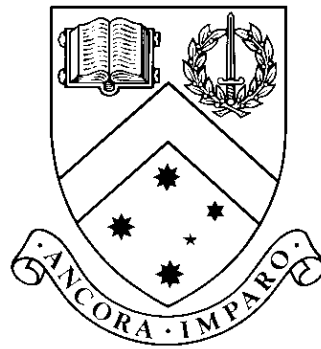


Lossless Compression of Magnetic Resonance Imaging Data

by

Chris Adamson



Thesis

Submitted by Chris Adamson

in partial fulfillment of the Requirements for the Degree of

Bachelor of Software Engineering with Honours (2150)

in the School of Computer Science and Software Engineering at

Monash University

Monash University

November, 2002

© Copyright

by

Chris Adamson

2002

For Lynda

Contents

List of Tables	vi
List of Figures	vii
Abstract	viii
Acknowledgments	x
1 Introduction	1
2 Image Compression	3
2.1 Prediction	4
2.1.1 Static Prediction	5
2.1.2 Adaptive Correction	8
2.1.3 Context Modeling	8
2.2 Coding	9
2.2.1 Huffman Coding	9
2.2.2 Rice Coding	10
2.2.3 Arithmetic Coding	11
2.3 Region Of Interest	12
3 Implementation	14

3.1	Data Sets	14
3.1.1	Data Interpretation	15
3.2	Universal Compression Schemes	17
3.3	Image Compression Schemes	17
3.4	Prediction	20
3.5	Post-processing of compressed data	21
3.6	Region of Interest Coding	24
3.7	Extraction of the brain	26
4	Conclusion	29
4.1	Software Engineering Notes	30
	References	31
	Appendix	33
.1	Data Sets	33
.2	Software Used	34

List of Tables

2.1	JPEG lossless predictors	7
3.1	Data set statistics	16
3.2	General-purpose algorithm comparison	18
3.3	Image compression algorithm comparison	19
3.4	Predictors investigated	20
3.5	Predictor comparison	22
3.6	RLE comparison	23
3.7	Near-lossless comparison	25
3.8	Brain extraction comparison	27

List of Figures

2.1	Causal Neighbourhood example	4
2.2	Rudimentary predictor transform images	5
2.3	Median Adaptive Predictor	6
2.4	Laplacian and Geometric distributions	10
2.5	Rice coding using parameter k	11
2.6	ROI examples	13
3.1	Pictorial representation of the data	15
3.2	Data interpretation formula	15
3.3	Plane	20
3.4	LOCO-I limitation example	24
3.5	Bit Oriented Run coding picture	24
3.6	RLE all thresholds graph	26
1	FMRIB Index	33
2	Howard Florey Index	34

Lossless Compression of Magnetic Resonance Imaging Data

Chris Adamson, BSE(Hons)
Monash University, 2002

Supervisor: Dr. Peter Tischer

Abstract

The use of digital media to store very large quantities of medical imaging data is becoming increasingly popular. The large amounts of data and often real-time processing requirements mean that any image compression scheme used would have to both compress well and operate quickly. There has been a great deal of research into lossless image compression and the latest schemes such as CALIC, TMW and LOCO-I have been shown to perform well on a wide range of generic continuous-tone images. However, there has not been a great deal of research into exploiting the predictable, common and hence compressible features of typical medical images. As such, investigation into this specific area could yield improved compression results and therefore lower the cost of storage.

This thesis investigates compression of 3D Magnetic Resonance Imaging (MRI) and 4D Functional MRI scans with a particular focus on the LOCO-I compression scheme. Possible improvements to the scheme with respect to prediction and theoretical compression limitations have been identified and discussed. The changes to the existing scheme are small, however in some cases they may deliver significant improvements in compression.

The distinct areas of foreground and background in MRI scans allow a single image to be compressed using multiple methods. This technique, known as Region of Interest (ROI) coding, increases compression in images with large regions of limited visually important features. The large proportion of background in MRI scans, compressed with a combination of lossless and lossy methods, yield significant compression gains while maintaining the visually important features of the image.

Lossless Compression of Magnetic Resonance Imaging Data

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Chris Adamson
November 11, 2002

Acknowledgments

I would like to thank my supervisor for all of the help he has provided me throughout the year and all the people who have helped me through the year. Your help has allowed me to continue through the year and has helped make this project and thesis possible.

Chris Adamson

Monash University

November 2002

Chapter 1

Introduction

The use of digital media to store medical images is becoming increasingly popular. Digital storage of medical images allows a device as simple as a desktop computer to process and analyze scans, which can assist in diagnoses and treatment. A computer is able to accurately and quickly identify small defects and changes in medical scans and in some cases this could lead to very early detection of problems. However, medical scans contain a massive amount of data and this makes storage and transmission expensive. Furthermore, often real-time processing requirements require any processing to be performed quickly.

An effective compression scheme would decrease the need to purchase large amounts of storage media and high capacity transmission lines. An efficient compression scheme would assist in delivering high throughput in processing applications.

In general terms, compression can be performed in lossless or lossy fashion. Lossless compression guarantees that the original data will be constructed exactly. A lossy compression scheme will generally be able to classify parts of an image or a sound as important or unimportant. By selectively throwing away unimportant information lossy schemes are generally able to deliver better compression.

There are existing lossless compression schemes, which are known as universal compression schemes, because they require no specific structure in input data. Because of this they can be used on any given set of data, regardless of any predefined structure or format. These compression schemes take advantage of general compressible features of data, such as long stretches of repeated text. In widespread use today, compression schemes such as GZIP, BZIP2 and ZIP are designed to perform well on totally unstructured, generic data.

There are many existing image compression schemes, each individual scheme generally tuned to perform well on certain types of images. Existing image compression schemes that perform lossless compression on colour images include GIF, JPEG (Lossless) and PNG. The focus of this thesis will be on those schemes that perform lossless compression on greyscale, continuous-tone images, these include CALIC, TMW and LOCO-I (JPEG-LS).

The now superceded JPEG (CCITT, 1992) scheme sparked a great deal of research into continuous-tone image compression. Since then, there has been significant improvements in prediction and modeling techniques, as well as improvements to coding techniques. This has allowed researchers such as Wu and Memon (Wu and Memon, 1997), Weinberger, Seroussi and Sapiro (Weinberger, Seroussi and Sapiro, 2000) and Meyer and Tischer (Meyer and Tischer, 1997) to make significant gains in the lossless compression of greyscale, continuous-tone images.

This thesis focuses on the compression of 3D Magnetic Resonance Imaging (MRI) and 4D Functional MRI scans using the LOCO-I compression scheme. The LOCO-I compression scheme delivers very high throughput however, it is tuned to perform well in the general case. As such, it tends not to exploit the common or predictable and hence compressible characteristics of medical images, which would lead to more compression.

Chapter 2 will discuss lossless image compression in general and present a number of techniques used in prediction and coding. Chapter 3 presents implementation and experiment details, including results and discussion for each part of the project. Chapter 4 will conclude the research and provide suggestions for further work.

Chapter 2

Image Compression

An image compression scheme consists of two components, an encoder and a decoder. The encoder is the portion of the scheme that takes an uncompressed image and encodes it into a more compact format. The decoder performs the opposite actions of the encoder, it takes the encoded image and attempts to reconstruct the original uncompressed image. This process will either be lossless, near-lossless or lossy, which will be determined by the particular needs of the user.

The two methods of lossless and near-lossless image compression both deal with the value or brightness of each individual pixel. Lossless compression guarantees that the value of each pixel in the reconstructed image will match its corresponding original value. When near-lossless compression is used, the reconstruction process may introduce errors in the reconstructed values, the maximum magnitude of these errors can usually be limited by the user.

Rather than deal with the value of every pixel, lossy compression schemes attempt to determine visually important components of an image. By maintaining visually important information, the reconstructed image has a similar “look” to the original. By throwing away visually unimportant information, the level of compression can be improved.

In some cases, areas of an image can be classified depending on their visual importance. For example, a brain scan can be classified into brain and background. The technique known as Region Of Interest (ROI) coding, alters the accuracy of compression for different regions of an image, in an attempt to increase the level of overall compression.

For many years now, people have been transferring images over communications networks. Transmission of large amounts of data proves costly, therefore the minimization of bandwidth use is an important issue. The highest percentage of total bandwidth use is occupied by the transmission of images, due to their large volume of data. The technique of progressive coding allows a user to begin to understand an image when only a relatively small amount of data has been sent.

The features just discussed may or may not be supported by individual image compression schemes. The particular features that a scheme supports generally makes it suitable for a particular purpose and each individual scheme will support various features in various ways. However, recent lossless compression schemes have mostly been composed using a two-stage process involving prediction and coding. These two stages, along with ROI coding will be discussed in detail in the following sections.

2.1 Prediction

It is assumed that an image is scanned using a raster-scan technique, that is to say the pixels are scanned from left to right, top to bottom. Therefore, at any time instance t it is assumed that all the previous pixels $x_0x_1 \dots x_{t-1}$ have been scanned. Both the encoder and the decoder scan the image in the same way, therefore if the encoder is encoding x_t , then it can assume that the decoder has decoded the pixels $x_0x_1 \dots x_{t-1}$.

The common scanning and decoding method used by both the encoder and decoder allow the value of x_t to be predicted in the same way by the encoder and decoder. The pixels that are used for this prediction cause the predictor to predict in a certain way, thus they are referred to as the causal neighbourhood (figure 2.1).

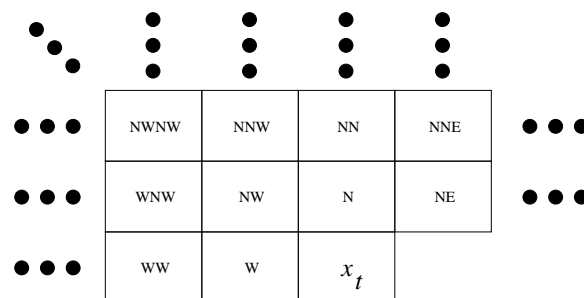


Figure 2.1: Example of a causal neighbourhood using “compass point” notation.

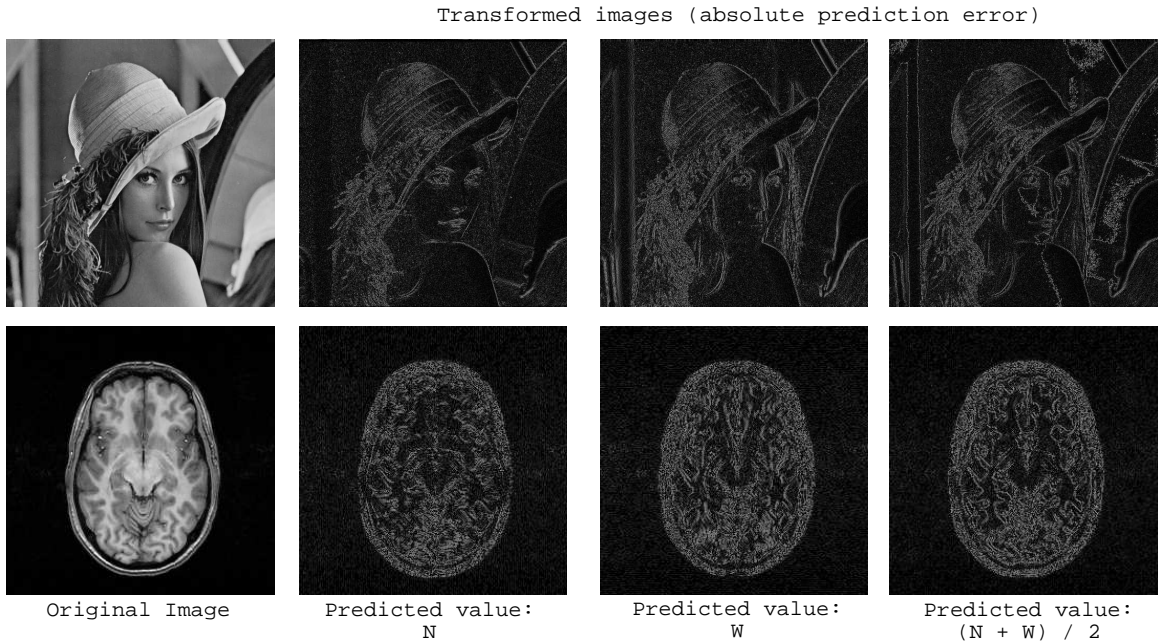


Figure 2.2: Examples of prediction transforms using some simple predictors.

The result of the prediction stage is a single predicted value for the pixel x_t . The original value of the pixel is subtracted from this value to give a prediction error, this is done for all the pixels in the image. The effect of this is that the image is transformed into another image, an image containing only prediction errors. Figure 2.2 displays the result of this prediction transform using a number of rudimentary predictors. The fact that the transformed images are very dark indicate that the predictor is predicting fairly accurately. Note that the areas of high contrast are also evident in the prediction error images, as these areas indicate bad predictions.

2.1.1 Static Prediction

The first step is to use a fixed prediction function that predicts the value of the current pixel.

An example of a prediction function is the Median Adaptive Predictor, originally proposed by Martucci in (Martucci, 1990) is presented in figure 2.3 and is the predictor used by LOCO-I (Weinberger et al., 2000).

$$\hat{x}_{MED} \triangleq \begin{cases} \min(W, N) & \text{if } NW \geq \max(W, N) \\ \max(W, N) & \text{if } NW \leq \min(W, N) \\ W + N - NW & \text{otherwise} \end{cases}$$

Figure 2.3: Median Adaptive Predictor

Another way of expressing this predictor is to say it returns the median of three sub-predictors ($N, W, W + N - NW$). Its adaptive nature and low complexity give a good tradeoff between accuracy and speed.

A predictor can take into account complex gradients¹ observed in the causal neighbourhood. The CALIC (Wu and Memon, 1997) scheme uses a Gradient Adjusted Predictor to perform this, which is presented below.

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

A prediction x_t is then made as such:

```

IF( $d_v - d_h > 80$ ) {sharp horizontal edge}
   $x_t = W$ 
ELSE IF( $d_v - d_h < -80$ ) {sharp vertical edge}
   $x_t = N$ 
ELSE
{
   $x_t = \frac{N+W}{2} + \frac{NE-NW}{4}$ 
  IF( $d_v - d_h > 32$ ) {horizontal edge}
     $x_t = \frac{x_t+W}{2}$ 
  ELSE IF( $d_v - d_h > 8$ ) {weak horizontal edge}
     $x_t = \frac{3x_t+W}{4}$ 
  ELSE IF( $d_v - d_h < -32$ ) {vertical edge}
     $x_t = \frac{x_t+N}{2}$ 
  ELSE IF( $d_v - d_h < -8$ ) {weak vertical edge}
     $x_t = \frac{3x_t+N}{4}$ 
}

```

¹Constant, directional value changes or slopes in a given direction.

The “magic numbers” of 80, 32 and 8 were chosen by Wu for 8-bit images, larger values are used in 16-bit images. Although it is more computationally intensive, it works well for a wider range of images.

Another technique is to run a number of predictors and use which one performs best. The PNG (Adler, Boutell, Bowler, Brunschen, Costello, Crocker, Dilger, Fromme, Gailly, Herborth, Jakulin, Kettler, Lane, Lehmann, Lilley, Martindale, Mortensen, Pickens, Poole, Randers-Pehrson, Roelofs, van Schaik, Schalnat, Schmidt, Stokes, Wegner and Wohl, 1999) scheme uses one of five predictors to produce the final predicted value. One predictor can be chosen for each line of the image. The number of the predictor being used must be transmitted. If all of the predictors must run on every pixel in the scan line to determine the best one, it is very intensive computationally. However, this method produces the best result for a given line in the image for the set of predictors available.

The baseline JPEG (CCITT, 1992) lossy mode uses a Discrete Cosine Transform (DCT) on blocks of pixels. The transform produces a set of 64 coefficients, one such coefficient gives the average brightness of the block. This is only value that is involved in the prediction step, the other 63 values must be stored unchanged. The baseline JPEG lossless mode uses one predictor, chosen by the user, from a table of 8 fixed predictors. These predictors are presented in table 2.1.

Selection-value	Prediction
0	0
1	W
2	N
3	NW
4	$W + N - NW$
5	$W + \frac{N-NW}{2}$
6	$N + \frac{W-NW}{2}$
7	$\frac{W+N}{2}$

Table 2.1: JPEG lossless mode prediction table. From (CCITT, 1992).

In the JPEG 2000 (ISO, 2000) scheme, a Discrete Wavelet Transform (DWT) is used to produce a set of high-pass and low-pass coefficients, these are used for the Inverse DWT, to reconstruct the original data. This is fairly computationally intensive, however it reconstructs images with a greater degree of quality than the original JPEG scheme.

Unlike the PNG scheme which uses only the result of one predictor, the TMW (Meyer and Tischer, 1997) scheme treats the output of each predictor as a probability distribution. Using

the sum of these distributions a global probability distribution for each pixel is produced. The number of predictors used is in fact given as a parameter by the user.

2.1.2 Adaptive Correction

Once a predicted value is obtained from the static predictor(s), a scheme can perform adaptive correction on that prediction. This stage is more open to the implementors to experiment, however there are a few common methods. The most straightforward method is to correct the predicted value by the current mean predicted value.

This tends to involve “learning” the behaviour of the prediction errors over time. Depending on the state of knowledge at that particular time, the encoder and decoder can correct the predicted value based on previous observed behaviour.

The LOCO-I observes the behaviour of the prediction errors over time and adjusts the predicted value based on the median observed prediction error from the causal neighbourhood.

2.1.3 Context Modeling

The context modeler is used to determine, from the causal neighbourhood, what context we are in. The best way this can be described is by using an example of English text. If letters were picked at random, we would expect ‘e’ to show up most often. As such, ‘e’ would always be our predicted value, as it is the most probable letter. However, if we look at the previous letter, the previous letter may change our mind. Provided we have seen a ‘q’, we would pick ‘u’ as ‘u’ is much more likely to be the next letter. Therefore in the context of there being a ‘q’, we adjust our predicted value to be a ‘u’. Similarly, in image compression, the context modifies the predicted value based on certain characteristics of the causal neighbourhood.

The context modeling can be done both in the prediction stage and the coding stage. The same information that the context modeler uses for the prediction stage can also be used in the coding stage. This means that we can make the prediction and the coding dependent on the context.

The LOCO-I scheme (Weinberger et al., 2000) maintains and updates context information over time, updating the predicted value and providing the coding parameter k .

The JPEG 2000 scheme (ISO, 2000) uses context modeling to assist in the coding phase. It passes data to an arithmetic coder (discussed in section 2.2.3, which it uses to steer its coding.

2.2 Coding

There are a number of coding techniques used by the schemes being investigated. There are optimal coding techniques, these include Rice, Huffman and Arithmetic coding. There are also other schemes such as Lempel-Ziv which is more a combination of a modeler and coder.

Coding is the procedure of creating a model and a code. The model is a stream of bits that describes how the data is encoded, the cost of incurred by having to store or transmit the model is known as the model cost. The code is a stream of bits that uniquely identifies the original symbols. As the code uniquely identifies all of the original symbols, coding will always be lossless.

2.2.1 Huffman Coding

Huffman coding (Huffman, 1952) is an established method of coding. It forms tables based on the probability of symbols occurring in a sequence. Traditionally, this has been done using a 2-pass approach, the entire sequence is scanned to compute each symbol's probability and then the Huffman table is constructed to compute how each symbol will be encoded.

There is a 1-pass version of Huffman coding that constructs a Huffman table as it reads the data, "learning" the structure over time. There is a trade-off here between compression and model cost. The main advantage in this method is efficiency as only one pass of the data is needed, which is important when large volumes of data are processed.

In cases where the probabilities are known beforehand or a good estimate of the probabilities can be made before looking at the data, a static Huffman table is used. This is known to both the encoder and the decoder, which eliminates model cost, but may make the compression non-optimal if the data is not similar to the data that was used to make the original Huffman table. If the data fits the table well, then the compression will be almost optimal and no model cost is incurred.

Huffman coding is an optimal coding scheme. That is to say the amount of bits that are produced is as close to the theoretical limit, the entropy, as possible. However, this only applies to the bits that identify the data, not the symbol table. The symbol table must be transmitted if it was formed independently by the encoder. Also, the theoretical limit of outputting at least 1 bit per symbol means that Huffman coding cannot achieve further compression on extremely compressible data.

The PNG (Adler et al., 1999) scheme uses Huffman coding combined with Lempel-Ziv (Ziv and Lempel, 1977) coding to generate compressed images. The CALIC-H (Wu and Memon, 1997) and JPEG (CCITT, 1992) schemes also use this coding method.

Huffman coding is more of a traditional coding technique and many other better techniques have been reported (Langdon, 1991; Howard and Vitter, 1993; Tischer, Worley, Maeder and Goodwin, 1993). Arithmetic coding schemes, which will be discussed later, generally achieve better compression than schemes that use Huffman coding.

2.2.2 Rice Coding

Rice coding (Rice, 1979) is a special case of Huffman coding. The special case arises when the probability distribution of the values in the input data is close to a Laplacian or a geometric distribution, given in figure 2.4. In this situation, Rice coding becomes a special case of Huffman coding.

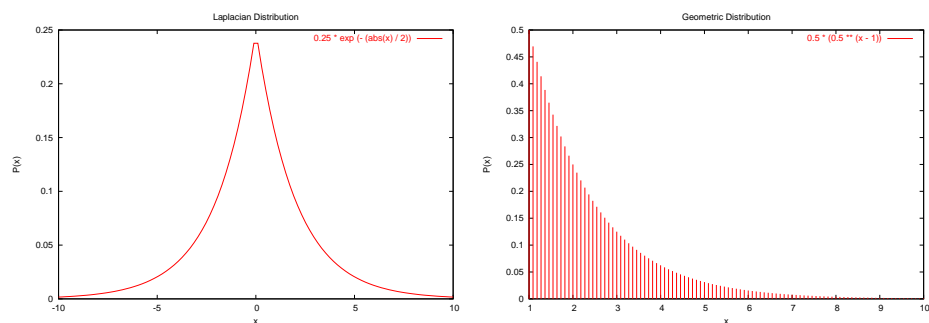


Figure 2.4: Laplacian and Geometric distribution examples.

The basic idea is that given a number n the coder shall output n 1's followed by a 0. A family of Rice codes can be constructed by using a coding parameter, k . k determines how many bits the encoder should treat as random. The parameter k splits an input symbol

into two parts. If the size of n in bits is defined as $|n|$, figure 2.5 demonstrates the coding procedure.

$$\hat{code} \triangleq \begin{cases} 0 \dots |n| - k - 1bits & \text{encoded in standard way} \\ |n| - k \dots |n|bits & \text{stored as are} \end{cases}$$

Figure 2.5: Formula for using the parameter k in Rice coding.

Rice coding does not need to store a histogram² or a symbol table as the codewords can be computed on-the-fly. This allows Rice coding to adapt well to larger bit depths, as large symbol tables do not have to be stored.

In Rice coding the symbol table does not have to be transferred and therefore the size of the output is generally nearly optimal. As the probabilities do not have to be computed, this scheme is quick and this was the main reason it was chosen in the only scheme mentioned in this document that uses it, LOCO-I (Weinberger, Seroussi and Sapiro, 1996b).

There are two problems with Rice coding that inhibit its compression abilities. The first problem is that it must output at least one bit for every symbol, this means it does not take advantage of highly compressible data. The other problem is that it is very inefficient if the value it must encode is large. By looking at Bit-Oriented Run Length Coding and a limiter on codeword length, these are both problems that can be minimized.

2.2.3 Arithmetic Coding

Arithmetic coding (Howard and Vitter, 1994) is based on representing a sequence of events as an interval between two numbers. This scheme is quite effective at compression, however it is computationally intensive. Although Arithmetic coding was originally designed to require floating-point arithmetic, it can be modified to use integer arithmetic³, which is considerably faster.

LOCO-A (Weinberger, Seroussi and Sapiro, 1996a), an extension to LOCO-I (Weinberger et al., 1996b), CALIC-A (Wu and Memon, 1997) and TMW (Meyer and Tischer, 1997) use arithmetic coding.

²A count of all symbols that have appeared.

³It has never actually been implemented in any standard using floating-point arithmetic.

Arithmetic coding has been shown as an effective method of compression in recent times. However, the computational complexity hampers the throughput of a scheme using Arithmetic coding.

2.3 Region Of Interest

In recent years, Content-Based Image Retrieval systems have sparked research into visually important features of images. From a compression standpoint, an image that can have regions classified as important or unimportant can be compressed using different methods. Visually unimportant regions can afford to be compressed with a lower degree of accuracy and as such be stored in a more compact fashion. The large area of background in typical medical images may allow significant gains in compression.

This research is focusing on the compression of brain scans and as such the Region Of Interest will focus on the importance of regions in brain scans. There are two main regions in a brain scan, the foreground and the background. The foreground consists of the brain material, this is the important region of the image and must be coded accurately. The background is the unimportant region of the image and a degree of error is allowable in the background.

The specification of the Region of Interest in an image is a problem. Presented in figure 2.6 is a few examples of regions of interest specifications possible on an example brain scan. The mask is the most accurate model at specifying a region of interest and would ensure the maximum compression benefits. But the mask takes the most information to represent and would add a significant model cost to the compressed file. The box includes some of the background in the important region, so it would not ensure the maximum benefit. However, the model cost is very cheap as a box takes little information to specify.

Considering the Region Of Interest must be known to the decoder when the file is decompressed, the decoder needs to know what the region of interest is. The specification of the region of interest must be stored with the compressed file, the space needed to do this is known as the model cost.

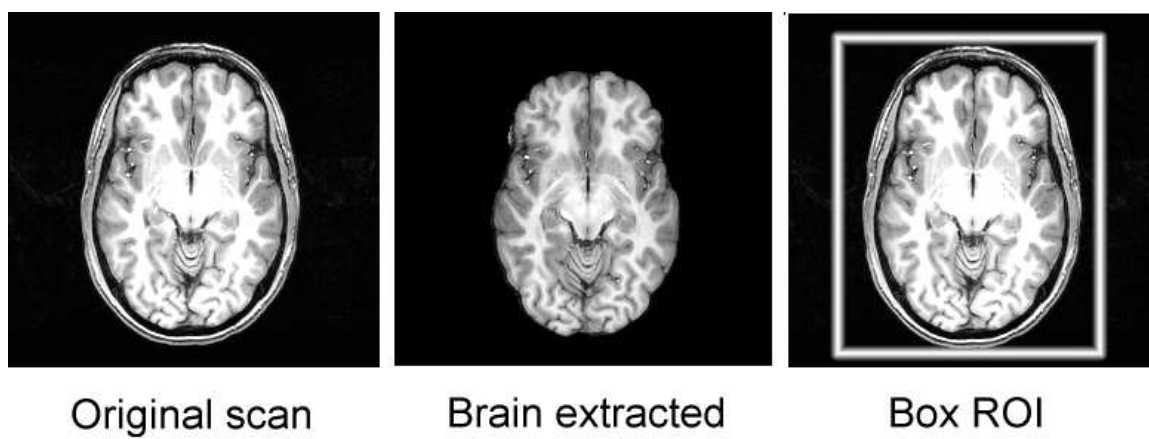


Figure 2.6: Region of Interest specification examples

Chapter 3

Implementation

This project will explore some of the techniques and ideas presented in Chapter 2. Section 3.1 presents the details of the data focused on this project. Sections 3.2 and 3.3 present an evaluation of universal and image compression schemes respectively. Section 3.4, 3.5, 3.6 and 3.7 present details of each component of research carried out in this project.

3.1 Data Sets

The data used for this project will be brain scans produced from MRI scanners. The data selected for the experiments should provide a representative sample of different types of possible data produced by typical scanners.

The main characteristic of data that affects compression is the presence of noise. In this case noise is generally distortion created by adverse conditions during a scan or by an error in the scanner. In some cases a scanner may distort all of its scans in the same way. Noise introduces unpredictability into the image, as such the prediction is done poorly and the compression suffers.

The FMRIB Image Analysis Group at Oxford University produced a suite of tools to process and analyze brain scans. Along with their software, they provided a test data set for use with their tools. It is this data set that will be used in this project.

Also, a test data set was obtained from the Howard Florey Institute, Melbourne. This data set contains scans of animal brains and was selected as it contains a level of noise and is representative of data collected in a real-life application.

Table 3.1 presents some statistics about the two data sets used. Note that only the 8 and 16 bit/voxel images could be processed by the programs that were used. The 32 and 64 bit/voxel images use floating-point representation for intensities, which no image compression scheme supports.

3.1.1 Data Interpretation

There are two distinct forms of scans that will be investigated. The 3D or Structural MRI scans are snapshots of the brain and a single scan is a 3D box composed of 2D slices. The 4D or Functional MRI scans are 3D scans of the brain over time, hence they are composed of 3D boxes. Figure 3.1 provides a pictorial representation of the scans.

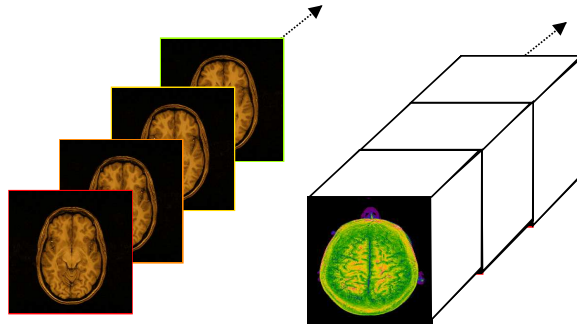


Figure 3.1: Pictorial representation of the data.

In order to process the scans with normal image compression schemes, they had to be interpreted with a single width and height i.e. they had to be 2D. Let $|d_i|$ be the magnitude of dimension i , figure 3.2.

$$\begin{aligned} \text{width} &= |d_1| \\ \text{height} &= |d_2| * |d_3| * |d_4| \end{aligned}$$

$$\begin{aligned} &\text{IF}(|d_4| \neq 1 \vee \text{width} \geq 65536 \vee \text{height} \geq 65536) \\ &\{ \\ &\quad \text{width} = |d_1| * |d_2| \\ &\quad \text{height} = |d_3| * |d_4| \\ &\} \end{aligned}$$

Figure 3.2: Width and height interpretation formula.

Image	Image Dimensions				Size (bytes)	Bits/ Voxel	Bits req'd for range
	1 st	2 nd	3 rd	4 th			
FMRIB data set							
epi	64	64	25	1	204800	16	16
fieldmap	64	64	25	2	1638400	64	N/A
fmri	64	64	21	180	30965760	16	15
structural	256	256	128	1	16777216	16	16
structural_brain	256	256	128	1	16777216	16	16
structural_brain_skull	256	256	128	1	16777216	16	8
structural_periph_render	256	256	128	1	16777216	16	16
structural_render	256	256	128	1	16777216	16	16
structural_susan	256	256	128	1	16777216	16	16
structural_talmask	256	256	128	1	8388608	8	1
str..._t..._segperiph	256	256	128	1	8388608	8	1
str..._t..._segvent	256	256	128	1	8388608	8	1
str..._t..._brain	256	256	128	1	16777216	16	16
str..._t..._b..._pve_0	256	256	128	1	33554432	32	1
str..._t..._b..._pve_0_segvent	256	256	128	1	33554432	32	1
str..._t..._b..._pve_1	256	256	128	1	33554432	32	1
structural_t..._b..._pve_1_segperiph	256	256	128	1	33554432	32	1
structural_t..._b..._pve_2	256	256	128	1	33554432	32	1
structural_t..._b..._seg	256	256	128	1	8388608	8	3
structural_vent_render	256	256	128	1	16777216	16	16
unwarped_epi	64	64	25	1	409600	32	16
unwrapped_phase	64	64	25	2	819200	32	8
Howard Florey data set							
S1_EPI	64	64	15	1	122880	16	12
S2_EPI	64	64	15	1	122880	16	12
S3_FIELDMAP	64	64	21	2	1376256	64	N/A
S4_SURFACECOIL	512	512	100	1	52428800	16	10
S6_MRA	256	256	55	1	7208960	16	14
S6_SURFACECOIL	512	512	100	1	52428800	16	10
T1.S1	124	256	256	1	16252928	16	10
T1.S2	256	256	124	1	16252928	16	9
T1.S3a	192	256	64	1	6291456	16	16
T1.S3b	256	256	64	1	8388608	16	16
T1.S4	256	256	124	1	16252928	16	9
T2.S1	512	512	15	1	7864320	16	11
T2.S2	512	512	15	1	7864320	16	11
T2.S4	512	512	15	1	7864320	16	11
T2.S5	512	512	15	1	7864320	16	11

Table 3.1: Data set statistics. N/A files did not have enough data to retrieve data range.

The width and height had to be limited to less than 65536 due to limitations in the LOCO-I implementation that was used.

3.2 Universal Compression Schemes

Currently, the most popular schemes used by people in the field are universal compression schemes. Schemes such as GZIP and BZIP2 are designed to handle totally unstructured, generic data. These schemes are designed to give the best average performance in the general case and it is expected that these schemes should achieve good results on the test data.

To provide a link to the image compression schemes, which will be discussed later, the generic compression schemes were compared to the LOCO-I scheme in Table 3.2.

The results in Table 3.2 demonstrate that the generic compression schemes achieve good results, with BZIP2 achieving the best results overall and GZIP the worst. However, the image scheme LOCO-I provided the best results in most cases and provided the best overall performance in both data sets.

From the results it can be seen that a specialized image compression scheme should achieve better compression on images than generic compression schemes.

3.3 Image Compression Schemes

To justify the decision to choose LOCO-I as the scheme for investigation, we shall now compare it to other image compression schemes. The image compression schemes chosen for investigation are PNG, CALIC-H, CALIC-A, JPEG 2000 and TMW. It is expected that some of these schemes will perform better than LOCO-I due to their increased complexity, however LOCO-I should still perform well.

Table 3.3 presents the results for the image compression scheme investigation. Although LOCO-I does not achieve the best compression results, it far outperformed all other schemes in execution time. The small loss in compression is a relatively small price to pay for the high throughput that LOCO-I provides. Therefore, in terms of the compression/speed tradeoff, it was found that the LOCO-I delivered the best tradeoff between speed and compression.

Image	LOCO-I	BZIP2	GZIP	RAR
epi	4.816	5.337	6.274	5.276
fmri	1.724	1.821	2.039	2.111
structural	5.092	5.460	6.402	5.404
structural_brain	0.593	0.684	0.767	0.729
structural_brain_skull	0.049	0.021	0.030	0.033
structural_periph_render	5.138	5.270	6.193	5.490
structural_render	5.214	5.237	6.182	5.493
structural_susan	2.990	3.676	5.348	3.675
structural_talmask	0.029	0.007	0.024	0.022
structural_talmask_segperiph	0.034	0.013	0.029	0.025
structural_talmask_segvent	0.011	0.003	0.012	0.008
structural_talmaskbrain	0.578	0.666	0.748	0.712
structural_talmaskbrain_seg	0.098	0.091	0.114	0.103
structural_vent_render	4.977	5.242	6.189	5.360
AVERAGE	1.269	1.350	1.625	1.404
S1_EPI	2.236	2.344	3.080	2.701
S2_EPI	2.302	2.426	3.217	2.700
S4_SURFACECOIL	0.857	0.956	1.397	1.091
S6_MRA	3.496	3.885	5.052	4.125
S6_SURFACECOIL	1.182	1.309	1.879	1.448
T1.S1	1.995	2.148	2.855	2.272
T1.S2	1.883	2.023	2.821	2.229
T1.S3a	5.141	5.628	6.591	5.541
T1.S3b	8.253	5.885	6.836	5.769
T1.S4	1.904	2.060	2.856	2.217
T2.S1	1.827	1.979	2.816	2.190
T2.S2	1.823	1.964	2.779	2.159
T2.S4	1.843	2.009	2.875	2.204
T2.S5	1.822	1.977	2.813	2.191
AVERAGE	1.849	1.897	2.566	2.036

Table 3.2: Comparison of general-purpose compression algorithms. Bits/original byte.

Image	LOCO-I	CALIC-H	CALIC-A	PNG	TMW
epi	4.816	4.846	4.939	6.026	8.050
fmri	1.724	1.940	1.774	2.002	2.848
structural	5.092	5.051	5.063	6.186	8.270
structural_brain	0.593	0.632	0.597	0.732	0.896
structural_brain_skull	0.049	0.070	0.020	0.060	0.304
structural_periph_render	5.138	5.081	5.081	6.094	8.220
structural_render	5.214	5.072	5.056	6.110	8.235
structural_susan	2.990	3.225	3.160	4.180	6.952
structural_talmask	0.029	0.097	0.013	0.014	0.021
structural_talmask_segperiph	0.034	0.107	0.015	0.018	0.023
structural_talmask_segvent	0.011	0.063	0.004	0.004	0.016
structural_talmaskbrain	0.578	0.617	0.582	0.714	0.873
structural_talmaskbrain_seg	0.098	0.199	0.078	0.099	0.063
structural_vent_render	4.977	4.936	4.945	6.026	8.219
AVERAGE	2.239	2.281	2.238	2.733	3.786
S1_EPI	2.236	2.357	2.185	3.060	6.199
S2_EPI	2.302	2.426	2.257	3.143	6.250
S4_SURFACECOIL	0.857	0.885	0.845	1.186	2.116
S6_MRA	3.496	3.515	3.475	4.696	7.670
S6_SURFACECOIL	1.182	1.208	1.159	1.598	2.911
T1.S1	1.995	1.970	1.944	2.666	5.985
T1.S2	1.883	1.873	1.879	2.592	5.844
T1.S3a	5.141	5.096	5.126	6.253	8.269
T1.S3b	8.253	8.250	8.212	6.515	5.532
T1.S4	1.904	1.899	1.908	2.623	5.871
T2.S1	1.827	1.843	1.794	2.506	4.933
T2.S2	1.823	1.838	1.790	2.490	4.922
T2.S4	1.843	1.862	1.813	2.520	4.947
T2.S5	1.822	1.839	1.789	2.503	4.928
AVERAGE	2.612	2.633	2.584	3.168	5.456

Table 3.3: Comparison of image compression algorithms. Bits/original byte. Note that the JPEG 2000 encoder was not used as it did not support 16 bit per pixel images.

The TMW compression results are not as good as they could be as a generic parameter file was used, however the lengthy execution time meant that its throughput was very low and hence it would not be suitable for real-time, medical processing applications.

3.4 Prediction

The prediction functions mentioned in section 2.1.1 are designed to give the best prediction for the data that it was expected to predict.

In the particular case of the Median Adaptive Predictor used by LOCO-I, it selects the median of $(W, N, W + N - NW)$. The third sub-predictor of those three: $W + N - NW$ is selected in the majority of cases and is known as the PLANE predictor. A plane in an image is where there is a constant gradient in both the north/south and east/west directions (shown in figure 3.3), the PLANE predictor will predict these perfectly. However, the presence of planes will only occur if and only if there is no noise in the image.

$$\begin{array}{cccc}
 v & v + h & v + 2h & \dots \\
 v + i & v + h + i & v + 2h + i & \dots \\
 v + 2i & v + h + 2i & v + 2h + 2i & \dots \\
 \vdots & \vdots & \vdots & \ddots
 \end{array}$$

Figure 3.3: The occurrence of a plane in an image.

A level of noise in a scan will be caused by either an anomaly in the scanning process or by a characteristic of the scanner. If the noise is caused by the scanner, then this level of noise should occur in all images produced by that scanner. Therefore, a prediction scheme that was more suited to noisy images should allow better compression in most cases.

Presented in Table 3.4 are the prediction schemes used in place of the existing prediction scheme in LOCO-I.

AVERAGE	PLANE	PIRSCH	CLARA
$\frac{N+W}{2}$	$N + W - NW$	$\frac{W}{2} + \frac{N}{4} + \frac{NE}{4}$	$\frac{N}{2} + \frac{N+NE}{4}$

Table 3.4: Static prediction schemes investigated in this project. The predictor labeled as CLARA is one of four predictors taken from the CLARA (Ueno and Ono, 1995) algorithm proposed by Mitsubishi for the JPEG-LS standard.

The PLANE and CLARA predictors are expected to perform poorly under the presence of noise, whereas the AVERAGE and PIRSCH predictors are expected to perform better under the influence of noise.

The results in Table 3.5 indicate that the existing scheme in LOCO-I performs best in the general case. However, in the Howard Florey data set the AVERAGE prediction scheme does perform better in most cases, its overall average being affected adversely by S1_EPI and S2_EPI. In most cases the AVERAGE prediction performing the best by around 0.02 bits/original byte, which on a 16 MB file saves a further 49 KB.

Using a different and possibly slightly more complex prediction scheme should improve compression overall by performing better predictions in the first phase of compression.

3.5 Post-processing of compressed data

In many cases it is possible to squeeze more compression out of a file that is already compressed by performing post-processing. This is possible generally because the compressor has not taken advantage of some compressible nature of the input data or it may have a theoretical limitation.

The LOCO-I scheme switches between two coding methods, Rice and run coding. Run coding is used when the predictor has been predicting perfectly for a specific time, therefore it requires certain conditions to be met. If the conditions for run coding are not met, LOCO-I will stay in normal Rice coding mode and it will not achieve better than 1 bit per symbol.

The criteria for changing to run mode are fairly basic and large portions of an image that could be run coded are not. Figure 3.4 demonstrates an image where LOCO-I misses out on run-coding the black section of the image due to the white strip.

To achieve better compression, the post-processing of the compressed data with bit-oriented run-length encoding was performed. The procedure involved looking for runs of 0s or 1s in the compressed file. Given a threshold t , if a run of 0s or 1s larger than the threshold is found, the bits after the threshold are encoded using Rice codes. Figure 3.5 presents a picture indicating the coding method, the symbol R indicating the rest or the remaining 0s or 1s after the threshold.

Table 3.6 presents results for selected thresholds and figure 3.6 presents results for all thresholds up to 32.

Image	LOCO-I (MAP)	AVERAGE	PIRSCH	PLANE	CLARA
epi	4.816	4.808	4.838	4.814	4.870
fmri	1.724	1.733	1.732	1.745	1.763
structural	5.092	5.100	5.121	5.102	5.164
structural_brain	0.593	0.603	0.606	0.603	0.598
structural_brain_skull	0.049	0.049	0.049	0.049	0.049
structural_periph_render	5.138	5.121	5.122	5.105	5.209
structural_render	5.214	5.201	5.206	5.192	5.319
structural_susan	2.990	3.040	3.043	3.084	2.968
structural_talmask	0.029	0.029	0.029	0.029	0.029
structural_talmask_segperiph	0.034	0.035	0.035	0.034	0.035
structural_talmask_segvent	0.011	0.011	0.011	0.011	0.011
structural_talmaskbrain	0.578	0.587	0.590	0.588	0.582
structural_talmaskbrain_seg	0.098	0.100	0.102	0.107	0.098
structural_vent_render	4.977	4.981	5.000	4.983	5.045
AVERAGE	1.269	1.272	1.275	1.286	1.275
S1_EPI	2.236	2.342	2.372	2.357	2.278
S2_EPI	2.302	2.404	2.425	2.421	2.345
S4_SURFACECOIL	0.857	0.855	0.857	0.859	0.857
S6_MRA	3.496	3.491	3.501	3.499	3.503
S6_SURFACECOIL	1.182	1.178	1.180	1.184	1.183
T1.S1	1.995	2.009	2.046	2.028	2.035
T1.S2	1.883	1.879	1.906	1.917	1.913
T1.S3a	5.141	5.161	5.186	5.159	5.199
T1.S3b	8.253	8.256	8.255	8.256	8.254
T1.S4	1.904	1.902	1.928	1.940	1.934
T2.S1	1.827	1.809	1.821	1.819	1.850
T2.S2	1.823	1.804	1.815	1.815	1.845
T2.S4	1.843	1.826	1.838	1.837	1.864
T2.S5	1.822	1.804	1.816	1.815	1.844
AVERAGE	1.849	1.847	1.858	1.863	1.857

Table 3.5: Comparison of LOCO-I scheme using different prediction schemes. Bits/original byte.

Image	LOCO-I	RLE 2	RLE 4	RLE 8	RLE 16	RLE 32
epi	4.816	7.954	5.566	4.861	4.816	4.815
fmri	1.724	2.846	1.997	1.741	1.724	1.724
structural	5.092	8.437	5.895	5.138	5.092	5.092
structural_brain	0.593	0.979	0.687	0.599	0.593	0.593
structural_brain_skull	0.049	0.080	0.056	0.050	0.049	0.049
structural_periph_render	5.138	8.492	5.935	5.176	5.134	5.137
structural_render	5.214	8.659	6.042	5.256	5.213	5.214
structural_susan	2.990	4.928	3.433	3.015	2.989	2.990
structural_talmask	0.029	0.045	0.033	0.029	0.029	0.029
structural_talmask_segperiph	0.034	0.054	0.040	0.035	0.034	0.034
structural_talmask_segvent	0.011	0.014	0.011	0.011	0.011	0.011
structural_talmaskbrain	0.578	0.953	0.669	0.584	0.578	0.578
structural_talmaskbrain_seg	0.098	0.155	0.113	0.100	0.098	0.098
structural_vent_render	4.977	8.247	5.761	5.021	4.977	4.977
AVERAGE	1.269	2.100	1.468	1.280	1.269	1.269
S1_EPI	2.236	3.588	2.542	2.247	2.233	2.236
S2_EPI	2.302	3.717	2.620	2.316	2.299	2.301
S4_SURFACECOIL	0.857	1.433	0.983	0.862	0.857	0.857
S6_MRA	3.496	5.813	4.034	3.522	3.496	3.496
S6_SURFACECOIL	1.182	1.975	1.358	1.189	1.182	1.182
T1.S1	1.995	3.310	2.298	2.014	1.995	1.995
T1.S2	1.883	3.102	2.165	1.903	1.883	1.883
T1.S3a	5.141	8.513	5.953	5.189	5.141	5.141
T1.S3b	8.253	13.769	9.569	8.320	8.253	8.253
T1.S4	1.904	3.139	2.190	1.924	1.904	1.904
T2.S1	1.827	3.036	2.112	1.840	1.827	1.827
T2.S2	1.823	3.029	2.107	1.835	1.823	1.823
T2.S4	1.843	3.063	2.129	1.856	1.843	1.843
T2.S5	1.822	3.028	2.105	1.835	1.822	1.822
AVERAGE	1.849	3.075	2.132	1.864	1.850	1.849

Table 3.6: Compression results from applying a post-processing stage. Bits/original byte.

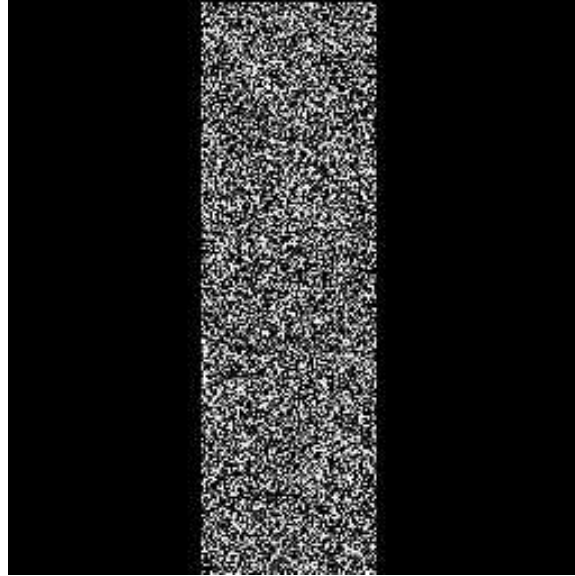


Figure 3.4: Limitation example, the black does not get compressed as well as it could be. LOCO-I achieves 2.589 bits per pixel, entropy of the original file was 1.11 bits per pixel.

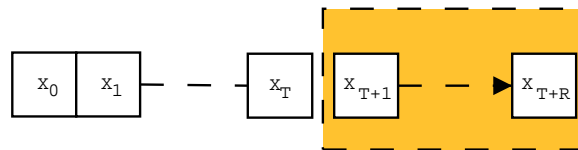


Figure 3.5: Pictorial representation of Bit-Oriented Run-Length Encoding.

The graph 3.6 showed that there were slight improvements, these occurred when the threshold was set at around 16. Only small improvements were expected as the nature of the data meant that LOCO-I should have compressed as well as possible, limiting further compression attempts.

3.6 Region of Interest Coding

Implementing Region Of Interest coding in LOCO-I was a difficult task. The implementation did not allow the degree of loss to be changed for individual pixels/voxels in an image. Therefore, the experiment involved the investigation of the LOCO-I near-lossless mode and its effectiveness on the test data.

Image	loss=0	loss=4	loss=8	loss=16	loss=32	loss=64
epi	4.816	3.225	2.767	2.297	1.830	1.381
fmri	1.724	1.301	1.179	1.054	0.921	0.772
structural	5.092	3.500	3.039	2.558	2.071	1.594
structural_brain	0.593	0.437	0.390	0.343	0.293	0.244
structural_brain_skull	0.049	0.039	0.035	0.030	0.026	0.014
structural_periph_render	5.138	3.570	3.113	2.635	2.151	1.677
structural_render	5.214	3.623	3.160	2.678	2.192	1.714
structural_susan	2.990	1.635	1.382	1.129	0.877	0.653
structural_talmask	0.029	0.004	0.004	0.004	0.004	0.004
structural_talmask_segperiph	0.034	0.004	0.004	0.004	0.004	0.004
structural_talmask_segvent	0.011	0.004	0.004	0.004	0.004	0.004
structural_talmaskbrain	0.578	0.428	0.381	0.334	0.286	0.237
structural_talmaskbrain_seg	0.098	0.004	0.004	0.004	0.004	0.004
structural_vent_render	4.977	3.411	2.957	2.483	2.005	1.537
AVERAGE	1.269	0.867	0.757	0.643	0.528	0.414
S1_EPI	2.236	1.055	0.841	0.640	0.467	0.326
S2_EPI	2.302	1.129	0.907	0.693	0.511	0.358
S4_SURFACECOIL	0.857	0.425	0.316	0.239	0.158	0.071
S6_MRA	3.496	1.924	1.489	1.058	0.717	0.377
S6_SURFACECOIL	1.182	0.577	0.422	0.311	0.187	0.077
T1.S1	1.995	0.546	0.367	0.233	0.085	0.010
T1.S2	1.883	0.488	0.320	0.202	0.094	0.027
T1.S3a	5.141	3.549	3.088	2.606	2.121	1.643
T1.S3b	8.253	6.659	6.208	5.732	5.244	4.750
T1.S4	1.904	0.516	0.339	0.206	0.076	0.013
T2.S1	1.827	0.750	0.429	0.246	0.153	0.089
T2.S2	1.823	0.746	0.427	0.238	0.143	0.080
T2.S4	1.843	0.760	0.445	0.266	0.165	0.095
T2.S5	1.822	0.747	0.429	0.244	0.152	0.090
AVERAGE	1.849	0.928	0.726	0.572	0.430	0.308

Table 3.7: Compression comparison using the near-lossless mode of LOCO-I. Bits/original byte.

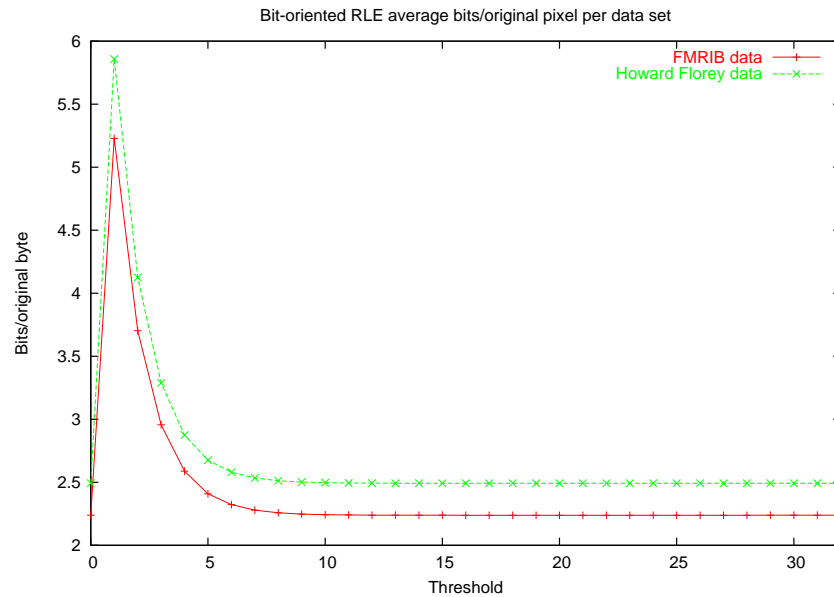


Figure 3.6: RLE graph for thresholds 0 ... 32

Table 3.7 demonstrates that the amount of compression significantly increases as the degree of loss is increased. This does not really provide any new information, however note that even setting the degree of loss at only 4, the compression improvement is significant. The majority of the improvement would be caused by the improved prediction of the background region. Therefore, implementing a Region Of Interest scheme for these brain scans should result in significant compression gains.

3.7 Extraction of the brain

The suite of software that accompanied the FMRIB data set contained a program that was able to identify brain material in a scan. The rest of the scan would be set to an arbitrary value, normally 0, leaving only the brain visible. Despite this having an important role in Region Of Interest coding, simply just compressing a scan with the non-brain area set to 0 should improve compression.

Table 3.8 shows that in most cases, the extraction of the brain produced significant compression gains. The brain extraction tool (BET) as it stands does take a while to run and if this must be run before compression, then the throughput of the whole process will be limited.

Image	LOCO-I	LOCO-I on Brain Extracted image
epi	4.816	1.330
fmri	1.724	1.721
structural	5.092	0.593
structural_brain	0.593	0.543
structural_brain_skull	0.049	0.002
structural_periph_render	5.138	0.650
structural_render	5.214	0.743
structural_susan	2.990	0.569
structural_talmask	0.029	0.004
structural_talmask_segperiph	0.034	0.004
structural_talmask_segvent	0.011	0.004
structural_talmaskbrain	0.578	0.536
structural_talmaskbrain_seg	0.098	0.004
structural_vent_render	4.977	0.511
AVERAGE	0.035	0.008
S1_EPI	2.236	0.829
S2_EPI	2.302	0.884
S4_SURFACECOIL	0.857	0.242
S6_MRA	3.496	0.936
S6_SURFACECOIL	1.182	0.195
T1.S1	1.995	0.452
T1.S2	1.883	0.437
T1.S3a	5.141	0.939
T1.S3b	8.253	1.064
T1.S4	1.904	0.454
T2.S1	1.827	0.535
T2.S2	1.823	0.538
T2.S4	1.843	0.570
T2.S5	1.822	0.524
AVERAGE	0.041	0.010

Table 3.8: Comparison of LOCO-I scheme using standard data and brain extracted data. Bits/original byte.

If the user is only interested in the brain material excluding the skull, which is removed, then the extraction of the brain before compression should lead to significant compression gains.

Note that although all of the images were put through BET, not all of them were true brain scans. Furthermore, it is not known how accurate BET is on non-human brains, therefore much of the brain in some of the non-human scans may have been removed.

Chapter 4

Conclusion

This thesis presented an investigation of Magnetic Resonance Imaging compression. This research was motivated by a need for an effective compression scheme for the vast amounts of data involved in the digital storage of medical images.

The investigation of universal compression schemes showed that LOCO-I, a specialized image compression scheme outperformed the popular universal compression schemes in both speed and compression. The JPEG-LS scheme has already been included in the DICOM (ASSOCIATION, n.d.) standard, a medical image format, which makes it relatively easy for practitioners to move over to using image compression.

The investigation of the image compression schemes revealed that the schemes investigated all provided good compression, however the speed/compression tradeoff needed to be taken into consideration. The LOCO-I scheme provided far superior speed to the other schemes and as such would be the best choice for real-time medical applications. The compression of the 32 bit and 64 bit images would require floating-point arithmetic capabilities, no image compression scheme has yet to implement this.

The investigation into the prediction schemes yielded slight improvements on the Howard Florey data set. The use of a different, perhaps slightly more computationally intensive prediction scheme, such as the GAP in CALIC, could yield better compression while not sacrificing a great deal on performance. The initial prediction function is only a small part of the prediction phase and as such the compression benefits may not be huge, but the amount of work needed to change the prediction function is minimal. Therefore, the payoff for changing the prediction function can be available quickly.

The Bit-Oriented Run Length Encoder did show some improvement with the threshold set at 16 and 32. However, this improvement was fairly small. This shows that the compression schemes that were used performed a fairly good job in the first instance, and further compression was not possible.

The Region Of Interest coding investigation was limited to using the near-lossless mode on the full scans, section 4.1 looks at this more closely. However, the magnitude of compression gains achieved by using a small degree of loss should allow a ROI-based scheme to achieve significant compression gains.

The extraction of the brain proved to be very beneficial from a compression standpoint. Having the background and skull removed from the scans increased their compressibility. Should a user not be interested in the background or the skull portion of a scan, pre-processing each scan with the brain extraction tool should yield smaller compressed files.

4.1 Software Engineering Notes

The JPEG-LS source code that was used in this project was version 2.2 of the SPMG/JPEG-LS group's implementation of JPEG-LS¹. In the interest of this research it was necessary to modify the source code to investigate the use of different predictors and Region Of Interest coding. This section presents some limitations of the research due to the source code used.

In the interest of producing an efficient and hence very fast executable, the developers chose to write the code in C. The heavy use of C pre-processor macros and the lack of documentation in the source code made it difficult to understand its operation. Despite the algorithm stating that the lossless mode is a special case of the lossy mode, the source code treats them as distinctly separate entities. Because of this it uses one set of instructions and function calls for each, making it difficult to change the lossy parameter adaptively.

¹This can be downloaded from <http://www.spmg.ece.ubc.ca>

References

- Adler, M., Boutell, T., Bowler, J., Brunschen, C., Costello, A. M., Crocker, L. D., Dilger, A., Fromme, O., Gailly, J., Herborth, C., Jakulin, A., Kettler, N., Lane, T., Lehmann, A., Lilley, C., Martindale, D., Mortensen, O., Pickens, K. S., Poole, R. P., Randers-Pehrson, G., Roelofs, G., van Schaik, W., Schalnat, G., Schmidt, P., Stokes, M., Wegner, T. and Wohl, J. (1999). PNG (portable network graphics) specification 1.2, (Online) <http://www.libpng.org/pub/png/pngdocs.html>.
- ASSOCIATION, N. E. M. (n.d.). Digital imaging and communications in medicine. <http://medical.nema.org/dicom/2001.html>.
- CCITT (1992). Information technology: Digital compression and coding of continuous-tone still images - requirements and guidelines, *Recommendation*, International Telecommunication Union. ITU T.81.
- Howard, P. and Vitter, J. S. (1993). Fast and efficient lossless image compression, *Data Compression Conference*, pp. 351–360.
- Howard, P. and Vitter, J. S. (1994). Arithmetic coding for data compression, *Proceedings of the IEEE*, Vol. 82, pp. 857–865.
- Huffman, D. (1952). A method for the construction of minimum redundancy codes, *Proceedings IRE*, Vol. 40, pp. 1098–1101.
- ISO (2000). Information technology – JPEG 2000 image coding system – part 1: Core coding system. ISO/IEC 15444.
- Langdon, G. G. (1991). Sunset: a hardware oriented algorithm for lossless compression of gray scale images, *Medical Imaging V: Image Capture, Formatting and Display 1444*: 272–282.

- Martucci, S. A. (1990). Reversible compression of HDTV images using median adaptive prediction and arithmetic coding, *IEEE International Symposium on Circuits and Systems*, pp. 1310–1313.
- Meyer, B. and Tischer, P. (1997). TMW - a new method for lossless image compression, *Proceedings International Picture Coding Symposium*, pp. 533–538.
- Rice, R. F. (1979). Some practical universal noiseless coding techniques, *JPL Pub 79-22*, Jet Propulsion Lab, Pasadena, CA.
- Tischer, P. E., Worley, R. T., Maeder, A. J. and Goodwin, M. (1993). Context-based lossless image compression, *Computer Journal* **36**: 68–77.
- Ueno, I. and Ono, F. (1995). CLARA: continuous-tone lossless coding with edge analysis and range amplitude detection. ISO working document ISO/IEC JTC1/SC29/WG1 N197.
- Weinberger, M. J., Seroussi, G. and Sapiro, G. (1996a). LOCO-A: an arithmetic coding extension of LOCO-I. ISO/IEC JTC1/SC29/WG1 document N342.
- Weinberger, M. J., Seroussi, G. and Sapiro, G. (1996b). LOCO-I: A low complexity, context-based lossless image compression algorithm, *Proceedings of the IEEE Data Compression Conference, Snowbird, Utah, March-April 1996*, p. 10.
- Weinberger, M. J., Seroussi, G. and Sapiro, G. (2000). The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS, *IEEE Transactions on Image Processing* **9**(8): 1309–1324.
- Wu, X. and Memon, N. D. (1997). Context-based adaptive lossless image compression, *IEEE Transactions on Communication* **45**: 437–444.
- Ziv, J. and Lempel, A. (1977). A universal algorithm for sequential data compression, *IEEE Transactions on Information Theory* **23**(3): 337–343.

Appendix

.1 Data Sets

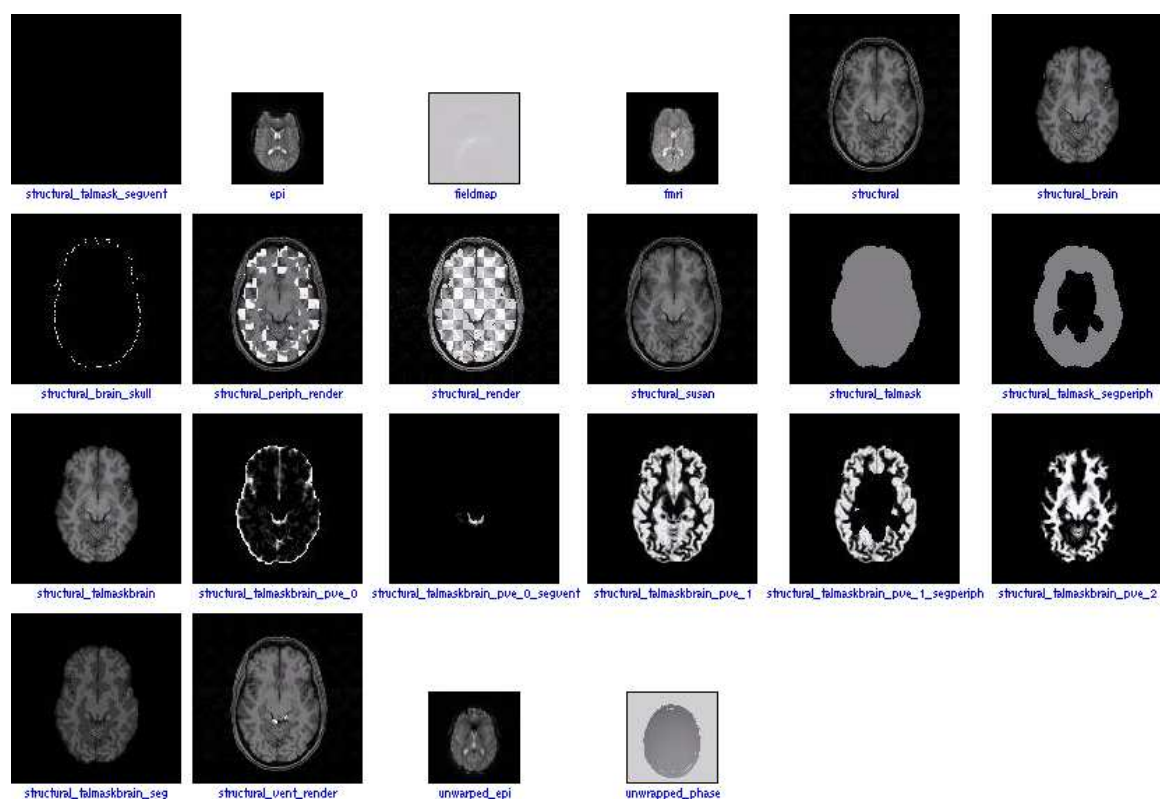


Figure 1: FMRIB Index (selected slices).

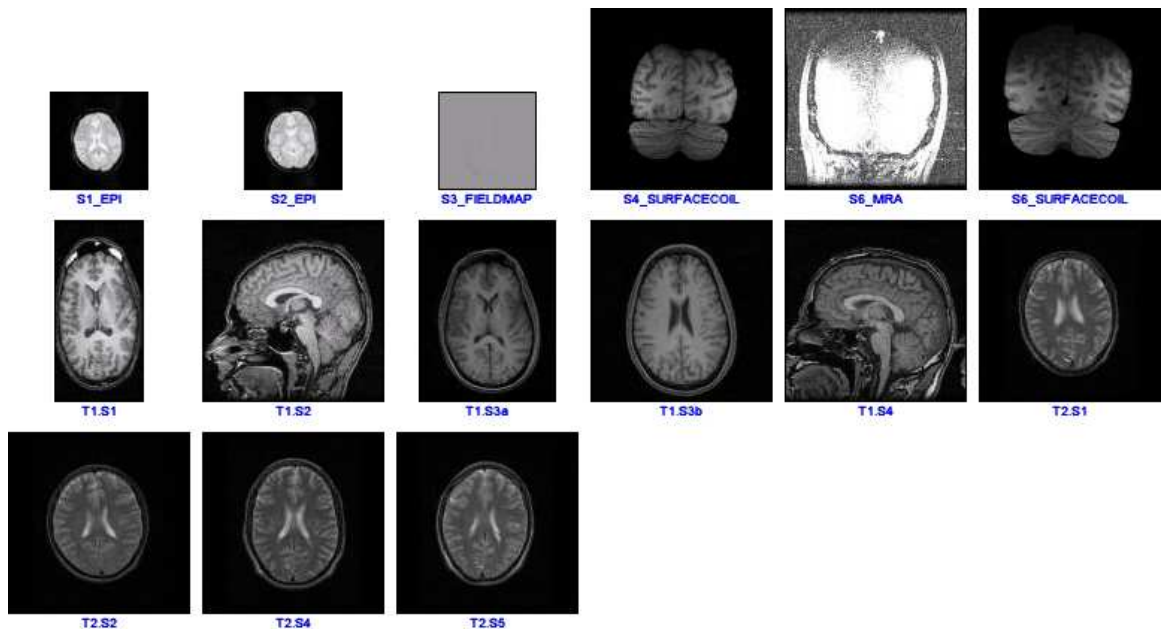


Figure 2: Howard Florey Index (selected slices).

.2 Software Used

- LOCO-I

Software name/version SPMG/JPEG-LS COMPRESSOR V.2.1

Author University of British Columbia

Web page <http://www.spmg.ece.ubc.ca>

- BZIP2

Software name/version bzip 1.0.2

Author Julian Seward

Web page <http://sourceware.cygnum.com/bzip2/>

- GZIP

Software name/version gzip 1.2.4a

Author Jean-loup Gailly

Web page <http://www.gzip.org>

- RAR

Software name/version RAR 3.00

Author Eugene Roshal

Web page <http://www.rar.cz>

- CALIC-H / CALIC-A

Software name/version CALIC

Author Xiaolin Wu

Web page ftp://ftp.csd.uwo.edu/pub/from_wu

- PNG

Software name/version pnmtopng 2.37.5

Author Bryan Henderson

Web page <http://netpbm.sourceforge.net>

- JPEG 2000

Software name/version JJ2000 4.1

Author Canon, Signal Processing Laboratory at Swiss Federal Institute of Technology, Ericsson

Web page <http://jj2000.epfl.ch/>

- TMW

Software name/version Tischer, Meyer, {Wombat, Wallace} 0.51

Author Bernie Meyer, Peter Tischer

Web page <http://www.csse.monash.edu.au/~bmeyer/tmw>

- FSL (BET, various tools)

Software name/version FMRIB Software Library

Author Functional Magnetic Resonance Imaging of the Brain Imaging Group, Oxford University

Web page <http://www.fmrib.ox.ac.uk/analysis>