Lossless image compression via bit-plane separation and multilayer context tree modeling

Alexey Podlasov
Pasi Fränti
University of Joensuu
Speech and Image Processing Unit
Department of Computer Science
P.O. Box 111
80101 Joensuu, Finland
E-mail: apodla@cs.joensuu.fi

Abstract. *Color separation and highly optimized context tree modeling for binary layers have provided the best compression results for color map images that consist of highly complex spatial structures but only a relatively few number of colors. We explore whether this kind of approach works on photographic and palette images as well. The main difficulty is that these images can have a much higher number of colors, and it is therefore much more difficult to exploit spatial dependencies via binary layers. The original contributions of this work include: 1. the application of context-tree-based compression (previously designed for map images) to natural and color palette images; 2. the consideration of four different methods for bit-plane separation; and 3. Extension of the two-layer context to a multilayer context for better utilization of the crosslayer correlations. The proposed combination is extensively compared to state of the art lossless image compression methods.* © *2006 SPIE and IS&T.* [DOI: 10.1117/1.2388255]

1 Introduction

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.

The earliest lossless compression methods used either dictionary-based methods or run-length encoding.[1] 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. *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.[1] 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.[2] Despite their apparent simplicity, 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.[3] 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 (CALIC) algorithm.[5] The recent JPEG2000[6] compression is based on wavelet transform, and although this algorithm is aimed at lossy 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 (see Fig. 1) but not so good on images that contain smaller amounts of color gradation (see Fig. 2), e.g., color palette images, web graphics, geographical maps, schemes, and diagrams. On the other hand, a method called the *piecewise-constant model* (PWC)[7] 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, as demonstrated in Fig. 3, and then compress the layers by a binary image compression method such as JBIG.[8] 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

Paper 05204RR received Nov. 23, 2005; revised manuscript received Jun. 22, 2006; accepted for publication Aug. 2, 2006; published online Nov. 22, 2006.
1017-9909/2006/15(4)/043009/11/$22.00

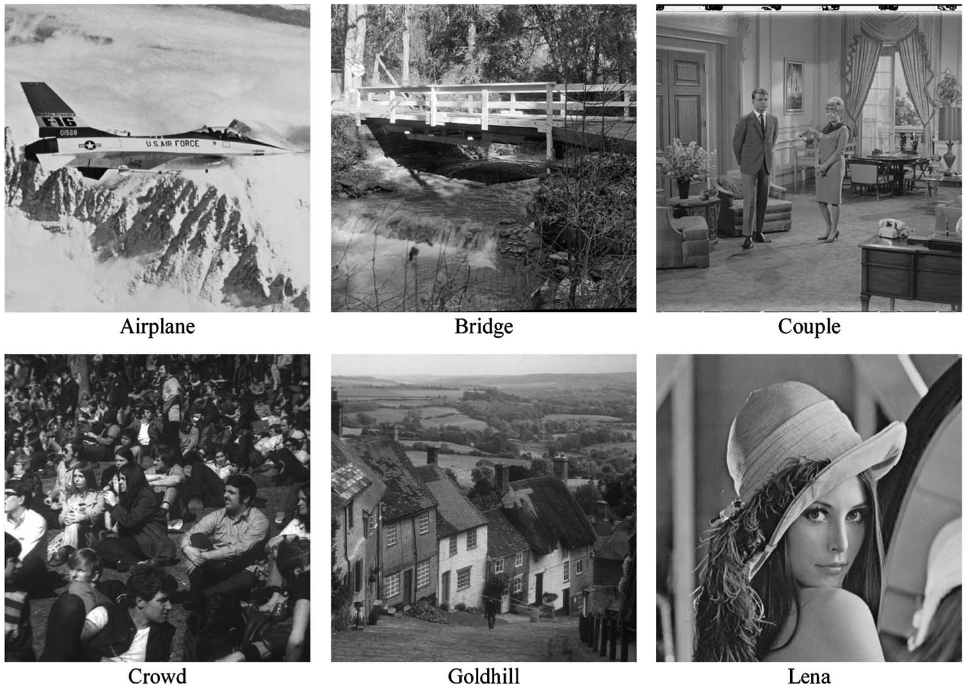

Fig. 1 Test set of natural images.

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* (EIDAC)[9] 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),[10] 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.[11] An attempt to spread the optimized context tree modeling to a multilayer case called *multilayer context tree* (MCT) modeling has been made in the case of multilayer geographical map images.[12] Optimal ordering of the layers was shown to give additional improvement.[13]

In general, the efficiency of the particular compression method depends on the utilization of *color* and *spatial* dependencies (see Fig. 4). 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 presented in Ref. 13, 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

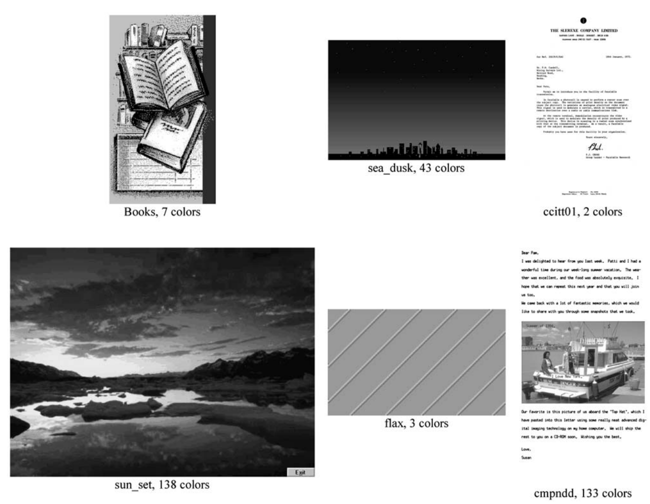

Fig. 2 Test set of simple images.

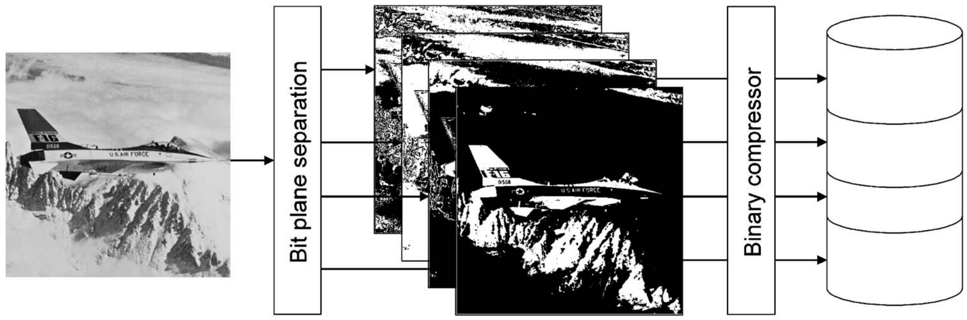

Fig. 3 Lossless compression of grayscale images by a binary-image-oriented compression.

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).

The rest of the work is organized as follows. The aspects concerning context modeling, context tree modeling, and multilayer context trees are described in Sec. 2. Different alternatives for bitplane decomposition are studied in Sec. 3. The performance of the proposed schemes is evaluated in Sec. 4 against the most competitive algorithms both for natural and palette images. Finally, conclusions are drawn in Sec. 5.

2 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. 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*[14] using implementation known as a QM-coder,[15] which was originally introduced for the JBIG standard.

2.1 *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 (see Fig. 5). 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,[16] in which the statistics are distributed over too many contexts, and thus affects the accuracy of the probability estimates.

2.2 *Context Tree*

The *context tree* (CT) concept[10] 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),[17] or optimized directly to the image to be compressed (semiadaptive approach).[10] 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 Sec. 4), this corresponds to a 10 to 25% proportion of the compressed file.

A variant called *free tree*[10] optimizes the location of the template pixels adaptively at each step of the tree construction. When a new child node is created, every possible

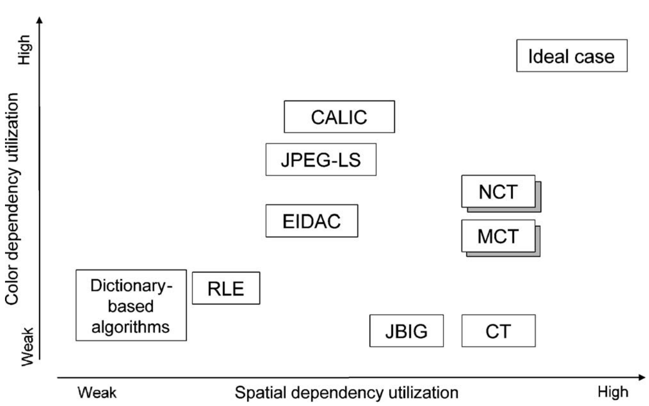

Fig. 4 Spatial and color dependency diagram. The algorithms considered in this work are emphasized by shadow.

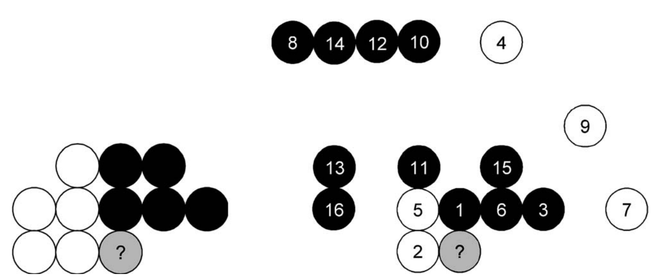

Fig. 5 Sample contexts defined by JBIG 10-pixel template (left), and the template optimized for a geographical image (right). The numbers refer to the order in which the pixels have been selected for this particular context.

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 H_N. The entropy is calculated as the sum of entropies of individual contexts:

$$H_N = -\sum_{j=1}^{N} p(C_j)(p_w^{C_j} \cdot \log_2 p_w^{C_j} + p_b^{C_j} \cdot \log_2 p_b^{C_j}),$$

where $p(C_j)$ is the probability of the context C_j, $p_w^{C_j}$ and $p_b^{C_j}$ are the probabilities of the white and black pixels in the context C_j, and N is the total number of contexts. The probabilities $p_w^{C_j}$ and $p_b^{C_j}$ are calculated on the basis of observed frequencies.[10] The position providing the best estimated-compression gain is included into the context rtemplate. The optimization, however, comes at the cost of additional computation time and increase in tree storage size. A sample context optimized by the free tree is demonstrated in Fig. 5.

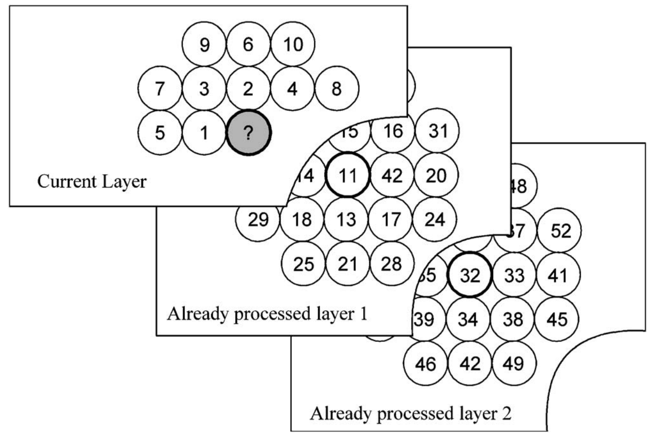

Fig. 6 Joint 52-pixel three-layer search area. The position of the current pixel is marked with "?" and the corresponding positions on the reference layers are emphasized with bold circles.

2.3 *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. 12 using a search area consisting of 40 pixels from the current layer, and 37 from the reference 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* problem[13] for maximal utilization of the interlayer dependencies. Again, the optimization comes at the price of a remarkable increase in the processing time.

2.4 *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. Figure 6 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 in Fig. 7.

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. In Ref. 13, 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 $O(n!)$, 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

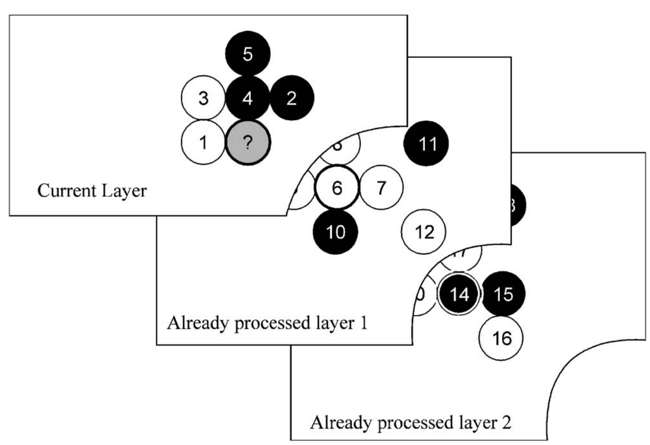

Fig. 7 A sample three-layer context constructed by the free-tree approach using the search area presented in Fig. 6. The black color represents 1 bit and white color represents 0 bits in the corresponding bit layer. The current pixels position is emphasized with a bolder circle.

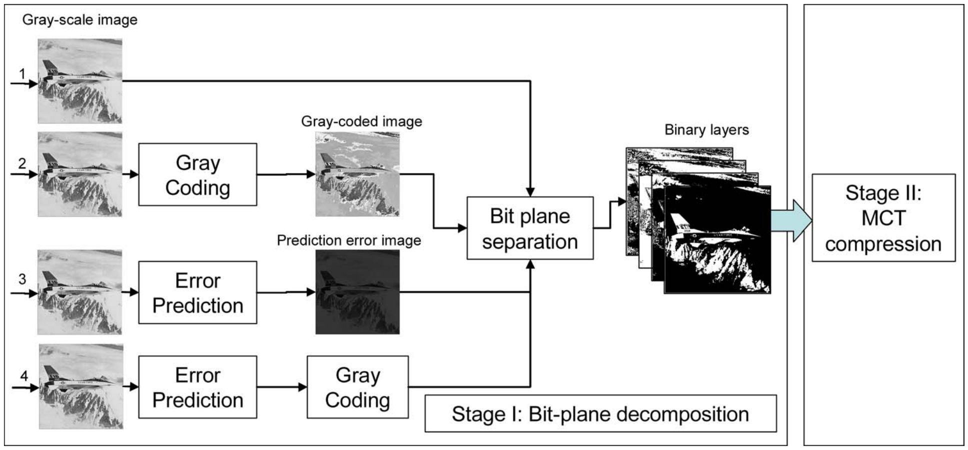

Fig. 8 Overall compression algorithm according to the different bit-plane decomposition schemes.

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.

3 Methods for Bit-Plane Separation

The proposed grayscale compression scheme consists of two independent lossless stages as shown in Fig. 8. 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 1 in Fig. 8), 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 (scheme 2 in Fig. 8),[18] 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. This transform is defined as

$$x \to G(x) = x \oplus (x \gg 1),$$

where $\oplus$ indicates the "exclusive-or" function, and » indicates the "binary shift-right" operation (i.e., $m \gg n = m/2^n$). 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 separation[20,21] (scheme 3 in Fig. 8). 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 fourth scheme (scheme 4 in Fig. 8) employs gray coding of the prediction error image with the following bit-plane separation. The bit layers produced by the four different bit plane separation schemes for the image Airplane are illustrated in Fig. 9.

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:

$$p(x,y) = \frac{(x,y-1)}{2} + \frac{(x-1,y)}{2} + \frac{(x+1,y-1)}{4} + \frac{(x+1,y+1)}{4},$$

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,[2] which we refer to here as a *median* predictor. Finally, for the third scheme we have chosen the gradient-adjusted prediction (GAP) algorithm used in CALIC,[5] which is the most complicated of the three predictors considered. This predictor is referred here to as *GAP*.

4 Experiments

We used two test image sets to evaluate the algorithms. The first set consists of five classical test images: Airplane,

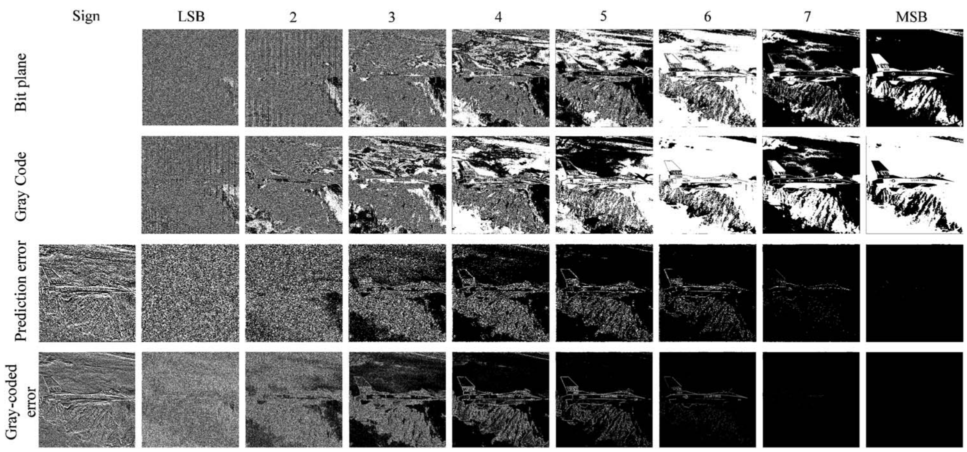

Fig. 9 Four bit-plane decomposition schemes applied to the image Airplane. Columns correspond to the bit planes starting with the sign bit, and continuing from the least significant bit to the most significant bit. Rows are different bit-plane decompositions.

Couple, Crowd, Goldhill, and Lena (see Fig. 1). All of them are 8-bit grayscale images of size 512×512 pixels. This test set represents a class of natural images that are typically photographic images of smooth color gradation. The second test set represents a class of palette images (see Fig. 2), where the number of colors is much smaller than the amount of pixels in the image. Such images can be Web graphics, schemes, maps, slides and engineering drawings, for example. This test set consists of eight images used in Ref. 7, Benjerry, Books, Ccit01, Cmpndd, Flax, Gate, Sea_dusk, and Sunset.

First, we evaluate the performance of the three prediction techniques: linear, median, and GAP predictors. Then we evaluate six variants of the proposed algorithms produced by the combination of the two context modeling schemes (MCT and NCT), and the three bit-plane decomposition schemes (BPS, GCS, and the best prediction-based scheme). Finally, we compare the best variants with the existing compressors. The competitive algorithms are:

- JBIG-GCS: JBIG applied to gray code separated layers[18,19]
- JBIG-PES: JBIG applied to prediction error separated layers
- EIDAC[9]
- CALIC[5]
- JPEG-LS[2]
- PWC-G: piecewise constant model optimized for grayscale images[7]
- PWC-P: piecewise constant model optimized for palette images[7]
- JPEG2000[6]
- PNG.

Results for EIDAC are taken from Ref. 9 and appear only for the set of palette images. The rest of the results are reported using publicly available implementations. All tests have been performed on a Pentium III 996-MHz computer with 384-MB memory and a Windows XP operating system.

4.1 *Choice of the Predictor*

We tested the performance of the three prediction techniques to choose the best for further comparison. Tables 1 and 2 summarize the overall compression performance of the PES and GCPES variants depending on the choice of predictor. In the case of natural images, the GAP predictor provides the best compression performance with all variants. The best performance (4.42 bpp) is obtained by the MCT-GCPES variant. In the case of palette images, the median predictor works better and the best result is obtained by MCT-PES (210,718 bytes). In the rest of the work, we apply the GAP predictor for natural images and the median predictor for palette images.

Table 1 Average compression results (bits per pixel) depending on the choice of predictor for the natural images.

	MCT		NCT	
Predictor	PES	GCPES	PES	CGPES
Linear	4.53	4.49	4.52	4.50
Median	4.49	4.48	4.50	4.48
GAP	4.45	**4.42**	4.44	4.43

Table 2 Total compression results (in bytes) depending on the choice of predictor for the palette images.

	MCT		NCT	
Predictor	PES	GCPES	PES	CGPES
Linear	282,280	274,525	296,877	290,815
Median	**210,718**	211,686	221,546	221,921
GAP	279,924	277,994	288,161	286,051

Table 3 Compression results (bits per pixel) for the natural images.

	MCT			NCT		
Image	BPS	GCS	GCPES	BPS	GCS	GCPES
Airplane	4.60	4.21	**4.05**	4.13	4.16	4.08
Couple	5.22	4.68	**4.54**	4.78	4.72	**4.54**
Crowd	4.76	4.38	**4.09**	4.13	4.14	4.24
Goldhill	5.68	5.10	4.96	5.02	5.08	**4.88**
Lena	5.25	4.62	4.46	4.58	4.59	**4.41**
Average	5.10	4.59	**4.42**	4.53	4.54	4.43

4.2 *Comparison of the Proposed Variants*

Here we evaluate the proposed algorithms over two test sets separately and choose the most efficient variants for further comparison. Table 3 presents the compression results for the natural image test set. The best result (4.42 bpp, on average) was obtained by MCT using both the prediction and gray coding (GCPES), but the difference from the corresponding variant of NCT is only marginal. The results also show that the choice of the bit-plane separation method is important when using MCT, as the best bit rate (4.42 bpp) is significantly better than if neither prediction nor gray coding were used (5.10 bpp). In the case of NCT, the choice of the bit-plane separation is less significant. This is because NTC can use all previous layers as references, and thus it exploits the interlayer dependencies better than MCT, which is limited to only one reference layer.

Table 4 presents results for the palette image test set. The best results (in total) are obtained by NCT without any prediction (BPS) or by using gray-coding (GCS). From this, we make three main observations. First, NCT performs better than MCT and is therefore the recommended variant for palette images. Second, the prediction-based bit-plane separation is extremely inefficient. Third, a noticeable exception is the simplest three-color image (flax), for which the MCT provides significantly better results. In this test set, the larger image files dominate the results. However, if we were to compress a large number of small images with very simple structure, then the preferred variant should be MCT.

4.3 *Comparison with Existing Methods*

The best variant of the proposed method (MCT-GCPES) is compared against existing methods in Table 5. As expected, the proposed algorithm outperforms the standard JBIG applied for separated binary layers. It also gives better results than PWC-P, which is a palette-image-oriented technique, and PNG, which is a dictionary-based method. However, the MCT fails to compete with the best grayscale oriented methods such as CALIC, JPEG-LS, and JPEG2000, as well

Table 4 Compression results (in bytes) for the palette images.

	MCT			NCT		
Image	BPS	GCS	PES	BPS	GCS	PES
Benjerry	4236	4173	6204	**2988**	3135	5071
Books	8749	10,145	14,610	**7948**	8486	15,041
Ccitt01	12,046	11,827	18,312	12,055	**11,990**	27,993
Cmpndd	68,215	62,330	70,645	**57,229**	59,080	70,710
Flax	82	**81**	142	156	146	213
Gate	24,381	22,512	25,144	**18,954**	19,937	25,082
Sea_dusk	**739**	748	1047	992	859	1203
Sunset	83,011	75,673	74,614	75,268	**73,914**	76,233
Total	201,459	187,452	210,718	**175,590**	177,547	221,546

Table 5 Compression results (bits per pixel) for the natural images.

	Proposed	Competitive							
Image	MCT-GCPES	JBIG-BPS	JBIG-GCS	CALIC	JPEG-LS	PWC-G	PWC-P	JPEG2K	PNG
Airplane	4.05	5.23	4.38	3.74	3.81	3.84	4.40	4.01	4.28
Couple	4.54	5.82	4.83	4.25	4.26	4.27	5.02	4.49	4.50
Crowd	4.09	5.35	4.57	3.77	3.91	3.93	4.46	4.19	4.52
Goldhill	4.96	6.17	5.26	4.64	4.71	4.71	5.33	4.81	4.88
Lena	4.46	5.66	4.72	4.11	4.23	4.33	4.96	4.28	4.60
Average	4.42	5.64	4.75	4.11	4.18	4.21	4.84	4.36	4.56

Table 6 Compression results (in bytes) for the palette images.

	Proposed	Competitive							
Image	NCT-BPS	JBIG-BPS	JBIG-GCS	EIDAC	CALIC	JPEG-LS	PWC-G	PWC-P	PNG
Benjerry	2988	7209	7104	**2659**	5939	6707	3960	3120	4846
Books	**7948**	23,277	14,927	8517	22,299	39,859	14,878	8972	15,019
Ccitt01	12,055	103,864	13,549	**5471**	22,547	35,840	20,619	14,056	46,772
Cmpndd	57,229	89,822	67,244	**48,305**	71,917	71,469	66,090	50,026	72,695
flax	156	1208	1143	**71**	760	3411	3485	1380	420
gate	18,954	31,020	26,198	16,662	25,038	27,656	23,127	**16,242**	24,922
Sea_dusk	992	2444	2344	**870**	1219	4061	941	1292	1986
Sunset	75,268	93,069	79,434	58,402	76,577	83,552	65,831	**49,256**	79,085
Total	175,590	351,913	211,943	**140,957**	226,296	272,555	198,931	144,344	245,745

Table 7 Compression results (bits per pixel) for Bridge image. Algorithms where prediction error modeling is used provide the worst results.

	Proposed			Competitive					
Image	MCT-BPS	MCT-GCS	MCT-PES	JBIG-BPS	JBIG-GCS	CALIC	JPEG-LS	PWC-G	PWC-P
Bridge	4.40	4.93	5.80	5.37	5.20	5.37	5.50	**4.09**	4.16

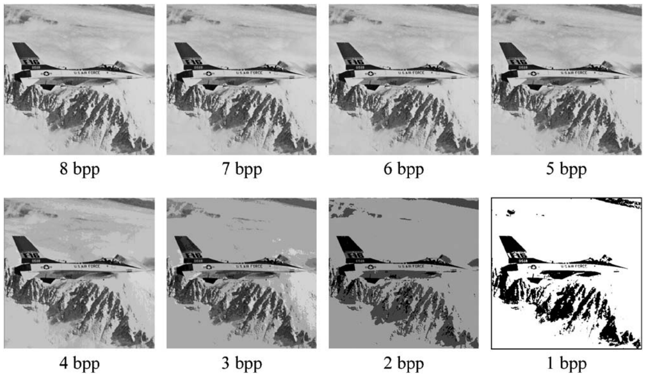

Fig. 10 Image Airplane with sequentially reduced graydepth.

as the PWC-G, which is also optimized for grayscale images. We conclude that the proposed method is most efficient when comparing to binary, dictionary-based, and palette-oriented compression algorithms, but the best grayscale-oriented techniques cannot be outperformed.

Similar comparisons for palette images are shown in Table 6. Again, the best variant of the proposed method (NCT-BPS) outperforms JBIG and all grayscale-oriented methods: CALIC, JPEG-LS, and PWC-G, as well as the dictionary-based PNG. Results for wavelet-based JPEG2000 are not presented, since this algorithm demonstrated extremely weak performance. On the other hand, the best palette-oriented algorithms such as EIDAC and PWC-P are more efficient.

Although error prediction applied to natural images is efficient in general, one can find an image where it fails to improve the compression performance. The Bridge image (see Fig. 1) is an example of such an image, as illustrated in Table 7. Note that MCT-PES failed, presenting the worst bit rate. The same holds for all competitive algorithms in which error prediction is used—CALIC and JPEG-LS. This example shows that even for a continuous-tone image case, there can be found counterexample where prediction error modeling fails to improve the compression.

4.4 *Grayscale Versus Palette Compression*

Competitive algorithms are designed to be applied to particular classes of images, either palette or photographic. These classes can be considered as images with the opposite characteristics: typical photographic images contain a lot of unique colors and have smooth color gradation, while palette images have only few colors and have sharp edges. We next study how the efficiency of the compression algorithms depends on how close the given image is to the class for which the algorithm is tuned to.

We designed an artificial test set to fill the gap between photographic and palette images by sequentially decreasing the gray depth of the original 8-bpp images. For each five images, we produced eight images with sequentially reduced gray depth, giving a set of 40 images in total. The process is illustrated in Fig. 10 for the image Airplane. We then compressed the images with two variants of the proposed algorithm: MCT (GCPES variant, the best variant for natural images) and NCT (BPS variant, the best variant for palette images), and compared them with CALIC, PWC-P, and PNG.

The results presented in Table 8 and illustrated in Fig. 11 show that, as expected, the best results for the 8-bpp images are obtained by CALIC and the worst by palette-optimized PWC-P. For images with 1 and 2 bpp, the situation inverts and the best results are shown by PWC-P and the worst by CALIC. The PNG presented an intermediate performance in both cases. NCT, on the other hand, has a slight edge over the other methods, performing best every-

Table 8 Compression results for gray depth reduction. Results are average bit rate over test set for every color depth separately.

	Proposed		Grayscale optimized	Palette optimized	Universal
Color depth, bpp.	MCT-GCPES	NCT-BPS	CALIC	PWC-P	PNG
1 (binary image)	0.42	0.16	0.23	0.18	0.43
2	0.55	0.18	0.39	0.23	0.55
3	2.28	0.50	1.10	0.53	1.11
4	3.46	1.00	2.13	1.07	1.65
5	4.30	1.72	3.14	1.86	2.66
6	4.50	2.59	3.85	2.89	3.37
7	4.46	3.56	4.08	3.97	4.01
8 (original image)	4.42	4.53	4.11	4.84	4.56

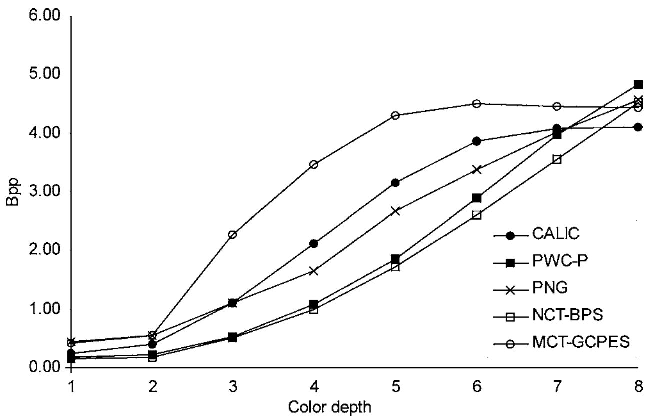

Fig. 11 Compression efficiency (bpp) depending on the color depth.

where else between 2 and 8 bpp. It seems to be the best choice when the images are not clearly of one type: photographic or palette images.

5 Conclusion

We study the efficiency of binary-oriented compression algorithms based on statistical probability estimation and arithmetic coding applied to grayscale and palette images. We consider two modeling schemes. The first scheme (MCT) uses two-layer free-tree modeling with optimized layer ordering. The second scheme (NCT) extended the context modeling to a true multilayer case with fixed ordering. We use four schemes for bit-plane decomposition: bit plane separation (BPS), gray code separation (GCS), prediction error separation (PES), and gray-coded prediction error separation (GCPES).

For prediction-based schemes, we evaluate three predictors: a simple linear scheme, median predictor employed by JPEG-LS, and gradient-adjusted prediction (GAP) used by CALIC. We find that the gray-coded GAP predictor together with MCT (MCT-GCPES) modeling provides the most efficient compression for natural images. The results also show that prediction-based bit-plane separation is inefficient for palette images. For this class of images, NCT with BPS separation (NCT-BPS) is the most efficient, though its advantage over NCT-GCS is minor. We conclude that NCT modeling is less dependent on the chosen bit-plane separation method.

The comparison with the existing compression algorithms on the Natural test set showed that MCT-GCPES outperforms JBIG, which is of similar nature, dictionary-based PNG, and palette-optimized PWC. Its performance is also close to that of lossless JPEG2000. However, other grayscale optimized algorithms—CALIC, JPEG-LS, and grayscale optimized PWC—are not outperformed. For this test set, we conclude that binary-based compression, even if applied with a very high degree of optimization, cannot outperform grayscale-oriented algorithms due to its binary nature.

The same comparison applied to a test set of palette images shows that NCT-BPS outperforms all binary-based techniques (JBIG-BPS and JBIG-GCS) as well as grayscale-optimized algorithms (CALIC, JPEG-LS, PWC-G, and universal PNG). Palette-optimized compressors EIDAC and PWC-P, however, are not outperformed, though the performance of the proposed method is closer to the best algorithm than to the worst.

The results of the palette test set inspired us to perform a detailed investigation of the algorithm's behavior depending on the amount of colors in the image. We designed eight test sets of images where color depth is sequentially decreased from 8 (grayscale case) to 1 bpp (binary case) and examined the performance of best palette (PWC-P and the proposed NCT-BPS), grayscale (CALIC and the proposed MCT-GCPES), and universal PNG. We found out that NCT-BPS performs closely to PWC-P and even outperforms it on some bit depths whereas MCT-GCPES loses to CALIC in all cases. From this observation, we conclude that first, bit-plane separation and binary modeling such as MCT-GCPES cannot be considered to be efficient for natural images, even if strong optimization is involved. Second, context-based compression techniques, such as NCT-BPS, could be considered efficient to be applied for compression of simple (palette) images and the optimization results in compression improvement.

Acknowledgments

The work of the first author was supported by the Finnish Foundation for Technical Development (Tekniikan Edistämissäätiö). The authors also thank Mantao Xu for his advice and support.

References

1. K. Sayood, *Introduction to Data Compression*, 2nd ed., Academic Press, San Diego (2000).
2. ISO/IEC JTC1/SC29/WG1, "JPEG-LS part-2 WD," ISO/IEC JTC1/SC29/WG1 N938, (1998).
3. G. G. Langdon and J. Rissanen, "Compression of black-white images with arithmetic coding," *IEEE Trans. Commun.* **29**(6), 858–867, (June 1981).
4. X. Wu, "Lossless compression of continuous-tone images via context selection, quantization and modeling," *IEEE Trans. Image Process.* **6**, 656–664, (1997).
5. X. Wu and N. Memon, "Context-based, adaptive, lossless image coding," *IEEE Trans. Commun.*, **45**(4), 437–444 (1997).
6. D. S. Taubman and M. W. Marcellin, *JPEG2000: Compression Fundamentals, Standards, and Practice*, Kluwer, Norwell, MA (2001).
7. P. J. Ausbeck, Jr., "The piecewise-constant image model," *Proc. IEEE*, **88**(11), 1779–1789, (Nov. 2000).
8. International Telegraph and Telephone Consulative Committee (CCITT), Progressive Bilevel Compression, Recommendation T.82, (1993); also appears as: International Organization for Standards/ International Electrotechnical Commission (ISO/IEC), Digital Compression and Coding of Continuous-Tone Still Images, International Standard 10918–1:1993.
9. Y. Yoo, Y. G. Kwon, and A. Ortega, "Embedded image-domain compression using context models," *Proc. IEEE Int. Conf. Image Process 1999 (ICIP 99)*, **1**, 477–481, (1999).
10. B. Martins and S. Forchhammer, "Tree coding of bilevel images," *IEEE Trans. Image Process.* **7**(4), 517–528, (Apr. 1998).
11. E. I. Ageenko, P. Kopylov, and P. Fränti, "On the size and shape of multilevel context templates for compression of map images," *IEEE Int. Conf. Image Process.* **3**, 458–461 (2001).
12. P. Kopylov and P. Fränti, "Context tree compression of multicomponent map images," *Proc. IEEE Data Compress. Conf.*, pp. 212–221 (2002).
13. P. Kopylov and P. Fränti "Compression of map images by multilayer context tree modeling," *IEEE Trans. Image Process.* **14**, 1–11, (Jan. 2005).
14. J. J. Rissanen and G. G. Langdon, "Arithmetic coding," *IBM J. Res. Dev.* **23**, 146–162 (1979).
15. W. Pennebaker and J. Mitchell, "Probability estimation for the Q-coder," *IBM J. Res. Dev.* **32**(6), 737–759, (1988).
16. M. J. Weinberger, J. Rissanen, and R. B. Arps, "Applications of universal context modeling to lossless compression of gray-scale images," *IEEE Trans. Image Process.* pp. 575–586, (Apr. 1996).

17. P. Fränti and E. Ageenko, "On the use of context tree for binary image compression," *Proc. IEEE Int. Conf. Image Process. (ICIP'99)* 3, 752–756, (1999).
18. M. Kunt and O. Johnsen, "Block coding of graphics: A tutorial review," *Proc. IEEE* **6**(11), pp 770–786, (1980).
19. M. Abdat and M. G. Bellanger, "Combining gray coding and JBIG for lossless image compression," *Proc. IEEE Int. Conf. Image Process. 1994 (ICIP 94)*, 3, 851–855, (Nov. 1994).
20. K. Denecker, W. Philips, and I. Lemahieu, "Universal binary context modeling of image residue for lossless image compression," *Proc. ProRRISC/IEEE Circ. Syst. Signal Process.* pp. 119–124 (1998).
21. B. J. Falkowski, "Compact representation of logic functions for lossless compression of gray-scale images," *IEE Proc. Comput. Digit. Tech.* **151**(3), 221–230, (May 2004).

Alexey Podlasov received his MSc degree in applied mathematics from Saint-Petersburg State University, Russia, in 2002, and his MSc degree in computer science from the University of Joensuu, Finland, in 2004. Currently, he is a doctoral student in computer science in the University of Joensuu. His research topics include processing and compression of map images.

Pasi Fränti received his MSc and PhD degrees in computer science in 1991 and 1994, respectively, from the University of Turku, Finland. From 1996 to 1999 he was a postdoctoral researcher of the Academy of Finland. Since 2000, he has been a professor in the University of Joensuu, Finland. His primary research interests are in image compression, clustering and speech technology.