Near-lossless image compression by relaxation-labelled prediction

Bruno Aiazzi\textsuperscript{a}, Luciano Alparone\textsuperscript{b}, Stefano Baronti\textsuperscript{a}. *

\textsuperscript{a}IFAC-CNR: Institute of Applied Physics “Nello Carrara”, National Research Council, Via Panciatichi, 64, 50127 Florence, Italy
\textsuperscript{b}DET: Department of Electronics and Telecommunications, Via Santa Marta, 3, 50139 Florence, Italy

Received 1 July 2001; accepted 30 January 2002

Abstract

This paper describes a differential pulse code modulation scheme suitable for lossless and near-lossless compression of monochrome still images. The proposed method is based on a classified linear-regression prediction followed by context-based arithmetic coding of the outcome residuals. Images are partitioned into blocks, typically $8 \times 8$, and a minimum mean square error linear predictor is calculated for each block. Given a preset number of classes, a clustering algorithm produces an initial guess of as many predictors to be fed to an iterative labelling procedure that classifies pixel blocks simultaneously refining the associated predictors. The final set of predictors is encoded, together with the labels identifying the class, and hence the predictor, to which each block belongs. A thorough performance comparison, both lossless and near-lossless, with advanced methods from the literature and both current and upcoming standards highlights the advantages of the proposed approach. The method provides impressive performances, especially on medical images. Coding time are affordable thanks to fast convergence of training and easy balance between compression and computation by varying the system’s parameters. Decoding is always real-time thanks to the absence of training.

© 2002 Elsevier Science B.V. All rights reserved.

Keywords: Differential pulse code modulation (DPCM); Data compression; Near-lossless image compression; Relaxation labelling; Statistical context modelling

1. Introduction

Image compression is gaining the attention of an ever-increasing audience because of the huge amount of data to be archived and/or transmitted. Compression methods can be either reversible, i.e. lossless [1,6,8,10,11,13,22,23,40], or irreversible (lossy), depending on whether images are exactly reconstructed after decoding or some distortion is introduced. When lossless coding is mandatory, however, compression ratios greater than two can be hardly obtained, mainly because of the acquisition and digitization noise [4,5,24,32].

Data compression consists of a de-correlation, aimed at generating a memory-less version of the correlated information source, followed by quantisation, which introduces a distortion to allow a reduction in the information rate, and entropy coding. If the de-correlation is achieved by means of an orthonormal transformation, e.g. the discrete cosine transform (DCT), or the discrete wavelet transform (DWT),...
the variance of quantisation errors in the transformed domain is preserved when the data are transformed back to the spatial domain. Thus, the mean square error (MSE) can be easily controlled through the step sizes of quantisers. However, quantisation errors in the transformed domain, which are likely to be uniformly distributed and are upper bounded by half of the step size, are spread by the inverse transformation and yield broad-tailed distributions, whose maximum absolute amplitude cannot be generally known a priori. Therefore, lossy encoders, e.g. that proposed by the Joint Photographic Expert Group (JPEG) [30], are unable to control the distortion but in the MSE sense, which means that relevant image features may be locally distorted or corrupted by an unpredictable and unquantifiable extent.

Noteworthy are lossy methods that allow to a priori settle the maximum reconstruction error, not only globally, but also locally. Out of the objective distortion measurements, the maximum absolute error, or peak error, or $L_\infty$ distance between original and decoded image, is capable to guarantee a quality that is uniform throughout the image. The current definition of near-lossless compression, established in the field of medical imaging [9,14,15], assumes that the $L_\infty$ error is user-defined, as well as small, e.g. $< 1\%$ of the full scale.

Differential pulse code modulation (DPCM) schemes, like JPEG in lossless mode [30], or JPEG-LS [36], an adjusted version of LOCO-I [35], are usually employed for either lossless, or near-lossless, intra-frame image compression. DPCM basically consists of a spatial prediction aimed at de-correlating the data, followed by entropy coding of the outcome prediction errors. Given a causal neighbourhood, i.e. a set of pixels that have previously been scanned, a prediction consists of taking a linear combination, or regression, of the values of such a neighbourhood, with coefficients calculated so as to yield minimum MSE (MMSE) over the whole image [29]. Such a prediction, however, is optimal only for stationary signals. To overcome this drawback, two variations have been proposed: adaptive DPCM (ADPCM), in which the coefficients of predictors are continuously recalculated from the incoming new data [29], and classified DPCM [16–19,25,28], in which a number of statistical classes are preliminarily recognized, an optimized predictor is calculated for each class, and such predictors are enabled, e.g. on a block basis, or blended, in order to attain the best space-varying prediction.

The remainder of this paper is organized as follows. Section 2 presents a causal DPCM procedure based on a classified MMSE prediction achieved through the iterative relaxation-labelling of a number of predictors, and suitable for the lossless/near-lossless compression of monochrome images. Section 3 reports experimental results on a variety of test images with comparisons with the most recent and advanced literature, as well as with current and upcoming standards. Coding performances and computing times are discussed varying with work parameters. Concluding remarks are drawn in Section 4.

2. Coding scheme

The proposed DPCM encoder is based on a classified linear-regression prediction followed by context-based arithmetic coding of the outcome residuals. The image is partitioned into blocks, typically $8 \times 8$, and an MMSE linear predictor is calculated for each block. Given a prefixed number of classes, a clustering algorithm produces an initial guess of as many classified predictors that are fed to an iterative labelling procedure which classifies pixel blocks simultaneously refining the associated predictors. In order to achieve reduction in bit-rate within the constraint of a near-lossless compression [20,21], prediction errors are quantised with odd valued step sizes, $A = 2E + 1$, with $E$ denoting the induced $L_\infty$ error. Quantized prediction errors are then arranged into activity classes based on the spatial context, which are entropy coded by means of arithmetic coding. Fig. 1 shows the flowchart of the encoder. As it appears, the refined predictors are transmitted along with the label of each block and the set of thresholds defining the context classes for entropy coding.

2.1. Basic definitions

Let $\mathcal{F} = \{(i,j) \in \mathbb{Z}^+ \mid 0 \leq i \leq I, \ 0 \leq j < J\}$ denote a Cartesian coordinate set and let $G = \{g(i,j) \mid (i,j) \in \mathcal{F}\} \cup g : \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ and $0 \leq g(\cdot, \cdot) \leq g_{fs}$ a discrete grey-scale image.
2.1.1. 2D causal neighbourhoods

The 2D set \( \mathcal{S} \) may be scanned left to right and top to bottom by successive lines, so as to yield an 1D set of coordinates \( \mathcal{S} = \{ n \in \mathbb{Z}^+ | \forall (i, j) \in \mathcal{S}, n = (J \cdot i + j) \} \) with \( 0 \leq n < N \), and \( N = I \cdot J \). Let us define as causal neighbourhood of the current pixel \( n^* \equiv (i^*, j^*) \) a circle of radius \( W \) in \( L_p \) (\( p \)-norm): \( \mathcal{C}_p^W(n^*) = \{(i,j) \in \mathcal{S} | \| (i,j) - (i^*,j^*) \|_p \leq W, J \cdot i + j = n < n^* = J \cdot i^* + j^* \} \).

\( \forall (i,j) \in \mathcal{C}_p^W(n^*) \), define the Euclidean distance from \( n^* \) as \( \delta = \|(i,j) - (i^*,j^*)\|_2 \). Thus, it is possible to sort \( \mathcal{C}_p^W(n^*) \) by increasing \( \delta \): \( \mathcal{C}_p^W(n^*) = \{(i_k,j_k) \in \mathcal{C}_p^W(n^*) | \| (i_k,j_k) - (i^*,j^*)\|_2 = \delta_k \leq \delta_{k+1} \} \).

Fig. 2 depicts examples of 2D causal neighbourhoods of radius \( W = 3 \) of the current pixel \( n^* \). Pixels are labelled for increasing Euclidean distance from \( n^* \). The causality constraint leads to discarding all pixels that have not been encountered along the raster scan before \( n^* \). The \( L_1 \) metrics yields a diamond-shaped neighbourhood (dark grey only); the \( L_2 \) metrics yields a roughly circular neighbourhood (both dark and light grey); eventually, the \( L_\infty \) metrics yields a square neighbourhood (all pixels highlighted). Trivially, \( \mathcal{C}_p^W \subseteq \mathcal{C}_p^{W+1}, \forall W, \text{ and } p \). Notice that \( \mathcal{C}_p^W \subseteq \mathcal{C}_p^{W} \cdots \subseteq \mathcal{C}_p^\infty \). The size of a causal neighbourhood is its number of elements, or cardinality of the set: \( \text{Card}(\mathcal{C}_p^W) \), with \( W(W+1) = \text{Card}(\mathcal{C}_p^W) \leq \text{Card}(\mathcal{C}_p^{W+1}) = 2W(W+1) \).

2.1.2. Linear-regression prediction

Prediction is based on a linear combination of surrounding pixels lying in \( \mathcal{C}_p^W(n) \). Let us define as prediction support of size \( S \) centred on \( n \), \( \mathcal{P}_S(n) \), a subset of \( \mathcal{C}_p^W(n) \) such that, for some \( D \in \mathbb{R} \), \( \mathcal{P}_S(n) = \{(i_k,j_k) \in \mathcal{C}_p^W(n) | \delta_k \leq D, k = 1, \ldots, S \} \). Let \( \mathbf{\psi}(n) = \{\psi_1(n), \psi_2(n), \ldots, \psi_S(n)\}^T \), with \( \psi_k(n) = g(i_k,j_k) \) where \( (i_k,j_k) \in \mathcal{P}_S(n) \). Thus, \( \mathbf{\psi}(n) \) denotes the vector containing the grey levels of the \( S \) samples lying within \( \mathcal{P}_S(n) \) sorted for increasing Euclidean distance from the pixel \( n \). \( \hat{\mathbf{\psi}}(0) \) is undefined: the first pixel cannot be predicted and must be encoded as it stands (PCM coding). On image edges, \( \hat{\mathbf{\psi}}(n) \) is padded with replicas of the nearest available samples. Let also \( \hat{\phi} = \{\phi_k \in \mathbb{R}, k = 1, \ldots, S\}^T \), with \( \sum_{k=1}^{S} \phi_k = 1 \), denote the vector comprising the \( S \)-coefficients of a linear predictor operating on the support \( \mathcal{P}_S \). Thus, a linear prediction for \( g(n) \) is defined as

\[
\hat{g}(n) = \sum_{k=1}^{S} \phi_k \cdot \psi_k(n) = \langle \hat{\phi}, \mathbf{\psi}(n) \rangle
\]

in which \( \langle \cdot, \cdot \rangle \) indicates scalar (inner) product.
2.2. Initialization

2.2.1. Calculation of block predictors

The determination of the block predictors is the key to the success of the coding process. It starts from observing that patterns of pixel values occurring within \( \mathcal{P}_S(n), n = 1, \ldots, N - 1 (\mathcal{P}_S(0) = \emptyset) \), reflect local spatial features of the image, e.g., edges, textures, and shadings. An efficient prediction should be capable of embodying and reflecting such features as much as possible. After preliminarily partitioning the input image into square blocks, e.g., \( 8 \times 8 \), a causal prediction support of size \( S \) is set, and the \( S \) coefficients of an MMSE linear predictor are calculated for each block by means of a least squares (LS) algorithm. Specifically, if \( B \) denotes a block of the partition, the LS algorithm is fed by the pairs \( \{ (\tilde{g}(n), g(n)) \mid n \in B \} \) to yield the associated predictor \( \tilde{g}_B \).

2.2.2. Clustering of predictors

The above process produces a large number of predictors, each optimized for a single block. The \( S \) coefficients of each predictor are arranged into an \( S \)-dimensional space. More exactly, since the coefficients of any predictor sum to one, all predictors lie on the hyper-plane passing through the versors of the coordinate axes. It can be noticed that statistically similar blocks exhibit similar predictors. Thus, the predictors found previously tend to cluster on the hyperplane, instead of being uniformly spread, as illustrated in Fig. 3.

A user provided number \( M \) of representative predictors is identified by a fuzzy clustering procedure. Such dominant predictors are calculated as centroids of as many clusters in the predictors space, according to a vector Euclidean metrics. Bezdek’s Fuzzy C Means (FCM) algorithm [12] was used. Thus, an \( S \times M \) matrix \( \Phi^{(0)} = \{ \tilde{g}_m^{(0)}, m = 1, \ldots, M \} \) containing the coefficients of the \( M \) predictors is produced. The superscript \( (0) \) highlights that such predictors are start-up values of an iterative refinement procedure.

2.3. Relaxation-labelling and predictors refinement

Once \( M \) predictors have been found out through fuzzy clustering, they are used to initialize an iterative procedure in which image blocks are assigned to \( M \) classes and an optimized predictor is obtained for each class.

**Step 0:** Classify blocks based on their mean square prediction error (MSPE). The label of the predictor minimizing MSPE for a block is assigned to the block itself. This operation has the effect of partitioning the set of blocks into \( M \) classes that are best matched by the predictors previously found out.

**Step 1:** Recalculate each of the \( M \) predictors from the data belonging to the blocks of each class. Then the set of predictors is thus designed so as to minimize MSPE for the current block partition into \( M \) classes.

**Step 2:** Reclassify blocks: the label of the new predictor minimizing MSPE for a block is assigned to the block itself. This operation has the effect of moving some blocks from one class to another, thus repartitioning the set of blocks into \( M \) new classes that are best matched by the current predictors.

**Step 3:** Check convergence; if realized, stop; otherwise, go to Step 1.

Convergence can be checked by thresholding either the percentage of blocks whose class is changed or, better, the decrement in cumulative MSPE associated. Another iteration is executed if either of such amounts, or both of them, exceed preset thresholds. Such an open-loop check is ruled by thresholds that can be calculated once through a closed-loop procedure, in which the coder of Fig. 1 is enabled to produce code bits at every iteration. Thresholds values...
corresponding to negligible further code benefits are found out accordingly.

2.4. Block-wise prediction and quantisation

Once all the blocks have been classified and labelled, together with the attached optimized predictors, the image is raster scanned and the predictors are activated based on the classes of crossed blocks. Thus, each pixel \( g(n) \) belonging to one block of the original partition, say \( B \), with \( B \) belonging to the \( m \)th class, is predicted as

\[
\hat{g}(n) = \langle \tilde{g}_m, \tilde{g}(n) \rangle.
\]

Prediction errors, \( e(n) = g(n) - \hat{g}(n) \), are uniformly quantised with a step size \( \Delta \) as \( e_A(n) = \text{round}[e(n)/\Delta] \) and fed to the context-coding section.

The operation of inverse quantisation \( \tilde{e}(n) = e_A(n) \cdot \Delta \) introduces an error, whose variance is approximately \( \Delta^2/12 \) and whose maximum absolute value is \( L_\infty = |\Delta/2| \). Therefore, since the MSE distortion is a quadratic function of the \( \Delta \), odd-valued step sizes yield lower \( L_\infty/MSE \) ratios than even sizes do (min–max quantiser).

2.5. Context-based arithmetic coding

Prediction errors should be similar to stationary white noise as much as possible. As a matter of fact, they are still spatially correlated to a certain extent and especially are non-stationary, i.e. they exhibit space-varying statistics. The better the prediction, however, the more noise-like prediction errors will be. Fig. 4 portrays prediction errors of the standard 512 \( \times \) 512 test image Lennagrey produced by the 7th predictor of lossless JPEG [30] and by RLPE, respectively. The former employs a 3-pel prediction with fixed coefficients. As it appears, the latter matches more closely samples of random noise: the spatial correlation is strongly reduced, but an inherent heterogeneity still survives, especially on the textures of the feather hat.

Following a trend established in the literature [31,33,36,38–40], prediction errors are entropy coded by means of a classified implementation of arithmetic coding [37]. For this purpose, they are classified into a predefined number of statistically homogeneous classes based on the spatial context. If such classes are statistically discriminated, then the entropy of a context-conditioned model of prediction errors will be lower than that derived from a stationary memory-less model of the de-correlated source [34].

In the present work, a context function of \( e(n) \) was defined and measured on prediction errors lying within the 2D causal neighbourhood \( \mathcal{N}_\infty^W(n) \supseteq \mathcal{P}_3(n) \), as the root mean square (RMS) of quantised prediction errors weighted by the reciprocal of their Euclidean distances from \( n \):

\[
c(n) \triangleq \sqrt{\frac{\sum_{k \in \mathcal{N}_\infty^W(n)} \delta_k^{-1} e_A^2(k) \sum_{k \in \mathcal{N}_\infty^W} \delta_k^{-1}}{\sum_{k \in \mathcal{N}_\infty^W} \delta_k^{-1}}}.
\]
The context function (3) captures the nonstationarity of prediction errors, regardless of their spatial correlation. Its effectiveness for entropy coding will be demonstrated in Section 3.

Again, causality of neighborhood is necessary in order to make the same information available both at the encoder and at the decoder. At the former, the probability density function (PDF) of \( c(n) \) is measured and partitioned into a number \( L \) of intervals chosen so as to be equally populated; thus, contexts are equally probable as well. This choice is motivated by the use of adaptive arithmetic coding [37] for encoding the errors belonging to each class. Adaptive entropy coding, in general, does not require previous knowledge of the statistics of the source, but benefits from a number of data large enough for training, which happens simultaneously with coding. From the PDF of context \( L-1 \) thresholds are calculated that define the decision intervals of each class. \( \Theta \), as well as \( \Phi \), is stored in the file header as side, or overhead, information. The source given by each class is further split into sign bit and magnitude [33]. The former is strictly random and is coded as it stands, the latter exhibits a reduced variance in each class and, thus, may be coded more efficiently.

It is noteworthy that the context-coding procedure described in this paper is independent of the particular method used to de-correlate the data. Unlike other schemes, e.g. CALIC [40], in which context-coding is embedded in the de-correlation procedure, the proposed method can be applied to any DPCM scheme, either lossless, or lossy.

2.6. Decoder

The decoder, shown in Fig. 5, can be summarized by the following steps:

- Retrieve predictors \( \Phi \), block labels \( A \), and context thresholds \( \Theta \) from the file header, as well as the first image sample \( g(0) \), which is PCM coded.
- Calculate context \( c(n) \) by using (3) from the previously decoded pixel values lying within \( \mathcal{N}_c \).
- Decode the encoded quantised prediction error \( e(n) \), after labelling its context class by thresholding \( c(n) \) through \( \{ \theta_l, l = 1, \ldots, L-1 \} \); multiply \( e_\Lambda(n) \) by \( \Lambda \) to yield \( \hat{e}(n) \).
- Calculate the output of the \( m \)th predictor, with \( m \) the prediction class of the current pixel given by its block label, from decoded pixels lying on \( \mathcal{P}_m \).
- Add \( \hat{g}(n) \) to the previously decoded \( \hat{e}(n) \) to yield \( \hat{g}(n) \).

Overhead information needed is: \( (S-1) \times M \) coefficients of predictors, \( L-1 \) thresholds for context decoding, and a label identifying one out of the \( M \) prediction classes for each block (log2 \( M \) bits/label, on an average, without further compression).

3. Experimental results and comparisons

3.1. Coder assessments

3.1.1. Lossless compression performance comparison

Several methods have been compared with the proposed relaxation-labelled prediction encoder (RLPE) working in lossless mode, i.e. with a step size \( \Lambda = 1 \). One is a 2D version [2] of the fuzzy DPCM with context (FDC) [3]. Three are advanced algorithms established in the literature: TMW [26], Said and Pearlman’s multiresolution encoder (S&P) [33], and CALIC [40]. The last entries are lossless JPEG (L-JPEG), i.e. predictive [30], in an implementation employing arithmetic coding, the new standard for lossless/near-lossless compression JPEG-LS [36], and the upcoming standard JPEG 2000 [41] in lossless mode (L-J2K). A test set comprising a number of widely known grey-scale images, plus two remote-sensing and two medical images, is shown in Fig. 6.
Table 1 reports the bit-rate needed for lossless encoding of the test images. Bit rates on disk, including overhead information, are reported in bits per pixel. RLPE largely outperforms all the other methods except TMW, which is superior by 1% on an average. The computational cost of the latter, however, is classified (presumably days of CPU time [26]) due to extremely massive training, aimed at constructing a model of the image. This explains the surprising result of 0.76 bit/pel on the noise-free synthetic test image Shapes. Due to the absence of noise, the model is extremely fitting. The complementary case is given by Noisesquare on which the performance of TMW is average: the simple image model is poorly learned because of the large noise superimposed. Surprisingly, in near-lossless experiments [27] the bit-rate produced by TMW on Shapes attains values that are even larger than that of the lossless case. The only possible explanation is that the reduction in bit-rate due to quantization is lower than the performance penalty occurring when the model, calculated from noise-free data, is forced to make a prediction starting from samples affected by quantization noise. The performance of RLPE is particularly impressive on the test X-ray image: a 14% gain over CALIC, and 23% over JPEG-LS.

3.1.2. Near-lossless performance comparison

A lossy performance comparison was carried out on Lenna (see Fig. 6) chosen as the most representative among the test images of Fig. 6, since its encoding rates are surprisingly similar to the average rates reported in the last entry of Table 1.

Rate distortion (RD) plots are shown in Fig. 7(a) for RLPE, for a non-causal, i.e. interpolation-based, DPCM known as enhanced Laplacian pyramid (ELP) [9,10] and for JPEG-LS [36], all working in near-lossless mode, i.e. $L_{\infty}$ error-constrained, as well as for the lossy wavelet-based JPEG 2000 [41].

RLPE is always superior to JPEG-LS, especially for low rates, and gains over JPEG 2000 for rates larger than 1.25 bit/pel. The rapidly decaying curve for decreasing rates shown by RLPE and by JPEG-LS is typical of all the causal DPCM schemes and is an effect of quantisation noise feedback in the prediction loop. The “noisy” data that are reconstructed at the decoder are utilized for prediction, thus...
Table 1
Bit rates (in bit/pixel) needed to encode the test images by several reversible compression methods: L-JPEG denotes best predictor; JPEG-LS utilizes Golomb–Rice entropy coding; all the other methods arithmetic coding. Best result for each image emphasized in boldface. TMW bit-rates, where unavailable, have been replaced with the best available ones (i.e. those of RLPE) to compute the average on the last row

<table>
<thead>
<tr>
<th>Image</th>
<th>RLPE</th>
<th>TMW</th>
<th>FDC</th>
<th>CALIC</th>
<th>S&amp; P</th>
<th>JPEG-LS</th>
<th>L-JPEG</th>
<th>L-J2K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airplane</td>
<td>3.71</td>
<td>n.a.</td>
<td>3.71</td>
<td>3.74</td>
<td>3.90</td>
<td>3.82</td>
<td>4.30</td>
<td>4.02</td>
</tr>
<tr>
<td>Baboon</td>
<td>5.74</td>
<td>5.73</td>
<td>5.87</td>
<td>5.88</td>
<td>5.92</td>
<td>6.04</td>
<td>6.26</td>
<td>6.11</td>
</tr>
<tr>
<td>Balloon</td>
<td>2.73</td>
<td>2.65</td>
<td>2.80</td>
<td>2.83</td>
<td>2.96</td>
<td>2.90</td>
<td>3.28</td>
<td>3.04</td>
</tr>
<tr>
<td>Barb</td>
<td>4.03</td>
<td>4.08</td>
<td>4.39</td>
<td>4.41</td>
<td>4.51</td>
<td>4.69</td>
<td>5.30</td>
<td>4.61</td>
</tr>
<tr>
<td>Barb2</td>
<td>4.39</td>
<td>4.38</td>
<td>4.55</td>
<td>4.53</td>
<td>4.70</td>
<td>4.69</td>
<td>5.24</td>
<td>4.80</td>
</tr>
<tr>
<td>Board</td>
<td>3.41</td>
<td>n.a.</td>
<td>3.53</td>
<td>3.56</td>
<td>3.81</td>
<td>3.68</td>
<td>4.23</td>
<td>3.78</td>
</tr>
<tr>
<td>Boats</td>
<td>3.68</td>
<td>3.61</td>
<td>3.84</td>
<td>3.83</td>
<td>4.01</td>
<td>3.93</td>
<td>4.47</td>
<td>4.07</td>
</tr>
<tr>
<td>Bridge</td>
<td>5.66</td>
<td>5.59</td>
<td>5.73</td>
<td>5.68</td>
<td>5.80</td>
<td>5.79</td>
<td>5.99</td>
<td>6.00</td>
</tr>
<tr>
<td>Cameraman</td>
<td>4.18</td>
<td>4.10</td>
<td>4.25</td>
<td>4.19</td>
<td>4.46</td>
<td>4.31</td>
<td>4.92</td>
<td>4.56</td>
</tr>
<tr>
<td>Couple</td>
<td>3.56</td>
<td>3.45</td>
<td>3.67</td>
<td>3.61</td>
<td>3.87</td>
<td>3.70</td>
<td>4.17</td>
<td>3.94</td>
</tr>
<tr>
<td>Elaine</td>
<td>4.44</td>
<td>n.a.</td>
<td>4.67</td>
<td>4.80</td>
<td>4.87</td>
<td>4.90</td>
<td>5.13</td>
<td>4.95</td>
</tr>
<tr>
<td>Girl</td>
<td>3.61</td>
<td>3.67</td>
<td>3.77</td>
<td>3.77</td>
<td>3.95</td>
<td>3.93</td>
<td>4.26</td>
<td>4.07</td>
</tr>
<tr>
<td>Goldhill</td>
<td>4.31</td>
<td>4.27</td>
<td>4.41</td>
<td>4.39</td>
<td>4.54</td>
<td>4.48</td>
<td>4.85</td>
<td>4.61</td>
</tr>
<tr>
<td>Harbor</td>
<td>4.44</td>
<td>n.a.</td>
<td>4.53</td>
<td>4.44</td>
<td>4.72</td>
<td>4.49</td>
<td>5.11</td>
<td>4.78</td>
</tr>
<tr>
<td>Hotel</td>
<td>4.24</td>
<td>4.17</td>
<td>4.30</td>
<td>4.25</td>
<td>4.52</td>
<td>4.39</td>
<td>4.94</td>
<td>4.59</td>
</tr>
<tr>
<td>Lennagrey</td>
<td>3.96</td>
<td>3.91</td>
<td>4.06</td>
<td>4.11</td>
<td>4.16</td>
<td>4.24</td>
<td>4.61</td>
<td>4.31</td>
</tr>
<tr>
<td>Noisesquare</td>
<td>5.26</td>
<td>5.54</td>
<td>5.33</td>
<td>5.44</td>
<td>5.55</td>
<td>5.68</td>
<td>5.59</td>
<td>5.65</td>
</tr>
<tr>
<td>Peppers</td>
<td>4.08</td>
<td>4.03</td>
<td>4.13</td>
<td>4.20</td>
<td>4.35</td>
<td>4.29</td>
<td>4.63</td>
<td>4.42</td>
</tr>
<tr>
<td>Shapes</td>
<td>1.30</td>
<td>0.76</td>
<td>1.45</td>
<td>1.14</td>
<td>2.13</td>
<td>1.21</td>
<td>1.35</td>
<td>1.93</td>
</tr>
<tr>
<td>Zelda</td>
<td>3.66</td>
<td>n.a.</td>
<td>3.78</td>
<td>3.86</td>
<td>3.89</td>
<td>4.01</td>
<td>4.24</td>
<td>4.00</td>
</tr>
<tr>
<td>AVIRIS (12b)</td>
<td>6.18</td>
<td>n.a.</td>
<td>6.21</td>
<td>6.32</td>
<td>6.38</td>
<td>6.37</td>
<td>6.43</td>
<td>6.41</td>
</tr>
<tr>
<td>Landsat TM</td>
<td>4.50</td>
<td>n.a.</td>
<td>4.55</td>
<td>4.54</td>
<td>4.63</td>
<td>4.65</td>
<td>4.97</td>
<td>4.80</td>
</tr>
<tr>
<td>X-Ray</td>
<td>1.27</td>
<td>1.33</td>
<td>1.40</td>
<td>1.45</td>
<td>1.71</td>
<td>1.56</td>
<td>1.91</td>
<td>1.64</td>
</tr>
<tr>
<td>CAT (10b)</td>
<td>2.60</td>
<td>2.54</td>
<td>2.65</td>
<td>2.85</td>
<td>2.95</td>
<td>2.87</td>
<td>3.11</td>
<td>2.99</td>
</tr>
<tr>
<td>Average</td>
<td>3.96</td>
<td>3.92</td>
<td>4.07</td>
<td>4.08</td>
<td>4.26</td>
<td>4.19</td>
<td>4.55</td>
<td>4.34</td>
</tr>
</tbody>
</table>

Fig. 7. Near-lossless performance comparison on test image Lennagrey. RLPE, enhanced Laplacian pyramid (ELP) working in near-lossless mode, JPEG-LS (near-lossless), and JPEG 2000 (lossy): (a) PSNR vs. bit-rate; (b) peak error vs. bit-rate.

making it poorer and poorer as the quality, and hence the bit-rate, decreases. This effect is also exhibited, though at rates even lower, by the DCT-based JPEG, and is due to DPCM coding of the dc component [29,30]. The non-causal prediction of ELP performs better for low rates, with an RD curve similar to that of JPEG 2000 (less than one dB lower), which, however, is not $L_{\infty}$-constrained.
The near-lossless plots, i.e. $L_\infty$ vs. bit-rate, shown in Fig. 7(b), demonstrate that the error-bounded encoders are far superior to JPEG 2000. RLPE and ELP share the best results, the former for rates higher than 0.75 bit/pel, the latter otherwise.

3.2. Coding performances varying with work parameters

More specific experiments are reported in Figs. 8(a)–(d). Fig. 8(a) shows the trend of bit-rate to number of predictors for different sizes of prediction support, relatively to the test image Lennagrey. The number of predictors should match the size of their support, to avoid performance saturation, i.e. flat trend. Incidentally, a unique MMSE 4-pel predictor yields a code rate of 4.19 bit/pel, which can be hardly improved by extending the size of the prediction support. Thanks to the moderate coding overhead, large neighbourhoods and/or number of predictors lead to a further reduction in code rate.

Fig. 8(b) shows the code rate as a function of the side of the square blocks of the image partition. The block sizes are powers of two, whose exponent is indicated on abscissa. As it appears, due to the trade-off between spatial adaptivity and coding overhead, the block size is somewhat critical. The optimum exponent is three, i.e. $8 \times 8$ blocks for most of pictures; four for medical images. As expected, the minimum in the bit-rate plot is more pronounced as the number of predictors is increased. A unique predictor is obviously unaffected by the block size.

The number of iterations of the refinement of predictors is non-crucial as well (see Fig. 8(c)). A single iteration guarantees a performance close to the best attainable one (more than five iterations) by <1%. Such a fast convergence is little sensitive to the number of iterations of the FCM algorithm used for clustering the predictors (10 to 20 is reasonable). The case of two predictors yielding rates slowly increasing with the number of iterations is somewhat puzzling. It can be
explained as a resonance effect in which an increasing number of blocks is moved from one class to another. The refinement step is more useful when the number of predictors is larger; thus, predictors can be better specialized to fit the features of the data.

Fig. 8(d) shows that the number of context classes is non-crucial. Practically, 8–32 classes are adequate for all the test images. Also, the optimal number of classes is independent of the number of predictors. Instead, the benefits of context-coding, represented by the difference in rate between zero exponent, i.e. a unique class, and, say, 16 classes, are more significant if the prediction is poorer, e.g. few predictors and/or small support.

3.3. Analysis of computational complexity

The total time spent per pixel, either at the encoder, or at the decoder, is given by the sum of contributions relative to the blocks of the diagrams either of Fig. 1, or of Fig. 5

\[ t_{ENC} = t_1 + (K + 1)t_C + Kt_R + t_p + t_E, \]

\[ t_{DEC} = t_E + t_p \]

in which \( K \) is the number of training iterations and

\[ t_1 = \alpha_1S, \]

\[ t_C = \alpha_C MS, \]

\[ t_R = \alpha_R S, \]

\[ t_p = \alpha_p S, \]

\[ t_E = t_{E0} + \alpha_E 2W(W + 1), \]

where \( \alpha_1, \alpha_C, \alpha_R, \alpha_p, \) and \( \alpha_E \) are proportionality constants.

Time for initialization, \( t_1 \), is roughly proportional to the number of coefficients of predictors, \( S \), since the clustering time, which is proportional to \( M \)—the number of clusters that are searched for—is small with respect to the time for calculation of block predictors. The most onerous step is calculation of average MSPE for each block. This operation is executed \( K + 1 \) times at the encoder only. Such a classification time, \( t_C \), is proportional to the overall number of coefficients tried, i.e. to the number of predictors \( M \) and to the size of predictors \( S \). The time cost for re-calculating predictors, \( t_R \), is roughly proportional to the length of predictor, \( S \), and not to the number of predictors (i.e. of classes), \( M \). In fact, as \( M \) increases, the overall number of pixels used for predictors calculation is unchanged. The pixel prediction at the decoder is identical to that at the encoder. Its cost \( t_p \) is proportional to the number of coefficients, \( S \). Time for context-coding, \( t_E \), consists of a constant amount \( t_{E0} \) for arithmetic coding, plus a term for context calculation, which is proportional to the size of the context support \( 2W(W + 1) \). Time for context-decoding is assumed to be the same as for context-coding. Actually, the latter requires a moderate extra effort for thresholds set-up.

Time values reported in Table 2, are congruent with the models (4) and (5). The benefits of context-coding are highlighted as well. Notice that an encoding time thirty times lower guarantees a code rate that is < 3% larger than that reported in the first entry of Table 2. It is noteworthy that decoding is performed in real time (mostly \(< 2 \) s), as no training is needed. The code was written in C language and slightly optimized.

Eventually, Fig. 9 shows the minimum attainable bit rates as a function of encoding times. Such a plot was obtained as the convex hull of a number of experiments, like those described in Figs. 8(a) and (c) and in Table 2. For achieving lower times, with the penalty of higher rates, the features of RLPE, with the exception of context-coding, have been progressively disabled. In the leftmost part of the plot, RLPE reduces first to a 4-pel AR DPCM, then to a lossless JPEG predictor, both with context coding. As it appears, the bit-rate saving is achieved at a roughly exponentially increasing computational cost. For encoding times tending to infinity, the bit-rate should asymptotically attain the entropy rate of the source. Unfortunately, the net reduction in bit-rate achieved vanishes because of the increased overhead of predictors and especially of labels. CALIC, JPEG-LS and JPEG 2000 do not have parameters to be adjusted for lossless compression. Thus, their time-rate plots reduce to a single point. It is noteworthy that the performances of both CALIC and JPEG-LS are achieved by RLPE in a slightly lower time, whereas the balance between performance and
Table 2
Tests for $512 \times 512$ Lennagrey of RLPE running on a Pentium III-650 MHz PC under Linux OS. Bit rates in bit/pixel and computing times reported for different work parameters. $8 \times 8$ blocks, 16 iterations of FCM for initialization, and 16 context classes used throughout. Bit-rate and time values in parentheses are obtained when context is disabled.

<table>
<thead>
<tr>
<th>Bit rate</th>
<th>$S$</th>
<th>$M$</th>
<th>$W$</th>
<th>$K$</th>
<th>$t_{ENC}$</th>
<th>$t_{DEC}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.96 (4.15)</td>
<td>16</td>
<td>16</td>
<td>4</td>
<td>5</td>
<td>106 s (105 s)</td>
<td>1.5 s (0.7 s)</td>
</tr>
<tr>
<td>3.99 (4.19)</td>
<td>16</td>
<td>16</td>
<td>4</td>
<td>1</td>
<td>27.8 s (26.6 s)</td>
<td>1.5 s (0.7 s)</td>
</tr>
<tr>
<td>4.03 (4.24)</td>
<td>16</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>15.4 s (14.4 s)</td>
<td>1.2 s (0.5 s)</td>
</tr>
<tr>
<td>4.04 (4.25)</td>
<td>4</td>
<td>16</td>
<td>3</td>
<td>1</td>
<td>5.8 s (4.8 s)</td>
<td>0.9 s (0.3 s)</td>
</tr>
<tr>
<td>4.08 (4.30)</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3.1 s (2.1 s)</td>
<td>0.9 s (0.3 s)</td>
</tr>
<tr>
<td>4.19 (4.43)</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>—</td>
<td>0.9 s (0.4 s)</td>
<td>0.7 s (0.3 s)</td>
</tr>
</tbody>
</table>

Fig. 9. Encoding time (log-scale) of RLPE, as convex hull of experiments in Figs. 8(a)–(d), compared with JPEG-LS, CALIC and lossless JPEG 2000 on the test image Lennagrey.

computational cost is much less favourable for JPEG 2000, which, however, has the advantage of a scalable decoding.

4. Concluding remarks

This paper introduced an original approach to error-bounded image compression based on a relaxation-labelled linear prediction aimed at de-correlating nonstationary 2D data. The superior performance of RLPE depends on its capability to capture the most relevant features of the data, that are exploited to make prediction be locally adaptive through the switching of a number of fitting prototype predictors. Prediction errors are partitioned into classes based on their spatial context, and each class is entropy coded through an adaptive arithmetic encoder. This strategy copes with imperfections in prediction, since it takes more and more advantage as prediction is poorer. Thus, its introduction actually damps differences in bit-rates which would be originated from deficiencies of prediction. The scheme exhibits favourable coding performances both lossless (rates 3% lower than those of CALIC, and 6% lower than JPEG-LS, averaged on 24 test images) and especially near-lossless (10 dB gain over JPEG-LS at 0.5 bit/pel on Lennagrey). The latter result is explained by considering that, unlike JPEG-LS and CALIC, the context model of RLPE is not based on spatial gradients, which are largely corrupted by quantization errors when a lossy coding is concerned. Computing times are affordable, thanks to fast convergence of training; decoding is strictly real-time. Its flexibility is guaranteed by varying size of prediction support and number of prototype predictors, thus allowing to achieve better and better results at the expense of computational cost. An extension of RLPE to 3D data is straightforward [7] by simply taking 3D prediction neighbourhoods.

Acknowledgements

The authors wish to warmly thank their former co-author P. Alba for his foresight and constructive efforts to apply fuzzy-logic techniques to data compression.

References


