LOCO-I: A Low Complexity, Context-Based, Lossless Image Compression Algorithm

Marcelo J. Weinberger, Gadiel Seroussi, and Guillermo Sapiro
Hewlett-Packard Laboratories, Palo Alto, CA 94304

LOCO-I: A Low Complexity, Context-Based, Lossless Image Compression Algorithm

Marcelo J. Weinberger, Gadiel Seroussi, and Guillermo Sapiro
Hewlett-Packard Laboratories, Palo Alto, CA 94304

Abstract. LOCO-I (LOw COmplexity LOssless COmpression for Images) is a novel lossless compression algorithm for continuous-tone images which combines the simplicity of Huffman coding with the compression potential of context models, thus "enjoying the best of both worlds." The algorithm is based on a simple fixed context model, which approaches the capability of the more complex universal context modeling techniques for capturing high-order dependencies. The model is tuned for efficient performance in conjunction with a collection of (context-conditioned) Huffman codes, which is realized with an adaptive, symbol-wise, Golomb-Rice code. LOCO-I attains, in one pass, and without recourse to the higher complexity arithmetic coders, compression ratios similar or superior to those obtained with state-of-the-art schemes based on arithmetic coding. In fact, LOCO-I is being considered by the ISO committee as a replacement for the current lossless standard in low-complexity applications.

1 Introduction

Lossless image compression schemes often consist of two distinct and independent components: *modeling* and *coding.* The modeling part can be formulated as an inductive inference problem, in which an image is observed pixel by pixel in some pre-defined order (e.g., raster-scan). At each time instant i, and after having scanned past data $x^i = x_1 x_2 \cdots x_i$, one wishes to make inferences on the next pixel value x_{i+1} by assigning a conditional probability distribution $p(\cdot|x^i)$ to it.[1] Ideally, the code length contributed by x_{i+1} is $-\log p(x_{i+1}|x^i)$ bits (hereafter, logarithms are taken to the base 2), which averages to the entropy of the probabilistic model. Thus, a skewed (low-entropy) probability distribution, which assigns a high probability value to the next pixel, is desirable. In a *sequential* formulation, the distribution $p(\cdot|x^i)$ is learned from the past and it is available to the decoder as it decodes the past string sequentially. Alternatively, in a two-pass scheme the conditional distribution can be learned from the whole image in a first pass and must be sent to the decoder as header information. In this case, the total code length includes the length of the header. Yet, both the second encoding pass and the (single-pass) decoding are subject to the same sequential formulation.

In principle, skewed distributions can be obtained through larger conditioning regions or "contexts." However, this implies a larger number of parameters in the statistical model, with an associated *model cost* [1] which could offset the entropy savings. In a sequential scheme, this cost can be interpreted as capturing the penalties of "context dilution" occurring when count statistics must be spread over too many contexts, thus affecting the accuracy of the corresponding estimates. In non-sequential (two-pass) schemes the model cost represents the code length required to encode the model parameters estimated in the first pass, which must be transmitted to the decoder. In both cases the model cost is proportional to the number of free parameters in the statistical model [2].

[1]Notice that pixel values are indexed with only one subscript, despite corresponding to a two-dimensional array. This subscript denotes the "time" index in the pre-defined order.

In state-of-the-art lossless image compression schemes, the above probability assignment is generally broken into the following components:

a. A prediction step, in which a deterministic value $\hat{x}_{i+1}$ is guessed for the next pixel x_{i+1} based on a subset of the available past sequence x^i (a *causal template*).
b. The determination of a *context* in which x_{i+1} occurs (again, a function of a past subsequence).
c. A probabilistic model for the *prediction residual* (or error signal) $e_{i+1} \stackrel{\Delta}{=} x_{i+1} - \hat{x}_{i+1}$, conditioned on the context of x_{i+1}.

The optimization of these steps, inspired on the ideas of *universal modeling*, is analyzed in [1], where a high complexity scheme is proposed. In this scheme, the prediction step is accomplished with an adaptively optimized, context-dependent linear predictor, and the statistical modeling is performed with an optimized number of parameters (variable-size quantized contexts). The modeled prediction residuals are *arithmetic encoded* in order to attain the ideal code length [3]. In [4], some of the optimizations performed in [1] are avoided, with no deterioration in the compression ratios. Yet, the modeling remains in the high complexity range. In various versions of the Sunset algorithm [5, 6], a further reduction in the modeling complexity, via a fixed predictor and a non-optimized context model, causes some deterioration in the compression ratios. Moreover, the models used in all these algorithms provide best results with arithmetic coding of prediction residuals, an operation that is considered too complex in many applications, especially for software implementations. Other alternatives have been designed with simplicity in mind and propose minor variations of traditional DPCM techniques [7], which include Huffman coding of prediction residuals obtained with some fixed predictor. Thus, these simpler techniques are fundamentally limited in their compression performance by the first order entropy of the prediction residuals.

LOCO-I combines the simplicity of Huffman (as opposed to arithmetic) coding with the compression potential of context models, thus "enjoying the best of both worlds." The algorithm uses a non-linear predictor with rudimentary edge detecting capability, and is based on a very simple context model, determined by quantized gradients. A small number of free statistical parameters is used to approach the capability of the more complex universal context modeling techniques for capturing high-order dependencies [1], without the drawbacks of "context dilution." This is achieved through a single-parameter probability distribution per context, efficiently implemented by an adaptive, symbol-wise, Golomb-Rice code [8, 9]. In addition, reduced redundancy is achieved by embedding an alphabet extension in "flat" regions.

A low complexity scheme based on Golomb-Rice coding, FELICS, was previously described in [10]. LOCO-I differs significantly from FELICS in that it follows a more traditional predictor-modeler-coder structure along the paradigm of [1], in the use of a novel and extremely simple explicit formula for Golomb-Rice parameter estimation (described in Section 2.3), as opposed to the search procedure described in [10], and in the use of an embedded alphabet extension to reduce code redundancy. As shown in the results Section 3, LOCO-I attains compression ratios that are significantly better than those of FELICS, while maintaining the same low complexity level. In fact, the results show that LOCO-I compresses similarly or better than state-of-the art schemes based on arithmetic coding, at a fraction of the complexity. LOCO-I is

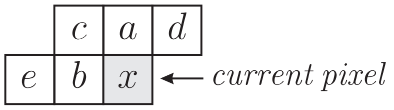

Figure 1: A causal template for LOCO-I

being considered by the ISO/IEC JTC1/SC29/WG1 committee as a replacement for the current lossless standard in low-complexity applications.

2 A description of LOCO-I

The prediction and modeling units in LOCO-I are based on the *causal template* depicted in Figure 1, where x denotes the current pixel, and a, b, c, d, and e are neighboring pixels in the relative positions shown in the figure. The dependence of a, b, c, d, and e on the time index i has been deleted from the notation for simplicity.

2.1 Prediction

Ideally, guessing the value of the current pixel x_{i+1} based on a, b, c, d, and e should be done by adaptively learning a model conditioned on the local edge direction. However, our complexity constraints rule out this possibility. Yet, a primitive edge detector is still desirable in order to approach the best possible predictor. The approach in LOCO-I consists on performing a primitive test to detect vertical or horizontal edges. If an edge is not detected, then the guessed value is $a+b-c$, as this would be the value of x_{i+1} if the current pixel belonged to the "plane" defined by the three neighboring pixels with "heights" a, b and c. This expresses the expected smoothness of the image in the absence of edges. Specifically, the LOCO-I predictor guesses:

$$\hat{x}_{i+1} \triangleq \begin{cases} \min(a, b) & \text{if } c \geq \max(a, b) \\ \max(a, b) & \text{if } c \leq \min(a, b) \\ a+b-c & \text{otherwise.} \end{cases} \tag{2}$$

Assuming, without loss of generality, that $a \leq b$, then the predictor of (2) can be interpreted as picking a in many cases where a vertical edge exists left of the current location, b in many cases of an horizontal edge above the current location, or a plane predictor $a+b-c$ if no edge has been detected.

The above predictor has been employed in image compression applications [11], although under a different interpretation. In [11], the guessed value is seen as the *median* of three fixed predictors, a, b, and $a+b-c$. Notice that the predictor of (2) does not employ either d or e, which will be used in context modeling.

2.2 Context Modeling

As noted in Section 1, reducing the number of parameters is a key objective in a context modeling scheme. In a sequential formulation, the goal is to avoid "context dilution," while in a two-pass scheme we wish to reduce unnecessary table overhead. The total number of parameters in the model depends on the number of free parameters defining the coding distribution at each context and on the number of contexts.

Coding distributions

It is a widely accepted observation [7] that the distribution of prediction residuals in continuous tone images can often be approximated by a Laplacian distribution, i.e. a two-sided exponential decay centered at zero. In principle, a prediction residual can take on any value ϵ in the range $-(\alpha-1) \leq \epsilon \leq \alpha-1$, where $\alpha = 2^\beta$ is the size of the input alphabet. Actually, given the predicted value $\hat{x}$ (known to both the encoder and decoder), ϵ can take on only α possible different values. One way to exploit this property is to take the actual value of the prediction residual and reduce it modulo α to a value between $-\alpha/2$ and $\alpha/2-1$. This has the effect of remapping large prediction residuals, which have very low probability in smooth images, to small prediction residuals. Thus, the "tails" of the distribution are merged with its central part, without significantly affecting the Laplace-like behavior of the central part. The above remapping consists of just taking the β least significant bits of $x - \hat{x}$, and interpreting them in 2's complement representation.[2]

In the sequel, we assume that prediction residuals ϵ are in the range $-\alpha/2 \leq \epsilon \leq \alpha/2 - 1$, and their distribution is assumed to decay exponentially away from zero.

In a one-pass scheme, the encoder cannot tune an optimal Huffman table to each possible distribution of prediction residuals, as the encoding needs to be done "on the fly." Adaptive construction of optimal tables is ruled out by the complexity constraints. Instead, for each context the encoder adaptively chooses the best among a limited set of Huffman codes, matched to exponentially decaying distributions, based on past performance. As these distributions are assumed to be centered at 0, a *single* parameter (e.g., the average of error magnitudes at the corresponding context) is sufficient to characterize each one of them. The adaptive computation of this parameter and the selection of the code are described in Section 2.3.

Context determination
The context that conditions the encoding of the current prediction residual in LOCO-I is built out of the differences $g_1=d-a$, $g_2=a-c$, $g_3=c-b$, and $g_4=b-e$. These differences represent the local gradient, thus capturing the level of activity (smoothness, edginess) surrounding a pixel, which governs the statistical behavior of prediction errors. Notice that this approach differs from the one adopted in the Sunset family [6] and other schemes, where the context is built out of the prediction errors incurred in previous encodings. By symmetry,[3] g_1, g_2, and g_3 influence the model in the same way. Since further parameter reduction is obviously needed, each difference g_j, $j = 1, 2, 3$, is quantized into a small number or approximately *equiprobable* regions (the same regions for $j = 1, 2, 3$). In a well-defined theoretical setting [1], this maximizes the *mutual information* between the current pixel and its context, an information-theoretic measure of the amount of information provided by the conditioning context on the pixel value to be modeled. Difference g_4, being farther away from the predicted pixel, is quantized more coarsely.

In principle, the number of regions into which each context difference is quantized should be adaptively optimized. However, the low complexity requirement dictates a fixed number of "equiprobable" regions. By symmetry, there is one region centered at the difference value 0, and if the interval $[r_1, r_2]$ represents a region, then so does $[-r_1, -r_2]$. Thus, the total number of quantization regions for g_j, $j = 1, 2, 3$, is an odd integer $2R + 1$, while g_4 is quantized into $2T + 1$ regions, $T < R$. This leads to a total of $(2T + 1) \times (2R + 1)^3$ different contexts. A further reduction in the number

[2]Other remappings are possible [12], which might give a slight compression advantage for smooth images. However, these remappings have a complexity cost, and are inferior for images containing a mixture of smooth regions and regions with many sharp transitions, e.g., compound documents.

[3]Here, we assume that the class of images to be encoded is essentially symmetric with respect to horizontal/vertical, left/right, and top/bottom transpositions, as well as the pixel "negation" transformation $x \rightarrow (\alpha-1-x)$.

of contexts is obtained after observing that, by symmetry, it is reasonable to assume

$$\mathrm{Prob}\{e_{i+1} = \Delta \,|\, C_i = [q_1, q_2, q_3, q_4]\} = \mathrm{Prob}\{e_{i+1} = -\Delta \,|\, C_i = [-q_1, -q_2, -q_3, -q_4]\}$$

where C_i represents the quantized context quartet and q_j, $j = 1, \cdots, 4$, are quantized differences corresponding, respectively, to g_j, $j = 1, \cdots, 4$. Hence, by merging contexts of opposite signs, the total number of contexts becomes $((2T+1)(2R+1)^3+1)/2$. As stated before, the coding distributions have only one free parameter per context. Therefore, a large number of contexts can be afforded without incurring an excessive model cost. Our implementation uses $R = 4$ and $T = 1$, resulting in 1094 contexts. Notice that, by merging symmetric contexts, the encoded value may actually be the opposite of the prediction residual. This is anticipated by the decoder, which flips the error sign if necessary to obtain the original error value.

To complete the definition of the contexts in LOCO-I, it remains to specify the values of the boundaries between quantization regions. For an 8-bit per pixel alphabet, the quantization regions for g_j, $j = 1, 2, 3$, are $\{0\}$, $\{1, 2\}$, $\{3, 4, 5, 6\}$, $\{7, 8, \cdots, 14\}$, $\{\, e \,|\, e \geq 15 \,\}$, and their corresponding negative counterparts. The three quantization regions for g_4 are $|g_4| < 5$, $g_4 \geq 5$, and $g_4 \leq -5$.

2.3 Coding

Golomb-Rice codes

Golomb codes were first described in [8], as a means for encoding nonnegative run lengths. Given a positive integer parameter m, the Golomb code G_m encodes an integer $n \geq 0$ in two parts: a *binary* representation of $n \bmod m$, and a *unary* representation of $\lfloor n/m \rfloor$.

Golomb codes are optimal [13] for exponentially decaying (*geometric*) probability distributions of the nonnegative integers, i.e. distributions of the form $Q(n) = (1 - \rho)\rho^n$, where $0 < \rho < 1$. For every distribution of this form, there exists a value of the parameter m such that G_m yields the shortest possible average code length over all uniquely decipherable codes for the nonnegative integers. The optimal value of m is given [13] by

$$m = \left\lceil \log(1 + \rho) \,/\, \log(\rho^{-1}) \right\rceil \,. \tag{3}$$

Rice [9] emphasized the special case of Golomb codes with $m = 2^k$. Choosing m to be a power of 2 leads to very simple encoding/decoding procedures: the code for n consists of the k least significant bits of n, followed by the number formed by the remaining higher order bits of n, in unary representation. The length of the encoding is $k + 1 + \lfloor n/2^k \rfloor$ (the simplicity of the case $m = 2^k$ was already noted in [8]). We refer to codes G_{2^k} as *Golomb-Rice* codes, and we denote them by R_k.

To match the assumption of a two-sided exponentially decaying distribution of prediction residuals, established in Section 2.2, to the optimality of Golomb codes for geometric distributions of nonzero integers, we need to make provisions for encoding of negative values. This is accomplished by means of the following mapping of prediction residuals ϵ in the range $-\alpha/2 \leq \epsilon \leq \alpha/2 - 1$ to values $M(\epsilon)$ in the range $0 \leq M(\epsilon) \leq \alpha - 1$:

$$M(\epsilon) = \begin{cases} 2\epsilon & \epsilon \geq 0, \\ 2|\epsilon| - 1 & \epsilon < 0. \end{cases} \,. \tag{4}$$

The mapping $M(\epsilon)$ orders prediction residuals, interleaving negative values and positive values in the sequence $0, -1, 1, -2, 2, ...$ If the values ϵ follow a Laplacian distribution centered at zero, then the distribution of $M(\epsilon)$ will be close to (but not exactly) geometric, which can then be encoded using an appropriate Golomb-Rice code.

Sequential parameter estimation
One of the crucial steps in schemes using Golomb-Rice coding is the determination of the optimal value of the code parameter k, i.e., in our case, the value yielding the shortest possible average code length for $M(\epsilon)$ under the Laplacian hypothesis for ϵ. Golomb-Rice-based schemes reported in the literature follow two basic approaches:

In a *block-oriented* approach [12], the image is divided into rectangular blocks of pixels (typical sizes are 6×6 and 8×8). For each block, an optimal value of the parameter k for the block is found, and is explicitly transmitted to the decoder together with the encoded block.

In a *sequential* approach (e.g., [10]), each pixel is encoded independently, and the encoder determines the code parameter based on the values of previous pixels. The decoder makes the same determination, after decoding the same past sequence.

With both approaches, the parameter k can be determined either by *exhaustive search*, or by an *explicit computation.* The exhaustive search approach is recommended in both [12] and [10], even though both mention ways of estimating the parameter k from observed values. In [12], an estimation formula based on the sum $F_0 = \sum_x L(R_0(x))$ for pixels x in the block is presented, where $L(y)$ denotes the length of a binary string y. The parameter k is determined according to ranges of values of F_0 (e.g. $k = 1$ if $5J/2 < F_0 \leq 9J/2$, where J is the block size). Reference [10] briefly mentions the possibility of estimating k by first estimating the mean and variance of the distribution over the observed sequence, and then deriving k from the mean and variance.

A sequential scheme is mandatory in a context-based method, as pixels encoded in a given context are not necessarily contiguous in the image and, thus, cannot be easily blocked. The method used in LOCO-I for estimating the value of the parameter k is *sequential*, *explicit*, and based on an estimate of the expectation $E[|\epsilon|]$ of the *magnitude* of prediction errors in the past observed sequence. Using this expectation rather than the variance or F_0 results in a surprisingly simple calculation for k.

Consider a discrete Laplacian distribution $P(\epsilon) = p_0 \gamma^{|\epsilon|}$ for prediction residuals ϵ in the range $-\alpha/2 \leq \epsilon \leq \alpha/2 - 1$. Here, we have $0 < \gamma < 1$, and p_0 is such that the distribution sums to 1. The expected prediction residual *magnitude* is given by

$$a_{\gamma,\alpha} \triangleq E[|\epsilon|] = \sum_{\epsilon=-\alpha/2}^{\alpha/2-1} p_0 \gamma^{|\epsilon|} |\epsilon| \; . \tag{5}$$

We are interested in the relation between the value of $a_{\gamma,\alpha}$ and the average code length $L_{\gamma,k}$ resulting from using the Golomb-Rice code R_k on the mapped prediction residuals $M(\epsilon)$. In particular, we seek to find the value k yielding the shortest code length. For an infinite alphabet, it can be shown [14, Theorem 3] that a good estimate for the optimal value of k is

$$k = \lceil \log_2 a_{\gamma,\alpha} \rceil \; . \tag{6}$$

Numerical calculations on a computer show that this result holds for all practical values of α. In order to implement the estimation dictated by (6), the encoder and the decoder maintain two variables per context: N, a count of prediction residuals seen so far, and A, the accumulated sum of magnitudes of prediction residuals seen so far. The expectation $a_{\gamma,\alpha}$ is estimated by the ratio A/N, and k is computed as

$$k = \min\{k' \,|\, 2^{k'} N \geq A\}.$$

In software, the computation of k can be realized by the following "one-liner" in C:

```
for ( k=0; (N<<k)<A; k++ );
```

To enhance adaptation to local image variations, and general non-stationarity of image statistics, we periodically reset the variables N and A. Specifically, we *half* the contents of N and A for a given context each time N attains a predetermined threshold N_0. In this way, the immediate past is given a larger weight than the remote past. Values of N_0 between 64 and 256 work well for typical images.

Bias cancelation

Golomb-Rice codes rely heavily on the distribution of prediction residuals being a two-sided, symmetric, exponential decay centered at zero. While these assumptions are usually satisfied in the case of memoryless models, the situation is quite different in the case of context-based models, where systematic, context-dependent biases in the prediction residuals are not uncommon. These systematic biases can produce a very significant deterioration in the compression performance of a Golomb-Rice coder.

To alleviate the effect of systematic biases, LOCO-I uses an error feedback aimed at "centering" the distributions of prediction residuals. Specifically, for each context, we keep a count N of context occurrences, a variable B accumulating the prediction errors encoded so far in the context, and a *correction value* C, which is added to the predicted value $\hat{x}$ to offset a possible prediction bias. Variable B accumulates the prediction error *after* correction. Variable C is initialized to zero, and it is incremented (resp. decremented) each time the corrected prediction residual averages 0.5 or more (resp. -0.5 or less). At this time, the accumulated corrected error B is also adjusted to reflect the change in C. This procedure, which can be implemented without divisions, tends to produce average prediction residuals in the interval $[-0.5, 0.5]$.

The bias cancelation mechanism above allows us to deal with another drawback of Golomb-Rice encoding of prediction residuals. Due to the order imposed by the mapping $M(\epsilon)$, the coder may assign a shorter code length to a negative value than to its positive counterpart. This results in the code actually being better matched to a probability distribution with average $-1/2$ for $k > 0$, and $-2/9$ for $k = 0$. Thus, we better match the actual distribution of corrected prediction errors to the code by slightly modifying the bias cancelation calculation so that it produces average corrected prediction residuals in the interval $[-1, 0]$. If $k = 0$ and $B/N < -0.5$ (i.e., the center of the distribution is closer to -1 than to 0), we encode the index $M(-1-\epsilon)$.

2.4 Embedded alphabet extension

Although optimal as a prefix code, a Huffman code may be far from matched to the distribution from which it was generated, if this distribution is very skewed. Huffman codes are characterized by a redundancy (i.e., the excess code length over the entropy) which vanishes only for distributions in which each probability is a negative power of two. A related fundamental limitation is that they require a minimum code length of one bit per encoding, which may produce a significant deterioration in the compression ratio for contexts with very skewed distributions. This is exactly the case in contexts representing smooth regions, for which a prediction error value of 0 is extremely likely. In traditional (non-conditioned) Huffman codes, this problem is addressed through

Image	LOCO-I	FELICS	Sunset CB9
Ballon	2.90	3.21	2.89
Barb 1	4.65	5.22	4.64
Barb 2	4.66	5.18	4.71
Board	3.64	4.03	3.72
Boats	3.92	4.37	3.99
Girl	3.90	4.35	3.90
Gold	4.47	4.75	4.60
Hotel	4.35	4.71	4.48
Zelda	3.87	4.17	3.79
Average	4.04	4.44	4.08

Table 1: Compression results on original JPEG image test set (bits/pixel)

alphabet extension. By encoding blocks of data as "super-symbols," distributions tend to be less skewed (i.e., the excess code length of the Huffman code is spread over many symbols, thus reducing the per-symbol redundancy). Golomb-Rice codes, being a subset of the class of Huffman codes, suffer from the same limitations.

LOCO-I addresses the redundancy of Huffman or Golomb/Rice codes due to very skewed distributions, by embedding an alphabet extension into the context conditioning. Specifically, the encoder enters a "run" mode when a context with $a = b = c = d$ is detected. Since the central region of quantization for the gradients g_1, g_2, g_3 is the singleton $\{ 0 \}$, the run condition is easily detected in the process of context quantization by checking for $[q_1, q_2, q_3] = [0, 0, 0]$. Once in run mode, a run of the symbol b is expected, and the run length (which may be zero) is encoded. When the run is broken by a non-matching symbol x, the encoder goes into an "end-of-run" state, where the difference $\epsilon = x - b$ is encoded (notice that we always have $\epsilon \neq 0$ in this state). Runs can also be broken by ends of lines or by reaching a pre-specified maximum length, in which case the encoder returns to normal context-based coding. Since all the decisions for switching in and out of the run state are based on past pixels, the decoder can reproduce the same decisions without any side information.

The encoding of runs in LOCO-I is also based on Golomb-Rice codes, applied to run lengths *ranked* by occurrence count. Thus, when a run of length r occurs, the *rank* of r is encoded rather than r itself (starting with rank 0 for the most frequent run length, and going up in rank number as frequency decreases). The ranking mechanism adds negligibly to the running time, as the ranking rapidly converges to a stable state.

3 Results

In this section we present compression results obtained with the scheme described in Section 2. In order to allow for comparisons with other relevant schemes reported in the literature over a wide variety of images, two sets of results are presented.

In Table 1, we compare the compression performance of LOCO-I with that of FELICS [10], and with the latest published results for Sunset, i.e., the CB9 algorithm [6], on a standard set of images used in developing the existing ISO/IEC 10918-1 lossless (JPEG) standard. These are generally "smooth" natural images. Compression ratios are given in bits/pixel for the luminance component of each image. The results for FELICS were extracted from [15].

Table 2 shows results of LOCO-I compared with FELICS, and with the two versions of the existing ISO/IEC 10918-1 lossless standard, i.e. the strongest one based on arithmetic coding, and the simplest one based on the default predictor followed by Huffman coding using default tables. The latter is probably the most widely used version of the lossless standard. The table also shows results for the arithmetic coding version of the CALIC algorithm [4], which attains the best compression ratios among schemes recently presented in response to the ISO/IEC JTC1/SC29/WG1 Call for Contributions for a new lossless image compression standard.

The images in Table 2 are the subset of 8 bits/pixel images from the standard set provided by the ISO/IEC JTC1/SC29/WG1 committee in its Call for Contributions. This is a richer set with a wider variety of images, including compound documents, aerial photographs, scanned, and computer generated images. Compression ratios are given in bits/pixel averaged over color planes. The results for FELICS were extracted, in this case, from [4].

The results in the tables show that LOCO-I significantly outperforms other one-pass schemes of comparable complexity (e.g., FELICS, JPEG-Huffman), and it attains compression ratios similar or superior to those of higher complexity schemes based on arithmetic coding (e.g. Sunset CB9, JPEG-Arithmetic). LOCO-I is within a few percentage points of the best available compression ratios (given by CALIC), at a complexity level close to an order of magnitude lower.

To further illustrate the very favorable location of LOCO-I in a conceptual compression/complexity chart, consider the following: going from lossless JPEG-Huffman to LOCO-I we obtain an average compression gain of 28% (as per Table 2), at a complexity increase estimated at 2X; going from LOCO-I to CALIC-Arithmetic we obtain a compression gain of 4% at a complexity increase close to an order of magnitude. Other schemes tested lie well above the polyline defined by these three points in the conceptual chart.

As for data rates in software implementations, LOCO-I benchmarks at about the same throughput as the UNIX *compress* utility, which is also the approximate throughput reported for FELICS in [10]. Measured compression data rates on an HP9000/735 workstation range from about 500 KB/s for natural images to about 2000 KB/s for compound documents and computer graphics images. LOCO-I decompression is about 10% slower than compression, making it a fairly symmetric system.

References

[1] M. J. Weinberger, J. Rissanen, and R. Arps, "Applications of universal context modeling to lossless compression of gray-scale images," *IEEE Trans. Image Processing*, Apr. 1996. To appear.

[2] J. Rissanen, "Universal coding, information, prediction, and estimation," *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 629–636, July 1984.

[3] J. Rissanen and G. G. Langdon, Jr., "Universal modeling and coding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 12–23, Jan. 1981.

[4] X. Wu, N. Memon, and K. Sayood, "A context-based, adaptive, lossless/nearly-lossless coding scheme for continuous-tone images (CALC)." A proposal submitted in response to the Call for Contributions for ISO/IEC JTC 1.29.12, 1995.

[5] S. Todd, G. G. Langdon, Jr., and J. Rissanen, "Parameter reduction and context selection for compression of the gray-scale images," *IBM Jl. Res. Develop.*, vol. 29 (2), pp. 188–193, Mar. 1985.

Image	LOCO-I	FELICS	Lossless JPEG Huffman	Lossless JPEG arithmetic	CALIC arithmetic
bike	3.59	4.06	4.34	3.92	3.50
cafe	4.80	5.31	5.74	5.35	4.68
woman	4.17	4.58	4.86	4.47	4.03
tools	5.07	5.42	5.71	5.47	4.92
bike3	4.37	4.67	5.18	4.78	4.21
cats	2.59	3.32	3.73	2.74	2.49
water	1.79	2.36	2.63	1.87	1.73
finger	5.63	6.11	5.95	5.85	5.39
us	2.67	3.28	3.77	2.52	2.45
chart	1.33	2.14	2.41	1.45	1.28
chart_s	2.74	3.44	4.06	3.07	2.66
compound1	1.30	2.39	2.75	1.50	1.20
compound2	1.35	2.40	2.71	1.54	1.22
aerial2	4.01	4.49	5.13	4.14	3.80
faxballs	0.97	1.74	1.73	0.84	0.72
gold	3.92	4.10	4.33	4.13	3.82
hotel	3.78	4.06	4.39	4.15	3.68
Average	3.18	3.76	4.08	3.40	3.05

Table 2: Compression results on new test set (in bits/pixel averaged over color planes)

[6] G. G. Langdon, Jr. and C. A. Haidinyak, "Experiments with lossless and virtually lossless image compression algorithms," in *Proc. SPIE*, vol. 2418, pp. 21–27, Feb. 1995.

[7] A. Netravali and J. O. Limb, "Picture coding: A review," *Proc. IEEE*, vol. 68, pp. 366–406, 1980.

[8] S. W. Golomb, "Run-length encodings," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 399–401, July 1966.

[9] R. F. Rice, "Some practical universal noiseless coding techniques," Tech. Rep. JPL-79-22, Jet Propulsion Laboratory, Pasadena, CA, Mar. 1979.

[10] P. G. Howard and J. S. Vitter, "Fast and efficient lossless image compression," in *Proc. of the 1993 Data Compression Conference*, (Snowbird, Utah, USA), pp. 351–360, Mar. 1993.

[11] S. A. Martucci, "Reversible compression of HDTV images using median adaptive prediction and arithmetic coding," in *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1310–1313, IEEE Press, 1990.

[12] R. F. Rice, "Some practical universal noiseless coding techniques - part iii," Tech. Rep. JPL-91-3, Jet Propulsion Laboratory, Pasadena, CA, Nov. 1991.

[13] R. Gallager and D. V. Voorhis, "Optimal source codes for geometrically distributed integer alphabets," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 228–230, Mar. 1975.

[14] N. Merhav, G. Seroussi, and M. J. Weinberger, "Modeling and low-complexity adaptive coding for image prediction residuals." To be presented at the 1996 Int'l Conference on Image Processing, Lausanne, Sept. 1996.

[15] N. D. Memon and K. Sayood, "Lossless image compression: A comparative study," in *Proc. SPIE (Still-Image Compression)*, vol. 2418, pp. 8–20, Feb. 1995.