A Lossless Compression Method for 3D Medical Images

V.Ragul¹, S.Sumathi², Dr.R.Kanthavel³, G.Meenakshi⁴
Student¹, Assistant Professor² ⁴, Professor and Head³
Department of ECE
Velammal Engineering College, Chennai, India

Abstract:
We present an innovative method for lossless compression of discrete-color images, such as map images, graphics, GIS, as well as binary images. This method comprises two main components. The first is a fixed-size codebook encompassing 8×8 bit blocks of two-tone data along with their corresponding Huffman codes and their relative probabilities of occurrence. The probabilities were obtained from a very large set of discrete color images which are also used for arithmetic coding. The second component is the row-column reduction coding, which will encode those blocks that are not in the codebook. The results show that our method compresses images from both categories (discrete color and binary images) with 90% in most case and higher than the JBIG-2 by 5%-20% for binary images, and by 2%-6.3% for discrete color images on average. The proposed system consists of two algorithms for lossless compression of the images. They are Code Book and Row Column Reduction Coding (RCRC). The proposed system has the advantage that it has high accuracy and the reconstructed image is equivalent to that of the original uncompressed image.

Keywords: Compression, Binary Images, RCRC, etc

I. INTRODUCTION

A digital image is an array of real or complex numbers represented by a finite number of bits. An image given in the form of a transparency, slide, photograph or an X-ray is first digitized and stored as a matrix of binary digits in computer memory. This digitized image can then be processed and/or displayed on a high-resolution television monitor. The objective of image compression is to reduce irrelevance and redundancy of the image data in order to be able to store or transmit data in an efficient form. Image compression refers to minimizing the size in bytes of a graphics file without degrading the quality of the image to an unacceptable level. The reduction in file size allows more images to be stored in a given amount of disk or memory space.

II. LOSSY COMPRESSION

Lossy compression also called as irreversible compression is the class of data encoding methods that uses inexact approximations and partial data discarding to represent the content. These techniques are used to reduce data size for storage, handling, and transmitting content. Lossy compression is more effective when used to compress graphical images and digitized voice where losses outside visual or aural perception can be tolerated. Lossy compression methods, especially when used at low bit rates, introduce compression artifacts. Lossy methods are especially suitable for natural images such as photographs in applications where minor (sometimes imperceptible) loss of fidelity is acceptable to achieve a substantial reduction in bit rate. The lossy compression that producible differences may be called visually lossless.

III. LOSSLESS COMPRESSION

Lossless compression is a class of data compression algorithms that allows the original data to be perfectly reconstructed from the compressed data. It is used mainly for compressing database records, spreadsheets or word processing files, where exact replication of the original is essential. Lossless compression is preferred for archival purposes and often for medical imaging, technical drawings, clip art, or comics. For most types of data, lossless compression techniques can reduce the space needed by only about 50%. For greater compression, one must use a lossy compression technique. Note, however, that only certain types of data graphics, audio, and video can tolerate lossy compression.
IV PROPOSED WORK

Central to the proposed method in this work is the idea of partitioning a binary image or the bi-level layers of discrete color images into non-overlapping 8 x 8 blocks. Partitioning binary images into blocks and encoding them, referred to as block coding, has been summarized, wherein images are divided into blocks of totally white (0-valued) pixels and non-white (1-valued) pixels. The former blocks are coded by one single bit equal to 0, whereas the latter are coded with a bit value equal to 1 followed by the content of the block in a row-wise order. Moreover, the hierarchical variant of the block coding relies in dividing a binary image into b x b blocks (typically, 16 x 16), which are then represented in a quad-tree structure. In this case, a 0-valued b x b block is encoded using bit 0, whereas other blocks are coded with bit 1 followed by recursively-encoded blocks of pixels with the based case being one single pixel. In and, it is suggested that block coding can be improved by resorting to Huffman coding or by employing context-based models within larger blocks, a hybrid compression method based on hierarchical block coding is proposed. Here, predictive modeling has been employed to construct an error image as a result of the difference between the predicted and original pixel values. Then, the error image is compressed using Huffman coding of bit patterns at the lowest hierarchical level. This work builds upon the ideas presented wherein block coding with Arithmetic coding has been employed. In the context of discrete-color images, lossless compression methods are generally classified into two categories: (i) Method is applied directly on the image, such as the graphics interchange format (GIF) or the portable network graphics (PNG) (ii) Methods applied on every layer extracted (or separated) from the image, such as TIFF-G4 and JBIG. In this research, we focus on the second category. Previous work in literature amounts to several lossless compression methods for map images based on layer separation. The standard JBIG2, which is specifically designed to compress bi-level data, employs context-based modeling along with arithmetic coding to compress binary layers. In a lossless compression technique based on semantic binary layers is proposed. Each binary layer is compressed using context-based statistical modeling and arithmetic coding, which is slightly different from the standard JBIG2. In a method, which utilizes interlayer correlation between color-separated layers is proposed. Context-based modeling and arithmetic coding are used to compress each layer. An extension of this method applied on layers separated into bit planes is given. Moreover, other lossless compression standards, such as GIF, PNG, or JPEG-LS are applied directly on the color image.

V. BLOCK CODE

Block coding has been devised primarily for coding of graphics, but it has subsequently been extended to multilevel pictures. All the proposed codes are simple suboptimum prefix codes. Their simplicity makes them suitable for real-time applications. Although blocks can be of any shape, higher efficiencies are obtained with two-dimensional blocks, thus exploiting the inherent two-dimensional correlation of pictures. According to the value of a preset parameter, block coding can be either information lossless or information lossy. In the former case, the original digitized picture can be exactly reconstructed from its coded version. In the latter case, where the compression is much higher, distortions possess easily identified features. An appropriate filtering can restore the decoded picture satisfactorily. With a slight increase in complexity, block coding can be made adaptive in a number of ways, leading to much higher compressions. For each case, comprehensive theoretical models are developed to predict the performances and to optimize the parameters. The dependence of the compression ratio on image resolution for each specific code is also examined.

VI. ROW-COLUMN REDUCTION CODING

The codebook component of the proposed method is efficacious in compressing the 6952 blocks it contains. These blocks, as seen in the previous section, are the most frequently occurring symbols as per the empirical distribution. Compared to the alphabet size of 264, the cardinality of the codebook is very small. Hence, there will be blocks from input images that cannot be compressed via the codebook. For that purpose, we designed the row column reduction coding (RCRC) to compress 8 * 8 blocks of a binary matrix. V, that are not in the codebook, C. In this section, we illustrate how the algorithm works. The RCRC is an iterative algorithm that removes redundancy between row vectors and column vectors of a block and proceeds as follows. For each 8 * 8 block b, c, b /c, RCRC generates a row reference vector (RRV), denoted as r, and a column reference vector (CRV), denoted as c. Vectors r and c may be viewed as 8-tuples which can acquire values ri = {0, 1}, ci = {0, 1}, i {1, . . . , 8}. These vectors are iteratively constructed by comparing pairs of row or column vectors from the block b. If rows or columns are identical in a given pair, then the first vector in the pair eliminates the second vector, thus reducing the block. If the two vectors are not identical, then they are both preserved. The eliminations or preservations of rows and columns are stored in the RRV and CRV, respectively, which are constructed in a similar way. The iterative construction procedure is exposed in what follows. Let bi, j denote row i of block b, for j = 1, 2, . . . , 8. RCRC 29 compares rows in pairs starting with the first two row vectors in the block: (b1, j, b2, j). If b1, j = b2, j, we assign a value of 1 to r1 and a value of 0 to r2 and we eliminate row b2, j from block b. Next, b1, j is compared to b3, j and if they are equal, a value of 0 is stored in r3 implying that row b3, j is eliminated from the block. If, however, b1, j = b3, j, then a value of 1 is assigned to r3 implying that the third row of block b is preserved. Thereafter, RCRC will create the new pair of rows (b3, j, b4, j) and the comparison proceeds with those two rows as above.

VII. CODEBOOK MODEL

The proposed method operates on a fixed-to-variable codebook, Where in the fixed part consists of 8 * 8 blocks and the variable part comprises codes corresponding to the blocks. In order to devise an efficient and practical codebook, we performed a frequency analysis on a sample of more than 25000 8 * 8 blocks obtained by partitioning 120 randomly chosen binary data samples. By studying the natural occurrence of 8* 8 blocks in a relatively large binary data sample, the Law of Large Numbers motivates us to devise a general (empirical) probability distribution of such blocks. In principle, this could be used to construct a universal static model, which can be employed for compressing efficiently (on average) all sorts of bi-level images. The images have different sizes and varied from complex topological shapes, such as fingerprints and natural sceneries, to bounded curves and filled regions. Also, the main criterion in constructing an unbiased data sample was that the images should convey the clear meaning they 21 were constructed to convey without
unintentional salt-and-pepper noise. The images dimensions vary from 149*96 to 900*611 bits, yielding approximately 250,000 8 * 8 blocks. This image set is different from those image used to test JBIG2, for example. Before proceeding to the frequency analysis, we preprocessed binary images in two phases. The new image dimensions are w* h. The newly padded vector entries are filled with the respective background bit. If the background color in the binary image is white (represented conventionally with 0), the new entries will be filled with 0-valued bits. Having gone through the two aforementioned phases, we performed the frequency analysis on the 250,000 8*8 blocks and we used these relative probabilities to construct the codebook model. In terms of the 22 coding paradigm, we resorted to both Huffman and Arithmetic coding, as will be illustrated below. From an information theoretic standpoint, we consider the images in the data sample, exposed above, to have been generated by the hypothetical source described in Section 1.1. As such, the set of 8 * 8 blocks may be characterized as a discrete stochastic process defined over a very large discrete alphabet of size equal to 2^264 symbols that represent all possible patterns of zeros and ones in an 8 *8 block. Essentially, one can study the distribution of 8 * 8 blocks for a relatively large data sample. It is, however, not possible to estimate empirical probabilities for all 2^264 8 * 8 blocks, and it is certainly not feasible or time efficient to construct a codebook containing all possible blocks and the corresponding Huffman codes. Therefore, one should consider devising a codebook comprising the most frequently occurring blocks. In general, the more one increases block dimensions, the smaller the waiting probability of observing all possible blocks becomes because the size of the alphabet increases exponentially. Thus, it would be reasonable to have an expected value of the number of trial samples required to observe all the possible 8 * 8 blocks. The latter problem of determining the waiting probability of observing a particular number of blocks and the expected number of samples needed may be viewed as an instance of the more general Coupon-Collector is Problem, which is elegantly posed and is summarized as follows: Let S denote the set of coupons (or items, in general) having cardinality |S| = N. The column-reduction operation is applied on the row reduced block, as depicted in Fig. 2. The column-reference vector (CRV) is shown on top of the block. In this case, the first column is identical to and eliminates columns 2 to 6. Also, column 7 eliminates column 8. This results in the reduced block, RB, shown on the right of the column-reduced block. For this example, the output of the RCRC is a concatenated string composed of the RRV (the first group of 8 bits), CRV (the second group of 8 bits), and the RB (the last 4 bits) displayed as a vector: 101000001000001010111, for a total of 20 bits. The compression ratio achieved for this block is (64–20)/64 = 68.75%. To further clarify the decoding process, we consider the row-column reduced block of the preceding example. The number of ones in the row-reference and column reference vectors shows the number of rows and columns of the reduced block respectively. The output 101000001000001010111 contains two ones in the first group of 8 bits (the RRV), and two ones in the second group of 8 bits (the CRV). This means that there are 2 rows and 2 columns in the reduced block. That is, the first two bits of the reduced block i.e., ‘10’ of the underlined bits, represent the first reduced row, and the second two bits, ‘11’ of the underlined portion, represent the second reduced row. Then, given the ones and zeros in the reference vectors, we construct the rows and columns of the original block. Fig. 3 shows the column reconstruction process based on the column-reference vector. In the CRV tells the decoder that columns 2 to 6 are exact copies of column 1, and column 8 is an exact copy of column 7. The block on the right depicts this operation. It shows the row reconstruction process to obtain the original block.

Haar Transform:

Haar wavelet split the input signal into two signals called averages related to approximation coefficients and differences related to detail coefficients If we have an input signal sj, which has 2j samples sj, k, is split into two signals sj-1 with 2j-1 averages sj-1,k and dj-1 with 2j-1 differences dj-1,k. The averages sj-1 as a resolution representation of the signal sj and of the differences dj-1 as the information needed to go from the representation back to the original input signal. Applying the same transform to the coarser signal sj-1 itself, and iteratively repeating this process we obtain the averages and differences of successive levels, until obtain the signal s0 on the very coarsest scale, which a single sample s0,0, which is the average of all the samples of the original signal, that is the DC component or zero frequency of the signal.

![Figure 2. Haar Transform single level step and its inverse](image)

The whole Haar transform applying a N x N matrix (N = 2n) to the signal sn. The cost of computing the transform requires O (N) operations. The Haar transform uses a predictor which is correct in case the original signal is a constant. It eliminates the zeroth order correlation. The order of the predictor is one. Similarly the order of the update operator is one as it preserves the zero order moment.

![Figure 3. Haar Transform single level input, output signal](image)

i.Short Time Fourier Transform

The Fourier Transform is a great tool in signal processing which breaks down a signal into its constituent sine signal of different frequency components. Thus the Fourier analysis transforms our input signal view from time domain to frequency domain. But in Fourier analysis, when a particular event took place (i.e.) during transformation to the frequency
domain, time domain information is lost. If the signals don’t shift much over stationary signals, this drawback isn’t that much considered, but most signals contain numerous transitory characteristic or non-stationary and we lost very important information. The Short-Time Fourier Transform (STFT) was build up to correct this difficulty in Fourier analysis, while adapting the Fourier Transform to analyze the windowed sections of the signal along the time rather than frequency. Thus, the STFT maps the signal into a 2D function of frequency and time in a sort of compromise between the frequency- and time-based views of a signal. Even though, the STFT has a fixed resolution we obtain limited precision information, and that precision is resolved by the size of the window. By this way, a wide window gives better frequency resolution but poor time resolution, while a good time resolution but poor frequency resolution is given by narrower window. These are known as wideband and narrowband transforms.

**Figure 4. STFT original and reconstructed signal**

**Figure 5. Filtered reference input signal**

*Spectrum Analyzer*

Spectrum analyzer measures the input signal is electrical, but, spectral compositions of the other signals, such as optical light waves and acoustic pressure waves; can be considered through the use of a correct transducer. By analyzing the different spectra of electrical signals, dominant frequency power, harmonics of the signal, distortion, bandwidth of the signal, and other spectral components of a signal can be examined which are not easily evident in terms of time domain waveforms. As real-time spectrum analyzers, using a hybrid technique where the incoming signal is first converted to a lower frequency using super heterodyne and then analyzing using fast Fourier transformation (FFT) technique. The analyzers sample the incoming radio frequency spectrum in the time domain and convert the information to the frequency domain using the FFT process. FFT’s are constructed in parallel, without any gap and overlapped so there are no gaps in the calculated RF spectrum and no information is missed.

**VIII. RESULTS AND CONCLUSION**

IX. CONCLUSION

In this paper, we have introduced an innovative method for lossless compression of discrete-color and binary images. This method has a low complexity and it is easy to implement. The experimental simulation results on more than 150 images show that the proposed method enjoys a high compression ratio that in many cases was higher than 95%. This method has been successfully implemented on two major image categories: (i) images that consist of a predetermined number of discrete colors, such as digital maps, graphs, and GIS
images (ii) Binary images. The results of a large number of test images show that our method has compression ratio that are comparable or higher than the standard JBIG-2 by 5% to 20% for binary images, and by 2% to 6.3% for discrete color images. This proposed method works best on mid-to-small size images such as those on the Internet.

X. REFERENCES


