Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Signal Processing: *Image Communication* 18 (2003) 443–454

SIGNAL PROCESSING: IMAGE COMMUNICATION

www.elsevier.com/locate/image

Gray-level-embedded lossless image compression

Mehmet Utku Celik[a], Gaurav Sharma[b,*], A. Murat Tekalp[a,c]

[a] *Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627-0126, USA*
[b] *Xerox Corporation, MS0128-27E, 800 Phillips Road, Webster, NY 14580, USA*
[c] *College of Engineering, Koc University, Istanbul, Turkey*

Received 25 June 2002; received in revised form 12 December 2002; accepted 28 January 2003

Abstract

A level-embedded lossless compression method for continuous-tone still images is presented. Level (bit-plane) scalability is achieved by separating the image into two layers before compression and excellent compression performance is obtained by exploiting both spatial and inter-level correlations. A comparison of the proposed scheme with a number of scalable and non-scalable lossless image compression algorithms is performed to benchmark its performance. The results indicate that the level-embedded compression incurs only a small penalty in compression efficiency over non-scalable lossless compression, while offering the significant benefit of level-scalability.
© 2003 Elsevier Science B.V. All rights reserved.

PACS: 07.05.Pj; 42.68.Sq; 42.79.Ls

Keywords: Level-scalability; CALIC; Lossless compression; Near-lossless compression

1. Introduction

Image compression is an important tool for reducing the bandwidth and storage requirements for practical imaging systems. Although most image processing applications can tolerate some information loss, in several areas—such as medical, satellite, and legal imaging—lossless compression algorithms are preferred. CALIC [10,11], JPEG-LS [4], and JPEG2000 [5] are among well-known lossless image compression algorithms. Among these CALIC provides best compression ratios over typical images, whereas, JPEG-LS is a low complexity alternative with competitive efficiency. The JPEG2000 standard, on the other hand, is a wavelet-based technique, which provides a unified approach for lossy-to-lossless compression.

In several applications, it is advantageous to have *scalable compression*, where a desired level of compression may be determined after the source has been compressed. This allows flexibility in determining the data rate to meet the bandwidth, memory and processing power constraints imposed by the operating environment—with a corresponding trade-off in image quality. Scalable compression is typically achieved by generating an *embedded bit-stream* which has the property that

*Corresponding author.
E-mail addresses: celik@ece.rochester.edu (M.U. Celik), g.sharma@ieee.org (G. Sharma), tekalp@ece.rochester.edu (A.M. Tekalp).
URLs: http://www.ece.rochester.edu/~celik, http://chester.xerox.com/~sharma, http://www.ece.rochester.edu/~tekalp.

any truncation of the (compressed) bit-stream also yields an efficient compressed representation of the source in a rate-distortion sense. That is, the distortion for the truncation of the embedded stream to a particular rate should be comparable to the distortion achievable for that rate by any compressed representation, embedded or otherwise.

In this paper, we develop a level-embedded compression scheme, which is a specific instance of scalable compression.[1] For an R bit-image, the method generates an embedded bit-stream that allows scalability in the pixel-wise peak (maximum) absolute error (PAE) ranging from 0 (lossless) to $2^{(R-1)}$ (single bit-thresholded representation) in suitably chosen steps. The method is useful in several applications, where data is acquired by a capture device with a high dynamic range or bit-depth. A lower bit-depth representation is often sufficient for most purposes and the higher bit-depth data is only required for specialized analysis/enhancement or archival purposes. For example, a digital camera may preserve an acquired image without loss at full bit-depth of the acquisition device, but truncate later if necessary, say in order to create space for additional images. If the full bit-depth image is stored in a conventional lossless compressed stream, a subsequent truncation of lower order bits requires a decompression and reconstruction of the image prior to truncation, followed by a compression of the resulting level-truncated image. If on the other hand, the compression scheme (and the corresponding bit-stream) is level-embedded, the truncation can effectively be performed in the bit-stream itself by dropping the segment of the stream corresponding to the truncated lower levels. The latter option is often much more desirable because of its memory and computational simplicity, which translate to lower power, time, and memory requirements.

JPEG2000 offers scalability in resolution and distortion by allowing reconstruction of lower resolution and/or lower signal-to-noise-ratio (SNR) images. The scalability in JPEG2000 is, however, different from the scalability provided by level-embedded compression. Resolution scalability in JPEG 2000 is a natural outcome of the discrete wavelet transform on which it is based. SNR scalability is implemented by aligning quantized bit-planes of sub-band wavelet coefficients according to their mean square error (MSE) significance and encoding these progressively in an order in which the data with the largest distortion reduction per average bit of compressed representation is coded first. This ensures that a truncation of the encoded bit-stream at any point closely approximates the MSE optimal reconstruction for the corresponding file size or rate. However, this truncation in the wavelet transform coefficient domain does not, in general, offer any reasonable PAE guarantees in the image pixel value domain. On the other hand, since the truncation in level-embedded compression occurs directly in the pixel domain, it offers tight MSE and PAE bounds in the image pixel domain. In this sense, the scalability in level-embedded compression corresponds to an embedding with respect to the L^{∞} norm in pixel domain whereas the SNR scalability of JPEG2000 roughly corresponds to an L^2 norm embedding. Finally we note that because it is a point-wise operation with no spatial interactions, level-embedded compression is free from spatial artifacts that may be encountered with JPEG2000 in certain images due to the spatial spread of the wavelet bases used.

As a result, in several applications, level-embedded scalability is more natural and acceptable than the scalability in JPEG2000. Document scanning applications offer a specific example, where one may require archival of complete gray level information, even though most users of the data may need only thresholded bi-level information. These dual needs are readily and efficiently met by using level-embedded compression. Another example is the use of digital photography for legal evidence, where level-embedded scalability may be more acceptable because the potential for spatial artifacts in alternate scalable compression schemes may cast doubts on the veracity of photographic evidence. The bit-depth truncation in level-embedded compression is analogous to using an acquisition device with a lower resolution

[1] A preliminary paper describing the method presented here appears in [2].

A/D converter and therefore likely to be more acceptable. In other non-critical applications, the JPEG2000 scalability based on wavelet domain truncation is often superior to level truncation, because it results in a smaller visual distortion. JPEG-LS, in its near-lossless compression mode, provides per pixel maximum absolute error guarantees without introducing any spatial artifacts, as in level-embedded compression. In this mode, however, JPEG-LS provides only lossy compression at a fixed level and not an embedded lossless stream that allows subsequent level scalability.

Level-embedded compression may be achieved through independent compression of individual bit-planes as in JBIG [3]. If the process is applied directly to the natural binary coded values of image pixels it performs extremely poorly in comparison to non-level-embedded methods. Gray-coding [6, pp. 177–178] of the sample values prior to compression of individual bit-planes provides a major improvement by increasing the correlations among spatially adjacent bits in a plane, but still fails to exploit correlations between the different bit-planes of an image and consequently incurs a significant penalty in compression performance over non-level-embedded methods. In this paper, we propose an alternative method for achieving level embedded compression which significantly reduces the penalty in compression performance by exploiting the correlations among different bit-planes as well as the correlations among neighboring pixels.[2]

The remainder of this paper is organized as follows. In Section 2, we develop the level-embedded compression scheme, with subsections detailing the underlying context modeling, and context adaptive prediction and entropy coding components. Experimental results obtained on test images using this compression scheme are presented in Section 3. Finally, concluding remarks and a discussion are included in Section 4.

[2] During the course of writing this paper, the authors became aware of recent independent work by Avcibas et al. [1] which provides an alternate algorithm for progressive lossless and near lossless image compression with similar functionality.

2. Level-embedded compression algorithm

We first describe the algorithm for the case of two embedding levels: a base layer corresponding to the higher levels and a residual layer comprising of the lower levels. The method is subsequently generalized to multiple levels in Section 2.4.

The image is separated into the base layer and a residual layer. The base layer is obtained by dividing each pixel value by a constant integer L ($B_L(s) = \lfloor s/L \rfloor$). L specifies the amplitude of the enhancement layer, which is the remainder, which is also called *the residual* ($r = s - L\lfloor s/L \rfloor$). We also call the quantity $L\lfloor s/L \rfloor$ as the quantized pixel, $Q_L(s)$. Note that the use of a power of 2 for L corresponds to partitioning of the images into more significant and less significant bit-planes, and other values generalize this notion to a partitioning into higher and lower levels.

Since the resulting base layer representing the most significant levels of the image is coded without any reference to the enhancement layer and its statistics closely resemble those of a full bit-depth image, any lossless compression algorithm is well suited for this layer. In this paper, we use the CALIC algorithm for the base layer compression. Details of the CALIC algorithm may be found in [10,11]. The compression of the enhancement layer is outlined in greater detail below.

Since the enhancement layer, or the residual signal, represents the lowest levels of a continuous-tone image, its compression is a challenging task. For small values of L, the residual typically has no structure, and its samples are virtually uniformly distributed and uncorrelated from sample to sample. Direct compression of the residual therefore is highly inefficient. However, if the rest of the image information is used as side-information, significant coding gains can be achieved in the compression of the residual, by exploiting the spatial correlation among pixel values and the correlation between high and low levels (bit-planes) of the image.

The proposed method for the compression of the enhancement layer has three main components: (i) prediction, (ii) context modeling and quantization, (iii) conditional entropy coding. The overall structure and components for the enhancement layer compression scheme are inspired by the CALIC algorithm [10] but adapted to the special case of encoding an enhancement layer rather than a full image. The prediction component is aimed at decreasing the redundancy in the enhancement layer data by exploiting both correlations with the already decoded base layer and with available spatial neighbors. The context modeling stage allows the prediction to adapt to locally varying statistics in the image and also enables the same adaptation for the conditional entropy coding. Finally, the conditional entropy coding uses context dependent probability models to encode the information into the smallest number of bits. The algorithm is presented below in pseudo-code.

```
1. $\hat{s}_O$ = Predict Current Pixel( );
2. $d, t$ = Determine Context D, T($\hat{s}_O$);
3. $\dot{s}_O$ = Refine Prediction($\hat{s}_O, d, t$);
4. $\theta$ = Determine Context $\Theta(\dot{s}_O)$;
5. If ($\theta \geqslant 0$),
       Encode/Decode Residual($r_O, d, \theta$);
   else,
       Encode/Decode Residual($L-1-r_O, d, |\theta|$);
```

2.1. Prediction

Prediction is based on a local neighborhood of a pixel which consists of its 8-connected neighbors. In this neighborhood, we denote the current (center) pixel(residual) position by O, and neighboring positions by standard map directions: W, NW, $N, \ldots$. The residual samples are encoded and decoded in the raster scan order, i.e. left-to-right and top-to-bottom. This order guarantees that residuals at positions W, NW, N, NE have already been reconstructed when the center residual, r_O, is being decoded. In addition, all quantized pixel values of the image, $Q_L(\mathbf{s})$, are known as side-information. Therefore, at a given position, pixel values $s = Q_L(s) + r$ at positions W, NW, N, NE and quantized pixel values $Q_L(s)$ at positions E, SE, S, SW are known. To simplify the notation, we define a reconstruction function $f(.)$, which gives the best known value of a neighboring pixel, exact value if known, or the quantized value plus $L/2$ (to compensate for the bias in the truncation $Q_L(.)$).

$$f(s_k) = \begin{cases} s_k & \text{if } k \in \{W, NW, N, NE\}, \\ Q_L(s_k) + \frac{L}{2} & \text{otherwise.} \end{cases} \tag{1}$$

A simple, linear prediction for the current pixel value is calculated using the nearest, 4-connected neighbors of a pixel.

$$\hat{s}_O = \frac{1}{4} \sum_{k \in \{W,N,E,S\}} f(s_k). \tag{2}$$

Since this predictor is often biased, resulting in a non-zero mean for the prediction error, $s_O - \hat{s}_O$, we refine this prediction and remove its bias using a feed-back loop, on a per-context basis as in [10]. The refined prediction is calculated as

$$\dot{s}_O = \text{round}(\hat{s}_O + \bar{\varepsilon}(d, t)), \tag{3}$$

where "round(·)" is the integer round, and $\bar{\varepsilon}(d, t)$ is the average of the prediction error ($\varepsilon = s_O - \dot{s}_O$) over all previous pixels in the given context (d, t). In order to avoid the propagation of rounding errors, the average prediction error is computed from the refined prediction instead of the raw prediction in Eq. (2). The resulting predictor $\dot{s}_O$ is a context-based, adaptive, non-linear predictor.

2.2. Context modeling and quantization

Typical natural images exhibit non-stationary characteristics with varying statistics in different regions. This causes significant degradation in performance of compression algorithms that model the image pixels with a single statistical model such as a universal probability distribution. If the pixels can be partitioned into a set of *contexts*, such that within each context the statistics are fairly regular, the statistics of the individual contexts (e.g. probability distributions) may be exploited in encoding the corresponding pixels (residuals) using conditional entropy coding. If the contexts and the corresponding statistical models are chosen appropriately, this process can yield significant improvements in coding efficiency. The context selection problem addresses the fundamental trade-off concerning the number of contexts. Increasing number of contexts better adapt to the local image statistics hence improve

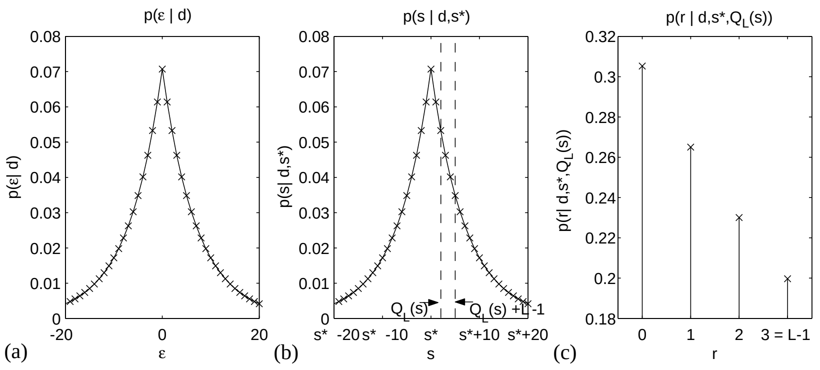

Fig. 1. (a) Prediction error PMF, $p(\varepsilon|d)$, under Laplacian assumption ($\sigma_d = 10$). (b) Corresponding pixel PMF $p(s = \dot{s} + \varepsilon|d, \dot{s})$. (c) Conditional PMF of the residual ($L = 4$), $p(r|d, \dot{s}, Q_L(s))$.

the coding efficiency. Since the corresponding conditional statistics often have to be learned on-the-fly observing the previously encoded (decoded) symbols, convergence of these statistics and thereby efficient compression is delayed when a large number contexts are used. The reduction in compression efficiency due to large number of contexts is known as the *context-dilution* problem. A good context model should avoid context-dilution by choosing the optimum number of contexts.

As a first step, we adopt a variant of d and t contexts from [10], which are defined as follows:

$$\Delta = \sum_{k\in\{W,NW,N,NE,E,SE,S,SW\}} \tfrac{1}{8}|f(s_k) - \hat{s}_O|, \tag{4}$$

$$d = Q(\Delta), \tag{5}$$

$$t_k = \begin{cases} 1 & \text{if } f(s_k) > \hat{s}_O, \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

$$t = t_W \| t_N \| t_E \| t_S, \tag{7}$$

where t is obtained by concatenating the individual t_k bits (16 values), and $Q(\Delta)$ is a scalar non-uniform quantizer with 8 levels, whose thresholds are experimentally determined so as to include an approximately equal number of pixels in each bin.[3] The context d corresponds to local activity as measured by the mean absolute error of the unrefined predictor Eq. (2) and t corresponds to a texture context that is based on the relations of the individual neighbors to the unrefined prediction.[4]

As described earlier in 2.1, for each pixel, the (d, t) context is determined and the prediction is refined by using the average prediction error for the previous pixels in the context as in Eq. (3). In the encoding step, the average prediction error for the context is then updated using the prediction error for the current pixel, in the decoding step, the pixel is first decoded and the update follows.

Typically, the probability distribution of the prediction error, $\varepsilon = s - \dot{s}$, can be approximated fairly well by a Laplacian distribution with zero mean and a small variance which is correlated with the context d [6–8, p. 33]. In order to make precise statements, for the following discussion, we assume that the prediction error distribution $p(\varepsilon|d)$ is exactly Laplacian with variance σ_d determined by d. The arguments and the ensuing conclusions and techniques, however, are largely applicable even when the true distributions deviate from this assumption. Fig. 1a shows a plot of the probability mass function (pmf) $p(\varepsilon|d)$ under this assumption. Given $\dot{s}$, the conditional probability distribution of pixel values $p(s = \dot{s} + \varepsilon|d, \dot{s})$ is obtained by shifting the prediction error distribution $p(\varepsilon|d)$ by $\dot{s}$. The corresponding pmf is illustrated in Fig. 1b.

[3] For the experimental results of Section 3, the quantizer $Q(\,)$'s threshold are $\{1, 2, 3, 4, 6, 10, 15\}$.

[4] In order to avoid context-dilution during coding, t contexts are used only during prediction and not while coding.

In base layer compression algorithms, pixel values are coded using these conditional probability distributions. However, the residual compression problem deviates from the base layer compression problem in two aspects: (i) The residual's probability distribution for entropy encoding should be obtained from the pixel statistics; and (ii) The quantized value of the pixel $Q_L(s)$ is known, and this knowledge should be exploited. We address these issues by introducing an additional context, θ, which is used only in the coding process and not in prediction.

In order to motivate the context θ, note that the known quantized value $Q_L(s)$ may be used as an additional context directly. A known quantized pixel value, $Q_L(s)$, limits the possible values of the pixel s to the range $[Q_L(s), Q_L(s)+L)$. This is illustrated in Fig. 1b as the region between the two vertical broken lines. The conditional pmf $p(r|d,\dot{s},Q_L(s))$ can therefore be obtained by normalizing this segment of the pmf to sum up to 1. Fig. 1c illustrates the conditional pmf $p(r|d,\dot{s},Q_L(s))$ obtained for the segment illustrated in Fig. 1b. Entropy coding the residual using this conditional pmf restricts the symbol set required thereby improving compression. Note, however, that there are typically a large number of possible values for $Q_L(s)$, which would cause significant context-dilution since a large number of samples would be required to learn the statistics for each of these contexts on the fly. The characteristics of the Laplacian distribution, however, allow for a significant reduction in the number of these contexts.

Since the Laplacian distribution decreases exponentially about its peak at $\dot{s}$, the conditional pmf $p(r|d,\dot{s},Q_L(s))$ can be determined from the relative positions of $\dot{s}$ and $Q_L(s)$. For instance, if $\dot{s} \leqslant Q_L(s)$, the peak is at $r=0$ and the pmf decreases exponentially and is identical for all cases corresponding to $\dot{s} \leqslant Q_L(s)$. This case corresponds to the one illustrated in Fig. 1b and c. This allows all the cases corresponding to $\dot{s} \leqslant Q_L(s)$ to be combined into a single composite context. Similarly, if $\dot{s} \geqslant Q_L(s)+L-1$, the peak is at $r=L-1$ and the distribution increases exponentially, which may all be combined into a single context as well. In other cases, when $Q_L(s) < \dot{s} < Q_L(s)+L-1$, the peak is at $r=\dot{s}-Q_L(\dot{s})$. Although total number of contexts after the above reductions is not large, it can be reduced further, if the symmetry of the Laplacian is exploited.

The symmetry of possible residual statistics is illustrated in Fig. 2. In particular, the distributions with peaks at r_θ and $L-1-r_\theta$ are mirror images of each other. If the residual values are re-mapped (flipped $r_{\text{new}} = L-1-r_{\text{old}}$) in one of these two contexts, the resulting distributions will be identical. As a result, we can merge these contexts without incurring any penalty. Furthermore, we encode the re-mapping instruction into the sign of the θ context. We assign each pair of symmetric distributions to an opposite sign, equal magnitude context value $(\pm\theta_i)$. During entropy encoding, first the residuals are re-mapped if necessary. Afterwards, the absolute value of θ is used as the coding context, together with d.

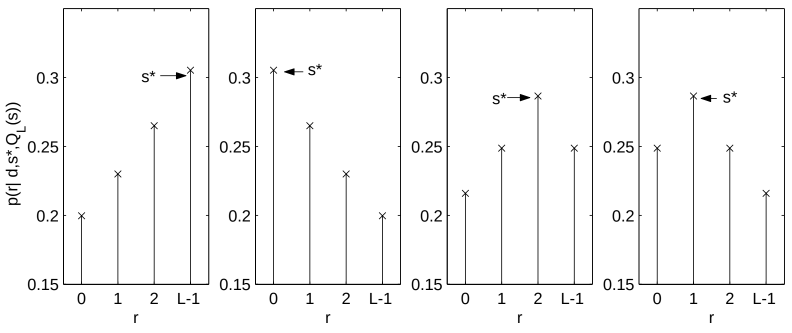

Fig. 2. Conditional PMFs $p(r|d,\dot{s},Q_L(s))$ for contexts $\theta=\{\pm 1,\pm 2\}$ $(L=4)$. Symmetric contexts are merged by re-mapping the residual values.

The θ contexts differentiate between statistically different (after incorporating all symmetries) residuals using the knowledge of $\dot{s}$ and $Q_L(s)$. This enables the conditional entropy coder to adapt to the corresponding probability distributions in order to achieve higher compression efficiency. Minimizing the number of such contexts allows the estimated conditional probabilities to converge to the underlying statistics faster. Therefore, it prevents context dilution and improves the compression efficiency.

Finally, we have empirically determined that assigning a separate θ context to the cases $\dot{s} = Q_L(s)$ and $\dot{s} = Q_L(s) + L - 1$ further enhances the compression efficiency. These cases have been formerly included in the context where $\dot{s} \leqslant Q_L(s)$ and $\dot{s} \geqslant Q_L(s) + L - 1$. We believe that the rounding in Eq. (3) partially randomizes the prediction when $Q_L(s) \approx \dot{s}$ and causes this phenomenon. The number of θ contexts and (d, θ) coding contexts become $\lfloor (L+1)/2 + 1 \rfloor$ and $8\lfloor (L+1)/2 + 1 \rfloor$, respectively.

2.3. Conditional entropy coding

At the final step, residual values are entropy coded using estimated probabilities conditioned on different contexts. In order to improve efficiency, we use a context-dependent adaptive arithmetic coder [9] as in [10]. In a context-dependent adaptive entropy coder, conditional probability distribution of residuals in each coding context (d, θ) is estimated from previously encoded(decoded) residual values. That is, the observed frequency of each residual value in a given context approximates its relative probability of occurrence. These frequency counts are passed to an arithmetic coder which allocates best code-lengths corresponding to given symbol probabilities.

2.4. Multi-level-embedded coding

The above description outlined level-embedded compression for two levels, a base layer and a single enhancement level. Multi-level embedded coding can be obtained as a straightforward extension by applying the algorithm recursively. In the first stage, the image is separated into a base layer B_1 and an enhancement layer r_1 using level L_1. In the second stage, the base layer B_1 is further separated into a base layer B_2 and enhancement layer r_2 using a (potentially different) level L_2. The process is continued for additional stages as desired. Each enhancement layer r_i is compressed using the corresponding base layer B_i, and last base layer B_n is compressed as earlier.

3. Experimental results

We evaluated the performance of the proposed scheme using the six 512×512 8-bit gray-scale images seen in Fig. 3.

Although the algorithm works for arbitrary values of the embedding level L, in order to allow comparison with bit-plane compression schemes, here we concentrate on bit-plane embedded coding, which corresponds to using $L = 2$. Furthermore, the recursive scheme outlined in Section 2.4 is used to obtain multi-level embeddings with more than one enhancement layer, each consisting of a bit-plane. The number of enhancement layers, i.e. embedded bit-planes, is varied from 1 to 7. One (1) enhancement layer corresponds to the case where the LSB-plane is the enhancement layer and 7 MSB-planes form the base layer. Likewise, seven (7) enhancement layers correspond to a fully scalable bit-stream, where all bit-planes can be reconstructed consecutively, starting with the most significant and moving down to the least significant. As indicated earlier, in each case, the corresponding base layer is compressed using CALIC algorithm.

In Table 1, the performance of the proposed algorithm is compared with that of state-of-the-art lossless compression methods. The methods included in this benchmarking include the regular (non-embedded) lossless compression methods: CALIC, JPEG2000, JPEG-LS; and embedded compression using JBIG (independent bit-planes), gray-coded JBIG-“JBIG(gray)”, and the level-embedded scheme proposed in this paper. The different level embeddings are denoted as L.E. 1, L.E. 2, ..., L.E. 7 for the cases corresponding to $1, 2, \ldots, 7$ enhancement layers. In our experiments, CALIC provided the best compression rates for

Fig. 3. Test images used for experiments. Each image is 512 × 512 in size and has 256 gray levels (8-bits).

Table 1
Performance of level-embedded compression scheme against different lossless compression methods. Percent increase with respect to CALIC is indicated

	Image	F-16	Mand	Boat	Barb	Gold	Lena	Avg.
	Comp. method	Best loss-less compression rate (baseline)						
	CALIC (bpp)	3.54	5.66	4.15	4.42	4.58	4.08	4.40
		Percent increase in bit-rate wrt baseline						
Regular	CALIC	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	JPEG2000	7.6	4.0	6.2	4.6	4.6	5.2	5.2
	JPEG-LS	1.9	2.8	2.4	6.2	1.8	3.4	3.1
Level-embedded	JBIG	46.6	26.2	35.8	36.3	33.6	39.7	35.5
	JBIG(gray)	17.5	11.2	15.8	17.6	13.7	15.8	15.0
	L.E. 1	2.0	0.2	1.6	1.4	0.7	1.1	1.1
	L.E. 2	4.1	0.9	4.1	4.1	2.2	3.7	3.0
	L.E. 3	7.0	2.2	6.3	6.3	4.7	5.8	5.1
	L.E. 4	10.6	3.4	9.9	10.1	6.6	8.6	7.8
	L.E. 5	13.7	5.3	12.4	14.0	8.5	11.7	10.5
	L.E. 6	15.9	6.6	14.5	17.5	10.7	14.4	12.8
	L.E. 7	18.8	7.6	16.4	20.0	12.6	17.5	14.9

non-embedded compression. Therefore, in Table 1, we tabulate results for all non-embedded schemes and the level-embedded scheme proposed here as the percentage increases in bit-rate with respect to the CALIC algorithm.

From the table, it is apparent that JPEG-LS and JPEG2000 offer fairly competitive performance to CALIC with only modest increases in bit-rate. Nonetheless, just like CALIC these methods are not bit-plane scalable. JPEG2000 provides resolution and distortion scalability but not bit-plane scalability. In its default mode, JBIG provides bit-plane scalability, however at a significant loss of coding efficiency (almost a 35% increase in bit-rate over CALIC, on average). The level-embedded compression scheme does significantly better than

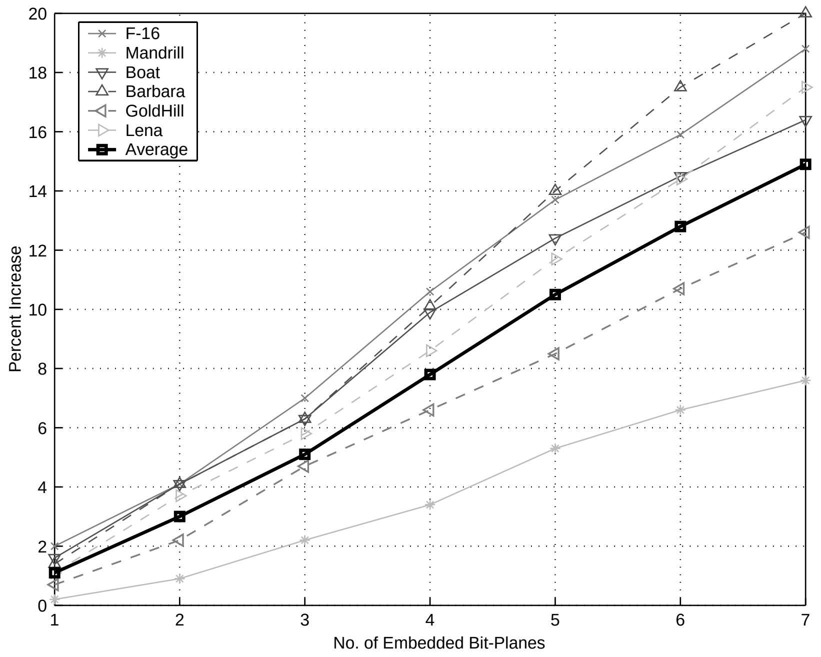

Fig. 4. Percent increase in data-rate over CALIC algorithm for a given number of enhancement layers.

JBIG in this mode. The performance of JBIG is significantly improved when pixel values are gray-coded [6, pp. 177–178] prior to separation into bit-planes. In the gray-coded representation, spatially adjacent bits have a significantly higher correlation than in the natural binary coded [6, p. 177] representation. This translates to improved compression performance. The results for the gray-coded JBIG compression are shown in the row labeled “JBIG(gray)” in the table. If the gray code bit-planes are suitably arranged,[5] for any chosen number of MSB bit-planes, there is a one-to-one correspondence between the values of these planes in the natural binary coded representation and the values of the same MSB planes in the gray-coded representation. This property allows for a bit-plane-embedded compressed stream to be constructed using JBIG(gray)—with some additional processing. In this case, JBIGs performance is comparable to that of the proposed method. Nevertheless, proposed scheme offers additional flexibility in that the number of embedded levels are not constrained to be equal to the number of bit-planes and a smaller number of embedded levels can be used if required with a corresponding improvement in compression efficiency. For a small number of embedding levels the penalty is quite small with up to 4 enhancement layers requiring under 8% increase in bit-rate over CALIC.

In Fig. 4 the relative performance results from Table 1 are plotted for the level-embedded compression scheme. From the figure, we see that the proposed method incurs a penalty which increases roughly linearly for each image with increase in the number of enhancement layers (embedded bit-planes). In a hypothetical application, where 2 bit-planes are embedded, for instance, to truncate 8-bits to 6-bits in a digital camera, the increase in bit-rate is 3% on the average. This number is quite competitive with the non-scalable JPEG-LS and CALIC algorithms in view of the added functionality. It is also better than the corresponding rate for the JPEG2000 algorithm. When all bit-planes are embedded the

[5]Note that the permutation of the bit-planes of a gray-code retains its gray-code properties.

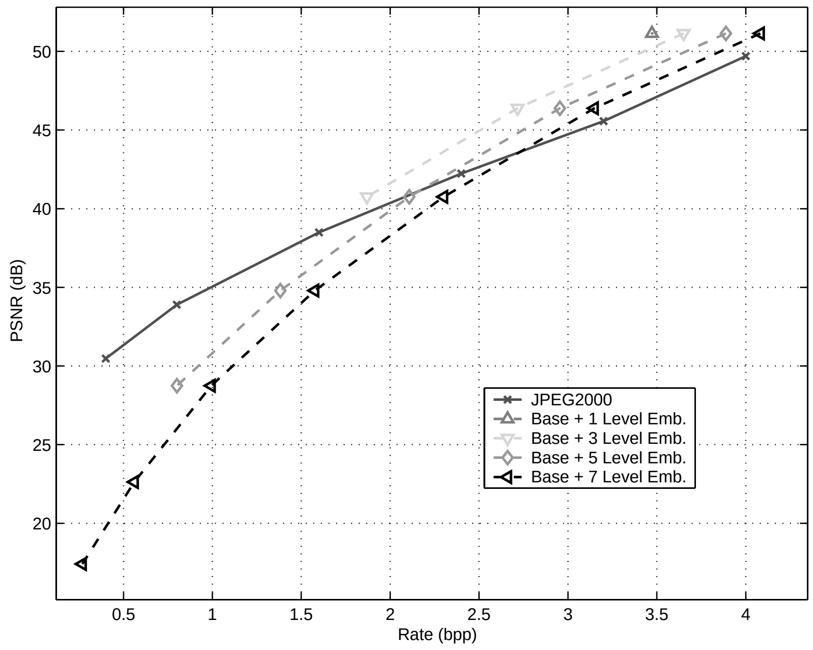

Fig. 5. Comparison of rate-distortion (MSE) performance (averaged over all images) for proposed level-embedded scheme and JPEG2000.

penalty increases to 15%. This is approximately equal to the JBIG(gray) and is considerably worse than the JPEG2000, where alternate scalability is provided. The degradation at higher levels of embedding is not a major concern because most applications of level-embedded compression are likely to require only a small number of embedded bit-planes.

Note that at first the loss of efficiency in level-embedded compression may seem non-intuitive because the information in the base layer is exploited in the compression of the enhancement layer. However, the information in the enhancement layers is not utilized in the compression of the base layer and subsequent enhancement layers—this can lead to inefficiency in the compression of the base layer. For instance if at a given pixel, multiple bit-planes in increasing order of significance are zero, it is likely that the subsequent higher order bit-planes will also be zero. This information cannot be exploited in level-embedded compression and therefore some penalty over regular loss-less compression is to be expected.

The rate distortion performance of the level-embedded scheme is compared against the rate-distortion performance for JPEG2000 in Fig. 5 for the MSE based peak signal-to-noise-ratio (PSNR) distortion metric and in Fig. 6 for the PAE distortion metric. The MSE, PAE, and PSNR metrics are defined as

$$D(\mathrm{MSE}) = \frac{1}{N}\sum_{i=0}^{N-1}(s_i - \hat{s}_i)^2, \tag{8}$$

$$D(\mathrm{PAE}) = \max_i \|s_i - \hat{s}_i\|, \tag{9}$$

$$\mathrm{PSNR}(dB) = 10\log_{10}\left(\frac{s_{\max}^2}{D(\mathrm{MSE})}\right), \tag{10}$$

where s_i and $\hat{s}_i$ represent the original and reconstructed values, respectively, at pixel position i and the maximum is over all pixels in the image. Note that, for the proposed scheme the reconstruction level is selected as the mid-point of the quantization interval. For instance, if only the most significant bit is received and its value is zero then actual value of the pixel lies in $[0, 128)$

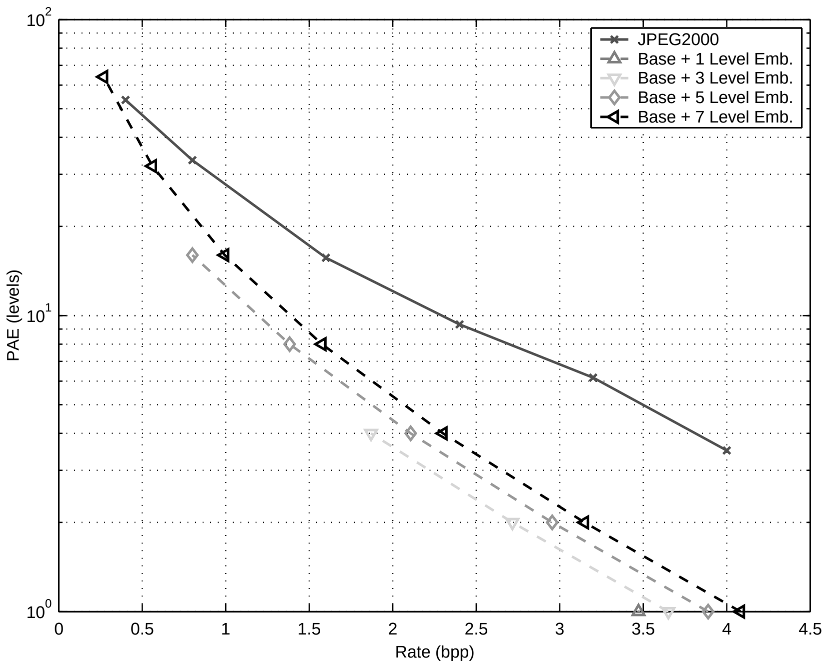

Fig. 6. Comparison of rate-distortion $\|\cdot\|_\infty$ performance (averaged over all images) for proposed level-embedded scheme and JPEG2000.

(for an 8-bpp image) and $\hat{s} = 64$ is selected as the reconstruction value.

Fig. 5 shows the rate in bits per pixel vs. the MSE PSNR for the proposed level-embedded compression scheme (for different choices of embedding) and for JPEG2000. In order to generate the points on these plots for JPEG2000, the rate for the JPEG2000 encoder was adjusted to roughly match the MSE distortion points for the different embedding levels. Note that for high rates the proposed scheme offers a better rate distortion performance but at lower rates JPEG2000 does better in the MSE distortion sense.

Fig. 6 shows a plot of the rate vs. peak absolute error (PAE) for the proposed level-embedded compression scheme (for different choices of embedding) and for JPEG2000. For JPEG2000, the compressed images used to generate Fig. 5 were used for estimating the PAE for this graph. As may be expected, the proposed level-embedded scheme offers a superior rate distortion performance in comparison to JPEG2000 wrt the PAE metric. For all points shown, the proposed embedding scheme has a significantly lower PAE than JPEG2000 for the same rate.

4. Conclusions

We present a level-embedded lossless image compression method, which enables bit-plane scalability, or more generally level scalability. In situations, where the resulting compressed bit-stream needs to be truncated to produce a lower bit-rate (and lower quality) image, the proposed scheme guarantees freedom from compression induced spatial artifacts and tight bounds on per pixel maximum error, making it especially suitable in certain medical, legal, and document imaging applications. Experimental results comparing the method with state-of-the-art lossless compression methods indicate that level scalability is achieved with only a small penalty in the compression efficiency over regular (non-level-embedded) compression schemes.

References

[1] I. Avcibas, N. Memon, B. Sankur, K. Sayood, A progressive lossless/near-lossless image compression algorithm, IEEE Signal Process. Lett. 9 (10) (2002) 312–314.

[2] M.U. Celik, A.M. Tekalp, G. Sharma, Level-embedded lossless image compression, in: Proceedings of IEEE ICASSP, Hong Kong, April 2003, accepted for presentation.

[3] ISO/IEC 11544, Information technology—coded representation of picture and audio information—progressive bi-level image compression, 1993.

[4] ISO/IEC 14495-1, Lossless and near-lossless compression of continuous-tone still images—baseline, 2000.

[5] ISO/IEC 15444-1, Information technology-JPEG 2000 image coding system-part 1: Core coding system, 2000.

[6] N.S. Jayant, P. Noll, Digital Coding of Waveforms: Principles and Applications to Speech and Video, Prentice-Hall, Englewood Cliffs, NJ, 1984.

[7] J.B. O'Neal, Predictive quantizing differential pulse code modulation for the transmission of television signals, Bell System Technical J. 45 (1966) 689–721.

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

[9] I.H. Witten, M. Radford, J.G. Cleary, Arithmetic coding for data compression, Comm. ACM 30 (6) (1987) 520–540.

[10] X. Wu, Lossless compression of continuous-tone images via context selection, quantization, and modelling, IEEE Trans. Image Process. 6 (5) (1997) 656–664.

[11] X. Wu, N. Memon, Context-based, adaptive, lossless image codec, IEEE Trans. Comm. 45 (4) (1997) 437–444.