Journal of Advances in Science and Technology

Vol. VIII, Issue No. XVI, February-2015, ISSN 2230-9659

LOSSLESS IMAGE COMPRESSION VIA BIT-PLANE SEPARATION AND MULTILAYER CONTEXT

AN INTERNATIONALLY INDEXED PEER REVIEWED & REFEREED JOURNAL

www.ignited.in

Journal of Advances in Science and Technology
Vol. VIII, Issue No. XVI, February-2015, ISSN 2230-9659

Lossless Image Compression via Bit-Plane Separation and Multilayer Context

Ranjeet Yadav

Research Scholar, Sathyabama University, Chennai, Tamilnadu

Abstract – Lossless image compression is needed for applications that cannot tolerate any degradation of original imagery data, e.g., medical applications such as mammography, angiography, and x-rays. It is essential that the decompressed image does not contain any degradation in quality, since it could lead to misdiagnosis and health injury. Satellite or geographical map images are another case where distortion caused by compression cannot be tolerated. In this paper we focused about Lossless Image Compression via Bit-Plane Separation and Multilayer Context.

Keywords: Image Compression, Decompressed, Pixels and Image

----------♦----------

INTRODUCTION

The earliest lossless compression methods used either dictionary-based methods or run-length encoding. However, these techniques do not exploit 2-D correlations in the image, and they are not very efficient for natural images that contain smooth color variations but do not have repeating patterns [1]. Predictive modeling, on the other hand, exploits spatial correlations by predicting the value of the current pixel by a function of its already coded neighboring pixels. The difference between the actual and predicted value, called prediction error, is then encoded. A simple linear prediction is used in the lossless mode of the JPEG still compression standard and a nonlinear predictor in the newer JPEG-LS standard [3]. Despite their apparent simply prediction-based techniques are quite effective and used in state of the art compression methods. Another approach is to use context modeling followed by arithmetic coding. In context-based models, every distinctive pixel combination of the neighborhood is considered as its own coding context. The probability distribution of the pixel values is estimated for each context separately based on past samples. In grayscale images, however, the number of possible pixel combinations is huge and only a small neighborhood can be used. The number of contexts must therefore be reduced by context quantization [4]. This approach, combined with predictive modeling, has been used in the context-based adaptive lossless image compression [5].

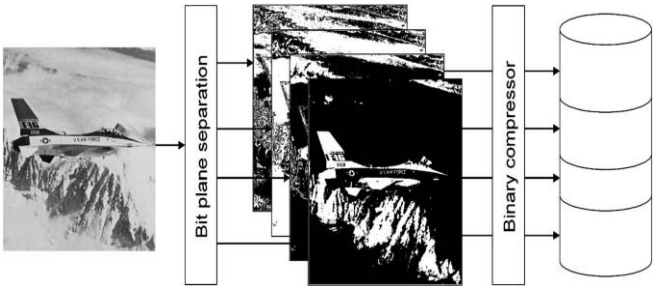
REVIEW OF LITERATURE:

The recent JPEG20006 compression is based on wavelet transform, and although this algorithm is aimed at loss compression, it also includes a lossless variant. The efficiency of the prediction scheme also depends on the type of image. For example, CALIC is efficient on photographic images but not so good on images that contain smaller amounts of color gradation see image, e.g., color palette images, web graphics and its recognition [2], geographical maps, schemes, and diagrams. On the other hand, a method called the piecewise-constant model has been optimized for this type of image. The algorithm is a two-pass method. In the first pass, it uses special classification to establish boundaries between constant color pieces in the image. In the second pass, the decisions are coded by a binary arithmetic coder. The method also takes advantage of uniform regions where the same context repeatedly appears. One approach for exploiting spatial correlations efficiently is to decompose the image into a set of binary layers, and then compress the layers by a binary image compression method such as JBIG. The advantage of this approach is that a much larger neighborhood can be applied in the context model than when operating on the grayscale values. The decompression process is reversed: the compressed file is decompressed into a set of layers, which are then combined back into the grayscale image. Unfortunately, JBIG is not very efficient when applied to bit-plane separated layers, as it is on images that are binary by their origin. Typically, the bit layers especially less significant bits lack predictable structure to be compressed well. This is because the bit-plane separation destroys the gray-level correlations of the original image, making the compressor unable to exploit them when coding the bit planes separately. In fact, interlayer dependencies are stronger than spatial dependencies within the layers. Embedded image-domain adaptive compression of simple images EIDAC9 therefore

uses a 3-D context model, where context pixels are selected not only from the current bit plane but also from the already processed layers. Another way to improve compression performance is to increase the size of the context template. A larger context can be achieved by a selective context expansion using context tree CT, which allocates memory only for contexts that are really present in the image. The size as well as the ordering of the pixels within the context can be optimized [6]. An attempt to spread the optimized context tree modeling to a multilayer case called multilayer context tree modeling has been made in the case of multilayer geographical map images [7]. Optimal ordering of the layers was shown to give additional improvement [8]. In general, the efficiency of the particular compression method depends on the utilization of color and spatial dependencies. Prediction-based algorithms concentrate mainly on color dependencies, since they are looking for correlation between gray values in a relatively small spatial neighborhood. On the other hand, binary image compression algorithms concentrate more on utilizing spatial dependency than color dependencies. Binary nature of the input data makes it possible to use a larger spatial context template, but when applied to bit-plane separated data, the compression efficiency is low, since there are more interlayer color dependencies than spatial dependencies among the neighboring bits. A successful compression method should utilize both types of dependencies. We study how well the bit-plane-based approach can work on natural and palette images. We apply the MCT method, but instead of the color separation, we perform bit-plane separation because of a higher number of colors in the images. We consider four different methods: a straightforward bit-plane separation as such, gray coding, a separate prediction step, as well as the combination of the last two. Furthermore, we extend the two layer context model to a multilayer context model for better utilization of the cross-layer dependencies. In general, one can use any previously compressed layer as the reference layers. The first layer is compressed as such, the second layer can use the first one as the reference layer, and the process continues so that the last layer can use all previous layers. We denote this extension as an N-layer context tree modeling NCT.

Test set of simple images.

The rest of the work is organized as follows. The aspects concerning context modeling, context tree modeling, and multilayer context trees are described. Different alternatives for bitplane decomposition are studied. The performance of the proposed schemes is evaluated against the most competitive algorithms both for natural and palette images. Finally, conclusions are drawn. Probability estimation can be achieved using a larger context template. However, it does not always result in compression improvement, because the number of contexts grows exponentially with the size of the template. This leads to the context dilution problem, in which the statistics are distributed over too many contexts, and thus affects the accuracy of the probability estimates.



Lossless compression of grayscale images by a binary-image-oriented compression.

MULTILAYER CONTEXT TREE MODELING:

Statistical image compression consists of two phases: modeling and coding. In the modeling phase, the probability distribution of the pixels to be compressed is adaptively estimated [9, 10]. The coding process assigns variable length code words to the pixels according to the probability model, so that shorter codes are assigned to more probable pixels and vice versa. The coding is performed by arithmetic coding

using implementation known as a QM-coder, which was originally introduced for the JBIG standard.

➢ **Context Modeling:**

The probability of a pixel is conditioned on a context, which is defined as the black-white configuration of the neighboring pixels within a local template. The index of the selected context and the pixel to be coded are then sent to the arithmetic coder. In principle, better probability estimation can be achieved using a larger context template. However, it does not always result in compression improvement, because the number of contexts grows exponentially with the size of the template. This leads to the context dilution problem, in which the statistics are distributed over too many contexts, and thus affects the accuracy of the probability estimates.

➢ **Context Tree:**

The context tree CT concept provides a more flexible approach for modeling the contexts so that a larger number of neighbor pixels can be taken into account without the context dilution problem. The contexts in CT are represented by a binary tree, in which the context is constructed pixel by pixel. The context selection is deterministic and only the leaves of the tree are used. The location of the next neighbor pixels and the depth of the individual branches of the tree depend on the combination of the already coded pixel values. The tree can be constructed beforehand using a training image static approach or optimized directly to the image to be compressed semi adaptive approach We use the latter approach because it optimizes the structure and size of the tree directly to the input image without any parameter tuning or prior training. The structure of the tree must then be stored in the compressed file, and it takes 1 bit per node. In the case of our test sets see images, this corresponds to a 10 to 25% proportion of the compressed file. A variant called free tree10 optimizes the location of the template pixels adaptively at each step of the tree construction, when a new child node is created, every possible.

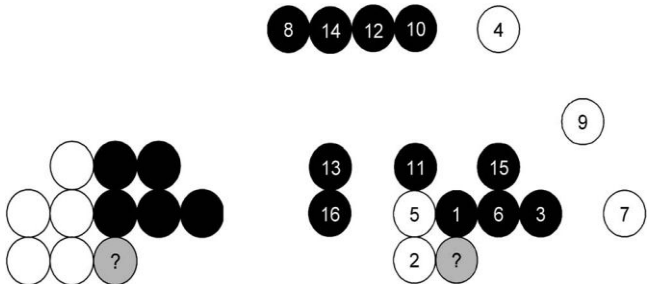


Image of Sample contexts defined by JBIG 10-pixel template left, and the template optimized for a geographical image right. The numbers refer to the order in which the pixels have been selected for this particular context, location for the next context pixel is considered within a predefined search area and the compression efficiency is estimated by the entropy of the current context model HN.

➢ **Two-Layer Context Tree:**

The CT modeling can be extended to the multilayer case, called MCT, by defining a context template where pixels from previously coded layers can also be included. In this way, information from other bit layers, called reference layers, can compensate the loss of color correlation caused by the bit-layer separation. A two-layer model was considered in Ref. 10 using a search area consisting of 40 pixels from the current layer. The pixels in the current layer can be located in the neighborhood area including already coded pixels, but the pixels in the reference layer can be located anywhere, since they are already known by the decoder, as the reference layer is always coded before the current one. Further optimization exploits the fact that the efficiency of the compression of any particular layer strongly depends on the choice of the reference layer. In general, we can select any predefined order on the basis of known or assumed dependencies. When image source is not known beforehand, the optimal order of the layers can be solved as a directed minimum spanning tree problem13 for maximal utilization of the interlayer dependencies. Again, the optimization comes at the price of a remarkable increase in the processing time.

➢ **N-Layer Context Tree:**

In this work we generalize the idea by considering the N-layer context tree, further referred as NCT. We consider all previously compressed layers as reference layers. When the first layer is compressed, the free-tree context template involves only already processed pixels of the current layer. After being compressed, this layer becomes a reference layer for the second one illustrates the search area used for the compression of the third layer. It consists of 52 pixel positions, of which ten are from the current layer and 42 are from the reference layers. Each template position is examined for the provided compression gain, and the most efficient position is included in the template at each step. The process then continues as long as further improvement will be achieved. A sample context is illustrated. The ordering of the layers affects the compression performance in NCT in the same way as in MCT. For example, when test image Airplane is compressed starting from the least significant bit LSB toward the most significant bit MSB, the obtained total code size would be 148,388 bytes. On the other hand, when compressed with reversed ordering from MSB toward LSB, the code size is 136,185 bytes. The optimal ordering was solved as a directed minimum spanning tree problem, which was possible because only one previous layer

was used as a reference layer. In the case of an N-layer context tree, similar formulation would lead to a traveling salesman problem. In this case, the optimal solution would take and an even faster heuristic would influence the processing time significantly because of a larger search area. Fortunately, the optimal ordering is not as critical as in the MCT, and therefore, we used a fixed order starting from MSB to LSB. A common property of the context-based techniques is that in the case when the statistical dependencies of the source are extremely weak, the code size produced by the compressor could be even greater than the size of the uncompressed file. This issue is especially essential for the compression of the less significant bit layers, which are quite noisy. In this situation, we transmit the uncompressed bit layer as such.

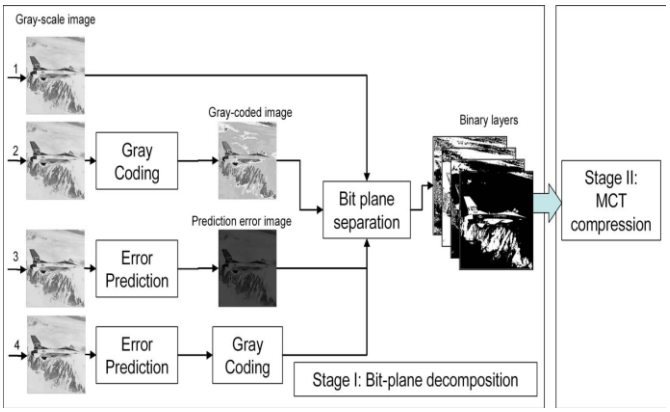
METHODS FOR BIT-PLANE SEPARATION:

The proposed grayscale compression scheme consists of two independent lossless stages. In the first stage, the grayscale image is decomposed into a set of binary images layers. In the second stage, the MCT or NCT compression method is applied. The decompression is performed in reverse order: first, an archive file is decompressed into a set of binary layers, which are then combined into a grayscale image. We consider the following four decomposition methods:

- Bit-plane separation (BPS)
- Gray code separation (GCS)
- Prediction error separation (PES)
- Gray code prediction error separation (GCPES).

The first scheme is a straightforward bit-plane separation scheme, which is a classical method for creating bit planes where each pixel value corresponds to a particular bit of the original grayscale image. The second scheme is a gray-code separation which codes the pixel intensities so that the change of pixel value by +1 or −1 causes the change of only 1 bit value in the corresponding bit layers.

For example, when the gray code is not applied, increasing value 127 01111111b by 1 gives 128 10000000b, which causes changes in all eight bit layers. On the other hand, the gray code for 127 is 64 01000000b, and for 128 it is 192 11000000b, which differ in 1 bit only. Gray coding has turned out to be an efficient preprocessing technique for improving compression performance.19 The third scheme uses a separate prediction step followed by bit plane separation20,21. The idea is to encode the prediction error, i.e., the difference between the predicted and the actual value of a pixel, instead of the original gray value. Error prediction is a lossless transformation converting a grayscale image of gray values varying from 0 to 255 into a so-called prediction error image, where every pixel represents the prediction error varying from −255 to +255. Therefore, when using this scheme, the grayscale image is decomposed into nine binary layers instead of eight as in the first two schemes. When the predictor is effective, the prediction error values are mostly concentrated around zero. Therefore, after bit plane decomposition, more significant bit planes contain a very small amount of variation, thus having low entropy and resulting in high compression ratio. The bit layers produced by the four different bit plane separation schemes for the image Airplane are illustrated. An important design question is the choice of prediction technique. In this works, we considered three popular predictors in order to choose the most efficient for further use. The first scheme is a simple linear predictor defined as: where (x, y) is the pixel value at coordinates x and y. This is referred to further as linear. The second technique is a slightly more complicated prediction method employed in the JPEG-LS compressor, which we refer to here as a median predictor. Finally, for the third scheme we have chosen the gradient-adjusted prediction algorithm used in CALIC, which is the most complicated of the three predictors considered. This predictor is referred here to as GAP.



Overall compression algorithm according to the different bit-plane decomposition schemes

CONCLUSION:

Image quality improvement/enhancement has been a concern throughout the area of image processing. Images are affected undesirable because it degrades image quality. An application of reduction in an image processing is a promising research fields. The accurate prediction of quality from an end-user perspective has received increased attention with the growing demand for compression and communication of digital image and video services over wired and wireless networks.

REFERENCES:

[1] S. Ablameyko, T. Pridmore, Machine Interpretation of Line Drawing Images, Springer, 2000.

[2] A.K. Chhabra, D. Dori (eds.), Graphics Recognition – Recent Advances, Lecture Notes on Computer Science, vol. 1941, Springer, Verlag, 2000.

[3] NLS: National Land Survey of Finland, Opastinsilta 12 C, P.O.Box 84, 00521 Helsinki, Finland. http://www.nls.fi/index_e.html.

[4] CIE, Colorimetry, CIE Pub. No. 15.2, Centr. Bureau CIE, Vienna, Austria, 1986.

[5] K. Sayood, Introduction to Data Compression, 2nd edition, Academic Press, 2000.

[6] D. Salomon, Data Compression: The complete reference, 3rd Edition, Springer Verlag, 2004.

[7] I.H. Witten, A. Moffat, T.C. Bell, Managing Gigabytes: Compressing and Indexing Documents and Images. 2nd Edition, Morgan Kaufmann, 1999.

[8] Graphics Interchange Format(sm), Version 89a, http://www.dcs.ed.ac.uk/home/mxr/gfx/2d/GIF89a.txt, 1990.

[9] S. W. Thomas, J. Mckie, S. Davies, K. Turkowski, J. A. Woods, and J. W. Orost. Compress (version 4.0) program and documentation, 1985.

[10] T. Welch, “A technique for high-performance data compression”, Computer Magazine, vol.17(6), pp. 8-19, 1984.