so is any other message that starts 'aabd...'. In order to decode the correct message the encoder and decoder must either agree on the length of the message or a special terminator symbol must be used. As the decoder will know the dimensions of the image being processed, the first option can be used for image compression applications.

The operation of the decoder is fairly simple; given a value from the encoder, it can simply read off the next symbol in the message until the correct number of symbols have been read. For example, the encoder could chose to send 0.25 to encode 'aabd', with an agreed message length of 4. As $0 \leq 0.25 < 0.6$, the first symbol in the message is taken as 'a'. Similarly, the rest of the message can be decoded.

Arithmetic coding as described so far, seems fairly simple. However, there are a number of technical issues with implementing the scheme as it is given above. The main problem is that the precision with which *low* and *high* must be stored increases without bound as the message gets longer. This problem can be removed by registering digits that *low* and *high* have in common. In the example above, after encoding the 'b', *low* and *high* have values 0.216 and 0.252 respectively. At this stage, the encoder knows that the encoded message must start with 0.2 and so this can be transmitted to the decoder. Having done this, the active range can be re-scaled to $0.16 \rightarrow 0.52$ by both the encoder and the decoder. In this way, the precision required to support Arithmetic coding can be kept within the limits of practical computing hardware. Further subtleties, such as avoiding underflow and overflow

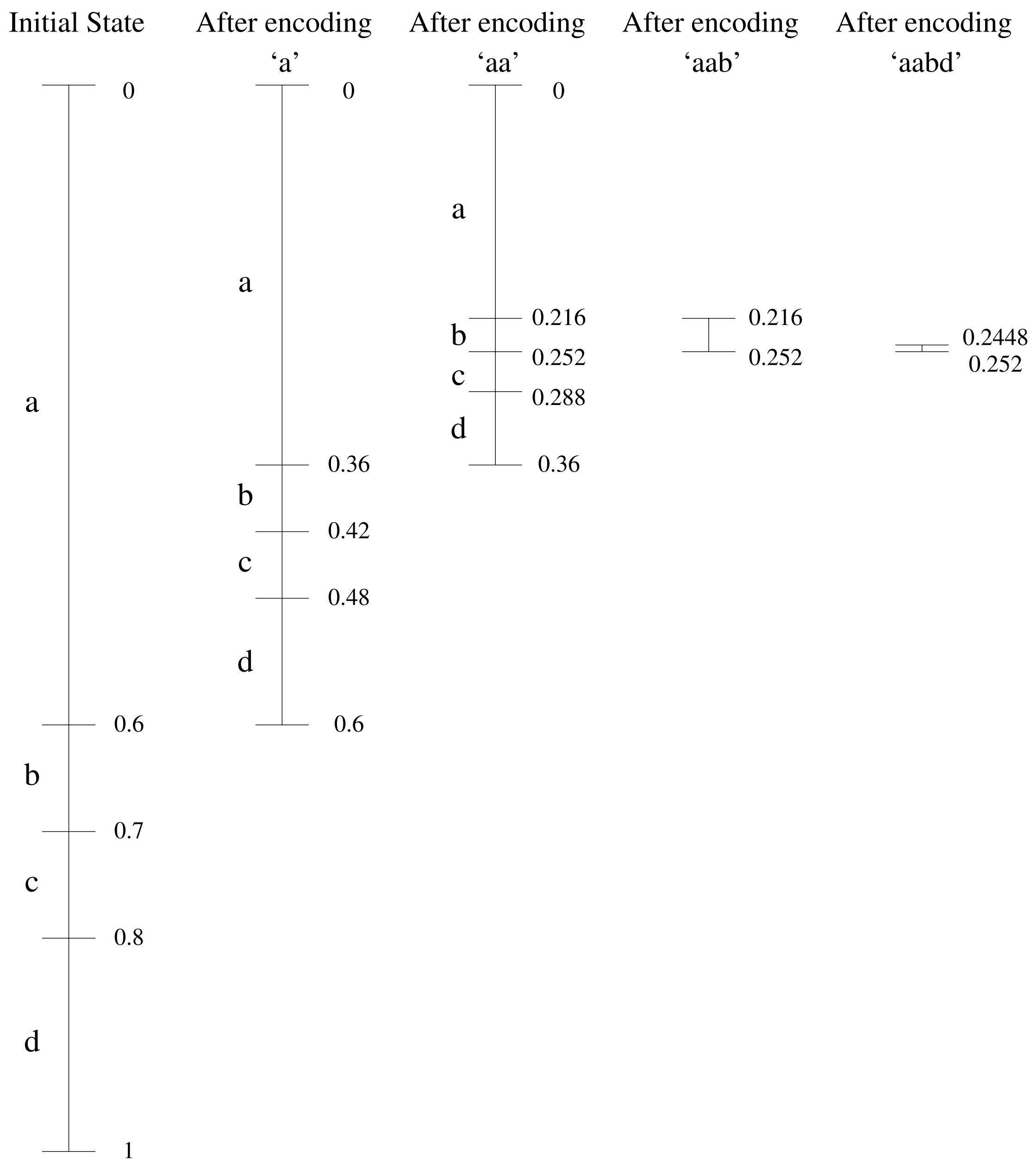


Figure A.2: An example of the Arithmetic coding process.

are left to more detailed texts[BCW90].

As well as coding efficiency, another benefit of Arithmetic coding is that it can adapt to changing statistics in a simple way. The symbol probabilities can easily be changed at each stage in the coding process, as long as the encoder and decoder adapt their symbol probabilities in the same manner. However despite its benefits, due to perceived implementation difficulties, computational complexity and a large number of patents, Arithmetic coding is not as widely used as it could be.

A.3 Lempel-Ziv Based Methods

Whereas Huffman and Arithmetic coding aim to store a symbol, or sequence thereof, using a variable number of bits, Lempel-Ziv methods aim to store a variable number of input symbols using a fixed number of bits. Abraham Lempel and Jakob Ziv originally suggested two schemes in 1977 and 1978, however the following discussion will focus on the later (LZ78) family of methods.

LZ78 is a dictionary based coding scheme. It replaces common sequences of symbols with an index into a dictionary. The interesting part is the build-up of the dictionary. Initially, the dictionary contains only single input symbols. As the input data is read, new symbol strings are added to the dictionary as they are encountered. A new string is a sequence of symbols that appears in the dictionary, with an additional final symbol, such that the whole string is not in the dictionary. Importantly though, on encountering a new string, the last symbol is not coded as part of that string. This necessarily leaves a known string that is in the dictionary. The last symbol from the new string is then the first symbol in the next string. Hence, by using backward adaptation the decoder can build the same dictionary as the encoder and the dictionary never needs to be explicitly sent.

Although LZ based methods are widely used in general data compression schemes, they are not frequently used in lossless image compression schemes. The reason for this, is that the noise inherent in natural images does not allow enough exactly repeating sequences of symbols to gain adequate compression.

Appendix B

Image Material used for Testing

The reduction in data size obtained by lossless image compression is fundamentally limited by the information content of the source image. Hence, results from lossless compression methods are only relevant to general performance, if the test set is a suitable sample of the imagery to be compressed.

Given the scope of image types this work intends to address, it is hard to envisage a *fully* comprehensive test set. However, efforts have been made to try and ensure a reasonable variety in the test imagery used. For the colour images there are examples of natural scenes, satellite imagery and computer generated images. The video sequences used include both slow and fast moving scenes and throughout the test set there is a mixture of low and high resolution items.

What follows is a brief commentary on every item in the test set that was used for the results in Chapters 4, 5 and 6. For reasons of space, the video sequences have only the first, middle and last frames shown. Likewise for multiview sequences only the far left, middle and far right views are shown. Also, all images have been reduced to some extent prior to display.

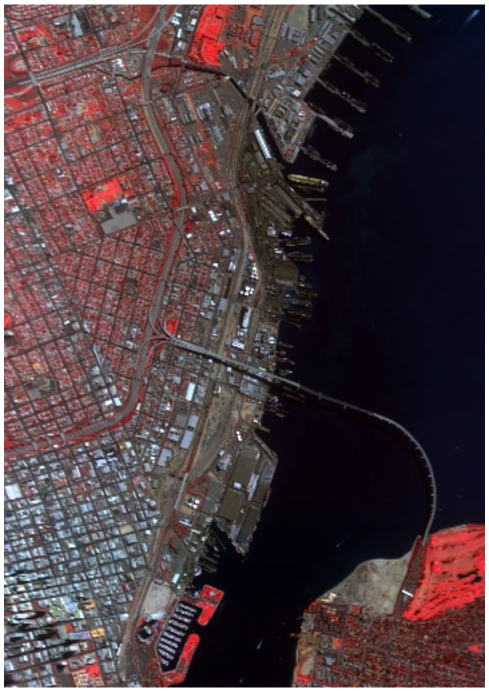


Figure B.1: **Air2 - (720 $\times$ 1024, 24bpp)** This is an often used example of an aerial image.

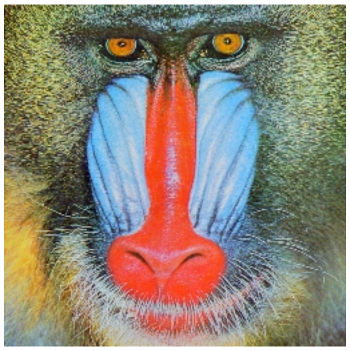


Figure B.2: **Baboon - (512 $\times$ 512, 24bpp)** A popular close-up of a baboon's face. The fine detail of the fur, makes for a very noise like image.

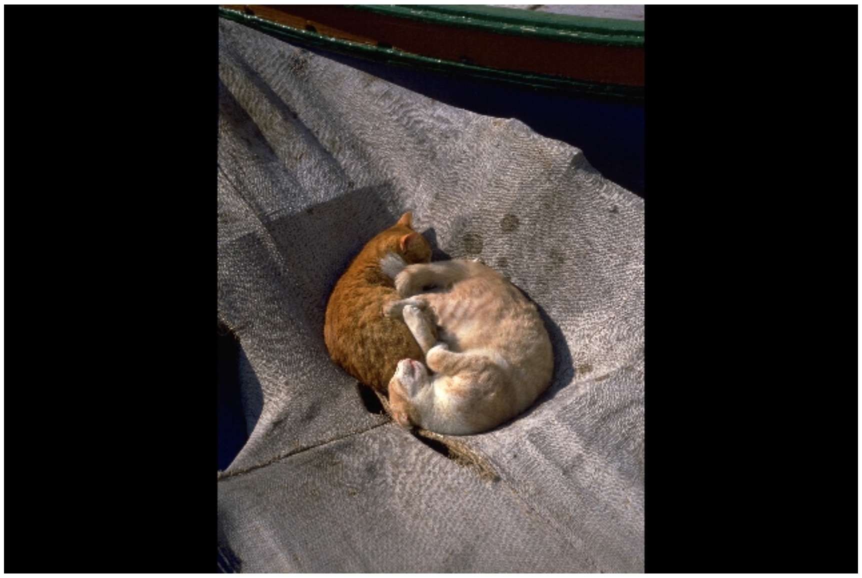
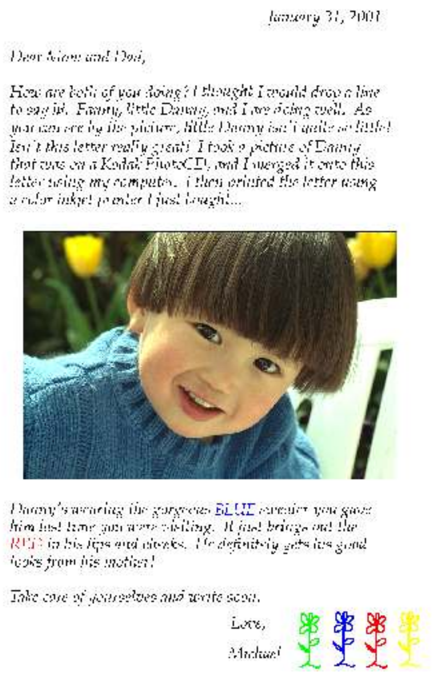


Figure B.3: **Cats - (3072 × 2048, 24bpp)** A high resolution image of two sleeping cats. The image includes large black borders at either side.



January 31, 2001

Dear Mom and Dad,

How are both of you doing? I thought I would drop a line to say hi. Fanny, little Danny, and I are doing well. As you can see by the picture, little Danny isn't quite so little! Isn't this letter really great! I took a picture of Danny that was on a Kodak PhotoCD, and I merged it onto this letter using my computer. I then printed the letter using a color inkjet printer I just bought...

Danny's wearing the gorgeous BLUE sweater you gave him last time you were visiting. It just brings out the RED in his lips and cheeks. He definitely gets his good looks from his mother!

Take care of yourselves and write soon.

Love,

Michael

Figure B.4: **Cmpnd2 - (1024 × 1400, 24bpp)** An often used example of a compound document. Most of the image is black and white (binary) text, but a full colour photograph makes up the central portion of the image.

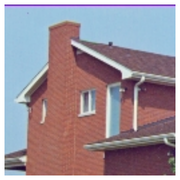


Figure B.5: **House - (256 × 256, 24bpp)** The side of a house.

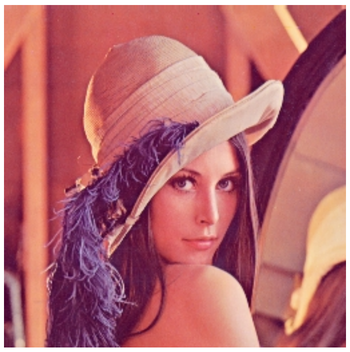


Figure B.6: **Lena - (512 × 512, 24bpp)** The ubiquitous Lena image is a tasteful section from a 1972 Playboy article. It has long been popular with the image processing and compression community, because of its mixture of smooth, texture and edge regions. This popularity has been self-sustaining, as the image is often used for comparison, because of its use in previous works.

Figure B.7: **Claire - (360 $\times$ 288, 168 frames, 8bpp)** A very frequently used head and shoulders video clip.

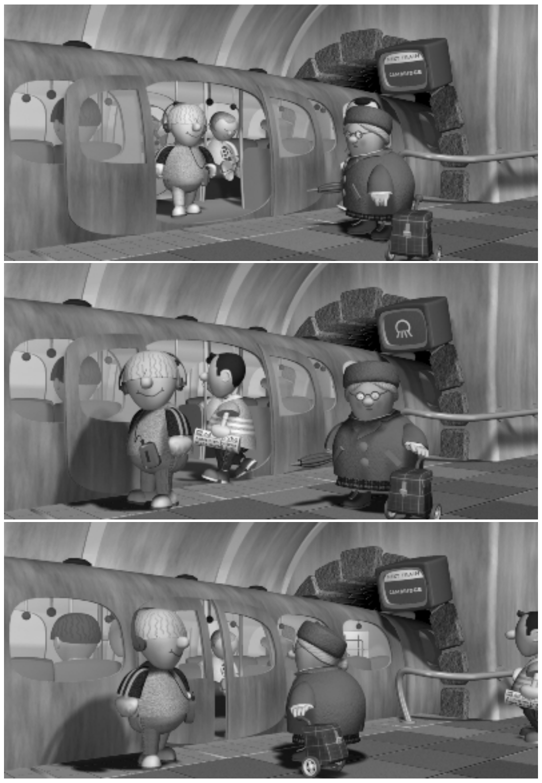


Figure B.8: **Granny - (800 × 384, 100 frames, 8bpp)** The Granny sequence was rendered for multiview display. It was produced by Pictures on the Wall, and is used with permission of Autostereo Displays Ltd. Here the central view has been converted to greyscale and is used as a video sequence. The middle 100 frames, from the full 300 frame sequence were used. In this range, there is motion of some foreground objects.

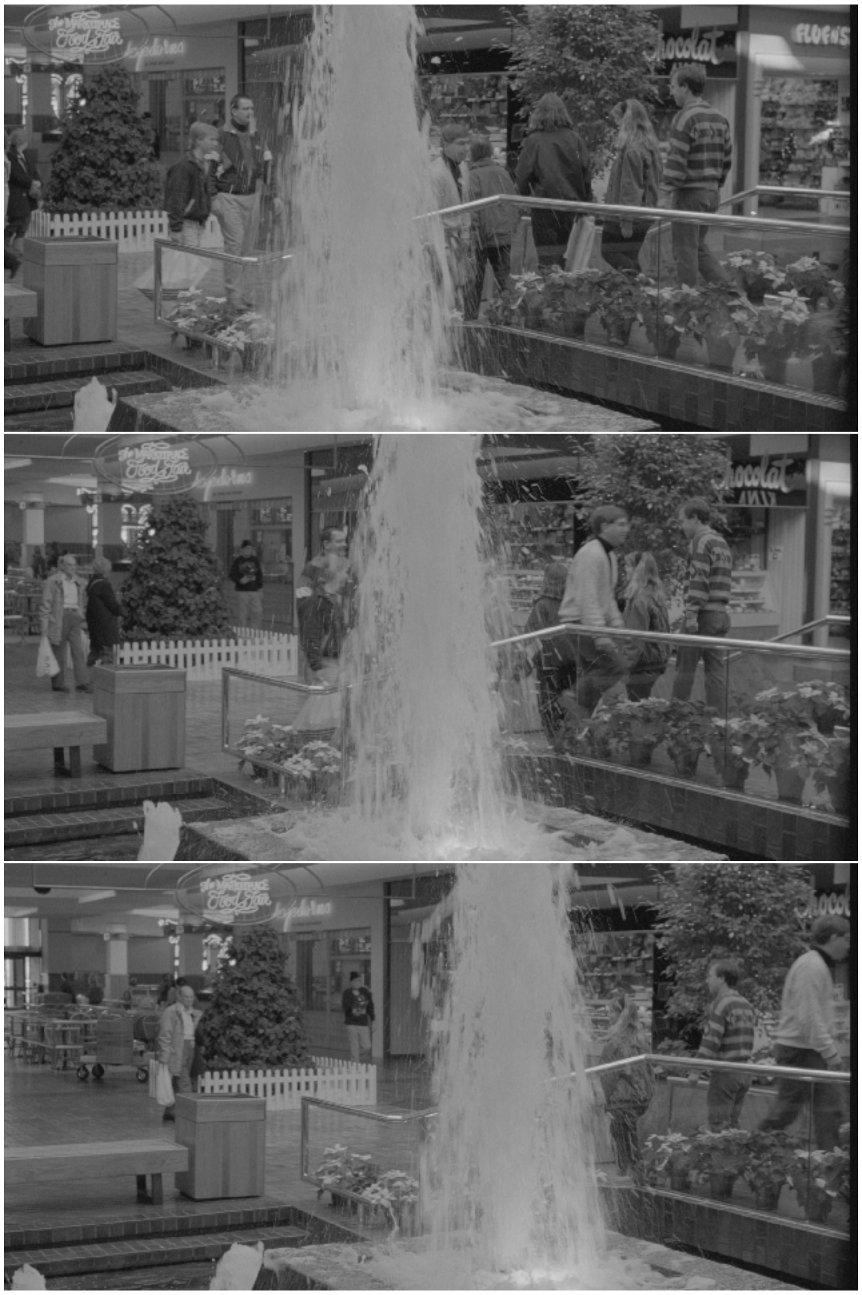


Figure B.9: **Mall - (2048 $\times$ 1024, 50 frames, 8bpp)** A high resolution scene in a shopping mall. The footage contains a lot of motion; people walking, camera pan and a large central water fountain.

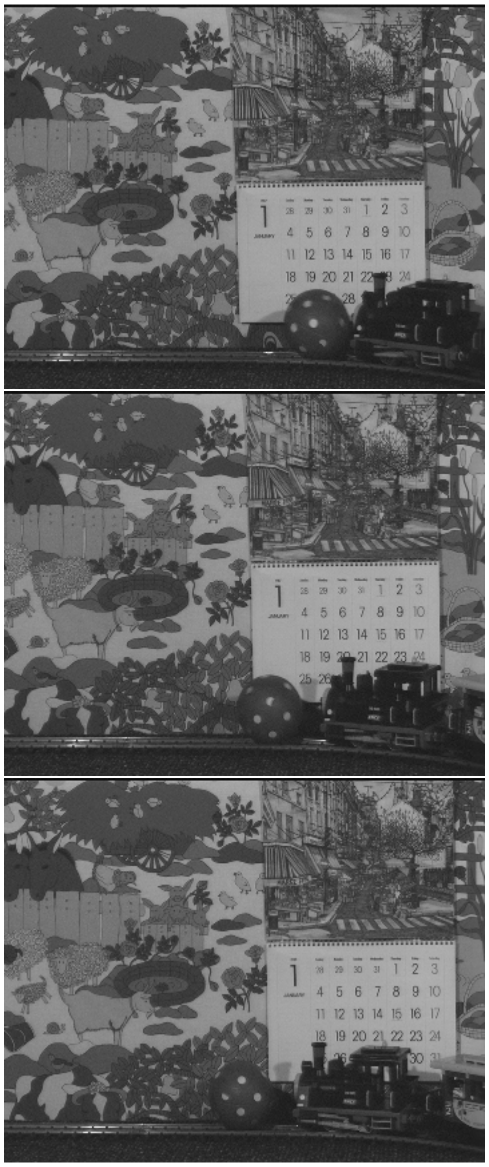


Figure B.10: **Mobile - (720 × 576, 40 frames, 8bpp)** This sequence contains motion of several independent objects and a slow camera pan.

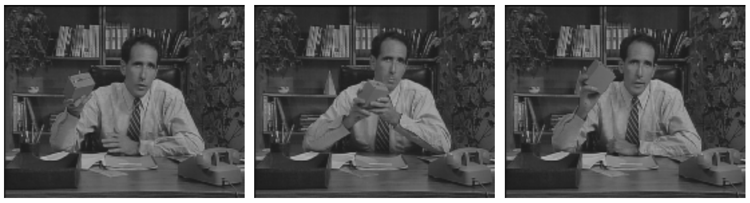


Figure B.11: **Salesman - (360 × 288, 100 frames, 8bpp)** A salesman sat behind a desk, gesticulating as he practices his art.

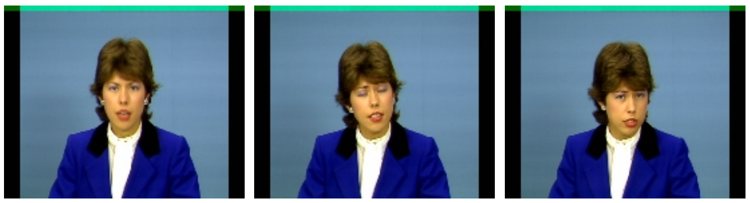


Figure B.12: **Claire - (360 × 288, 168 frames, 24bpp)** As above, but in colour.

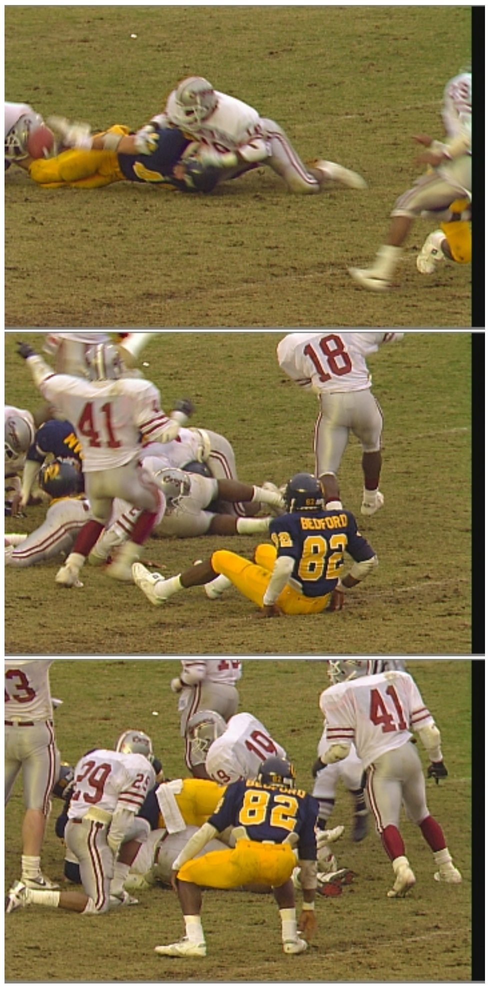


Figure B.13: **Football - (720 × 486, 97 frames, 24bpp)** An action sequence from an American football game. The footage is affected by de-interlacing artefacts.

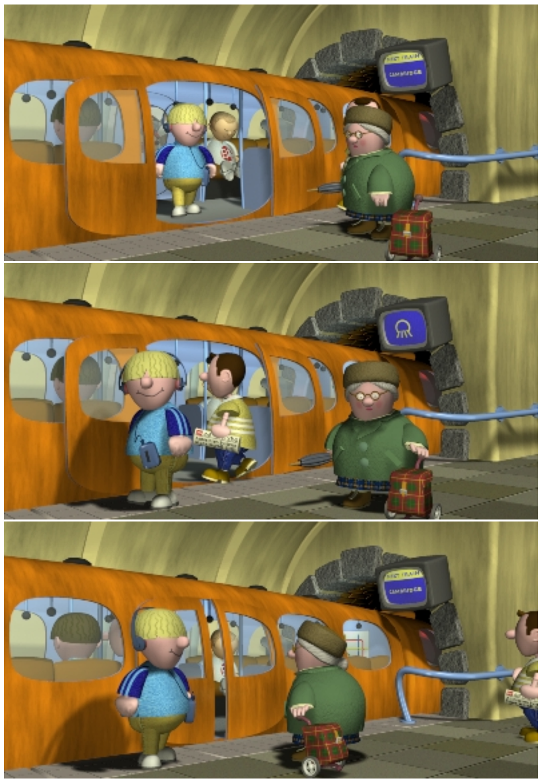


Figure B.14: **Granny - (800 $\times$ 384, 100 frames, 24bpp)** As above, but in colour.

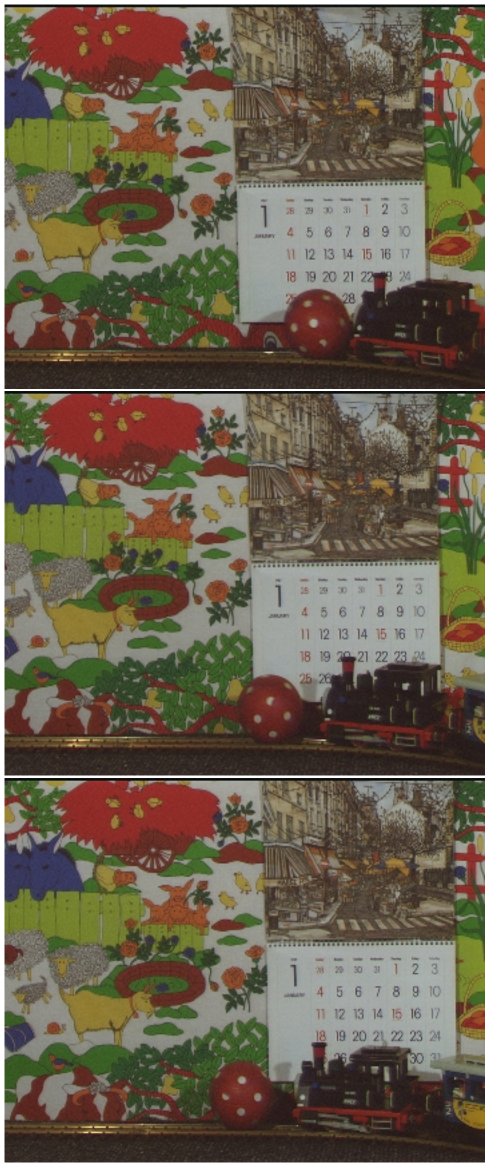


Figure B.15: **Mobile - (720 $\times$ 576, 40 frames, 24bpp)** As above, but in colour.

Figure B.16: **Susie - (720 $\times$ 480, 150 frames, 24bpp)** A relatively high resolution head and shoulders clip.

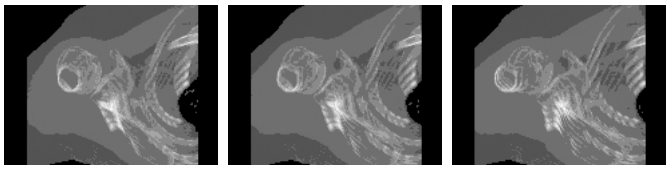


Figure B.17: **CTFlesh - (320 $\times$ 240, 16 views, 8bpp)** Raw CT data was rendered into a series of views for 3D display. The rendering used thresholds to highlight the flesh aspect of the scan, although the bone structure is quite visible.

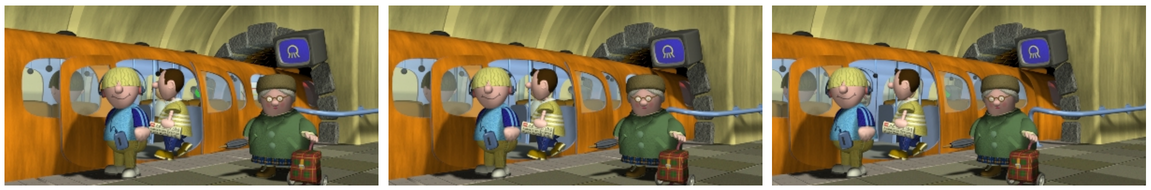


Figure B.18: **Granny - (800 $\times$ 384, 10 views, 24bpp)** The central ten views, taken from the temporal mid-point of the Granny sequence.

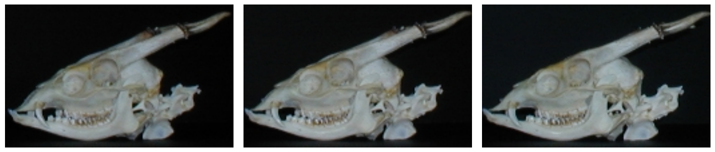


Figure B.19: **Skull - (342 $\times$ 214, 9 views, 24bpp)** Nine views of an animal skull, taken for autostereo display. The sequence suffers from slight vertical mis-alignments, incurred during the manual part of the capture process.

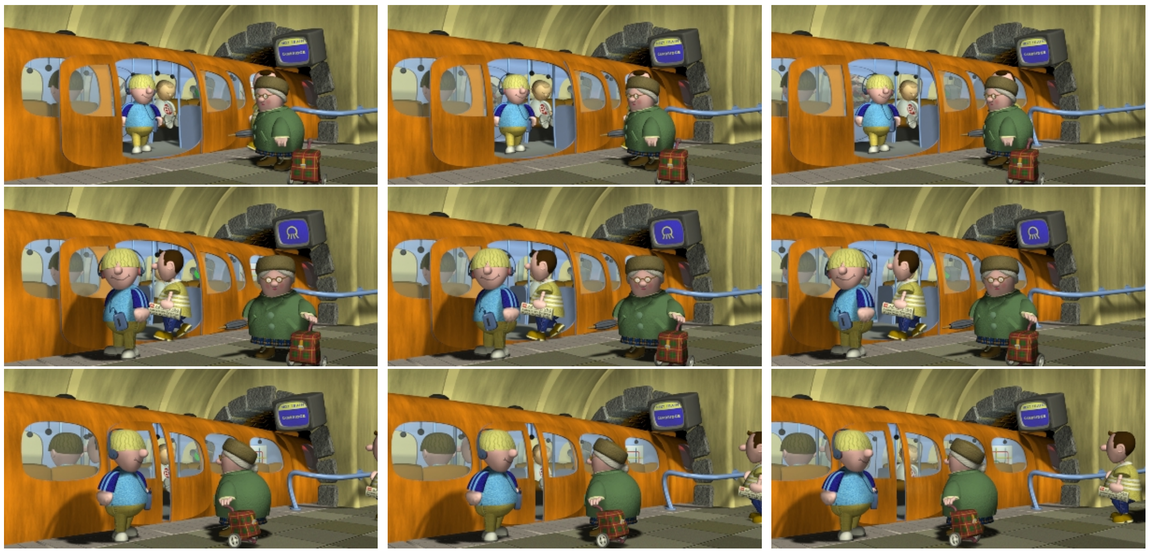


Figure B.20: **Granny - (800 $\times$ 384, 10 views, 100 frames, 24bpp)** A sequence comprising the central ten views, and one hundred frames from the Granny sequence.

Glossary

bit	Contraction of *bi*nary dig*it*.
bpp	Bits Per Pixel
DCT	Discrete Cosine Transform
gigabyte (GB)	$2^{30} \approx 10^9$ bytes.
GOP	Group Of Pictures
JPEG	Joint Photographic Experts Group.
kilobyte (KB)	$2^{10} = 1024 \approx 10^3$ bytes.
MAE	Mean Absolute Error
MAP	Median Adaptive Predictor
megabyte (MB)	$2^{20} = 1048576 \approx 10^6$ bytes.
MED	Median Edge Detection
MPEG	Moving Picture Experts Group
MSE	Mean Square Error
terabyte (TB)	$2^{40} \approx 10^{12}$ bytes.

Bibliography

[197] ISO/IEC JTC 1/SC 29/WG 1. CD14495, lossless and near-lossless coding of continous tone still images, May 1997. DRAFT release.

[Abo95] G. Abousleman. Compression of hyperspectral imagery using hybrid dpcm/dct and entropy-constrained trellis coded quantization. *Proc. Data Compression Conference*, pages 322–331, March 1995.

[AF97] E. Ageenko and P. Fränti. Storage system for document imaging applications. *Proc. International Picture Coding Symposium, Berlin*, pages 361–364, September 1997.

[AMH97] G. Abousleman, M. Marcellin, and B. Hunt. Hyperspectral image compression using entropy-constrained predictive trellis coded quantization. *IEEE Trans. on Image Processing*, 6(4):566–573, April 1997.

[APL98] S. Assche, W. Philips, and I. Lemahieu. Lossless compression of pre-press images using a novel color decorrelation technique. *Proc. SPIE Very High Resolution and Quality Imaging III*, 3308:85–92, January 1998.

[BCW90] T. Bell, J. Cleary, and I. Witten. *Text Compression.* Prentice Hall, 1990.

[Bie88] M. Bierling. Displacement estimation by hierarchical blockmatching. *Proc. SPIE Visual Communications and Image Processing*, 1001:942–951, January 1988.

[BM97] M. Baker and A. Maeder. Region-based motion compensation for video compression using a quadtree approach. *Proc. International Picture Coding Symposium, Berlin*, pages 585–590, September 1997.

[Cla95] R. Clarke. *Digital Compression of Still Images and Video.* Signal Processing and its Applications. Academic Press, 1995.

[CP89] W. Choi and R. Park. Motion vector coding with conditional transmission. *Signal Processing*, 18:259–267, 1989.

[Csi97] P. Csillag. Motion-based segmentation of image sequences based on general 2d motion model using adaptive pel-recursive motion estimation. *Proc. International Picture Coding Symposium, Berlin*, pages 491–495, September 1997.

[Cut52] C. Cutler. Differential quantization of communication signals. *US Patent No. 2,605,361*, July 1952.

[DD99] S. Dubuisson and F. Davoine. Motion compensation using adaptive rectangular partitions. *Proc. International Conference on Image Processing*, November 1999.

[DML97] N. Dodgson, J. Moore, and S. Lang. Time-multiplexed autostereoscopic camera system. *Proc. SPIE Stereoscopic Displays and Virtual Reality Systems IV*, 3012:72–83, February 1997.

[FH94a] T. Fujii and H. Harashima. Data compression of an autostereoscopic 3-d image. *Proc. SPIE Stereoscopic Displays and Virtual Reality Systems*, 2177:108–118, February 1994.

[FH94b] T. Fujii and H. Harashima. Data compression of an autostereoscopic 3-d image. *Proc. SPIE Visual Communications and Image Processing*, 2308:930–941, September 1994.

[Fra96] R. Franich. *Disparity Estimation in Stereoscopic Digital Images.* PhD thesis, Delft University of Technology, September 1996.

[FvDFH93] Foley, van Dam, Feiner, and Hughes. *Computer Graphics - Practice and Principles.* Addison Wesley, 1993.

[GW92] R. Gonzalez and R. Woods. *Digital Image Processing.* Addison Wesley, 1992.

[Har52] C. Harrison. Experiments with linear prediction in television. *Bell Systems Technical Journal*, 31:764–783, July 1952.

[HP99] S. Han and C. Podilchuk. Efficient encoding of dense motion fields for motion-compensated video compression. *Proc. International Conference on Image Processing*, November 1999.

[Huf52] D. Huffman. A method for the construction of minimum redundancy codes. *Proc. of the Institute of Radio Engineers*, 40:1098–1101, 1952.

[HV91] P. Howard and J. Vitter. New methods for lossless image compression using arithmetic coding. *Proc. Data Compression Conference*, pages 257–266, March 1991.

[HV93] P. Howard and J. Vitter. Fast and efficient lossless image compression. *Proc. Data Compression Conference*, pages 351–360, March 1993.

[ITU96] ITU-T Rec. T.84 - Information Technology. Digital compression and coding of continuous-tone still images: Extensions, July 1996.

[JJ81] J. Jain and A. Jain. Displacement measurement and its application in interframe image coding. *IEEE Trans. on Communications*, 29:1799–1806, December 1981.

[KCR98] I. Kozintsev, J. Chou, and K. Ramchandran. Image transmission using arithmetic coding based continuous error detection. *Proc. Data Compression Conference*, pages 339–348, March 1998.

[Lan88] G. Langdon. Compression of multilevel signals. U.S. Patent 4,749,983, June 1988. Filed April 1986.

[Lan91] G. Langdon. Sunset: A hardware-oriented algorithm for lossless compression of grey scale images. *Proc. SPIE Medical Imaging V: Image Capture, Formatting and Display*, 1444:272–282, March 1991.

[Lee95] C-C. Lee. Lossless adaptive differential coding of images. *Proc. SPIE Still-Image Compression*, 2418:2–7, March 1995.

[LF96] L. Liu and E. Feig. A block-based gradient descent search algorithm for block based motion estimation in video coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 6(4):419–422, August 1996.

[LGS92] G. Langdon, A. Gulati, and E. Seiler. On the jpeg model for lossless image compression. *Proc. Data Compression Conference*, pages 172–180, March 1992.

[LL97] J. Lu and M. Liou. A simple and efficient search algorithm for block-matching motion estimation. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(2):429–433, April 1997.

[M$^+$93] D. McAllister et al. *Stereo Computer Graphics*. Princeton, 1993.

[Mar90] S. Martucci. Reversible compression of hdtv images using median adaptive prediction and arithmetic coding. *Proc. IEEE International Symposium on Circuits and Systems*, pages 1310–1313, 1990.

[MDTL96] J. Moore, N. Dodgson, A. Travis, and S. Lang. Time-multiplexed colour autostereoscopic display. *Proc. SPIE Stereoscopic Displays and Virtual Reality Systems III*, 2653:10–19, January 1996.

[MH96] G. Marosi and E. Hendriks. Recursive disparity estimation algorithm for real time stereoscopic video applications. *Proc. International Conference on Image Processing*, pages 887–890, 1996.

[MKC97] A. Merriam, A. Kellner, and P. Cosman. Lossless image compression over lossy packet networks. *International Conference on Signal Processing Applications and Technology (ICSPAT '97)*, 1997.

[MPFD96] J. Mitchell, W. Pennebaker, C. Fogg, and L. Didier. *MPEG Video Compression Standard.* Chapman Hall, 1996.

[MPG85] H. Musmann, P. Pirsch, and H. Grallert. Advances in picture coding. *Proc. of the IEEE*, 73(4):523–548, April 1985.

[MS95a] N. Memon and K. Sayood. Asymmetric lossless image compression. *Proc. Data Compression Conference*, page 457, March 1995.

[MS95b] N. Memon and K. Sayood. Lossless compression of rgb color images. *Optical Engineering*, 34(6):1711–1717, June 1995.

[MS95c] N. Memon and K. Sayood. Lossless image compression: A comparative study. *Proc. SPIE Still-Image Compression*, 2418:8–20, March 1995.

[MS96] N. Memon and K. Sayood. Lossless compression of video sequences. *IEEE Trans. on Communications*, 44(10):1340–1345, October 1996.

[MT97] B. Meyer and P. Tischer. TMW- a new method for lossless image compression. *Proc. International Picture Coding Symposium, Berlin*, pages 533–538, September 1997.

[MWSM97] N. Memon, X. Wu, V. Sippy, and G. Miller. An interband coding extension of the new lossless jpeg standard. *Proc. SPIE Visual Communications and Image Processing*, 3024:47–58, January 1997.

[Nel91] M. Nelson. *The Data Compression Book.* Prentice Hall, 1991.

[NH94] T. Naemura and H. Harashima. Fractal coding of a multi-view 3-d image. *IEEE International Conference on Image Processing*, pages 107–111, 1994.

[NKH96a] T. Naemura, M. Kaneko, and H. Harashima. 3-d object based coding of multi-view images. *Picture Coding Symposium*, pages 459–464, 1996.

[NKH96b] T. Naemura, M. Kaneko, and H. Harashima. 3-d segmentation of multi-view images based on disparity estimation. *Proc. SPIE Visual Communications and Image Processing*, 2727:1173–1184, February 1996.

[OA97] G. Oliveira and A. Alcaim. On fast motion compensation algorithms for video coding. *Proc. International Picture Coding Symposium, Berlin*, pages 467–472, September 1997.

[Oli52] B. Oliver. Efficient coding. *Bell Systems Technical Journal*, 31:724–750, July 1952.

[O'N66] J. O'Neal. Predictive quantizing systems (differential pulse code modulation) for the transmission of television signals. *Bell Systems Technical Journal*, 45:689–721, May 1966.

[ÖS93] T. Özkan and E. Salari. Coding of stereoscopic images. *Proc. SPIE Image and Video Processing*, 1903:228–235, February 1993.

[Pen99a] A. Penrose. Error resilient lossless image coding. In *International Conference on Image Processing*, September 1999.

[Pen99b] A. Penrose. Extending lossless image compression. In *Eurographics UK*, Cambridge, April 1999.

[Per92] M. Perkins. Data compression of stereopairs. *IEEE Trans. on Communications*, 40(4):684–696, April 1992.

[PW72] W. Peterson and E. Weldon. *Error-Correcting Codes.* The MIT Press, 1972.

[Qiu99] G. Qiu. A progressively predictive image pyramid for efficient lossless coding. *IEEE Trans. on Image Processing*, 8(1):109–115, January 1999.

[RC96] R. Roger and M. Cavenor. Lossless compression of AVIRIS images. *IEEE Trans. on Image Processing*, 5(5):713–719, May 1996.

[RD93] M. Rabbani and S. Dianat. Differential pulse code modulation image compression using artificial neural networks. *Proc. SPIE Image and Video Processing*, 2903:197–203, February 1993.

[RH97] A. Redert and E. Hendriks. Disparity map coding for 3d teleconferencing applications. *Proc. SPIE Visual Communications and Image Processing*, 3024, February 1997.

[Rim92] S. Rimmer. *Supercharged Bitmapped Graphics.* McGraw-Hill, 1992.

[RK96] D. Redmill and N. Kingsbury. The erec: An error-resilient technique for coding variable-length blocks of data. *IEEE Trans. on Image Processing*, 5(4):565–574, April 1996.

[Rob97] J. Robinson. Efficient general-purpose image compression with binary tree predictive coding. *IEEE Trans. on Image Processing*, 6(4):601–608, April 1997.

[SA92] S. Sayood and K. Anderson. A differential lossless image compression scheme. *IEEE Trans. on Signal Processing*, 40(1):236–241, January 1992.

[Sed92] R. Sedgewick. *Algorithms in C++*. Addison-Wesley, 1992.

[SGSJ94] M. Siegel, P. Gunatilake, S. Sethuraman, and A. Jordan. Compression of stereo image pairs and streams. *Proc. SPIE Stereoscopic Displays and Virtual Reality Systems*, 2177:258–268, February 1994.

[Sha48] C. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:398–403, July 1948.

[SHB93] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision*. Chapman and Hall, 1993.

[SN94] M. Slyz and D. Neuhoff. A nonlinear vq-based predictive lossless image coder. *Proc. Data Compression Conference*, pages 304–310, March 1994.

[SP96] A. Said and W. Pearlman. An image multiresolution representation for lossless and lossy compression. *IEEE Trans. on Image Processing*, 5(9):1303–1310, September 1996.

[SSMJ97] M. Siegel, S. Sethuraman, J. McVeigh, and A. Jordan. Compression and interpolation of 3d-stereoscopic and multi-view video. *Proc. SPIE Stereoscopic Displays and Virtual Reality Systems IV*, 3012:227–238, February 1997.

[STM97] T. Seemann, P. Tischer, and B. Meyer. History-based blending of image sub-predictors. *Proc. International Picture Coding Symposium, Berlin*, pages 147–151, September 1997.

[Tat94] S. Tate. Band ordering in lossless compression of multispectral images. *Proc. Data Compression Conference*, pages 311–320, March 1994.

[TGS97] D. Tzovaras, N. Grammalidis, and M. Strintzis. Object-based coding of stereo image sequences using joint 3-d motion/disparity compensation. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(2):312–327, April 1997.

[TLR85] S. Todd, G. Langdon, and J. Rissanen. Parameter reduction and context selection for compression of the gray-scale images. *IBM J. Res. Develop.*, 29(2):188–193, March 1985.

[Tur94] M. Turner. *The Contour Tree Image Encoding Technique and File Format*. PhD thesis, University of Cambridge, Computer Laboratory, July 1994.

[Ver98] S. Verdú. Fifty years of shannon theory. *IEEE Trans. on Information Theory*, 44(6):2057–2078, October 1998.

[Wal91] G. Wallace. The jpeg still picture compression standard. *Comms. of the ACM*, 34(4):30–44, April 1991.

[WC90] Q. Wang and R. Clarke. Motion compensated sequence coding using image pyramids. *Electronics Letters*, 26(9):575–576, April 1990.

[WM97] X. Wu and N. Memon. Context-based, adaptive, lossless image codec. *IEEE Trans. on Communications*, 45(4), April 1997.

[WMC98] X. Wu, N. Memom, and W. Choi. Lossless interframe image compression via context modeling. *Proc. Data Compression Conference*, pages 378–387, March 1998.

[WO97] W. Woo and A. Ortega. Stereo image compression based on disparity field segmentation. *Proc. SPIE Visual Communications and Image Processing*, 3024:391–402, February 1997.

[WRA96] M. Weinberger, J. Rissanen, and R. Arps. Applications of universal context modeling to lossless compression of gray-scale images. *IEEE Trans. on Image Processing*, 5(4):575–586, April 1996.

[WSG99] T. Wiegand, E. Steinbach, and B. Girod. Long-term memory prediction using affine motion compensation. *Proc. International Conference on Image Processing*, November 1999.

[WSS96] M. Weinberger, G. Seroussi, and G. Sapiro. LOCO-I: A low complexity, context-based, lossless image compression algorithm. *Proc. Data Compression Conference*, pages 140–149, March 1996.

[Wu96] X. Wu. An algorithmic study on lossless image compression. *Proc. Data Compression Conference*, pages 150–159, March 1996.

[YNL97] C. Yap, K. Ngan, and R. Liyanapathirana. Error resilient combined source-channel coder for mobile video. *Proc. International Picture Coding Symposium, Berlin*, pages 413–418, September 1997.

[Yu95] S. Yu. High-order entropy coding of medical image data using different binary-decomposed representations. *Proc. SPIE Optical Engineering Midwest '95*, 2622:879–884, August 1995.

[Zha82] C. Zhang. Ein neuer adaptiver pradiktor fur die dpcm-codierung von fernsehsignalen. *Frequenz*, 36:161–184, 1982.