Image compression using topological maps and MLP

Gilles Burel, Jean-Yves Catros

IEEE Int. Conf. on Neural Networks, Vol. 2, pp. 727-731, March 28th - April 1st, 1993, San Francisco, California, USA

Image Compression using Topological Maps and MLP

Gilles BUREL & Jean-Yves CATROS
Thomson CSF, Laboratoires Electroniques de Rennes
Avenue de Belle Fontaine, 35510 Cesson-Sévigné, France

***Abstract*— Image compression is an essential task for image storage and transmission. We propose a compression technique in which an MLP predictor takes advantage of the topological properties of the Kohonen algorithm. The Kohonen algorithm creates a code-book which is used for Vector Quantization of the source image. Then, an MLP is trained to predict references to code-book, allowing further compression. Even with difficult images, the result is a reduction of 15% to 20% of the bit rate compared with classical Vector Quantization techniques, for the same quality of decoded images.**

I. Introduction

Image compression is an essential task for image storage and transmission. As these domains have become increasingly important, the theory and practice of image compression have received increased attention. Since the bandwidth of many communication systems, or mass memory storage capacities are relatively inextensible, some kind of data compression is required to face the growing amount of information that people want to transmit or store.

Existing compression techniques dedicated to static images can be grouped in 3 great families, on the basis of their specific nature :

1. **Predictive techniques** take profit of image redundancy to predict the luminance of a pixel according to its neighbourhood.
2. **Vector Quantization** takes advantage of image redundancy to define a code-book for image blocks.
3. **Transform Coding** performs compression of image blocks by energy preserving transformations that pack maximum information on a minimum number of samples.

A good review of image compression techniques can be found in [5] [7]. Predictive techniques are the easiest to implement, but they show some lack of robustness and do not achieve high compression rates (generally, they only take profit of redundancies between adjacent pixels). Transform Coding or Vector Quantization techniques are preferently used when higher compression rates are required. Their better performances are mainly due to the fact that they take into account large image blocks, taking profit this way of a larger amount of redundancies. A comparative study is provided in [4]. It shows the interest of Vector Quantization techniques when high compression rates are required (bit rate under 1 bit per pixel). For smaller compression rates (bit rate above 2 bits per pixel), transform coding performs better.

In the neural network field, Cotrell, Munro, and Zipser [3] have proposed a method for image compression that lies in the Transform Coding family. It is based on a 3-layers perceptron, that performs identity mapping of image blocks via a small hidden layer. Its performances are the same as those of the classical Discrete Cosinus Transform technique. Hence, it should be used preferently for small compression rates. For high compression rates, techniques of the VQ family should be prefered.

The method we propose [2] lies in the Vector Quantization family. We use Kohonen's topological maps [8] for block compression of digital images by Vector Quantization, and an MLP trained by backpropagation [10] to perform further compression by prediction of references to code-book. The paper is organized as follows. In section 2, we shortly describe the principle of VQ techniques, and compare the performances of neural and non-neural VQ algorithms. In section 3, we explain the proposed approach, that takes advantage of topological properties of Kohonen algorithm to design an MLP predictor for further compression. We also describe the structures of corresponding coder and decoder, and we discuss experimental results.

II. Principle of Vector Quantization techniques

Vector Quantization is a compression technique that has been widely used ([1], [6] for instance). The method consists in dividing the image in small blocks, and replacing each block by an index. The index is a reference to a code-book of standard blocks: it indicates which block of the code-book is the closest to the image block. The code-book is generally built using the LBG algorithm [9].

Let us note K the number of pixels in a block, and M the size of the code-book. Assuming the availability of the code-book on the decoder side, the compression achieved is $(log_2 M)/K$ bits per pixel. To maintain a small distortion on a lot of images, a huge code-book is required. Furthermore, it is necessary to use large blocks sizes in order to take profit of a large number of redundancies. That's why we prefer an approach in which the code-book is adapted to each image (or to each group of images). Such a point of view has already been suggested in [6]. It allows to use small code-books, and quite small block sizes. For the experiment described below, we use a code-book of 256 blocks of 3x3 pixels.

The counterpart is the need to transmit or store the code-book with each image, or each group of similar images. But we will show below that it isn't very costly. There is also the need to adapt the code-book for each image, or group of images. This may be a drawback for TV applications, due to real time constraints. Hence, our method is more dedicated to image storage, and transmission of satellite images or aerial images (more generally transmission of individual images rather than video sequences). Furthermore, whithin the context of compression of TV images, it is better to use techniques based on movement estimation.

We describe in the next section experiments done with a data-base of 4 difficult images, often used to evaluate compression algorithms: “foot”, “Kiel”, “calendar”, and “boat” (figure 1).

To compare the performances of VQ algorithms, we have built a data-base of 15716 blocks of 3x3 pixels, randomly extracted from images “foot”, “Kiel”, and “calendar”.

Figure 2 shows the mean square error of each algorithm versus the number of iterations. KH stands for the Kohonen algorithm [8]. It is clear that all these algorithms reach approximately the same performances in terms of mean-square error. The exception is the k-means algorithm, which is likely to be trapped in a local minimum. Hence, we want to emphasis the idea that the advantage of Kohonen algorithm over LBG is not its performances as a Vector Quantizer, but its hability to preserve topology. The objective of the next section is to show how an MLP can take profit of these topological properties to increase the compression rate without degradation of the decoded image.

III. Taking profit of topological properties

A. Basic ideas

Figures 3 shows the images of indexes obtained using code-books built by Kohonen algorithm (KH), and LBG. The topological properties of the Kohonen algorithm clearly appear: The image obtained with KH is very coherent, while the image obtained with LBG looks like noise.

To take profit of preservation of topology, we propose to compress the image of indexes itself. The idea is to use a predictor to provide an estimation of an index knowing its causal neighbourhood. An MLP [10] trained by backpropagation seems to be a good candidate to achieve that task. If we obtain a good prediction, it will be possible to compress the image of indexes itself by transmitting only the prediction error. We will perform reversible compression of the image of indexes, because we don't want to degrade it. So, the reconstructed original image will be exactly the same whether we use the predictor or not. This will allow to measure the gain provided by use of prediction simply by comparing the resulting bit rates.

The MLP receives on input the causal neighbourhood $\mathcal{N} = \{C_1, C_2, C_3, C_4\}$ of the index C to predict, and provides on output an estimation $\widehat{C}$ of C.

C_1	C_2	C_3
C_4	$C?$	

We have evaluated the performances of various MLP architectures. 2-layers MLP offer generally quite poor performances. A good compromise between performances and complexity is obtained with a 4-layers MLP (4+12+8+1 neurons). The neuron model is a linear summator (with bias) followed by a non-linear function (hyperbolic tangent), exept for the output neuron which is linear, and the input neurons which do nothing.

The interest of topology preservation can be easily justified on a mathematical point of view. The best possible predictor according to a quadratic criteria is the predictor which minimizes the mean square prediction error $E\{(\widehat{C} - C)^2\}$, where E designs the mathematical expectancy.

$$E\{(\widehat{C}-C)^2\} = E_{\mathcal{N}}E_{C|\mathcal{N}}\{(\widehat{C}-C)^2|\mathcal{N}\}$$

but

$$E_{C|\mathcal{N}}\{(\widehat{C}-C)^2|\mathcal{N}\} = (\widehat{C}-\bar{C})^2 + var_{C|\mathcal{N}}\{C|\mathcal{N}\}$$

Where $\bar{C}$ designs $E_{C|\mathcal{N}}\{C|\mathcal{N}\}$. So the best predictor is:

$$\widehat{C}(\mathcal{N}) = E_{C|\mathcal{N}}\{C|\mathcal{N}\}$$

and its mean square error is:

$$E\{(\widehat{C}-C)^2\} = E_{\mathcal{N}}\left\{var_{C|\mathcal{N}}\{C|\mathcal{N}\}\right\}$$

Hence, it is obvious that no good prediction can be obtained if topology isn't preserved, because $var_{C|\mathcal{N}}\{C|\mathcal{N}\}$ would always be high. And, of course, since we have shown that $E_{\mathcal{N}}\left\{var_{C|\mathcal{N}}\{C|\mathcal{N}\}\right\}$ is the theoretical lower bound of mean square prediction error, neither the MLP nor any other predictor can provide a better prediction than that.

On the contrary, if topology is preserved, $var_{C|\mathcal{N}}\{C|\mathcal{N}\}$ will generally be small, hence there is no theoretical obstacle to good prediction.

B. The structure of the coder and the decoder

Figures 4 and 5 show the structure of the coder and decoder, and the messages format. To compress a source image (SI), the coder computes a topological code-book using Kohonen algorithm (to achieve very fast convergence, the Kohonen network is initialized with a code-book computed on a large set of images). Then, it computes the image of indexes (II). It computes the weights of the MLP predictor (again, to achieve very fast convergence, the MLP is initialized with weights computed on a large set of images of indexes). It computes the prediction errors (PE), then the corresponding optimum Huffman code, and computes the message. The message contains a header: bit P indicates if the predictor is used or not, bits UC, UM, UH indicate respectively if the code-book, the MLP, and/or the Huffman follow (for update). Then, if P=1, the prediction errors follow, else the image of indexes follows.

The decoder reads the header, updates the code-book, the predictor, and/or the Huffman codes if UC, UM, and/or UH is/are set. Then, if P=1, it reconstructs the image of indexes using the predictor and the prediction errors, else it reads directly the image of indexes. The image (RI) is then reconstructed from the image of indexes thanks to the code-book.

C. Experimental results

The table below shows some results obtained on the previously mentioned images. These images, often used to test compression algorithms, are difficult images. So, these results provide a lower bound of expected results on more classical images.

image	foot	Kiel	calendar	boat
header	3071	3063	3053	3080
data	32567	38440	33247	21547
message	35668	41503	36300	24667
source	358425	414720	358425	254016
bpp (P=0)	0.94	0.93	0.94	0.96
bpp(P=1)	0.79	0.80	0.81	0.77
gain	16%	14%	14%	20%

Lines 2 to 4 indicate respectively the size (in bytes) of the header, the data field, and the message (header+data), when the predictor is used. The size of the header correspond to the worst possible case (UC=UM=UH=1, which means that everything is updated), in order to indicate the lower bounds of possible performances. Line 5 reminds the size in bytes of the source image. Lines 6 and 7 indicate respectively the bit rate in bits per pixel when VQ is used alone, and when both VQ and prediction are used. The last line shows the gain in bit rate provided by use of the MLP predictor (we remind that it corresponds to the worst possible case because we have assumed that UC=UM=UH=1).
Photographies of the reconstructed images are not provided here, because differences with source images can be visually detected only on high quality video screens.

IV. Conclusion

We have proposed an approach for image compression that takes profit of both Vector Quantization properties and topological properties of Kohonen algorithm. An MLP predictor is used to compress the image of indexes provided by Vector Quantization, and that is possible only because topology has been preserved. Even in the worst possible configurations (difficult images plus need to update all the internal parameters), the benefit of that approach over classical Vector Quantization is a reduction of 15% to 20% of the bit rate.

Further work could include definition of a variant of the MLP model to achieve better prediction (this model will use non-linear synapses), extensive tests on large bases of images, and sudy of potential for image perception as well as compression/communication.

References

[1] R. ARAVIND, Allen GERSHO, “Image compression based on Vector Quantization with finite memory”, Optical Engineering, July 1987, vol 26, n^o7

[2] Gilles BUREL, “Procédé de compression d’images”, French patent n^o 91-11609, september 20th, 1991

[3] G.W.COTTREL, P.MUNRO, D.ZIPSER, “Image compression by backpropagation”, ICS Report 8702, February 1987, University of California, San Diego

[4] Thomas R. FISCHER, Mary E. BLAIN, “A comparison of Vector Quantization Subband and Tranform Coding of Imagery”, International Symposium on Circuits and Systems, 1989, pp 1520-1523

[5] Robert FORCHHEIMER, Torbjörn KRONANDER, “Image coding - From waveforms to animation”, IEEE trans. on ASSP, vol 37, n^o12, December 1989

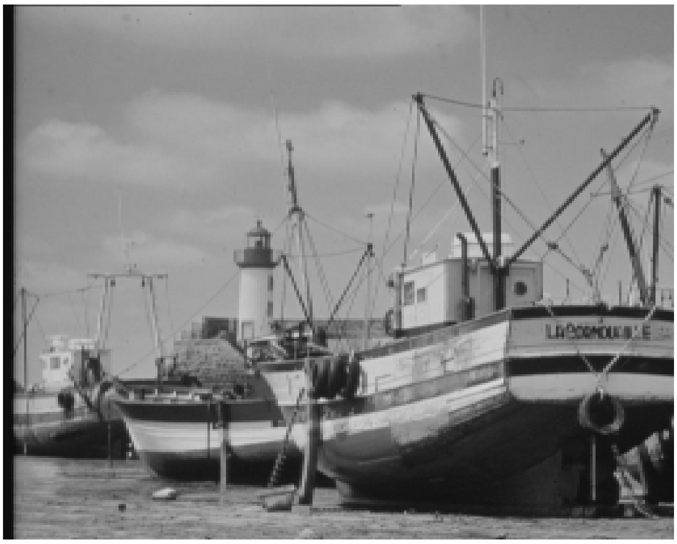
[6] Morris GOLDBERG, Paul R. BOUCHER, “Image compression using adaptative Vector Quantization”, IEEE trans. on Communications, vol COM-34, n^o2, February 1986

[7] A.K. JAIN, “Image Data Compression : A Review”, Proceedings of the IEEE, vol 69, n°3, March 1981

[8] Teuvo KOHONEN, “Self-Organization and Associative Memory”, Springer-Verlag, 1984

[9] Y. LINDE, A. BUZO, R.M. GRAY, “An algorithmm for Vector Quantizer design”, IEEE Trans. on Communications, vol COM-28, n°1, January 1980

[10] D.E. RUMELHART, G.E. HINTON, R.J. WILLIAMS, “Learning internal representations by error backpropagation”, Parallel Distributed Processing, D.E. RUMELHART and J.L. Mc CLELLAND, Chap8, Bradford book - MIT Press - 1986



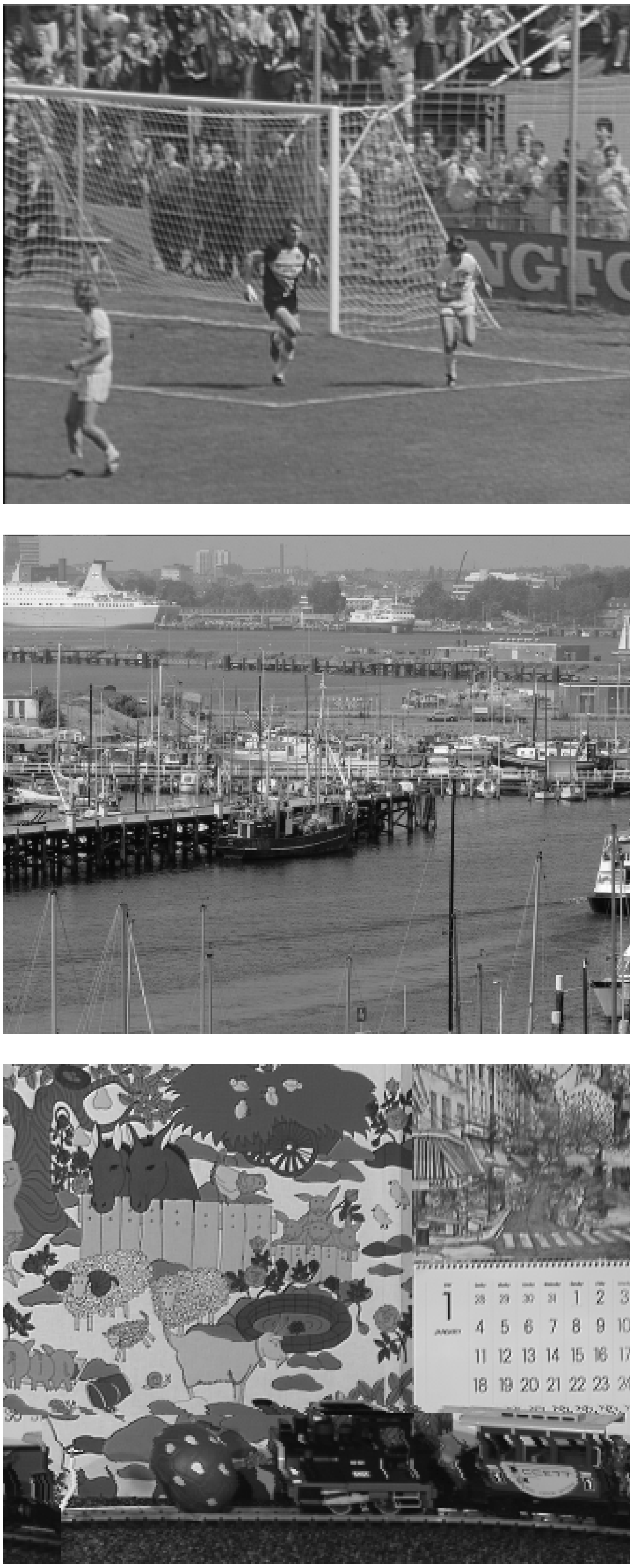


Figure 1: From left to right and top to bottom: images “boat”, “foot”, “Kiel”, and “calendar”

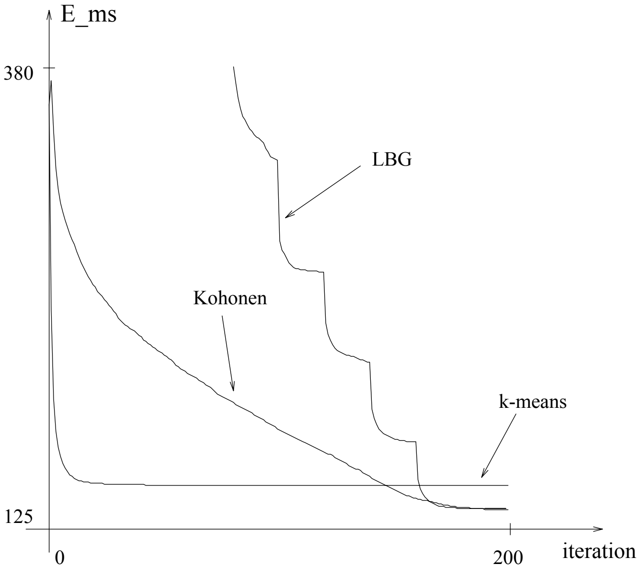


Figure 2: Comparison of algorithms KH, LBG and k-means

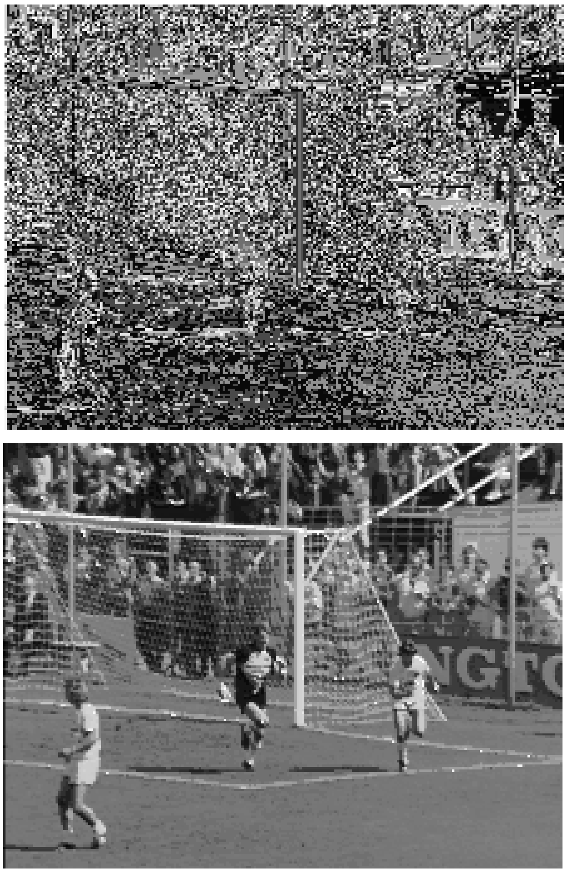


Figure 3: images of indexes of “foot” obtained by LBG (top) and KH (bottom)

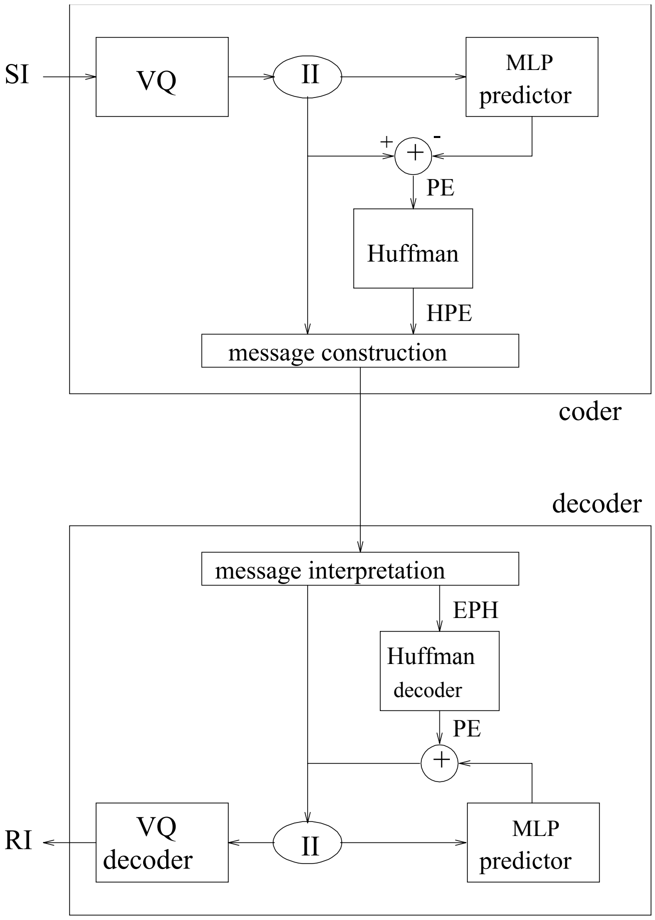
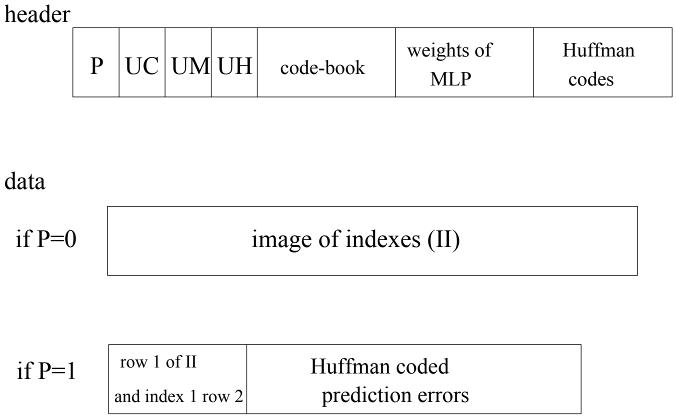


Figure 4: Structure of the coder and the decoder



header

P	UC	UM	UH	code-book	weights of MLP	Huffman codes

data

if P=0 image of indexes (II)

if P=1 row 1 of II and index 1 row 2 Huffman coded prediction errors

Figure 5: Format for compressed image