Chapter 2:

Literature review

2. Literature review

This chapter presents a literature review on the subjects related to the dissertation. Data compression is introduced in Section 2.1, followed by image compression in Section 2.2. The characteristics of several types of images are described in Section 2.3. The role of redundancy in enabling data compression is examined in Section 2.4. Information theory, which provides a theoretical basis to data compression, is introduced in Section 2.5. A general image compression model is then described in Section 2.6. Section 2.7 and 2.8 is a brief survey on common compression methods and standards. Finally, the issue of compressing road map images is discussed in Section 2.9.

2.1 Data compression

Data compression is the process of converting an input data stream (the source stream or original data) into another data stream (the output, encoded or compressed bitstream) that has a smaller size (Salomon, 2000). It reduces the amount of data that is required to represent a given quantity of information.

There is a difference between *data* and *information*. *Information* signifies what we want to know, while *data* is the means used to convey the information (Gonzalez & Woods, 1992). A given amount information can be represented by various amounts of data, depending on the representation being used. The data may contain *redundancy*, which provides no relevant information or repeats what is already known. Data compression seeks to find the minimum amount of data required to represent the information by eliminating redundancy in the data.

After the data has been compressed, the original data can be obtained using a process called *decompression*, which is the inverse of compression. The reconstructed data may or may not be identical to the original data, depending on whether any loss of information has occurred during the compression process. Therefore, compression techniques can be broadly classified as *lossless* or *lossy*. In lossless compression, the reconstructed data is numerically identical to the original data, whereas in lossy compression it is not identical.

Compression can be applied to data representing all sorts of information, for example audio, still images, video, and text. Many methods for compressing data have been developed. Their suitability depends on the characteristics of the data to be compressed.

2.2 Image compression

A still image can be represented as a two-dimensional array of coefficients which are non-varying in time. Each coefficient represents the intensity level or colour.

Each element of the array is called a *pixel*. The pixels have a range of values, represented by p bits per pixel. For example, in a grayscale image, 8 bits per pixel may be used. The pixels will thus have values in the range of 0 to 255, where each value represents different gray levels.

For colour images, a colour model is used to specify the colour. Popular models include RGB, CMY, HSI and YIQ (Gonzalez & Woods, 1992). In the RGB model, for example, each pixel's colour is specified by three components R, G, and B, corresponding respectively to the three primary colours, red, green and blue. If each component is represented by 8 bits, as is normally the case, each pixel is then represented by 24 bits.

An image stored in uncompressed form is called *raw data*. Consider an image with the height of H pixels and the width of W pixels (see Figure 2.1). There is a total of H x W pixels, where each pixel is represented by p bits. The size of the data required to represent the image, S, is given by:

$$S = H \times W \times p \tag{2.1}$$

Example: Let H = W = 256, p = 8. Therefore, S = 256 x 256 x 8 = 524,288 bits = 65.536 bytes.

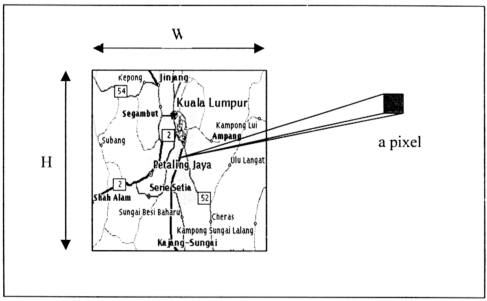

Figure 2.1: An example of an image and a pixel

Compression is capable of reducing the amount of data required to represent an image. As in data compression, image compression techniques can be broadly classified as either *lossless* or *lossy*. In lossless compression, the reconstructed image is exactly identical to the original image. In lossy compression, some amount of the original information is lost, causing the degradation of image quality. Depending on how much information is lost, the degradation may or may not be perceptually noticeable by the human vision. The goal in lossy compression is to reduce the information in such a way that the degradation is either acceptable or not perceivable to the user.

2.3 Image types

Images can be classified into various types (Salomon, 2000). They include:

1. *Bi-level/monochromatic*

   The pixels have either one of two colours, which are usually black and white. Each pixel can thus be represented by one bit. An example is a facsimile document.

2. *Grayscale*

   The pixels have different levels of gray. The values of the pixel are the integers between 0 and 2^p-1, where p is the number of bits per pixel, e.g. p = 8.

3. *Continuous-tone*

   The adjacent pixels have similar values and are hard to distinguish visually. There are areas with smooth colour variations, with the fine details being represented as sharp edges in between the smooth variations. Continuous-tone images are normally natural images taken by photograph. They can be grayscale or colour.

4. *Discrete-tone/graphical/synthetic*

   This is normally an artificially generated image. It may have a few or many colours, but does not have the noise and blurring of natural image. Instead it has sharp, well-defined edges, and adjacent pixels have either the same value or differ significantly.

5. *Cartoon-like*

   This consists largely of uniform areas, where each area has uniform colors but adjacent areas may have very different colours.

Their different characteristics mean that the images have redundancies that need to be exploited in different ways. Therefore different methods are employed to compress the different types of images.

2.4 Redundancy

Compression techniques are based on one fundamental principle, which is removing redundancy from the original data. Any non-random collection of data has some structure, and this structure can be exploited to achieve a smaller representation of the data such that no structure is discernible (Salomon, 2000).

According to Gonzalez & Woods (1992), there are three basic types of redundancy in still images:

1. *Coding redundancy*

   This occurs when codes are assigned to a set of events such that more bits than necessary are used to represent each event. In the case of an image, the event is the value representing the pixel colour or intensity. It is closely related to the concept of entropy (see the next section).

2. *Inter-pixel redundancy*

   The value of a pixel can be predicted from its neighbouring pixels. The information carried by a pixel is relatively small because the pixel value can be guessed based on neighbour's values or some other criteria, therefore less code bits can be assigned to it. The neighbouring pixels are said to be *correlated.* This type of redundancy is also known as *spatial* or *geometric redundancy.*

3. *Psycho-visual redundancy*

   Due to the nature of the human visual system, some information is not important and can be eliminated without perceptible loss of image quality. This type of redundancy is exploited in lossy compression.

2.5 Information theory

An important concept in data compression is information theory. Information theory was developed by Claude Shannon and provides a mathematical basis to data compression (Stallings, 2002; Sayood, 2000). Information theory can be used to define a theoretical limit on the amount of compression without loss of information. Two important concepts in information theory are *information* and *entropy*.

As mentioned previously, *information* signifies what we want to know about something. In information theory, information is equivalent to the *reduction of uncertainty* about an event. For example, most of us already know that the earth is round. Therefore a satellite photograph that shows such a fact conveys no information to us at all, since we already know it for certain. However, to someone living in prehistoric times, the photograph would have conveyed a large amount of information, since he did not know whether the earth is round or some other shape. The photograph provides visual proof that reduces uncertainty as to what the shape of the earth is.

Given an event x, which is an outcome of a random experiment, where the probability of the occurrence of the event is denoted P(x), a quantity used to define the amount of information is given by (Stallings, 2002):

$$I(x) = \log (1/P(x)) = - \log P(x) \qquad (2.2)$$

I(x) is known as the *self-information* associated with the occurrence of the event x. The unit depends on the logarithm's base, which is arbitrarily chosen. In the case of base 2, the unit is in *bits*. From the equation, it can be seen that if the probability of the event is low, the amount of self-information associated with it is high. On the other hand, if the probability of the event is high, the amount of self-information associated with it is low.

The second important concept is *entropy*. Let a set of independent events x_i, i = 1 to N, be the possible outcomes of some experiment *E* such that

$$\bigcup_{i=1}^{N} x_i = E \tag{2.3}$$

where *E* is the sample space and N is the number of possible outcomes (Sayood, 2000). The average self-information associated with the random experiment is given by (Stallings, 2002):

$$H(E) = \sum_{i=1}^{N} P(x_i)\, I(x_i)$$

$$= -\sum_{i=1}^{N} P(x_i) \log P(x_i) \tag{2.4}$$

The quantity H is known as the *entropy* associated with the experiment E.

Let the experiment be a source that outputs the symbols x_i, from the set *A* (called the *alphabet*). Shannon showed that the entropy is a measure of the average number of bits required to represent the output of the source. In other words, the theoretical bound on the average number of bits required to encode the output without any loss of information, as in the case of a lossless compression scheme, is equal to the entropy of the source. Looking at it another way, if the average number of bits used is more than required, as given by the entropy, then the data is said to contain coding redundancy (as mentioned in the previous section). An effective compression method is thus one which gives the average number of bits approaching as close as possible the entropy limit.

2.6 A general image compression model

Figure 2.2 shows a general image compression model. The source data is a sequence of symbols from the set $\{s_1, s_2 \ldots s_N\}$ and is the input to the compressor. There are three independent operations involved in the compressor (Gonzalez & Woods, 1992):

1. *Mapper*

   The mapper is an optional component. It maps the data into a format designed to reduce inter-pixel redundancies in the input image and is a reversible process. The mapping can be in some form of one-to-one mapping, prediction, decomposition or transformation.

2. *Quantizer*

   The quantizer is only present in lossy compression methods. It reduces the accuracy of the mapper's output in accordance to some pre-established fidelity criterion. This causes irreversible loss in information and reduces psycho-visual redundancy. It is omitted for a lossless compression system.

3. *Symbol encoder*

   The encoder creates a fixed or variable length code to represent the output symbols from the quantizer. The data is encoded and output as a bitstream.

The decompressor performs the inverse operations, except that a dequantizer is not included, since the quantization is an irreversible process.

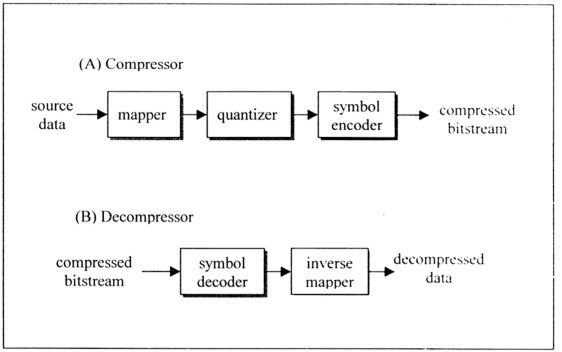

Figure 2.2: A general image compression model

2.7 Compression methods

Figure 2.3 shows a taxonomy of compression methods for both lossless and lossy compression. Various compression methods under both categories are highlighted next.

2.7.1 Lossless compression

Lossless compression methods can be categorized into the following techniques (Salomon, 2000; Sayood, 2000):

1. Run-length techniques
2. Statistical techniques
3. Dictionary techniques
4. Lossless predictive techniques

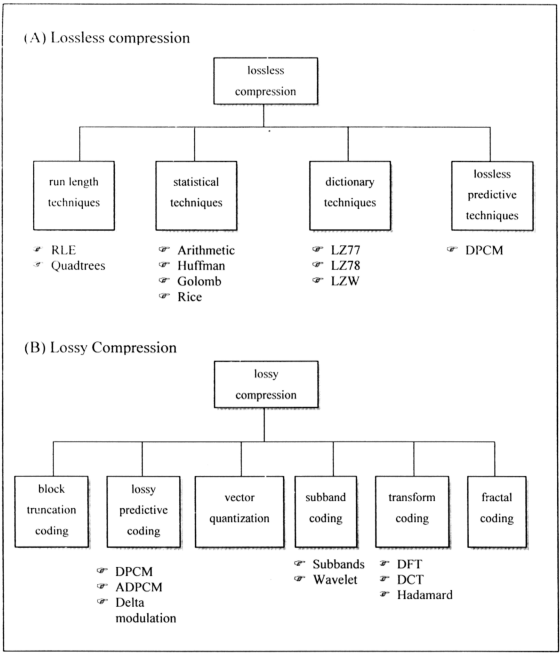

Figure 2.3: A taxonomy of image compression methods

2.7.1.1 Run length techniques

Run Length Encoding (RLE) is a simple technique that can be used for compressing data containing long runs of similar symbols. A sequence of identical symbols is replaced by the symbol and an indication of how many times the symbol repeats itself consecutively.

An example implementation is, given a sequence of r identical symbols s, where r is more than 3, then replace the sequence with the marker #sr, where # is a special control character to indicate the marker. Therefore, if the sequence of symbols is abbbbbbbbcde, it becomes a#b8cde after compression.

2.7.1.2 Statistical techniques

Statistical techniques attempt to compress data by using variable-length encoding. Symbols that occur more frequently (i.e. having higher probabilities) are assigned less bits, while symbols that occur less frequently are assigned more bits. In this way, the overall average number of bits per symbol can be expected to decrease, hence resulting in compression. Encoding is based on the symbol's probability, which is estimated using a model of the data.

Several well-known codes using statistical techniques include unary codes, Golomb, Rice, Turnstall, Shannon-Fano, Huffman and arithmetic codes. Huffman and arithmetic codes can give code lengths that are very close to the entropy. Hence, they are often used in combination with other compression techniques, for example, as the symbol encoder in lossy compression methods. They are also employed in many image compression standards.

- Huffman coding

Huffman coding is a popular algorithm that can be used to find the shortest codes to represent a set of symbols, given the symbols' probabilities (Sayood, 2000). It is an optimal code, which gives an average number of bits per symbol bounded below by the entropy of the source and bounded above by the entropy plus one bit:

$$H(E) \leq \text{average number of bits per symbol} < H(E) + 1 \qquad (2.5)$$

To achieve closer to the lower bound defined by the entropy, hence increasing compression, the symbols can be blocked together. If K symbols are blocked together and treated as a single symbol, the average number of bits per symbol is given by:

$$H(E) \leq \text{average number of bits per symbol} < H(E) + 1/K \quad (2.6)$$

Given p_m is the largest probability among the symbols' probabilities, a tighter upper bound for the Huffman code is given by:

$$\text{for } p_m \geq 0.5: \quad H(E) \leq \text{average number of bits per symbol} < H(E) + p_m \quad (2.7)$$

$$\text{for } p_m < 0.5: \quad H(E) \leq \text{average number of bits per symbol} < H(E) + p_m + 0.086 \quad (2.8)$$

There are however several drawbacks:

1. Huffman coding approaches the lower bound given by the entropy only when the probabilities of the symbols are negative powers of two. For other probability distributions, the average number of bits per symbol can be as large as that given by equation 2.7 or 2.8. When the probability distribution is highly skewed, p_m is large and it does not compress effectively. For example if p_m is 0.8, the average number of bits can be as large as 0.8 bit above the entropy, as given by equation 2.7.
2. Although compression can be increased by blocking K symbols together, the size of the alphabet increases exponentially. If K symbols are blocked together, the new size of the alphabet is N^K, where N is the size of the original alphabet. This may limit its practicality, because the codes for all the possible symbols have to be found.
3. In the case of binary alphabets, it produces no compression at all, unless the symbols are blocked together.

- Arithmetic coding

Arithmetic coding is another commonly-used compression algorithm. It assigns a single code to the whole data sequence instead of assigning codes for each symbol. Like Huffman coding, arithmetic coding is capable of yielding the average number of bits per symbol close to the entropy (Sayood, 2000). If L is the length of the data sequence,

$$H(E) \leq \text{average number of bits per symbol} < H(E) + 2/L \qquad (2.9)$$

Therefore, for long sequences of symbols, arithmetic coding guarantees compression performance very close to the entropy bound. Thus, it is suitable when the probability distribution is highly skewed, unlike Huffman codes. Given a probability model, arithmetic coding promises close to optimum compression.

The other advantages of arithmetic coding are:

1. Because improved modeling will enhance the compression effectiveness, finding an appropriate model is important. Arithmetic coding allows the probability modeling of the data and the coding process to be separated so that they are independent from each other (Rissanen & Langdon, 1981). Thus, complex or sophisticated modeling techniques can be easily tried without affecting the coding process. Several probability models can also be used together in the compressor with a single common arithmetic coder.
2. Adaptive modeling occurs when the model is built as the compression process goes along. This is useful, for example, when different data has different probability distributions, or the probability distribution changes. For arithmetic coding, adaptive modeling can be used easily. For Huffman coding, it is somewhat more complex (Langdon, 1981).

2.7.1.3 *Dictionary techniques*

In this technique, a dictionary of frequently occurring sequences of symbols is built. To compress the data, the frequently occurring sequences are represented by their indices in the dictionary.

Normally the dictionary needs to be sent along with the compressed data to the decoder. To be effective, the size of dictionary must be much smaller than number of possible patterns, so this method is useful for sources that generate a relatively small number of frequently occurring sequences compared to all possible sequences.

The dictionary can be static or adaptive. In the former case, the frequently occurring sequences of symbols are fixed based on some prior knowledge of the data. In the latter case, the dictionary is built during the encoding process. Methods based on the dictionary technique are LZ77, LZ78 and their variants.

2.7.1.4 *Lossless predictive techniques*

This technique is based on exploiting inter-pixel redundancies of pixels which are near to each other. A prediction of the pixel's value is made, based on the values of the nearby pixels. The difference between the actual and predicted value, that is, the prediction error, is computed to obtain a residual image. The residual image will have a much less dynamic range of pixel values if the neighbouring pixels have values that are close to each other (e.g. in a continuous-tone image). This image can then be efficiently encoded using a statistical technique such as Huffman coding.

2.7.2 Lossy compression

Lossy image compression methods take advantage of the characteristics of the human visual system (Subramanya, 2001). Human vision is more sensitive to the lower frequencies than to the higher frequencies of an image. The higher frequencies which are visually insignificant can be allocated less bits or discarded altogether in such a way that the loss in image quality is imperceptible or barely so, resulting in compression.

Several techniques are used for lossy compression (Subramanya, 2001):

1. Block truncation coding
2. Lossy predictive coding
3. Vector quantization
4. Subband coding
5. Transform coding
6. Fractal coding

Since lossy compression is not used and beyond the scope of this dissertation, no further explanation of these techniques will be given here. However, interested readers should refer to Subramanya (2001), Sayood (2000) and Salomon (2000).

2.8 Image compression standards

Over the years, several image compression standards have been developed. Standards play an important role to provide interoperability of compression schemes across different applications, operating systems and hardware platforms (Subramanya, 2001).

The major standards for still images include:

1. *Group 3* and *Group 4* facsimile compression standards – for the compression of bi-level images, specifically facsimiles.
2. *International Standards Organization's Joint Bi-level Image Group (JBIG)* – targeted for bi-level images, specifically facsimiles. There are two standards JBIG1 and JBIG2. JBIG2 is the newest version, featuring improvements on JBIG1.
3. *International Standards Organization's Joint Photographic Experts Group (JPEG)* – targeted for continuous-tone images, which may either be grayscale or colour images. There are several JPEG standards:
    - *JPEG* – this is the original JPEG standard created in the late 1980s. It provides for both lossy and lossless compression. A transform-based technique, Discrete Cosine Transform (DCT), is used in the lossy mode.
    - *JPEG-LS* – this is a JPEG standard defined especially for lossless and near-lossless compression of continuous-tone images. It uses predictive coding technique.
    - *JPEG-2000* – the latest JPEG standard aims to address several issues that are not provided in current standards, such as better low bit-rate compression and superior performance for both lossless and lossy compression. It is targeted for different types of still images with different characteristics (Santa-Cruz *et al.*, 2000). The standard uses wavelet-based coding.

2.9 Compressing road map images

Figure 2.4 shows an example of a road map image.

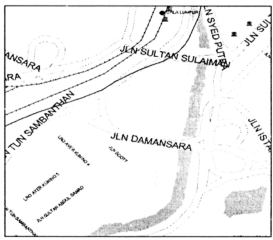

Figure 2.4: An example of a road map

The following assumptions can be made regarding the characteristics of a road map image:

1. It composes of a few number of colours. Maps in general use only a few colours of the available colours, since using too many colours causes difficulty in distinguishing the patterns (Mitchell, 1999).
2. It displays similar characteristic to a discrete-tone image, as described in Section 2.3. It does not contain regions with smoothly varying colours such as in a continuous tone image.
3. It has large uniform areas or regions, overlaid with text, lines and symbols

However, most image compression standards, as described in the previous section, are designed to compress continuous-tone or bi-level images.

Using a transform-based lossy compression method such as JPEG results in blurring of sharp edges which are visually important in palette images (for example, the text may become illegible). Also, a transform type encoder produces a significant number of coefficients which have to be represented with higher precision than the original pixels, causing inefficiency to the compression (Yoo *et al.*, 1998). The image also contains too few colours to make use of linear predictive models such as those used by JPEG-LS (Ausbeck, 1998).

A commonly-used approach is to use a *palette image* format. An image having a few colours can be stored efficiently using such a format as opposed to the bitmapped format. A palette image format contains a colour table which is a list of the colours used (the *palette*). Each pixel value is represented by an index pointing to the colour table, instead of the actual value itself. The colour table is stored in the header of the file. Examples of palette image formats are the *Graphics Interchange Format* (GIF) and the *Tagged Image File Format* (TIFF).

Two examples shown in Figure 2.5 demonstrate the difference between using a lossless and lossy method. Figure 2.5 (A) and (B) shows a couple of road maps saved in GIF and JPEG file format respectively. GIF uses lossless compression while JPEG uses lossy compression. The JPEG file is compressed until its size is the same as the GIF file. The effect is degraded image quality, with discolouration, blurred edges and poor text legibility.

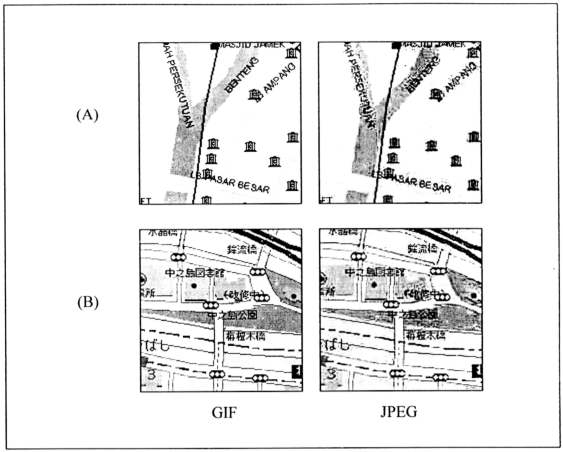

Figure 2.5: Degradation caused by lossy compression

Figure 2.6 shows an example showing the difference between a palette format and a bitmapped format. In the bitmapped format, a pixel is represented by the actual R, G and B component values. If each component requires 8 bits, then each pixel requires 24 bits. On the other hand, in the palette format, the pixel is represented by an index, which can be, for example, 8 bit, in the case of a 256-colour table. This presents a savings of three times, since the size of the colour table is negligible in most cases. For an image with the width of 600 pixels and height of 400 pixels, the size of the palette image is 240,768 bytes compared with 720,000 bytes for a bitmapped image.

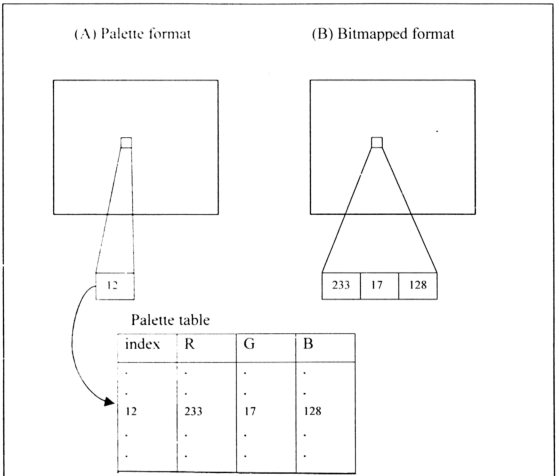

Figure 2.6: Difference between palette and bitmapped format.

Compression can be applied on the pixel data of a palette image to further reduce the size. For example, in GIF, a dictionary-based technique, the Lempel-Ziv-Welch (LZW) algorithm, is used (CompuServe Inc., 1989). However, it has been pointed out that

dictionary-based techniques do not compress images effectively because they use one-dimensional coding, thus fail to exploit the spatial redundancy fully in both the vertical and horizontal directions (Ausbeck, 1998; Gilbert & Brodersen, 1998). Efforts in finding suitable lossless compression methods for palette images, and map images in particular, have been done by researchers. A large number employed statistical techniques, combined with finding a suitable two-dimensional model (Forchhammer & Salinas, 2002; Franti *et al.*, 2002; Ageenko *et al.*, 2001; Ausbeck, 2000; Rauschenbach, 2000; Jensen & Forchhammer, 1999; Ratnakar, 1998; Yoo *et al.*, 1998). Most of these, except Rauschenbach (2000), used arithmetic coding.

As mentioned in Section 2.7.1.2, one advantage of arithmetic coding is that, given a model, it guarantees compression very close to the entropy limit, whereas Huffman coding does so only under certain conditions.

The second advantage is that modeling, which affects compression effectiveness, can be done separately, independent from coding. This allows the use of more complex modeling techniques without the need to be concerned about the coding process. Thus, in this dissertation, the focus will be on implementation of the modeling. The coding unit will use a readily available arithmetic coder implementation by Moffat *et al.* (1998). This implementation offers several improvements over the widely-used implementation by Witten *et al.* (1987), and will be described in the next chapter.

Finally, adaptive modeling can be implemented easily in arithmetic coding. This will be useful since it is expected that different images will have different probability distributions. For example, it is unlikely that two different images will have the same probability distribution of colours.

Therefore, this dissertation investigates statistical techniques to compress road map images, using arithmetic coding with adaptive context-based statistics modeling.

Context-based modeling is used to exploit the two-dimensional spatial redundancy of the images. Two methods involving context-based modeling were investigated in this dissertation:

1. Bitplane coding - the image is decomposed into bitplanes, and compression is applied to each bitplane using fixed order context modeling and a binary arithmetic coder. This approach was also used by the JBIG standard and several other researchers (Ageenko *et al.*, 2001; Jensen & Forchhammer, 1999; Yoo *et al.*, 1998)
2. Prediction by Partial Matching (PPM) - this is a state-of-the-art method for text compression and can be extended for image compression (Forchhammer & Salinas, 2002). PPM uses variable order context modeling and a multisymbol arithmetic coder.

Bitplane coding and PPM can also be used in a combined method, where they each operate independently on different planes of the image. This method was also investigated.