Variable Size Block-Based Histogram Packing For Lossless Coding of Color-Quantized Images

Armando J. Pinho
Dept. de Electrónica e Telecomunicações / IEETA
Universidade de Aveiro
3810–193 Aveiro, Portugal
ap@det.ua.pt

António J. R. Neves
Dept. de Electrónica e Telecomunicações / IEETA
Universidade de Aveiro
3810–193 Aveiro, Portugal
an@ieeta.pt

ABSTRACT
Previous work has shown that the lossless compression of color-quantized images, by general purpose continuous-tone encoders, can be improved if histogram packing is performed on a block by block basis, prior to compression. In this paper, we extend this idea, proposing an algorithm that adaptively finds a more appropriate region size for histogram packing. With this new algorithm, a balance between the effectiveness of the histogram packing operation and the overhead spent in representing the mapping tables is automatically sought, providing a further increase in the lossless compression gains.

KEY WORDS
Lossless image compression, histogram packing, JPEG-LS, color-quantized images

1 Introduction

Color-quantized images are usually represented by a matrix of indexes (the index image) and by a color table conveying the color information of the image, such that each image index points to a color entry in the table. A number of specialized coding techniques, tuned for compressing color-quantized images, have been proposed (see, for example, [1, 2, 3, 4]). Nevertheless, it remains an important issue to investigate preprocessing methods capable of preparing this class of images for competitive coding by general purpose continuous-tone encoders and, most specially, by those that are standards.

In fact, recent results have shown that the JPEG-LS [5, 6] / JPEG 2000 [7, 8, 9] lossless compression of the index images can be improved if a certain preprocessing step is performed [10, 11, 12]. This preprocessing consists of an order preserving remapping of the used indexes into a contiguous subset of the integers. With this operation, the gaps in the histogram of the indexes are eliminated and, consequently, the compression improves [13].

As shown in previous work [12], due to the fact that, frequently, color-quantized images exhibit globally dense histograms, the use of global histogram packing is pointless. However, locally, this characteristic generally does not hold, and histogram sparseness can be exploited quite effectively. In the work reported in [11, 12], a fixed-size block-based histogram packing approach was proposed, showing important compression gains. In this paper, we go forward with this idea, proposing an adaptive technique which is capable of automatically adjusting the size of the region being processed. With this new advance, we have been able to increase the lossless compression gains further.

In the remainder of this paper we briefly overview block-based histogram packing. Then, we describe the algorithm for variable size block-based histogram packing that is proposed in this paper. Next, we present experimental results comparing this new approach with the previously proposed fixed-size block-based technique and, finally, we discuss the results obtained and draw some conclusions.

2 Block-based histogram packing

Block-based histogram packing consists on applying off-line histogram packing to each block of an image partition (see Fig. 1). In the work reported in [11, 12], blocks of 32×32 pixels have been used.

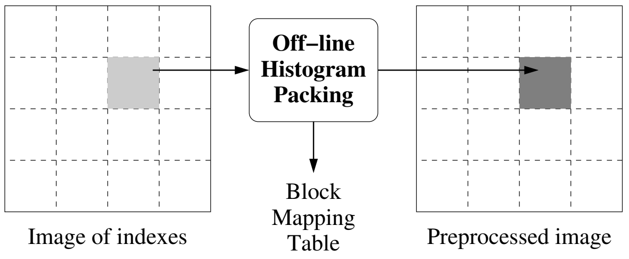


Figure 1. Block-based histogram packing.

Off-line histogram packing is obtained through the construction of an one-to-one order-preserving mapping h, from the block index values $\mathcal{I}$ into a contiguous subset of $\mathbb{N}_0$:

$$h = (I_0 \mapsto 0, I_1 \mapsto 1, \ldots, I_{M-1} \mapsto M-1),$$

where $I_i \in \mathcal{I}$, assuming, without loss of generality, that $I_i < I_j, \forall_{i<j}$, and where M denotes the number of different indexes found in the block.

Therefore, after constructing and applying this mapping to a block $\mathbf{B}$, a new block $\widetilde{\mathbf{B}} = h(\mathbf{B})$ is generated, hopefully more "compression-friendly" than $\mathbf{B}$. Compression gains are obtained if the storage required by the compressed version of $\widetilde{\mathbf{B}}$, together with the overhead required to invert the process (i.e., the list of indexes present in the original block), is smaller than the storage required by the compressed version of $\mathbf{B}$.

3 Adaptive histogram packing

3.1 Motivation

One of the problems overlooked by strategies based on fixed-size block processing is the known fact that characteristics of image data vary across the image. This means, for example, that a given block size may be appropriate for some region of the image, but might be too large or too small for some other region.

A way to overcome this limitation is by allowing the processing of regions of arbitrary shape and size. However, this is generally avoided in order to alleviate the overhead required by the representation of the regions, i.e., the number of bits needed for partition coding. A somewhat intermediate approach relaxes the arbitrary shape and size requirement into a less bit-expensive variable size and possibly variable shape approach. Here, "variable" means a reasonable, usually small, number of possibilities.

In this paper, we propose a greedy variable size region approach, which adaptively seeks an appropriate size for a region in which histogram packing is performed. Therefore, this new approach is an evolution of the method previously addressed in [11, 12], where fixed-size blocks of 32×32 pixels have been used.

3.2 The proposed algorithm

The new method relies on a process that sequentially aggregates elementary blocks, of a given predefined size, in order to construct a larger region where histogram packing is performed. Figure 2 sketches the idea, showing how a new elementary block $\mathbf{B}$ might be aggregated to region $\mathbf{R}_k^n$ (n identifies the region being created, whereas k indicates how many elementary blocks already belong to that region).

The decision of aggregating block $\mathbf{B}$ to the current region depends on the truthfulness of the relation

$$b_{k+1}^n - (b_k^n + b) < 0, \qquad (1)$$

that measures the balance between an estimate of the number of bits required to encode the region if the block is aggregated, (b_{k+1}^n), and the estimate of the number of bits needed if not, ($b_k^n + b$).

These estimates are calculated using the entropy of the residuals of a first order predictor, assuming that the

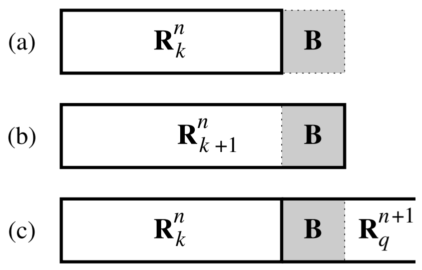


Figure 2. Construction of regions for histogram packing: (a) the elementary block $\mathbf{B}$ is the next candidate to integrate region $\mathbf{R}_k^n$; (b) if (1) is verified, them $\mathbf{B}$ is aggregated to region $\mathbf{R}_k^n$, which is relabeled to $\mathbf{R}_{k+1}^n$; (c) in this case (1) is not verified, triggering the beginning of a new region, $\mathbf{R}_q^{n+1}$ (at this stage, the value of q is unknown).

pixels are raster scanned, and taking into account the overhead required for storing the mapping table (M bits, for M-color images). Therefore, we may rewrite (1) as

$$(k+1)N^2 H_{k+1}^n + o - (kN^2 H_k^n + o + b) < 0$$

or

$$(k+1)N^2 H_{k+1}^n - kN^2 H_k^n - b < 0 \qquad (2)$$

where H_{k+1}^n and H_k^n are, respectively, the entropies (calculated as mentioned above) of the region with $k+1$ and k elementary blocks, and o is the overhead required by the mapping table. We assume, without loss of generality, that the elementary blocks are squares of $N \times N$ pixels.

Term b in (1) and (2) denotes an estimate of the number of bits that block $\mathbf{B}$ would claim if, instead of being aggregated to region $\mathbf{R}_k^n$ (see Fig. 2b), it would be allocated to a new region, $\mathbf{R}_q^{n+1}$ (situation depicted in Fig. 2c). The main problem in obtaining this estimate is that, based only on past data, we are unable to know the size of the next region, i.e., the value of q.

In fact, the size of the next region is needed not only to estimate its entropy, but also to know how the overhead of the mapping table will be distributed among the pixels of the region, i.e., the value of o/q. Therefore, deciding when stop growing a given region $\mathbf{R}^n$ depends on the knowledge of the size of the next region, $\mathbf{R}^{n+1}$, which, by itself, depends on the size of region $\mathbf{R}^{n+2}$, and so on. At each decision step, this recursion goes until the end of the image, generally leading to a computational problem not solvable in practical time.

To alleviate the computational burden involved in the computation of b, we impose two limitations in the algorithm: (1) the maximum size of the region is limited to some value K, typically 16 elementary blocks; (2) the recursion does not proceed until the end of the image but, instead, until a point which is Q elementary blocks ahead of $\mathbf{B}$, typically 8. Notice that limitation (2) has the role of imposing an artificial end-of-image to the recursion process. On the other hand, restriction (1) has the additional goal of limiting the number of bits required to encode the size of the regions, which is $\lceil \log_2 K \rceil$.

4 Experimental results

The experimental results that we present in this paper are based on a set of 23 true color images (768 columns $\times$ 512 rows) that we refer to as the "Kodak" images[1]. Using version 1.2.3 of the "gimp" program,[2] each image was color-quantized based on an image-dependent palette of 128 or 256 colors (generated by "gimp") with and without Floyd-Steinberg color dithering. After color quantization, but before applying histogram packing, the palettes of the color-quantized images have been reordered according to increasing luminance [14]. Finally, compression was obtained using a JPEG-LS encoder with default parameters[3].

We provide compression results concerning: (1) luminance-reordered images with fixed block size histogram packing (method described in [11, 12]); (2) luminance-reordered images with adaptive region size histogram packing. The total overhead generated by method (1) is constant, depending only on the number of image blocks and the size of the mapping table (M bits, for images with M colors). As in [11, 12], we used blocks of 32×32 pixels, which implies 384 blocks per image and, therefore, 98 304 bits of overhead.

The total number of overhead bits for the adaptive method depends on how many regions are generated for a given image. For each region, the overhead is given by the sum of the 128 or 256 bits of the mapping table and the $\lceil \log_2 K \rceil$ bits used for indicating the number of elementary blocks forming that region. In these experiments, we used $K = 16$, $Q = 8$ (depth of recursion) and elementary blocks of 26×26 pixels. The results presented in Table 1 include, besides the size of the encoded index image, the (uncompressed) size of the color table and all required overhead information needed for recovering the original image.

In Fig. 3b we provide an example of how the proposed algorithm transforms the luminance-reordered index image, in this case of the Kodak image "07" (displayed in Fig. 3a), whereas Fig. 3c shows the image partition found by the algorithm.

5 Discussion

Table 1 shows that, for all the images used in the test, the compression performance has always improved. As expected, the gain varies from image to image, showing the ability of the proposed technique for exploiting zones of stationarity, where large blocks can be used, but without compromising zones where small blocks are required.

[1]These images can be obtained from http://www.cipr.rpi.edu/resource/stills/kodak.html.

[2]http://www.gimp.org.

[3]http://spmg.ece.ubc.ca.

From the results, we observe that larger gains are generally attained for images having less colors. This is also an expected behavior, because the gains provided by the proposed method are due to a well-balanced tradeoff between mapping table overheads and effectiveness of the histogram packing operation: on one hand, larger blocks imply less bits of overhead, but may impair the effectiveness of histogram packing; on the other hand, small blocks provide better compression results of the packed image, but imply more bits to represent the larger number of mapping tables that are required. Since the overhead in constant for fixed-size block-based histogram packing, then the impact of better using the overhead bits will be potentially larger for images with less colors.

6 Conclusion

In this paper, we extended previous work, by investigating how the introduction of variable block-size capabilities into the histogram packing algorithm could improve the compression of color-indexed images. From the experimental results, we conclude that this new approach is able to provide further reduction, although moderate, in the bit-rates previously attained. It should be noticed, however, that none of the test images suffered a decrease in compression by this new technique, which suggests its robustness.

References

[1] V. Ratnakar, RAPP: Lossless image compression with runs of adaptive pixel patterns, *Proc. of the 32nd Asilomar Conf. on Signals, Systems, and Computers, 1998, 2*, 1998, 1251–1255.

[2] Y. Yoo, Y. G. Kwon, and A. Ortega, Embedded image-domain compression using context models, *Proc. of the 6th IEEE Int. Conf. on Image Processing, ICIP-99, I*, Kobe, Japan, Oct. 1999, pp. 477–481.

[3] P. J. Ausbeck Jr., The piecewise-constant image model, *Proceedings of the IEEE, 88*(11), Nov. 2000, 1779–1789.

[4] X. Chen, S. Kwong, and J.-F. Feng, A new compression scheme for color-quantized images, *IEEE Trans. on Circuits and Systems for Video Technology, 12*(10), Oct. 2002, 904–908.

[5] *Information technology - Lossless and near-lossless compression of continuous-tone still images*, ISO/IEC 14495–1 and ITU Recommendation T.87, 1999.

[6] M. J. Weinberger, G. Seroussi, and G. Sapiro, The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS, *IEEE Trans. on Image Processing, 9*(8), Aug. 2000, 1309–1324.

Image	128 Colors						256 Colors					
	Without dithering			With dithering			Without dithering			With dithering		
	Fixed packing	Adaptive packing	% gain	Fixed packing	Adaptive packing	% gain	Fixed packing	Adaptive packing	% gain	Fixed packing	Adaptive packing	% gain
01	4.529	4.477	1.2	4.776	4.721	1.2	5.286	5.264	0.4	5.518	5.497	0.4
02	4.581	4.562	0.4	4.873	4.852	0.4	5.383	5.375	0.1	5.684	5.670	0.3
03	1.980	1.853	6.4	2.935	2.849	2.9	2.428	2.333	3.9	3.430	3.348	2.4
04	3.125	3.053	2.3	3.676	3.609	1.8	3.726	3.677	1.3	4.219	4.176	1.0
05	3.924	3.893	0.8	4.479	4.449	0.7	4.612	4.570	0.9	5.101	5.075	0.5
06	3.624	3.545	2.2	3.981	3.906	1.9	4.268	4.208	1.4	4.619	4.567	1.1
07	2.624	2.561	2.4	3.377	3.315	1.8	3.255	3.197	1.8	3.971	3.899	1.8
08	4.151	4.114	0.9	4.381	4.344	0.8	4.869	4.830	0.8	4.993	4.953	0.8
09	2.945	2.847	3.3	3.437	3.365	2.1	3.650	3.578	2.0	4.089	4.021	1.7
10	3.088	3.030	1.9	3.695	3.634	1.6	3.805	3.764	1.1	4.315	4.278	0.9
11	3.500	3.437	1.8	3.771	3.706	1.7	4.119	4.056	1.5	4.484	4.454	0.7
12	2.649	2.545	3.9	3.413	3.353	1.8	3.298	3.207	2.8	3.965	3.910	1.4
13	5.072	5.039	0.6	5.223	5.196	0.5	5.764	5.739	0.4	5.987	5.965	0.4
14	3.428	3.325	3.0	3.865	3.795	1.8	4.219	4.145	1.8	4.586	4.523	1.4
15	2.643	2.576	2.6	3.184	3.130	1.7	3.241	3.194	1.4	3.824	3.773	1.3
16	3.259	3.184	2.3	3.772	3.731	1.1	3.913	3.867	1.2	4.417	4.367	1.1
17	3.253	3.182	2.2	3.627	3.576	1.4	3.951	3.911	1.0	4.360	4.312	1.1
18	4.238	4.188	1.2	4.504	4.455	1.1	4.918	4.871	0.9	5.195	5.148	0.9
19	3.343	3.267	2.3	3.799	3.738	1.6	4.105	4.046	1.4	4.467	4.419	1.1
20	2.599	2.504	3.7	2.903	2.811	3.2	3.195	3.103	2.9	3.456	3.369	2.5
21	3.569	3.475	2.6	3.976	3.906	1.7	4.333	4.270	1.4	4.693	4.627	1.4
22	3.646	3.577	1.9	4.032	3.969	1.6	4.361	4.301	1.4	4.708	4.646	1.3
23	2.059	1.994	3.2	3.191	3.132	1.9	2.624	2.566	2.2	3.634	3.569	1.8

Table 1. Lossless compression results, in bits per pixel, of the color-indexed "Kodak" images. Compression results are provided for: (1) luminance-reordered images with fixed block size histogram packing; (2) luminance-reordered images with variable region size histogram packing. The percentage indicates gains attained by the new method in relation to the fixed-size approach. Color table sizes as well as the overheads generated by the histogram packing procedure are included in the compression results.

[7] *Information technology - JPEG 2000 image coding system*, ISO/IEC International Standard 15444–1, ITU-T Recommendation T.800, 2000.

[8] A. Skodras, C. Christopoulos, and T. Ebrahimi, The JPEG 2000 still image compression standard, *IEEE Signal Processing Magazine, 18*(5), Sept. 2001, 36–58.

[9] D. S. Taubman and M. W. Marcellin, *JPEG 2000: image compression fundamentals, standards and practice* (Kluwer Academic Publishers, 2002).

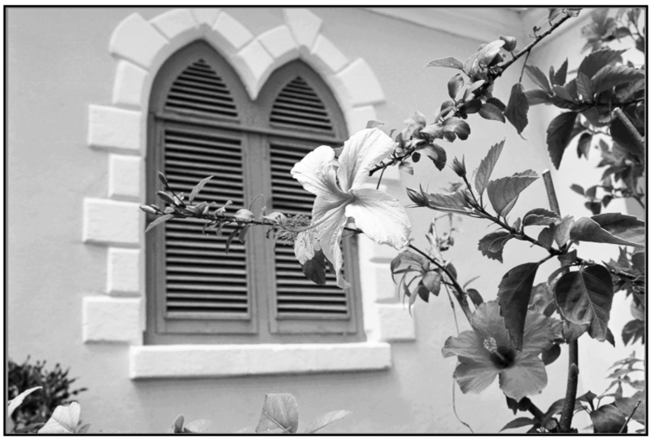
[10] A. J. Pinho, An online preprocessing technique for improving the lossless compression of images with sparse histograms, *IEEE Signal Processing Letters, 9*(1), Jan. 2002, 5–7.

[11] A. J. Pinho and A. J. R. Neves, Block-based histogram packing of color-quantized images, *Proc. of the IEEE Int. Conf. on Multimedia and Expo, ICME-2003, 1*, Baltimore, MD, July 2003, 341–344.

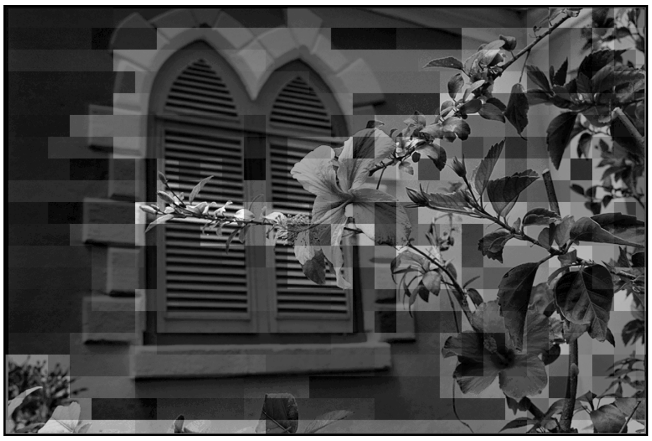
[12] A. J. Pinho and A. J. R. Neves, JPEG 2000 coding of color-quantized images, *Proc. of the IEEE Int. Conf. on Image Processing, ICIP-2003, 2*, Barcelona, Spain, Sept. 2003, 181–184.

[13] P. J. S. G. Ferreira and A. J. Pinho, Why does histogram packing improve lossless compression rates?, *IEEE Signal Processing Letters, 9*(8), Aug. 2002, 259–261.

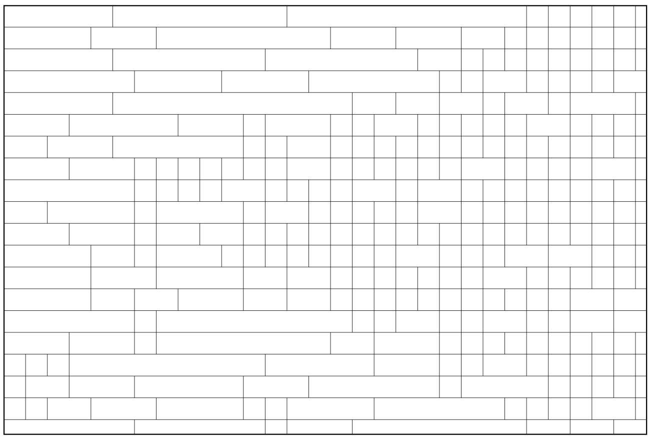
[14] A. Zaccarin and B. Liu, A novel approach for coding color quantized images, *IEEE Trans. on Image Processing, 2*(4), Oct. 1993, 442–453.



(a)



(b)



(c)

Figure 3. Example of the variable size block-based histogram packing approach applied to the Kodak image “07”: (a) Index image after palette reordering based on luminance (b) The luminance-reordered image of indexes after adaptive packing. (c) The regions created by the algorithm.