Volume 3, Issue 4, April 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Image Compression Techniques based On Wavelet and Huffman Coding

Ms. Pallavi M. Sune
Computer Science & Engineering, Amravati university
India.

Prof. Vijaya K. Shandilya.
Computer Science & Engineering, Amravati university
India.

***Abstract*— Image compression is now essential for applications such as transmission and storage in data bases. Compression becomes very easy techniques to apply without much technical requirement. In this paper we discuss about the image classification, wavelet compression, and converted an image into an array using Delphi image control tool. Image control can be used to display a graphical image Icon, Bitmap, metafile, GIF, JPEG, etc then an algorithm is created in Delphi to implement Huffman coding. Hence image compression has proved to be a valuable technique as one solution. The Wavelet Compression Engine was used in this study.**

***Keywords*— image, image compression techniques, Huffman coding, wavelete.**

I. INTRODUCTION

Image compression plays a very important role in application like televideo-conferencing, remote sensing, document and medical imaging, facsimile transmission (FAX) which depend on the efficient manipulation, storage and transmission of binary, gray scale or color images. Image compression technique can be classified into two categories lossless and lossy schemes.[1] In lossless method, the exact original data can be recovered while in lossy schemes only close approximation of the original data can be obtained. The signal representing images are usually in analog form, storage and transmission by computer applications, they are converted from analog to digital form. Digital image basically a 2-D dimensional array of pixels. Digital image require huge amounts of space for storage and large bandwidths for transmission. The primary goal of image compression is to minimize the memory footprint of image data so that storage and transmission times are minimized. Storage capacity can be limited, as in the case with digital cameras. Storage can be costly, as is the case when creating large warehouses of image data. Transmission of image data is also a central concern in many image processing systems. Recent studies of web use, for example, we have estimated that images and video account for approximately 85% of all internet traffic. Reducing the memory footprint of image data will correspondingly reduce internet bandwidth consumption more importantly however, since most web documents contain image data it is vital that the image data be transferred over the network within the reasonable time frame.

II. IMAGE COMPRESSION

Image compression refers to the reduction of the size of the data that images contain. Generally, image compression schemes exploit certain data redundancies to convert the image to a smaller form. A typical image compression system is shown in Figure (**1**). The data reduction, or compression, is performed by a device known as the encoder. The encoder reduces the data size of the original image X. The compressed image Y is the output which passes through a channel (usually an actual transmission channel or a storage system) to the decoder. The decoder reconstructs, or decompresses, the image Z from the compressed data. The ratio of the size (amount of data or bandwidth) of the original image to the size of the compressed image is known as the compression ratio or compression rate. The compression ratio can also be expressed in bpp (bits per pixel). The term bit rate is a general term for bpp. The higher the compression rate, the greater is the reduction of data [2]. Depending on the application, the channel may be affected by noise which results in distortion of the compressed image during transmission. If so, the channel is known as an error-prone channel; otherwise, it is errorless. In Fig. (**1**), the channel is assumed to be error-free, hence Y is the input to the decoder. Data compression schemes can be divided into two broad classes:

a) Lossless compression schemes, in which Z is identical to X
b) Lossy compression schemes, which generally provide much higher compression than lossless compression but allow Z to be different from X.

A compression algorithm can be evaluated in a number of different ways. We can measure the relative complexity of the algorithm, the memory required to implement the algorithm, how fast the algorithm performs on a given machine, the amount of compression, and how closely the reconstructed image resembles the original image. The main goal of data compression algorithm is to represent any given data at low bit rates [3].

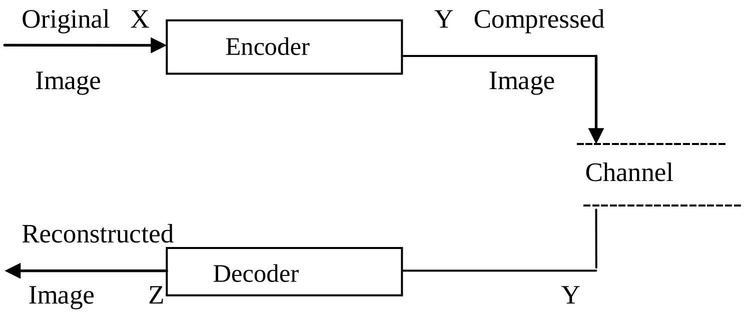

Figure 1 **Block diagram of image compression system.**

III. Image Classification

Based on the Normalized Compression Distance (NCD) presented in [3] and [4] which is one applications of Kolmogorov complexity, in [2] the authors perform image classification. The Normalized Information Distance (NID) is a similarity measure proportional to the length of the shortest program that represents *x* given *y*, as far as the shortest program that represents *y* given *x*.

$$NID(x,y) = \frac{\max\{K(x|y),K(y|x)\}}{\max\{K(x),K(y)\}} = \frac{K(x,y)-\min\{K(x),K(y)\}}{\max\{K(x),K(y)\}} \quad (1)$$

Since the Kolmogorov complexity *K(x)* is a non-calculable function, in [10] the authors approximate *K(x)* with *C(x)* where *C(x)* is the compression factor of *x*. Based on this approach we obtain the Normalized Compression Distance NCD (Normalized Compression Distance).

$$NCD(x,y) = \frac{C(x,y)-\min\{C(x),C(y)\}}{\max\{C(x),C(y)\}} \quad (2)$$

Where *C(x,y) i*s an approximation of the Kolmogorov complexity *K(x,y)* and represents the file size by compressing the concatenation of *x* and *y.* For image classification using the approach based on Normalized compression Distance (*NCD*); the first step is calculate the distance matrix between the images based on *NCD* using the equation 5, thus *dij* = *NCD(Ii,Ij)*. Thus, we can calculate the distance matrix *D* between all images *Ii* as:

$$D = \begin{pmatrix} d11 & d12 & dij \cdots din \\ d21 & d22 & d2j \cdots d2n \\ \vdots & \vdots & \vdots \quad \vdots \\ dn1 & dn2 & dnj \cdots \end{pmatrix}$$

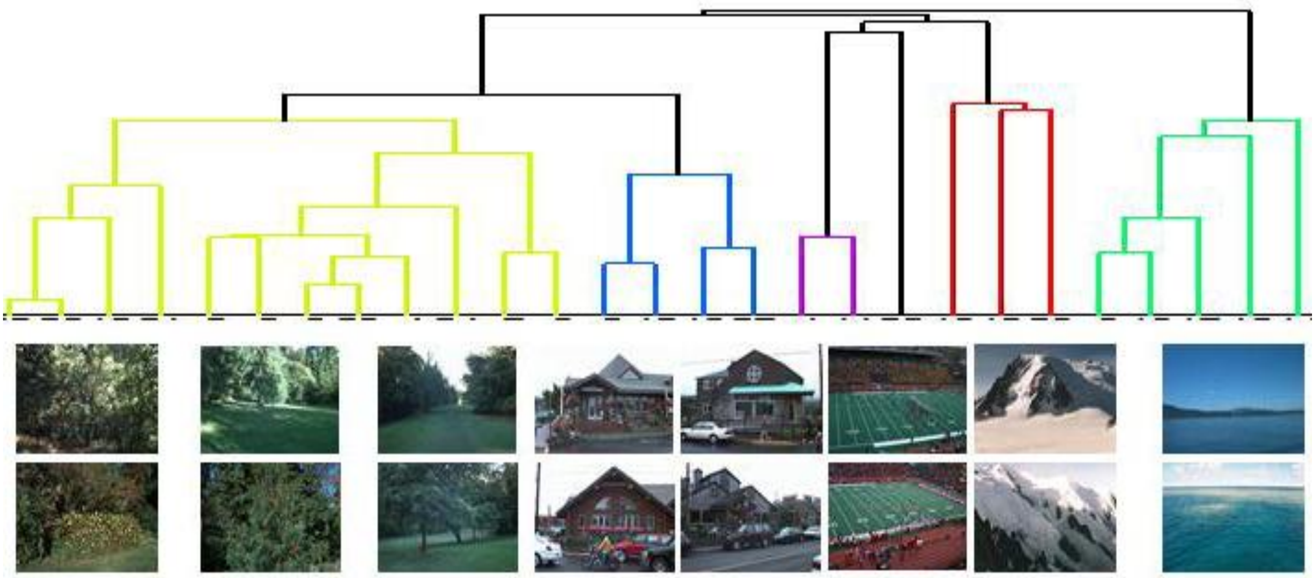

Figure2. Classification of Image

The matrix D is a square matrix of size n x n, where n is the number of images to classify. Finally we apply a hierarchical classification method as a dendrogram

IV. Wavelet And Image Compression

Wavelet transform image compression involves the use of a new field of applied mathematics often called 'wavelet theory' or simply "wavelets". Wavelet compression is a subset of a larger class of techniques generally referred to as "transform-based compression". The first step in a transform-based technique typically involves a lossless mathematical transform to provide a sparse representation of an input image. The transformed data are then quantized, in order to achieve the desired level of compression. Transform domain values that are quantized can never be restored to their original accuracy, but such quantization is necessary in order to achieve higher compression ratios. The greater the reduction in precision or quantization, the greater the compression ratio and the larger the error introduced into the compressed image [5]. The last step in transform-based compression is often referred to as "entropy coding" and involves the application of standard lossless compression techniques that may include run length encoding (RLE), Huffman coding, or arithmetic encoding. However, the Wavelet Compression Engine (standard edition 2.5) which we used in this study makes it practical to store a large amount of data. This standard uses a lossy compression method and wavelet image format (WIF) which has the power to reduce an image size from 1 Mbyte to 8KB without losing the image quality. The Compression Engine Pro also allows compression of multiple image files simultaneously, using batch compression. Furthermore, this standard supports many image types. For best results, it is recommended that one begins with images in uncompressed formats such as BMP or TIFF, but even with compressed formats such as JP

V. Huffman Coding

Proposed by DR. David A. Huffman in 1952. "A method for the construction for minimum redundancy code." Huffman code is technique for compressing data. Huffman made significant contributions in several areas. Mostly information theory and coding signal design for radar and communication & design procedures for asynchronous logical circuits. Huffman coding is a form of statistical coding which attempt to reduce the amounts of bits required representing the string of symbols to vary in length.[6] Shorter codes are assigned to the most frequently used symbols & longer codes to the symbol which appear less frequently in the string. Code word length is no longer fixed like ASCII.

A. Algorithm and Working Of Huffman Coding

The most popular techniques for removing coding redundancy is due to Huffman Algorithm of Huffman. It is a fixed-to-variable length code, that is, it maps fixed length input symbols to variable length code words. It reduces the average codeword length by assigning shorter codeword's to highly frequent symbols and longer codeword's to rarely occurring symbols. [7] In this paper we have converted an image into an array using Delphi image control tool

Following steps describes working and algorithm for Huffman coding. [8], [9], [10]

1. Read a BMP image using image box control in Delphi language. The TImage control can be used to display a graphical image-Icon (ICO), Bitmap (BMP), Metafile (WMF), GIF, JPEG, etc. This control will read an image and convert them in a text file.
2. Call a function that will sort or prioritize characters based on frequency count of each character in file.
3. Call a function that will create an initial heap. Then reheap that tree according to occurrence of each node in the tree, lower the occurrence earlier it is attached in heap. Create a new node where the left child is the lowest in the sorted list and the right is the second lowest in the sorted list.
4. Build Huffman code tree based on prioritized list. Chop-off those two elements in the sorted list as they are now part of one node and add the probabilities. The result is the probability for the new node.
5. Perform insertion sort on the list with the new node.
6. Repeat STEPS 3,4,5 UNTIL you only have 1 node left.
7. Perform a traversal of tree to generate code table. This will determine code for each element of tree in the following way.

The code for each symbol may be obtained by tracing path to the symbol from the root of the tree. A 1 is assigned for a branch in one direction and a 0 is assigned for branch in the other direction. For example a symbol which is reached by branching right twice, then left once may be represented by the pattern '110'. The figure below depicts codes for nodes of a sample tree.

```
        *
       / \
     (0)   (1)
          / \
      (10) (11)
           /  \
      (110) (111)
```

8. Once a Huffman tree is built, canonical Huffman codes, which require less information to rebuild, may be generated by the following steps: step1. Remember the lengths of the codes resulting from a Huffman tree generated per above. Step2. Sort the symbols to be encoded by the lengths of their codes (use symbol value to breakties). Step3. Initialize the current code to all zeros and assign code values to symbol from longest to shortest code as follows:

a. If the current code length is greater than the length of the code for the current symbol, right shift off the extra bits.
b. Assign the code to the current symbol.
c. Increment the code value.
d. Get the symbol with the next longest code.
e. Repeat from A until all symbols are assigned codes.

9. Encoding Data-once a Huffman code has been generated, data may be encoded simply by replacing each symbol with its code.
10. The original image is reconstructed i.e. decompression is done by using Huffman Decoding.
11. Generate a tree equivalent to the encoding tree. If you know the Huffman code for some encoded data, decoding may be accomplished by reading the encoded data one bit at a time. Once the bits read match a code for symbol, write out the symbol and start collecting bits again.
12. Read input character wise and left to the tree until last element is reached in the tree.
13. Output the character encodes in the leaf and returns to the root, and continues the step 12, until all the codes of corresponding symbols is known.

Figure3. **Original Grayscale Image**

Figure4. Reconstructed Grayscale Image

Fig.3 shows original Grayscale bit map Image then is successfully reconstructed using this Huffman coding method and the regenerated image is exactly same as original image that is represented by Fig.4. For grayscale image this method gives 60% saving in size.

VI. Conclusion

In this paper we have to present classification of image, and Normalized Information Distance (NID) is measure, and wavelet of image compression. The best technique Huffman coding in lossless compression, in that the image uncompressed need have some specific knowledge of the symbol of probabilities in the compressed files and this can need more bit to encode the file also Huffman coding required two passes if the information is unavailable compressing the file: find frequency of each symbol and construct tree Huffman to compress the file.

ACKNOWLEDGMENT

We would like to thank our Inspiration our principle Dr. Sidharth Ladhake sir, our project Guide Prof. V. K. Shandilya madam, and all teacher staffs and friends who supported us for the completion of project.

References

[1] Rafale C. Gonzalez, Richard E Woods “Digital image processing”, Second Edition page 410.

[2] M. R. Quispe-Ayala, K. Asalde-Alvarez, A. Roman-Gonzalez, “Image Classification Using Data Compression Techniques”; 2010 IEEE 26th Convention of Electrical and Electronics Engineers in Israel – IEEEI 2010; Eilat – Israel; November 2010, pp. 349-353.

[3] M. Li and P. Vitanyi, “The Similarity Metric”, IEEE Transaction on Information Theory, vol. 50, N° 12, 2004, pp. 3250-3264.

[4] R. Cilibrasi, P. M. B. Vitanyi; “Clustering by Compression”, *IEEE Transaction on Information Theory*, vol. 51, N° 4, April 2005, pp 1523 - 1545.

[5] R. Edmund, “Wavelet Compression of Medical Imagery”. *Tele-medicine Journal*, Vol. 2, 1996, pp. 131-137.

[6] Mamta Sharma, compression Using Huffman Coding, IJCSNS International Journal of Computer Science and Network Security, VOL.10 NO.5, May 2010,pp 133-141.

[7] D. A. Huffman. A Method for the construction of minimum redundancy codes. In proceedings of the IRE, volume 40, 1952

[8] D.A. Huffman, “A Method for the Construction of Minimum-Redundancy Codes”, Proceedings of the I.R.E., September 1952, pp 1098-1102. Huffman original article.

[9] D. E. Knuth- Dynamic Huffman Coding- Journal of Algorithms, 6, 1983 pp. 163-180.

[10] Lakhani, G, Modified JPEG Huffman coding, IEEE Transactions Image processing, 12(2), 2003 pp. 159-169.